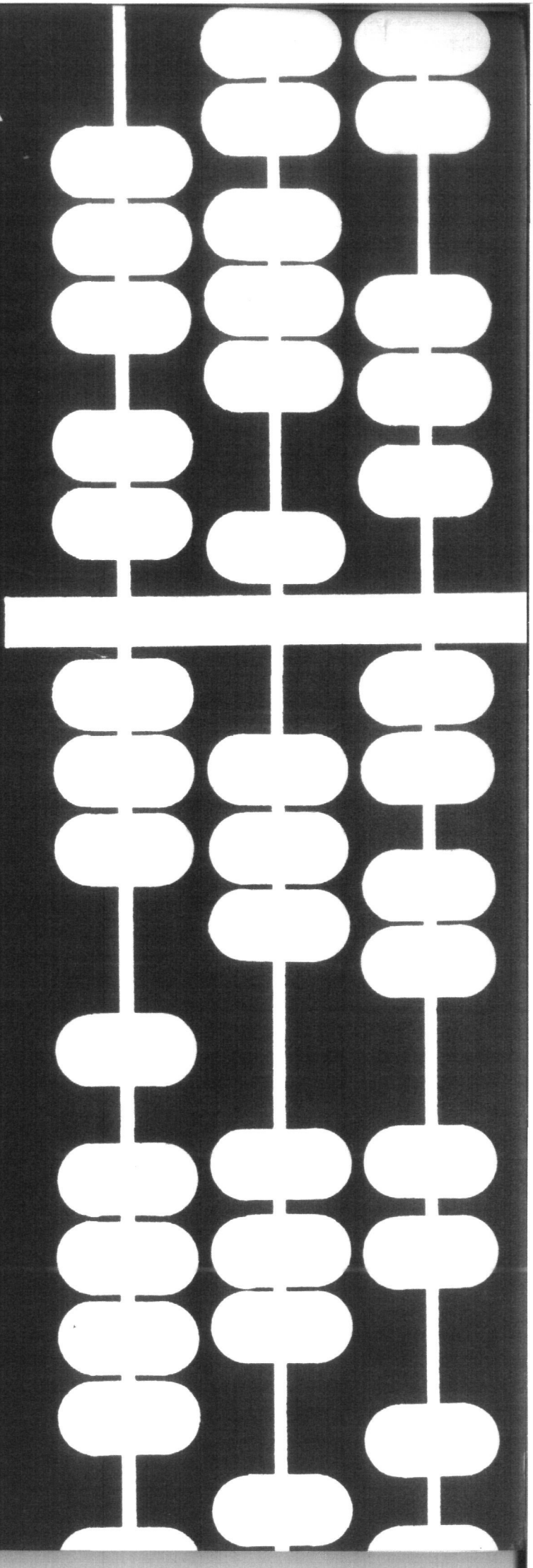
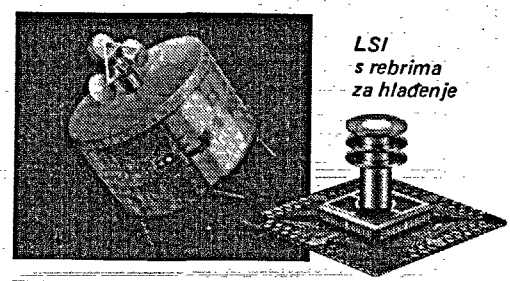
INFORMATICA



1978 3 YU ISSN 0350-5596

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



LSI s rebrima za hlađenje

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

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

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

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

Za dalje informacije obratite se na:

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

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

FUJITSU

Fujitsu Limited·Tokyo, Japan

INFORMATICA

Journal of Computing and Informatics

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

Headquarters : 61000 Ljubljana, Institut "Jožef Stefan", Jamova 39, Phone: (061) 263261, Cable: JOSTIN Ljubljana, Telex: 31 269 YU JOSTIN

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

Printed by: Tiskarna Kresija, Ljubljana

DESIGN: Rasto Kirn

Volume 2 , 1978 — no. 3

CONTENTS

INFORMATICA

časopis za tehnologijo računalništva in probleme informatike
časopis za računarsku tehnologiju i probleme informatike
spisanie za tehnologija na smetanjeto i problemi od oblasta na informatikata

YU ISSN 0350-5596

Letnik 2 , 1978 — številka 3

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

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

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

Pri financiranju revije sodeluje tudi Raziskovalna skupnost Slovenije.

Na podlagi mnenja Republiškega sekretariata za prosveto in kulturo št. 4210-7/78 z dne 19.1.1978, je časopis INFORMATICA strokovni časopis, ki je oproščen temeljnega davka od prometa proizvodov.

Tisk: Tiskarna KRESIJA, Ljubljana

Grafična oprema: Rasto Kirn

VSEBINA

navodilo za pripravo članka

Avtorje prosimo, da pošljejo uredništvu naslov in kratek povzetek članka ter navedejo približen obseg članka (število strani A 4 formata). Uredništvo bo nato poslalo avtorjem ustrezno število formularjev z navodilom.

Članek tipkajte na priložene dvokolonske formularje. Če potrebujete dodatne formularje, lahko uporabite bel papir istih dimenzij. Pri tem pa se morate držati predpisanega formata, vendar pa ga ne vrišite na papir.

Bodite natančni pri tipkanju in temeljiti pri korigiranju. Vaš članek bo s foto postopkom pomanjšan in pripravljen za tisk brez kakršnihkoli dodatnik korektur.

Uporabljajte kvaliteten pisalni stroj. Če le tekst dopušča uporabljajte enojni presledek. Črni trak je obvezen.

Članek tipkajte v prostor obrobljen z modrimi črtami. Tipkajte do črt - ne preko njih. Odstavek ločite z dvojnim presledkom in brez zamikanja prve vrstice novega odstavka.

Prva stran članka :
a) v sredino zgornjega okvira na prvi strani napišite naslov članka z velikimi črkami;
b) v sredino pod naslov članka napišite imena avtorjev, ime podjetja, mesto, državo;
c) na označenem mestu čez oba stolpca napišite povzetek članka v jeziku, v katerem je napisan članek. Povzetek naj ne bo daljši od 10 vrst.
d) če članek ni v angleščini, ampak v katerem od jugoslovanskih jezikov izpustite 2 cm in napišite povzetek tudi v angleščini. Pred povzetkom napišite angleški naslov članka z velikimi črkami. Povzetek naj ne bo daljši od 10 vrst. Če je članek v tujem jeziku napišite povzetek tudi v enem od jugoslovanskih jezikov;
e) izpustite 2 cm in pričnite v levo kolono pisati članek.

Druga in naslednje strani članka:
Kot je označeno na formularju začnite tipkati tekst druge in naslednjih strani v zgornjem levem kotu,

Naslovi poglavij:
naslove ločuje od ostalega teksta dvojni presledek.

Če nekaterih znakov ne morete vpisati s strojem jih čitljivo vpišite s črnim črnilom ali svinčnikom. Ne uporabljajte modrega črnila, ker se z njim napisani znaki ne bodo preslikali.

Ilustracije morajo biti ostre, jasne in črno bele. Če jih vključite v tekst, se morajo skladati s predpisanim formatom. Lahko pa jih vstavite tudi na konec članka, vendar morajo v tem primeru ostati v mejah skupnega dvokolonskega formata. Vse ilustracije morate (nalepiti) vstaviti sami na ustrezno mesto.

Napake pri tipkanju se lahko popravljajo s korekcijsko folijo ali belim tušem. Napačne besede, stavke ali odstavke pa lahko ponovno natipkate na neprozoren papir in ga pazljivo nalepite na mesto napake.

V zgornjem desnem kotu izven modro označenega roba oštevilčite strani članka s svinčnikom, tako da jih je mogoče zbrisati.

Časopis INFORMATICA
Uredništvo, Institut Jožef Stefan, Jamova 39, Ljubljana

Naročam se na časopis INFORMATICA. Predplačilo bom izvršil po prejemu vaše položnice.

Cenik: letna naročnina za delovne organizacije 300,00 din, za posameznika 100,00 din.

Časopis mi pošiljajte na naslov ☐ stanovanja ☐ delovne organizacije.

Priimek.....

Ime.....

Naslov stanovanja

Ulica.....

Poštna številka_____ Kraj.....

Naslov delovne organizacije

Delovna organizacija.....

.....

Ulica.....

Poštna številka_____ Kraj.....

Datum..... Podpis:

.....

instructions for preparation of a manuscript

Authors are invited to send in the address and short summary of their articles and indicate the opproximate size of their contributions (in terms of A 4 paper). Subsequently they will receive the outor's kits.

Type your manuscript on the enclosed two-column-format manuscript paper. If you require additional manuscript paper you can use similar-size white paper and keep the proposed format but in that case please do not draw the format limits on the paper.

Be accurate in your typing and through in your proof reading. This manuscript will be photographically reduced for reproduction without any proof reading or corrections before printing.

Use a good typewriter. If the text allows it, use single spacing. Use a black ribbon only.

Keep your copy within the blue margin lines on the paper, typing to the lines, but not beyond them. Double space between paragraphs.

First page manuscript:
a) Give title of the paper in the upper box on the first page. Use block letters.
b) Under the title give author's names, company name, city and state - all centered.
c) As it is marked, begin the abstract of the paper. Type over both the columns. The abstract should be written in the language of the paper and should not excesed 10 lines.
d) If the paper is not in English, drop 2 cm after having written the abstract in the language of the paper and write the abstract in English as well. In front of the abstract put the English title of the paper. Use block letters for the title. The lenght of the abstract should not be greater than 10 lines.
e) Drop 2 cm and begin the text of the paper in the left column.

Second and succeeding pages of the manuscript:
As it is marked on the paper, begin the text of the second and succeeding pages in the left upper corner.

Format of the subject headings:
Headings are separated from text by double spacing.

If some characters are not available on your typwriter write them legibly in black ink or with a pencil. Do not use blue ink, because it shows poorly.

Illustrations must be black and white, sharp and clear. If you incorporate your illustrations into the text keep the proposed format. Illustration can also be placed at the end of all text material provided, however, that they are kept within the margin lines of the full size two-column format. All illustrations must be placed into appropriate positons in the text by the author.

Typing errors may be corrected by using white correction paint or by retyping the word, sentence or paragraph on a piece of äpaque, white paper and pasting it nearly over errors

Use pencil to number each page on the upper-right-hand corner of the manuscript, outside the blue margin lines so that the numbers may be erased.

računalniška proizvodnja

j.škrubej
r.faleskini

UDK 681.3.002

ELEKTROTEHNA, TOZD Digital, Ljubljana
RAČUNALNIŠKI CENTER UNIVERZE, Ljubljana

Članek govori na splošno o relaciji med profesionalno elektroniko in računalniško proizvodnjo. Neelektronske komponente računalniške proizvodnje niso opisane sistematično, ker smo želeli samo poudariti, da je med računalništvom in elektroniko le dialektična zveza.

COMPUTER PRODUCTION
The article deals with the relation between professional electronics and computer production. The non-electronic components of computer production are not described systematicaly as only the dialectic relation between computers and electronics had to be stressed.

Vključevanje računalništva v profesionalno elektroniko se je v naših razmerah ohranilo v različnih pristopih k delitvi industrijskih dejavnosti, kljub dejstvu, da je na zahodu že dolgo računalništvo samostojna industrijska panoga. Sama razprava o tem v bistvu nominalističnem vprašanju niti ne bi bila smiselna, ko se iz nekega imena ali poimenovanja, ki je tudi v resnici včasih bilo smiselno, ne bi po načelu NOMEN EST OMEN vlekle tudi različne konsekvence.

Smatramo, da je potrebno na vse delitve industrije v panoge gledati dialektično, kar pomeni, da se bodo pod istim pojmom v različnih časih in različnih krajih pojavljale različne vsebine, saj je industrijska dejavnost živa v tem smislu, da se rojeva, razvija in propada.

Razvoj in vzpon računalništva v zahodnem svetu ni bil niti povsem neodvisen od razvoja elektronike niti ni bil od njega povsem odvisen. Elektronika je bila ves čas samo ena od komponent njegovega razvoja.

Brez dvoma pa v računalništvu v današnjem smislu ne moremo govoriti ne da bi upoštevali druge pomembne komponente, ki jih navadno skušamo uokviriti v pojme programska oprema, informatika itd., vključno s posebnimi teorijami deterministčnih tehničnih sistemov, sistemsko analizo, modeliranjem, simulacijami, analogno digitalnimi konverzijami, teorijo relacij med procesi in stanji itd.

Gre nadalje za kibernetiko kot vedo o upravljanju relativno izločenih dinamičnih sistemov, za informatiko kot vedo o zbiranju, procesiranju in uporabi informacij, o delovanju, analizi in sintezi računalniško podprtih informacijskih sistemov. Gre tudi za sistemsko hevristiko, kot vedo o načinih reševanja problemov, za teorije o posebnih jezikih za opisovanje in reševanje problemov in posebne, nove matematične panoge.

V zvezi s tem moramo opozoriti, da predstavlja elektronika samo en del tiste komponente računalnika, ki ji s tujko pravimo hardware in da drugi del hardwara kakor tudi kompleten software ne spadata v elektroniko.

Računalnik je smiselno gledati vsaj kot kompleten produkt z njegovimi hardwarskimi in softwarskimi komponentami, verjetno pa v sodobnih pogojih največkrat kar kot podsistem v določenem avtomatiziranem proizvodnem procesu ali v računalniško podprtem informacijskem sistemu. Če ga gledamo tako, se pokaže računalnik v drugi luči, kjer so neelektronske

komponente vsaj tako pomembne kot elektronske, če pa gledamo njihovo vrednost glede na količino vloženega dela ali glede na cene na svetovnem trgu pa brez dvoma več. Verjetno je smiselno postavljati ravno ti dve merili (količino vloženega dela in njegovo ceno na svetovnem trgu) pri presojanju, kdo je proizvajalec računalnikov in kdo ni.

Gre namreč za to, da mora računalniška proizvodnja pri nas pomagati gospodarstvu kot celoti, da torej ni samo sebi namen, temveč bo imela zelo velik vpliv na avtomatizacijo in kibernetizacijo naše proizvodnje in na razvoj informacijskih sistemov. Gledano s širšega vidika znanstveno tehnološke revolucije, lahko ugotovimo, da tuji in naši družboslovci gledajo na računalništvo kot na specifično komponento znanstveno tehnološke revolucije - ob drugih, med katerimi je tudi elektronika (npr. dr. M. Pečujlić: Prihodnost, ki se je začela, Obzorja revolucije; dr. A. Kirn: Marxovo razumevanje znanosti in tehnike). Pri računalništvu gre za komponento znanstveno tehnološke revolucije, ki seveda ni neodvisna od drugih in katere izločitev je lahko samo pogojna - zaradi potreb analize in obravnavanja, ni pa neodvisna ne npr. od razvoja elektronike niti ne npr. od razvoja splošne teorije sistemov, kibernetike in matematike. Seveda velja isto tudi za elektroniko, tudi njen razvoj ni neodvisen od računalništva, kemije, optike itd.

Ugotavljamo pa, da je možno graditi približno enake računalnike, vzeto v najširšem smislu, torej vključno programsko opremo, oziroma kot podsisteme nekih avtomatiziranih proizvodnih procesov ali računalniško podprtih informacijskih sistemov, s pomočjo različnih elektronskih razvojnih stopenj (stopenj integracije elektronskih elementov). To pomeni, da računalniki niso v celoti absolutno odvisni od razvoja elektronike, ampak je ta odvisnost samo delna, čeprav nikakor ni nebistvena.

Smiselno je gledati na problem še z vidika primerjave z Zahodom. Trdimo, da je postalo podjetje IBM pomemben proizvajalec elektronskih komponent zato, ker proizvaja računalnike in ne obratno.

Vsa ta dejstva: torej, da ima računalnik poleg elektronskih še druge pomembne komponente, da zaradi svojega vpliva v znanstveno tehnološki revoluciji brez dvoma predstavlja pomembno samostojno komponento govorijo za to, da je smiselno tudi pri nas v času, ko se začenja razvoj domače proizvodnje, začeti gledati na računalništvo kot na samostojno panogo, ne glede na neke statične modele delitve industrije na panoge in ne glede na to, da so bili prvi poskusi proizvodnje računalnikov ravno v elektronski industriji (ZUSE v ISKRI). Še bolj kot to pa je pomembno, da se pusti razvoj proizvodnje računalnikov tudi zainteresiranim organizacijam izven profesionalne elektronike, ker bo novo delitev panog avtomatično prinesel sam razvoj te proizvodnje.

Potrebe naše družbe po računalniški tehnologiji, ki so pripeljale do začetkov proizvodnje računalniških sistemov, so tako velike, da jih po ocenah strokovnjakov v naslednjih petih letih ne more zadovoljiti z razpoložljivimi kadri in proizvodnimi potenciali niti jugoslovanska proizvodnja kot celota, kaj šele posamezna proizvajalna OZD. Kakršne koli monopolne težnje bi v tem obdobju objektivno bile lahko le monopolne težnje tujega kapitala. Samoupravno sporazumevanje in delitev dela je edina alternativa za vse potencialne jugoslovanske proizvajalce elementov strojne in programske opreme za računalniške in informacijske sisteme.

Proizvodnja računalniških sistemov ima svojo specifiko, ki se kaže v veliki kompleksnosti problematike, ki jo mora proizvajalec obvladati. Računalniški sistemi so zelo občutljiv element širših informacijskih ali pa proizvodnih sistemov, zato je enako pomembno kot to, da jih sproduciramo, tudi to, da jih vso življenjsko dobo vzdržujemo in izboljšujemo. Pogojno bi proizvodnjo računalniške opreme lahko razdelili na naslednji način:

- Proizvodnja splošne sistemske strojne opreme (hardware) od elementov, podsklopov, sklopov, do določene konfiguracije kot celote.
- Proizvodnja sistemske programske opreme (sistemski software).
- Vključitev strojne in programske opreme v določen širši proizvodnji, informacijski ali drugi širši sistem. Taka vključitev zahteva izdelavo specialne strojne opreme (vmesniki, dajalniki informacij, kontrolne naprave) in specialne programske opreme (aplikacijski software).
- Raziskovalno delo naj zagotavlja stik in razvoj s svetovnim napredkom na področju strojne in programske opreme in razvoj

specifičnih aplikacij, značilnih za proizvodnje, informacijske in druge računalniško podprte sisteme v naši družbi.

- Hiter razvoj računalniške tehnologije v kvalitativnem in kvantitativnem smislu zahteva permanentno in rastočo skrb za kadre; tudi uporaba računalnikov zahteva veliko znanja.
- Potrebno je stalno vzdrževanje strojne in programske opreme. To vzdrževanje je zelo zahtevno in odgovorno, saj so računalniške, strojne in programske komponente pogosto ključni elementi v nekih širših, za družbo zelo pomembnih sistemih.

Na področju proizvodnje standardne strojne opreme vidimo predvsem možnosti, da bi jugoslovanski proizvajalci na osnovi svojih dolgoletnih izkušenj na področju elektronike in fine mehanike lahko začeli proizvajati periferno opremo, ki bi jo bilo možno z ustreznimi prilagoditvenimi strojnimi in programskimi elementi (vmesniki) priključevati na vse obstoječe in nove sisteme domače in tuje proizvodnje.

Sodelovanje na področju raziskovalnega dela in razvojnega dela vidimo v koordinaciji interesov vseh proizvajalcev in družbenih faktorjev v smislu podpore takega raziskovalnega in razvojnega dela, ki bi dajalo na družbeno relevatnih področjih koristne rezultate za vse proizvajalce. Konkretno rečeno, če se razvija neka periferna naprava, naj ima raziskovalna organizacija možnost in interes, da bodo vmesniki razviti za računalniške sisteme vseh proizvajalcev, čeprav bo to periferno napravo eden proizvajal in vgrajeval, drugi pa samo vgrajeval v svoje sisteme. Podobno naj se aplikativna programska oprema piše in dokumentira standarizirano, da jo bo možno prenesti na računalniške sisteme vseh proizvajalcev ne glede na to, kateri od proizvajalcev bo pobudnik razvoja te programske opreme pri neki raziskovalni instituciji.

Na področju izobraževalne dejavnosti bi morali skoordinirati interese proizvajalcev in družbene interese. Prizadevati bi si morali, da bi se v SR Sloveniji čimprej formiral center usmerjenega izobraževanja za področje računalništva. Tak center, na katerem bi morali izšolati večino kadrov za uporabnike in proizvajalce od najnižjega do najvišjega nivoja, bi naša družba potrebovala že brez proizvodnje, v pogojih proizvodnje pa so potrebe po njem in s temi zvezane aspiracije še večje.

Računalniški sistem nima značaja široke potrošnje, temveč je namenjen za uporabo v industriji in gospodarstvu na osnovi katerega se lahko poveča ali zmanjša reproduktivna sposobnost gospodarstva. Zato moramo predvsem uporabnikom računalnikov v naslednjem obdobju posvetiti največjo pozornost in dati ustrezno prioriteto. Kajti pravilna uporaba računalniških sistemov prinaša družbi daleč večje rezultate kot pa sama proizvodnja računalniških sistemov. Zelo važno je povezati vse izkušnje v gospodarstvu po posameznih branžah, kajti nekateri uporabniki imajo že razvite programske in tehnološke rešitve, ki so na svetovnem nivoju.

Zaradi specifičnosti računalniške proizvodnje in pomembnosti uporabe računalniških sistemov ter ob dejstvu, da sama elektronika v računalniškem sistemu s hitrim razvojem tehnologije ni najbolj pomemben del sistema, bi morali proizvodnjo računalnikov in uporabo obravnavati posebej in ne v sklopu elektronske in elektro industrije.

Če tega ne bomo v začetku pravilno opredelili, bo še naprej vladalo zgrešeno prepričanje, da so edino proizvajalci komponent poklicani za proizvodnjo računalniških sistemov, kar pa v svetu ni tako. Poznamo veliko primerov, ko je klasična elektro in elektronska industrija začela s proizvodnjo, pa je po nekaj letih propadla in s tem zapravila velika sredstva. Ravno zaradi specifičnosti proizvodnje računalniških sistemov je edina alternativa delitev dela in specializacija v samoupravno povezanem gospodarstvu.

informacijski sistem s r slovenije

m.krisper

UDK 681.3 (497.12)

REPUBLIŠKI KOMITE ZA DRUŽBENO PLANIRANJE
IN INFORMACIJSKI SISTEM, Ljubljana

Članek obravnava opredelitev, lastnosti, strukturo in organizacijske rešitve Informacijskega sistema SR Slovenije. Skladno določilom Ustave SFR Jugoslavije in Ustave SR Slovenije ter ravni razvoja našega družbeno političnega sistema mora Informacijski sistem SR Slovenije zagotoviti delovnim ljudem in občanom ter njihovim asociacijam celovite, resnične in pravočasne informacije, potrebne za njihovo ustvarjalno vključevanje v procese planiranja, odločanja in upravljanja ter družbenega dogovarjanja in samoupravnega sporazumevanja.

Information System of SR Slovenia

The paper deals with the definition, characteristics and organisational solutions of the information system of SR Slovenia. According to the regulations of the Constitution of SFR Yugoslavia and the Constitution of SR Slovenia and to the level of the developement of our sociopolitical system the information system of SR Slovenia has to offer the working people and all citizens and their associations integral, true and due information that are necessary for their creative participation in planning, decison making and control as well as social negotiations and self management agreements.

Uvod

Glede na doseženo stopnjo socialističnega samoupravljanja je nujno potrebna preobrazba organizacije in prakse informiranja ter s tem graditve učinkovitega družbenega sistema informiranja.

Na Izvršnem svetu Skupščine SR Slovenije je v razpravi gradivo " Razvoj informacijskega sistema SR Slovenije " kot bistvenega dela družbenega sistema informiranja. Prav tako je na zveznem nivoju v tem letu potekalo oz. še teče več akcij za opredelitev družbenega sistema informiranja. Tako je bil sprejet tudi osnutek Zvezne resolucije o osnovah družbenega sistema informiranja, zvezni svet za vprašanja družbene ureditve pa je tudi razpravljal o tej problematiki.

V vseh omenjenih razpravah je prišlo do razčiščevanja mnenj in stališč o vlogi, pomenu in vsebini družbenega sistema informiranja, kar predstavlja solidno osnovo za opredelitev strukture, vsebine, organiziranosti in nosilcev družbenega sistema informiranja, predvsem pa njegove osnovne komponente, informacijskega sistema SR Slovenije.

V tem članku Informacijski sistem SR Slovenije ni obravnavan z znanstvenega vidika, ampak je njegov namen podati predlog opredelitve, strukture in organizacijskih rešitev.

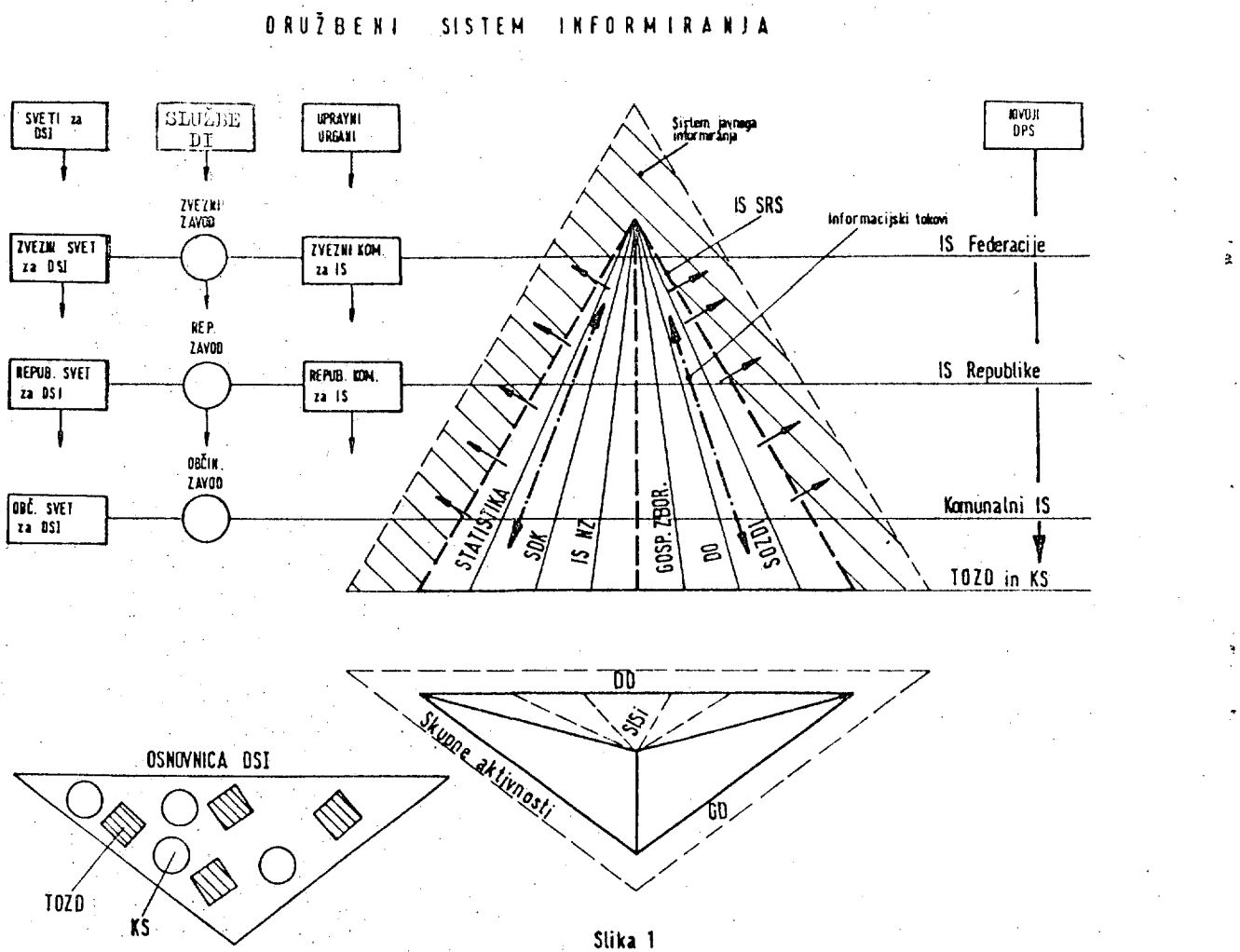
1. Definicija, struktura in vsebina informacijskega sistema SR Slovenije kot sestavnega dela družbenega sistema informiranja.

Informacijski sistem SR Slovenije kot ekzaktna osnova družbenega sistema informiranja, zagotavlja usklajeno evidentiranje, zbiranje, obdelavo, prenos in izkazovanje podatkov in dejstev, pomembnih za procese planiranja, odločanja in upravljanja delavcev in občanov ter njihovih samoupravnih in političnih organizacij.

Z informacijskim sistemom SR Slovenije je potrebno zagotoviti :

- podružbljanje dejavnosti s področja informacijskega sistema, ker ta dejavnost presega ožje tehnične in strokovne okvire,
- samoupravno organiziranje, programiranje dela in medsebojno povezovanje nosilcev dejavnosti,
- razpolaganje z vsemi podatki, ki so pomembni za usklajevanje odnosov v družbeni reprodukciji in usmerjanje družbenega razvoja,
- celovite, objektivne in pravočasne podatke in informacije, ki bodo dostopne vsem, ki jih bodo potrebovali, z izjemo tistih podatkov in informacij, katerih javnost bo z zakonom omejena,
- hiter in učinkovit pretok informacij in podatkov, pomembnih za vse uporabnike informacijskega sistema SR Slovenije,
- preprečitev ponavljanja in neracionalnih postopkov pri zbiranju, obdelavi in prenosu podatkov in informacij,
- povezavo z informacijskim sistemom SFR Jugoslavije,
- usklajenost s konceptom splošnega ljudskega odpora in družbene samozaščite,
- preprečitev možnega monopola in zlorabe informacij in podatkov od strani nosilcev informacijskih sistemov v okviru informacijskega sistema SR Slovenije ali uporabnikov.

Informacijski sistem republike lahko pogojno predstavimo kot informacijski piramido, kot je to prikazano na sliki 1.



Slika 1

Informacijska piramida ima dve osnovni koordinati-horizontalno in vertikalno. Elementi piramide so :osnovna ploskev, tri stranske ploskve in horizontalne prerezne ploskve.

Osnovno ploskev sestavljajo informacijski sistemi temeljnih organizacij združenega dela in krajevnih skupnosti kot osnovnih celic našega družbeno političnega sistema, ki pa so istočasno izvori in ponori informacijskih tokov.

Horizontalne prerezne ploskve predstavljajo informacijski sistemi na nivojih družbeno političnih skupnosti od občine do federacije.

Stranske ploskve predstavljajo skupine informacijskih sistemov za :
- gospodarstvo
- družbene dejavnosti
- skupne aktivnosti,

ki pa imajo vertikalen značaj.

Ovojnico informacijske piramide informacijskega sistema SR Slovenije predstavlja sistem javnega informiranja, ki kot kaže sl. 1. posreduje informacije iz informacijske piramide javnosti. Skupaj, informacijska piramida (informacijski sistem republike) in ovojnica (sistem javnega informiranja) tvorita družbeni sistem informiranje.

Poleg osnovne ploskve informacijske piramide, ki je že definirana, je potrebno definirati še ostale horizontalne prerezne ploskve, ki ustrezajo nivojem družbeno političnih skupnosti :
- komunalni informacijski sistemi na nivoju občin so sestavljeni iz informacijskih sistemov temeljnih organizacij združenega dela in občinskega informacijskega sistema, podrobneje opisanega v 1.1.1.Informacijski sistemi krajevnih skupnosti so vključeni v občinski informacijski sistem, ker zaradi njihove organizacijske in kadrovske šibkosti nimajo lastne tehnične baze,
- informacijski sistemi na ravni republike,ki zajemajo iz komunalnih informacijskih sistemov potrebne informacije oz. posredujejo potrebne informacije nižjim nivojem,
- informacijski sistemi federacije se definirajo analogno.

Informacijski sistemi z vertikalnim značajem, ki formirajo tri stranske ploskve po dejavnostih so :

za gospodarske dejavnosti :
- informacijski sistemi delovnih organizacij, SOZD-ov in asociacij združenega dela,
- informacijski sistemi poslovnih bank,
- informacijski sistemi samoupravnih interesnih skupnosti v gospodarstvu,
- informacijski sistemi splošnih združenj gospodarstva v Gospodarski zbornici,

za družbene dejavnosti :
- informacijski sistem izobraževalne skupnosti,
- informacijski sistem raziskovalne skupnosti,
- informacijski sistem zdravstvene skupnosti,
- informacijski sistem skupnosti pokojninskega in invalidskega zavarovanja in
- drugih samoupravnih interesnih skupnosti,

splošni informacijski sistemi za skupne aktivnosti :
- statistični sistem,
- informacijski sistem SDK,
- informacijski sistem Narodne banke,
- informacijski sistem notranjih zadev,
- informacijski sistem narodne obrambe,
- indok sistem,
- itd.

Večina naštetih informacijskih sistemov je institucionaliziranih na ravni republike oz. federacije, napajajo pa se z informacijami vseh horizontalnih nivojev. Kljub nakazanim informacijskim tokovom v informacijski piramidi(sl. 1) od osnovne ploskve proti vrhu in nazaj (izvori in ponori v temeljnih organizacijah združenega dela in krajevnih skupnostih), pa našteti institucionalizirani informacijski sistemi še vedno koncentrirajo in obdelujejo informacije predvsem na nivoju in za potrebe republike oz. federacije. Obratne smeri toka informacij proti osnovni ploskvi še ni, vsaj ne v zadotsni meri.

Ker novi koncept poudarja pomen horizontalnih informacijskih sistemov na nivoju vseh družbeno političnih skupnosti, bo te institucije (vertikalne) treba ustrezno reformirati tako, da bodo ne samo zajemale, temveč tudi posredovale informacije nazaj občinam, delovnim organizacijam in krajevnim skupnostim.

Informacijski sistem SR Slovenije se, kot že ponazarja piramida, gradi od spodaj navzgor " integralnost " sistema pa je mišljena v smislu funkcionalnih povezav njegovih komponent. Sistem mora zagotavljati samostojnost posameznih informacijskih sistemov v okviru Informacijskega sistema SR Slovenije z istočasnim povezovanjem in usklajevanjem vseh komponent v celovit in medsebojno povezan Informacijski sistem SR Slovenije.

Ta samostojnost izhaja iz dejstva, da mora vsak nosilec procesov planiranja, odločanja in upravljanja na nekem področju delovanja sam graditi lastni informacijski sistem, ki bo lahko optimalno zadovoljeval njegove specifične potrebe s tem, da bo smisleno uporabil obstoječe osnovne oz. posebne baze podatkov, ki so že zgrajene in vključene v informacijski sistem SR Slovenije.

Pri tem je poseben poudarek na komunalnem informacijskem sistemu (1.1), ki ne bo samo pasivno posredoval informacije vertikalnim institucijam, kot so statistika, SDK,Republiški sekretariat za notranje zadeve itd.,temveč bo sam formiral ustrezne baze podatkov za svoje potrebe, naprej pa posredoval agregate. Kar pa je najbolj pomembno, bo dobival vse potrebne informacije iz teh institucij.Vse baze podatkov seveda ne bo smotrno formirati na nivoju občine (in v vseh občinah), prav tako si te porazdelitve baz podatkov po vertikali ni treba predstavljati preveč fizično, vsekakor pa gre za to, da je ažuriranje in dostop do podatkov za potrebe občin v njihovi pristojnosti.

Omenjena preobrazba " državnih " inatitucionaliziranih informacijskih sistemov gre torej v smeri integralnega povezovanja informacijskih sistemov z vertikalnim in horizontalnim značajem.

Sodobna računalniška tehnologija kot infrastruktura informacijskega sistema SR Slovenije v polni meri in na osnovi načel porazdeljenih obdelav in baz podatkov omogoča vzpostavitev tako zasnovanega informacijskega sistema. Pri tem je potrebno upoštevati naslednja osnovna načela :
- potrebe po informacijah višjih nivojev morajo biti vgrajene v nižje nivoje,
- na nivoju občin formirati porazdeljene baze podatkov za lokalne potrebe s tem, da se višjim nivojem posredujejo samo potrebni oz. agregirani podatki,
- vsi informacijski sistemi, ki so dosedaj služili predvsem potrebam odločanja na ravni republike oz. federacije, se morajo usmeriti tudi v posredovanje informacij za potrebe občin, delovnih organizacij in krajevnih skupnosti.

Nosilci informacijskih sistemov oz. organi in organizacije, pooblaščeni za zbiranje, obdelavo in izkazovanje podatkov morajo delovati v skladu s sprejetimi programi, enotno metodologijo in tehničnimi standardi, predvsem pa v smeri preprečevanja prekrivanja in ponavljanja enakih podatkov ter podvajanja baz oz. delov baz podatkov.

Glede na kompleksnost in dinamičnost našega družbeno političnega sistema in s tem potrebe po pravočasnem in kvalitetnem uravnavanju procesov v sistemu, mora informacijski sistem SR Slovenije zagotoviti informacije v " realnem " času, torej mora biti evidentiranje,zbiranje in obdelava podatkov o pojavih in dejstvih izvršena tedaj, ko ta nastanejo.

Za potrebe planiranja, predvsem dolgoročnejših in srednjeročnejših nalog pa mora informacijski sistem zagotoviti ustrezne retrospektivne obdelave oz. napovedovanja na osnovi statističnih analiz oz. raziskav, ki pa morajo temeljiti na istih evidencah oz. registrih po načelu enkratnosti, racionalnosti ter enotne metodologije zbiranja istovrstnih podatkov in formiranja baz podatkov.

Dinamičnost informacijskega sistema kot funkcionalnega predpogoja procesov odločanja in upravljanja je kategorija, ki jo bo treba vgraditi v obstoječe informacijske sisteme, ki so predvsem statični, naravnani zlasti za statistično uporabo in temeljijo le izjemoma na v realnem času vzdrževanih registrih in evidencah. Tako informacijski sistem ne gre fizično enačiti s statistiko in evidencami, katerim manjka ravno aktivna oz. dinamična komponenta. Kompleksnost informacijskega sistema v naši socialistični in samoupravni družbi je v primerjavi z informacijskimi sistemi v nesamoupravni družbi neprimerno večja. V slednji informacijski sistem deluje samo za potrebe autokratskih oz. vodstvenih in političnih struktur, medtem ko so delovni ljudje samo sprejemniki instruktivnih informacij, oddajajo pa informacije o izvrševanju nalog.

Informacijski sistem v samoupravni družbi posreduje informacije delovnim ljudem in občanom ter njihovim asociacijam s tem,da so sami tudi aktivno vključeni v procese odločanja in upravljanja.

Za vzpostavitev tako zastavljenega informacijskega sistema SR Slovenije bo treba sprejeti ustrezne organizacijske rešitve oz. formirati ustrezne organe in institucije, ki bodo sposobne realizirati predlagani koncept.

1.1. Komunalni informacijski sistem

Komunalni informacijski sistem je kot je prikazano na sl. 2 a, sestavljen iz informacijskih sistemov temeljnih organizacij združenega dela in občinskega informacijskega sistema.

Informacijski sistem temeljnih organizacij združenega dela (oz. delovnih organizacij) ki je shematsko prikazan na sl. 2b, je težje obravnavati zaradi raznolikosti organizacije posameznih delovnih organizacij, vendar pa so posamezne splošne značilnosti opisane v podpoglavju 1.1.1.

Posebno pomembne so povezave tega sistema v okviru komunalnega informacijskega sistema oz. povezave z ustreznimi nivoji vertikalnih institucij (statistika, SDK,itd.). Mišljena je povezava na nivoju računalniških medijev ali telekomunikacijskih zvez z računskimi centri(centrom) omenjenih informacijskih sistemov, kar seveda predpostavlja unifikacijo, standardizacijo itd.

Občinski informacijski sistem je shematsko prikazan na sl. 2c, opisan pa v 1.1.2., vendar ga je kot osnovnega na najnižjem (občinskem)nivoju družbenopolitičnih skupnosti treba podrobneje obravnavati.

1.1.1. Informacijski sistemi temeljnih organizacij združenega dela

Potreba po modernizaciji informacijskih sistemov nastopa kot logična posledica stalnih zahtev našega gospodarskega razvoja po višji produktivnosti, po konkurenčnosti na tujih trgih in po sledenju sodobne tehnologije.

S sprejemom zakona o združenem delu se je tudi število potrebnih informacij za odločanje v temeljnih organizacija združenega dela ter ostalih oblikah organiziranja združenega dela bistveno povečalo. Z modernizacijo informacijskih sistemov oz. uvajanjem AOP bomo zmanjšali pritisk na dodatno zaposlovanje administrativnih in drugih režijskih delavcev. Zelo pogosto je namreč izvedba določil zakona o združenem delu nujno povezana z modernizacijo informacijskih sistemov.

Uporaba opreme za AOP za podporo informacijskega sistema v proizvodnji še ni zaživela, najpogosteje se računalniki uporabljajo za avtomatiziranje računovodske in finančne funkcije oz. obračun osebnih dohodkov. Za učinkovitejše procese odločanja, planiranja in vodenja bo uporabo računalniške opreme potrebno vključiti v neposredno opravljanje funkcij, kot so npr. :
- planiranje kapacitet,
- izdaja delovnih nalogov,
- terminiranje operacij,
- spremljanje proizvodnje.

z možnostjo povratne zveze oz. sprotnega uravnavanja. Postopno bo potrebno v delovnih organizacijah zgraditi integrirane informacijske sisteme s postopnim uvajanjem in povezovanjem posameznih podsistemov,,kot so vodenje proizvodnje, nabava, prodaja z vodenjem skladišč, planiranje in razvoj, finančni podsistem. Osnova podsistemov informacijskih sistemov je sistemska baza podatkov z osnovnimi bazami podatkov o :
- materialih, podsklopih in sklopih,
- strojih in napravah (delovnih mestih),
- tehnoloških postopkih,
- delavcih.

Za tovrstne sisteme ponujajo računalniške firme programske rešitve ali predlagajo koncepte. Te rešitve je nemogoče brez predelav prilagoditi našim razmeram, zato pa bo potrebno vložiti veliko lastnega truda in v ta namen formirati ustrezne softwareske hiše.

Pri tem je pomembna vloga Gospodarske zbornice Slovenije, ki mora postati nosilec povezovanja, dogovarjanja in sporazumevanja bodisi na organizacijskem ali strokovnem nivoju. Na njeno iniciativo bi se tudi lahko formirala biblioteka enotne programske opreme za tiste elemente poslovnih informacijskih sistemov, ki bodo služili povezavi z informacijskim sistemom SR Slovenije.

1.1.2. Občinski informacijski sistem

Novi pogledi na oblikovanje informacijskih sistemov so se uveljavili v razvitih državah najprej na ravni lokalnih organov oblasti,to je v občinah in mestih. Ti informacijski sistemi se običajno označujejo kot urbani informacijski sistemi ali kot občinski informacijski sistemi ali pa kot informacijski sistemi na področju uprave. Občinski informacijski sistem ima največji pomen ker je najbližji virom elementarnih podatkov, iz katerih so sestavljeni bodisi posamični skupni podatki, bodisi agregati podatkov ali pa informacije za potrebe uprave in družbe nasploh. Uprava na ravni občine ima svoje lastne potrebe po podatkih ter vsa ustrezna pooblaštila za zbiranje podatkov in za ustanavljanje mehanizma za pridobivanje podatkov. V okviru integriranega občinskega informacijskega sistema je mogoče opraviti tudi delitev na najpomembnejše dele, ki ta sistem sestavljajo. Ti deli so v odnosu na občinski informacijski sistem njegovi podsistemi, obenem pa pomenijo sami zase zaokrožene sisteme.

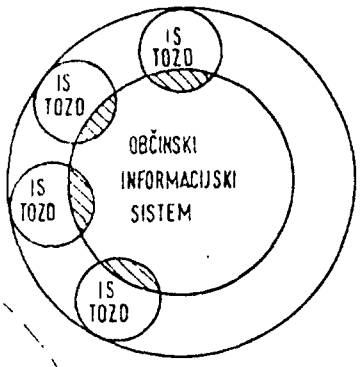
1.1.2.1 Podsistemi občinskega informacijskega sistema

Na podlagi znanstvenih raziskav in praktičnih izkušenj, ki so se izoblikovale ob prvih avtomatskih obdelavah podatkov in ob kasnejših uvajanjih informacijskih sistemov v občinski upravi, je danes mogoče opredeliti področja, ki sestavljajo občinski informacijski sistem. Smiselno gre za naslednje sisteme :
- podsistem prebivalstva
- podsistem delovnih organizacij
- prostorski podsistem
- finančni podistem.

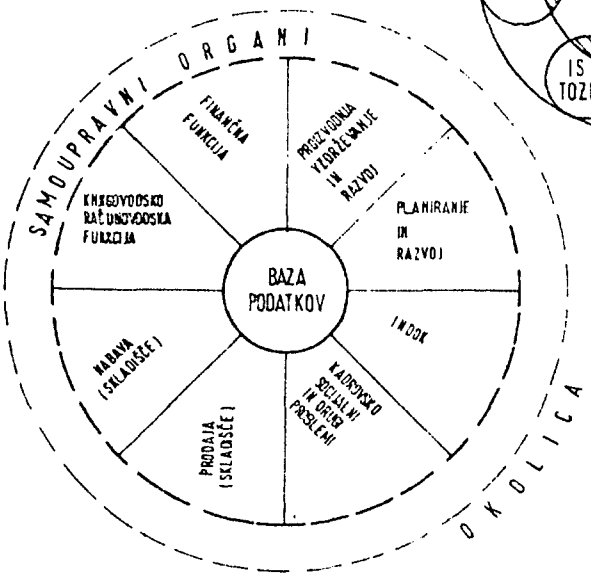
Prednjim štirim informacijskim sistemom, za k katere je značilna visoka stopnja integracije podatkov - najprej znotraj vsakega sistema, nato pa med vsemi sistemi skupaj, je mogoče dodati še dve specifični področji. To sta:
- INDOK (informacijsko dokumentacijski) sistem
- sistem znanstvenih metod in tehnik.

Ta slednja sistema temeljita na integrirani obdelavi podatkov. Vendar sta tako tesno navezana na lokalno področje, za potrebe katerega ju uporabljamo, da ju upravičeno lahko uvrstimo v občinski informacijski sistem.

KOMUNALNI INFORMACIJSKI SISTEM

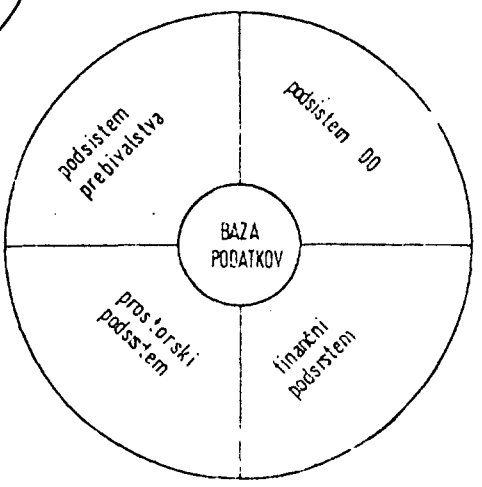


Slika 2a



INFORMACIJSKI SISTEM DO

Slika 2b



OBČINSKI INFORMACIJSKI SISTEM

Slika 2c

Podsistem prebivalstva je na prvem mestu med informacijskimi sistemi (podsistemi) v občinskem informacijskem sistemu, ker je najobsežnejši in ker daje najustreznejše izhodišče za začetek avtomatskih obdelav na področju občinske uprave. Omogoča tudi najenostavnejšo povezavo z drugimi informacijskimi sistemi.

Podsistem prebivalstva zajema tudi vse osnovne podatke, ki se nanašajo na prebivalce določenega teritorija. Pri tem ne gre samo za statistične in demografske podatke o prebivalstvu, ampak tudi za podatke s področja zdravstva, socialnega zavarovanja in socialne varnosti, izobraževanja in vzgoje, skratka vse podatke v zvezi z upravnimi organi in opravili javnih organizacij z območja občine, ki se nanašajo na njene občane in druge prebivalce. Temeljna informacijska enota v sistemu je fizična oseba. Zaradi tega nekateri avtorji govore tudi o informacijskem sistemu fizičnih oseb, v prvih začetkih pa se je uporabljal izraz register prebivalstva ali banka podatkov o prebivalstvu. Vsako fizično osebo v sistemu identificiramo s pomočjo posebnega znaka, ki ga imenujemo osebna matična številka.

Podsistem delovnih organizacij pomeni z družbeno ekonomskih vidikov izredno pomemben del modernega informacijskega sistema v upravi.

Osnovna zamisel za vzpostavitev tega sistema je pravzaprav registriranje podatkov o najrazličnejših dejavnostih , ki se odvijajo na določenem teritoriju. Ker so vse dejavnosti, ki se odvijajo dejansko institucionalizirane v obliki raznih organizacij, pomenijo organizacijske subjekte in na ta način tudi temeljne informacijske enote v okviru sistema - vsaka organizacija je identificirana z ustrezno registrsko številko.

Prostorski podsistem je med vsemi (pod)sistemi najbolj kompleksen. Za integriranje podatkov znotraj tega sistema so potrebne dolgotrajne priprave in študije. Razlog za to je treba iskati v okoliščini, da gre za značilno interdisciplinarno področje,kjer je treba uresničiti tesno sodelovanje strokovnjakov s področja upravnega prava, s področja urbanizma in gradbeništva ter prostorskega planiranja ter s področja informatike.

Prostorski podsistem, ki deluje na določenem teritoriju v lokalni upravi, zajema podatke o zemljišču ter o vseh fiksnih objektih, ki se nahajajo nad ali pod zemeljsko površino. Takšni objekti so npr. stanovanjske, poslovne in druge zgradbe, ceste, mostovi, električno omrežje nad površino, komunalne naprave in instalacije pod površino itd. V bolj izpopolnjeni verziji lahko vsebuje tudi podatke o

naravnih danostih. Temeljna informacijska enota v sistemu je parcela,dopolnilne enote pa so zgradbe ali bloki zgradb ter sečišča koordinatnega sistema.

Finančni podsistem podpira vse upravne naloge in opravila, ki so neposredno povezane s pridobivanjem ali dodeljevanjem finančnih sredstev, skratka s spremljanjem stanja in sprememb v plačilnem prometu (bančnih računih). Ta dejavnost se odvija prek službe za finančno knjigovodstvo. Temeljna identifikacijska enota v finančnem informacijskem sistemu je račun uporabnika pri službi družbenega knjigovodstva ali v poslovni banki. Identifikacijska oznaka te enote je številka računa.

Vsi podatki formirajo ustrezno bazo oz. baze podatkov, ki so vir podatkov (porazdeljene baze podatkov) informacijskih sistemov z vertikalnim značajem (SDK, statistika, Republiški sekretariat za notranje zadeve, poslovne banke itd.) po principu kontrolirane redudance (vsi enaki podatki se nahajajo samo ne enem mestu, seveda v okviru tehničnih možnosti).

2. Organizacijske rešitve

V okviru družbenega sistema informiranja deluje vrsta v poglavju 1. obravnavanih institucij: statistika, SDK, Narodna banka, Republiški sekretariat za notranje zadeve itd. Poleg teh so se formirale tudi INDOK službe, ustrezne službe samoupravnih interesnih skupnosti, delovnih organizacij ipd.

Te institucije se v skladu s spremembami družbeno ekonomskih in političnih odnosov transformirajo, vendar je stopnja te transformacije različna. Potrebno je vzpostaviti mehanizem dejanske samoupravne transformacije v smislu funkcionalnih povezav vseh teh institucij v enoten informacijski sistem republike, ki bo optimalno služil potrebam graditve samoupravnih družbeno ekonomskih odnosov.

V ta namen so potrebne ustrezne organizacijske rešitve v smislu formiranja ustreznih družbeno političnih,upravnih in operativnih (strokovnih)teles oz. institucij. Potrebno bo tudi vzpostaviti samoupravno organiziranost nosilcev informacijskih sistemov in uporabnikov, in sicer v smislu oblikovanja samoupravne interesne skupnosti za informacije, v kateri bi bil uveljavljen bolj učinkovit in celovit interes uporabnikov informacij.

Predlogi, ki sledijo, izhajajo iz razprave in stališč o družbenem sistemu informiranja Zveznega sveta za vprašanja družbene ureditve in sprejetega osnutka zvezne resolucije o temeljih družbenega sistema informiranja.

2.1. Svet za družbeni sistem informiranja

Kot družbeno telo bi bilo potrebno na nivojih družbeno političnih skupnosti, torej v republiki in občini formirati Svet za družbeni sistem informiranja. Svet naj bi bil koorinacijski odbor, ki bi usmerjal delo vseh dejavnikov v okviru sistema, sprejemal program dela, skrbel za zagotovitev objektivnosti, resničnosti in samoupravne naravnanosti operativnih in strokovnih služb v okviru sistema.

Svet naj bi se formiral skladno z zakonom o svetih in bi bil za svoje delo odgovoren skupščini družbeno politične skupnosti. Člane sveta bi imenovala skupščina družbeno politične skupnosti.

2.2. Republiški komite za informacijski sistem

Za opravljanje upravne funkcije na področju informacijskih sistemov je potrebno formirati ustrezni republiški organ : Republiški komite za informacijski sistem. Komite bi se formiral z zakonom o državni upravi.

2.3. Zavod za informatiko in informacijski sistem

Za opravljanje strokovne funkcije iz širšega kompleksa informatike in informacijskih sistemov vključno z obravnavanjem problematike izobraževanja,raziskovalnega dela domače proizvodnje AOP opreme, komunikacijskega omrežja za prenos podatkov itd. je potrebno formirati Zavod za informatiko in informacijski sistem.

Zavod bi v okviru svoje strokovne funkcije direktno izvajal in kordiniral ukrepe in akcije za razvoj informacijskega sistema SR Slovenije oz. informacijskih sistemov, predlagal predpise in norme obnašanja vseh dejavnikov v okviru informacijskih sistemov, koordiniral razvoj sodelujočih v okviru informacijskega sistema SR Slovenije, skrbel za izgradnjo in delovanje komunikacijskega omrežja ter neposredno opravljal in koordiniral strokovne naloge za delovanje tehnološke in vsebinske komponente informacijskega sistema SR Slovenije.

2.4. Služba za družbeno informiranje

Za optimalno delovanje informacijskega sistema SR Slovenije je potrebno formirati avtonomne, neodvisne organizacije za opravljanje operativnih in strokovnih funkcij od občine do republike (in federacije), ki bi bile odgovorne neposredno skupščinam družbeno političnih skupnosti (preko Sveta za družbeni sistem informiranja.

V ta namen je potrebno formirati Službe za družbeno informiranje (ali Centre ali ....).

Financiranje službe bi bilo iz proračuna na osnovi družbenih dogovorov in sporazumov družbeno političnih skupnosti, družbeno političnih organizacij itd., oz. z opravljanjem storitev za uporabnike.

Službe bi se formirale z Zakonom o družbenem sistemu informiranja, bile pa bi medsebojno povezane po vertikali in horizontali.

Službe bi imele pravico in možnost, zagotovljeno z Zakonom o družbenem sistemu informiranja, da iz statistike ter informacijskih sistemov SDK, Republiškega sekretariata za notranje zadeve, delovnih organizacij, občin itd. prevzemajo podatke oz. informacije in jih posredujejo vsem, ki jih potrebujejo za opravljanje funkcij planiranja, upravljanja in odločanja. Službi se mora dovoliti neposreden in posreden dostop do informacij. Vsi podatki morajo biti dostopni javnosti oz. uporabnikom, razen tistih, katerih javnost je omejena z Zakonom in sicer brezplačno ali za plačilo.

V te namene služba ne bi razpolagala z lastno banko podatkov, razen meta banke podatkov (banka podatkov o obstoječih bazah podatkov), ki bo omogočila dostop do vseh informacij v okviru informacijskega sistema SR Slovenije. V ta namen bodo Službe morale biti opremljene z ustrezno računalniško in komunikacijsko opremo in bodo predstavljale vozlišča informacijske mreže informacijskega sistema SR Slovenije.

Služba bi v okviru svoje neodvisnosti imela pravico preverjanja podatkov pri institucijah, ki jih ji posredujejo, da bi lahko prevzemala ustrezno odgovornost za njihovo resničnost.

Obveznost Službe bi bila, da posreduje informacije vsem, ki jih potrebujejo v skladu z Zakonom predpisanimi omejitvami.

Uporabniki bi bili : občani, delovne organizacije, družbeno politične skupnosti, upravni organi, družbeno politične organizacije itd.

Da ne bi Službe postale centri odtujene moči, bi si skupščine družbeno političnih skupnosti, katerim bodo odgovorne, preko Svetov in ustrezno formiranih organov (komisij) morale zagotoviti ustrezen družbeni nadzor. Ta nadzor bi bil po drugi strani tudi v smislu zagotovitve neodvisnosti Službe in družbene podpore.

3. Tehnološka baza informacijskega sistema SR Slovenije

Informacijski sistem SR Slovenije se mora graditi na temelju skupno dogovorjenih osnov, kot so : minimalna vsebina, ki je skupnega pomena za celotno skupnost, enotna metodologija, klasifikacija in nomenklatura ter tehnični standardi za povezovanje v informacijsko mrežo za prenos podatkov in informacij.

Informacijski sistem SR Slovenije se bo gradil v skladu s sodobnimi znanstvenimi dosežki na osnovi porazdeljenih x/ in medsebojno povezanih baz podatkov ter informacijske mreže za obdelavo, prenos preklapljanje in izkazovanje podatkov in informacij.

Informacijsko mrežo iz prejšnjih odstavkov tvorijo :
- računalniški sistemi, ki delujejo na principih porazdeljenih obdelav in baz podatkov ter tako omogočajo formiranje, shranjevanje in pristop k tem bazam podatkov,
- mreža komunikacijskih procesorjev s koncentratorji in vodi, ki omogočajo prenos in preklapljanje podatkov in informacij, oz. povezavo med računalniškimi sistemi.

Za vzpostavitev tako zasnovane mreže bo potrebno da :
- računalniški sistemi, ki bodo medsebojno povezani in s katerimi bodo upravljali vsi nosilci informacijskih sistemov zadovoljujejo tehnične pogoje za vključitev v informacijsko mrežo SR Slovenije,
- komunikacijske elemente informacijske mreže bo potrebno graditi v sodelovanju in okviru ustreznega programa skupnosti PTT prometa SR Slovenije.

x/ Pojem porazdeljena obdelava na računalniškem sistemu vključenem v informacijsko mrežo pomeni, da ta sistem obdeluje samo podatke za potrebe okolja, v katerem je postavljen,povezave pa so v smislu posredovanja oz. sprejemanja podatkov in informacij od drugih sistemov, vključenih v mrežo. Podobno se za potrebe okolja, v katerem je ta sistem postavljen formira potrebna (porazdeljena) baza podatkov.

Zaključek

Razprave o gradivu Razvoj Informacijskega sistema SR Slovenije v organih Izvršnega sveta Skupščine SR Slovenije so v glavnem zaključene, tako da bo gradivo v naslednji fazi obravnavano v Skupščini SR Slovenije. Predloženi koncept bo na ta način doživel preverjanje v delegatski bazi.Javna razprava bo pokazala ustreznost začrtanih usmeritev ter jih dopolnila s v družbeni praksi verificiranimi potrebami.

L I T E R A T U R A

/1/ Razvoj Informacijskega sistema SR Slovenije. Republiški komite za družbeno planiranje in informacijski sistem,Ljubljana,okt.1978

/2/ Predlog za izdajo zakona o temeljih informacijskega sistema SR Slovenije. Republiški komite za družbeno planiranje in informacijski sistem, Ljubljana,okt.1978

/3/ Zasnova družbenega dogovora o razvoju informatike in izgradnji informacijskega sistema SR Slovenije. Republiški komite za družbeno planiranje in informacijski sistem,Ljubljana,okt.1978

/4/ Razprave IX. in XII. seje Zveznega sveta za vprašanja družbene ureditve o družbenem sistemu informiranja. Brioni, nov. 1977 in april 1978

/5/ Družbeni sistem informiranja v združenem delu. XII. jugoslovanski simpozij,INTERBIRO 78, Zagreb, okt.1978

/6/ Problemtika povezovanja informacijskih sistemov v javni upravi in varstva osebne integritete občanov. Inštitut za javno upravo pri Pravni fakulteti v Ljubljani,Ljubljana,1977

/7/ Projektiranje i izgradnja informacionog sistema SR Hrvatske. Republički savjet za informatiku SR Hrvatske, Zagreb 1977

razširitev mikroračunalniškega sistema

a.p.železnikar

UDK 681.3 - 181.4

Odsek za računalništvo in informatiko
Institut "Jožef Stefan"
Ljubljana

Članek opisuje razširitev mikroračunalniškega sistema s procesorjem Z-80 v obliki posebnega (dodatnega) modula, ki ga sestavljajo avtonomna ura realnega časa, 32k zlogov pomnilnika tipa RAM in 5k zlogov pomnilnika tipa EPROM. Modul je priključen na standardno vodilo sistema Z-80 ter je z vodilom povezan preko krmiljenih ojačevalnikov treh stanj. Ura realnega časa steče s programsko nastavitvijo ter uporablja vse štiri kanale časovnika/števnika Z-80-CTC; resetiranje ure je programsko (z naslovom vrat) ter je neodvisno od sistemskega resetiranja. Pomnilnik tipa RAM za 32k zlogov je sestavljen iz 16k-bitnih pomnilnih integriranih vezij; njegovo delovanje preizkusimo s testirnim programom RAMTE1, ki je v članku opisan. Pomnilnik tipa EPROM za 5k zlogov uporabniškega operacijskega sistema sestavljajo integrirana vezja tipa 2708 (1k zlogov). Članek opisuje delovanje, gradnjo in preizkušanje enot (ura, pomnilnika RAM in EPROM) zgrajenega modula (dvojni evropski format plošče). Modul je mogoče npr. takoj uporabiti kot razširitev računalnika na eni plošči SDB-80 (Mostek).

An Extension of Microcomputer System. This article describes an extension of Z-80 microcomputer system by a hardware (and partly software) module; this module consists of an autonomous real time clock, 32k bytes of RAM, and 5k bytes of EPROM. This module is connected to a standard Z-80 bus via controlled three-state drivers. The real time clock starts by a programmed setting and uses the four channels of counter/timer circuit Z-80-CTC; clock reset is programmed (using port address) and independent from the system reset. 32k bytes RAM uses 16k bit memory integrated circuits; testing of new RAM is accomplished by the RAMTE1 program being described in the article. EPROM for 5k bytes of user oriented operating system uses 2708 IC's (1k bytes each). The article deals with operation, building and testing of units (real time clock, RAM and EPROM) of the extension module. This module can be immediately used as the SDB-80 (Mostek) extension.

1. Uvod

Osnovni mikroračunalniški sistem je navadno sestavljen iz centralne procesne enote, nekaj tisoč lokacij pomnilnika tipa ROM in RAM, časovniškoštevniškega vezja ter enot za serijske in paralelne V/I kanale. Tak osnovni sistem je zgrajen na eni sami (tiskani) plošči, ima pa svoje vhode in izhode izolirane, npr. z ojačevalniki treh stanj. K osnovnemu sistemu sodi tudi majhen operacijski sistem obsega enega do nekaj tisoč zlogov, ki zmore razem monitorja še zbirnik, urejevalnik ter nalagalnik za povezovanje in premeščanje programskih modulov.

Opisana osnovna mikroračunalniška konfiguracija je dokaj primerna za razvoj programske opreme, še zlasti tiste, za katero želimo imeti dobro vzdrževalno dokumentacijo, ko gradimo odprte (ne do kraja zaključene) uporabniške programske sisteme. Naraščanje uporabniških zahtev pripelje tudi do potrebe dodajanja materialne opreme zaradi pomanjkanja virov ali zaradi njihove (obstoječe) neustreznosti. Tako se pojavi zahteva za povečanje števila V/I kanalov (serijskih in paralelnih), za dodajanje perifernih krmilnikov (DMA, gibki diski, kasetni pogoni, časovniki, prekinjevalniki, pretvorniki itn.), za povečanje uporabniškega operacijskega sistema in uporabniških podatkov nujno potrebni dodatni pomnilnik (npr. tipa EPROM in RAM) itn.

V članku opisana razširitev zajema tako dodatno "tiskano" (bolje ožičeno) ploščo, ki vsebuje avtonomno delujočo uro realnega časa, s sodobnim časovniškim krmilnikom in prekinitvenim mehanizmom; nadalje imamo na tej plošči še 32k zlogov dodatnega dinamičnega pomnilnika tipa RAM ter 5k zlogov pomnilnika tipa EPROM, ki rabi za razširitev obstoječega operacijskega sistema. Razširitev se nanaša na sisteme s procesorjem Z-80 in dodani modul je prilagojen standardnim sistemskim zahtevam (uporablja celotno vodilo, sprejema signale za osposobitev svojih podenot, oddaja signale za osposobitev in preprečitev delovanja enot izven modula, vsebuje programirani "reset" itn.). V modulu uporabljena integrirana vezja so "klasične" vrste. In kar je tudi pomembno: modul je bil praktično preizkušen, je zanesljiv in njegovo gradnjo je moč ponoviti.

Pri razvoju zapletenih modulov za mikroračunalniški sistem moramo z vso pozornostjo upoštevati časovne diagrame procesorja in perifernih krmilnikov. Le z natančnim študijem časovnih diagramov signalov postane načrtovanje zanesljivo in razumljivo, še zlasti tedaj, ko moramo povezovati materialno in programsko opremo oziroma predvideti možnosti reševanja uporabniških nalog.

2. Razširitev pomnilnika tipa RAM

Pomnilnik tipa RAM, ki je razširitev pomnilnika na glavni plošči (plošča s CPU itn.), obsega 32k zlogov, tj. 16 integriranih vezij dinamičnega RAMa z oznako MK 4116; to pomnilno vezje je bilo opisano v članku (2). Pomnilni prostor za razširitev se nahaja v dveh zaporednih naslovnih intervalih

(4000,7FFF) in (8000,BFFF)

kjer so intervalne meje zapisane heksadecimalno. S tem je določen kodirnik, ki mora upoštevati bita A14 in A15 (gornja naslovna bita).

Signali, ki jih moramo upoštevati v krmilniku za dinamični RAM, pa so še tile:

- MRQ (memory request): ta signal se pojavi ob vsaki zahtevi po pomnilniku, torej takrat, ko imamo ukaze tipa "naloži iz pomnilnika" in "shrani v pomnilnik";

- RFSH (refresh) je signal, ki se pojavi skupaj s signalom MRQ na začetku vsakega pomnilniškega osveževalnega cikla;

- WR (write) je signal, ki določa vpis podatkov iz podatkovnega vodila v pomnilnik, njegova odsotnost pa lahko pomeni izpis podatkov iz pomnilnika (ob prisotnosti drugih ustreznih signalov);

- RAMDIS (RAM disable) je vhodni signal, s katerim je mogoče izključiti dodatni pomnilnik v naslovnem intervalu (4000, BFFF) ali pa tudi segmente tega intervala (v primeru prekrivanja);

- DØ,...,D7 je podatkovno vodilo;

- AØ,...,A13 je preostanek naslovnega vodila;

- RD (read) je signal, ko CPU čita;

- DIN' (data input) je signal, s katerim se odpirajo vrata za signale s podatkovnega vodila na glavno (CPU) ploščo;

- DIE (data input enable) je signal drugih krmilnikov na dodatnem modulu.

Bločno shemo krmilnika za dinamični RAM imamo na sliki 1, podrobnosti pa so razvidne iz skupne logične sheme modula na sliki 5.

Časovni diagrami krmilnika so standardni in so že bili opisani v članku (2). Preizkus zgrajenega pomnilnika bo opisan v poglavju 8.

Na koncu dodajmo še praktičen nasvet. Novi dinamični pomnilnik bo obratoval zanesljivo le, če smo zadostno blokirali napajalne napetosti na samih sponkah integriranih vezij 4116 (to velja zlasti za +12V in -5V, pa tudi za +5 V) in če smo ustrezno ojačili napajalne vode (dovolj veliki preseki žic za +12V, +5V, -5V in GND).

3. Razširitev pomnilnika tipa EPROM

Opisani pomnilnik tipa EPROM je razširitev obstoječega pomnilnika za operacijski sistem, ki že vsebuje 10k zlogov programske opreme v sistemu s ploščo SDB-80 (Mostek). Razširitev se pokriva z že naslovljenima, toda nezasedenima naslovnima intervaloma

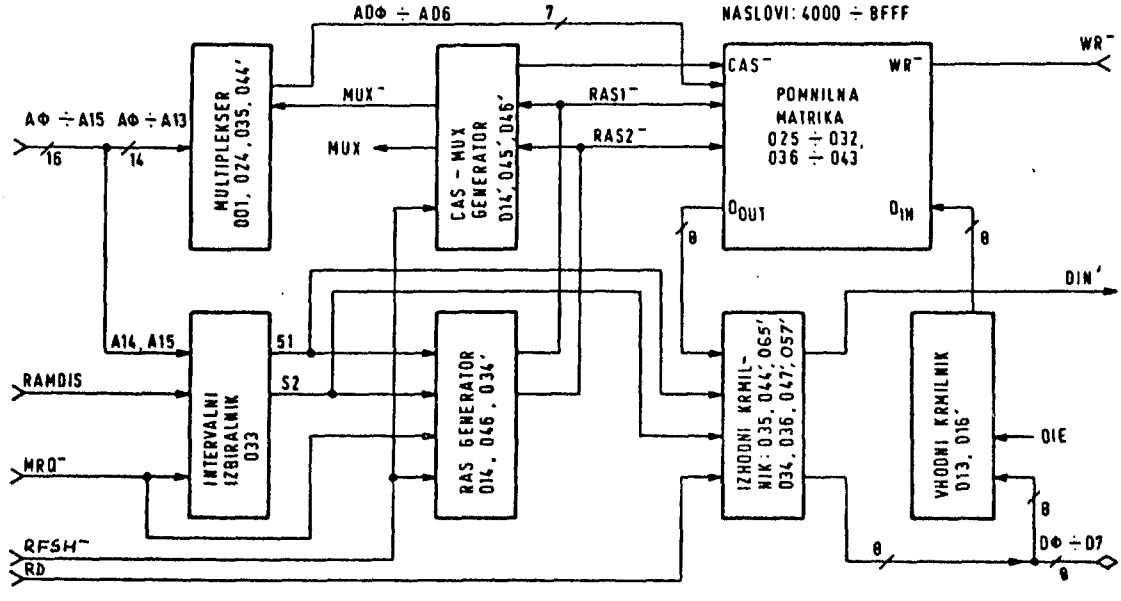
(E800, EFFF) in (F800, FBFF);

zaradi takega stanja je potrebno generirati signal MD- (Memory Disable) za glavno ploščo, ki v primeru naslovitve intervala

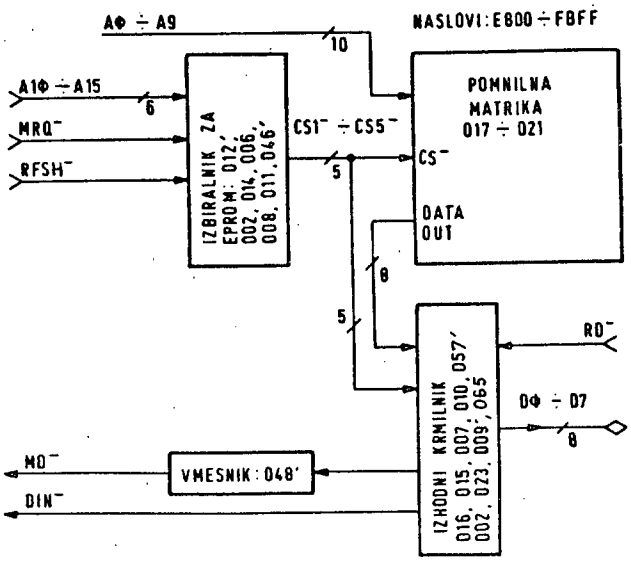
(E800, FBFF)

enostavno onesposobi pomnilnik na glavni plošči.

Uporaba signala MD- omogoča, da na nekaterih modulih izvedemo naslavljanje enostavneje (vzamemo večje intervale, kot so potrebni in dobimo s tem enostavnejše vezje); ko pa vstavimo dodatni pomnilnik na nekem drugem modulu, korigiramo to pomanjkljivost tako, da generiramo na tem modulu signal MD- za modul z enostavnejšim vezjem. S takim primerom se srečamo tudi v našem modulu, ko imamo signal MD- za glavno ploščo, kot kaže slika 2.



Slika 1. Shema krmilnika in pomnilne matrike za dinamični RAM; številke v pravokotnikih, npr. 001, 002, ... označujejo integrirana vezja na kasnejših slikah; vendar pomeni npr. 035' le del integriranega vezja z oznako 035 na spodnjem desnem vogalu tega elementa na sliki 5; signal DIE (Data Input Enable) nastane zaradi potreb drugih krmilnikov na tem modulu in ima pri aktiviranju krmilnika za RAM svojo nazivno vrednost.



Slika 2. Shema krmilnika in pomnilne matrike za pomnilno vezje tipa EPROM; številke v pravokotnikih označujejo integrirana vezja na sliki 5, s črtico označene številke pa dele teh vezij.

Izbiralnik za EPROM je izveden standardno, pri čemer s signalom RFSH- onemogočimo aktiviranje EPROMa med osveževanjem. Izbirni signali CS1-,...,CS5- (Chip Select) za integrirana vezja 2708 (1k-zložni EPROM) so uporabljeni skupaj s signalom RD- za generiranje MD- in DIN-. Signal DIN- omogoča vstop podatkov D0,...,D7 v vezje glavne plošče. Izhodni krmilnik na sliki 2 je uporabljen tudi za uro realnega časa, ki bo opisana v naslednjem poglavju; v tem primeru imamo optimizacijo vezja. Del multiplekserja iz slike 1 se v primeru osposobitve pomnilnika z EPROMi uporablja za ojačevanje naslovnih signalov A0,...,A5 z vodila (ko dobimo A0',...,A5' na sliki 5); tudi v tem primeru gre za optimizacijo vezja. Preizkus delovanja pomnilnika tipa EPROM bo opisan v poglavju 7.

Podobno kot za pomnilnike RAM velja tudi za pomnilnike EPROM, da bo njihovo delovanje zanesljivo le tedaj, ko so napajalni vodi (+12V, +5V, -5V, GND) medsebojno blokirani ter ustrezno ojačeni; ta integrirana vezja imajo tudi nekoliko večjo disipacijo, tako da je odvajanje toplote priporočljivo.

4. Avtonomna ura realnega časa

Ura realnega časa, ki jo želimo imeti, ima lastnost materialno avtonomnega (neodvisnega) delovanja. Vezje za uro naj bo zgrajeno tako, da ura steče po nastavitvi začetnega časa ter da jo potem ni mogoče ustaviti s sistemskim signalom RESET-. Seveda pa mora biti omogočena t. i. programska ustavitev: ko se je pojavil določeni naslov vrat, se ura realnega časa resetira in se tako vrednosti njenih štev nih registrov nastavijo na začetno vrednost (na vrednost časovne konstante, ki je bila vstavljena).

Razen opisane lastnosti pa mora imeti vezje ure še možnost prekinjevanja uporabniških in sistemskih programov in tudi značilno lastnost, da jo je moč odčitavati, jo ponovno nastavljati ter tudi korigirati natančnost njenega teka.

Avtonomna ura realnega časa (kratko AURČ), ki zadošča vsem naštetim zahtevam, je bila zgrajena z uporabo vezja Z-80-CTC (Counter Timer Circuit) s pripadajočim urnim krmilnikom. Lastnosti vezja CTC so bile delno opisane (brez opisa prekinitvenega mehanizma) v članku (3).

Časovni diagrami procesorja Z-80-CPU ter časovnika Z-80-CTC kažejo, da projektiranje krmilnika ure ni kritično glede na zakasnitve, ki nastanejo z zaporedno povezavo več TTL ali MOS elementov integriranih vezij. Uporabijo se lahko normalni tipi (sufiks N) ali nekoliko hitrejši, z manjšo porabo energije (LS tip: low power Schottky); v našem primeru smo uporabili integrirana vezja tipa LS.

Iz časovnih diagramov vezja CTC povzamemo tele izhodiščne podatke za načrtovanje AURČ:

1° Pisalni cikel: podatki se vpisujejo v CTC s pomočjo ustreznih stanj naslovnega, podatkovnega in krmilnega vodila; imamo tole situacijo v trenutku vpisa:

CE- (Chip Enable) = 0
CS0 (Channel Select) = A0
CS1 = A1 (tj. izbira kanala v CTC)
IORQ- (Input/Output Request) = 0
RD- (Read) = 1
podatek vpisan v CTC: D0,D1,...,D7
avtomatično se generira čakalna perioda T_W

Tu so A0,A1,... naslovni in D0,D1,... podatkovni signali.

2° Čitalni cikel: podatki se čitajo iz CTC v CPU in v trenutku čitanja velja:

CE- = 0, CS0 = A0, CS1 = A1,
IORQ- = 0, RD- = 0,
čitalni podatki iz CTC: D0,D1,...,D7
avtomatično se generira čakalna perioda T_W

3° Potrditev prekinitvenega cikla: po generiranju signala prekinitve INT- (Interrupt), ki ga CTC ob prekinitvi pošlje v CPU, odgovori CPU po nekaj periodah s potrditvijo sprejema prekinitve tako, da generira:

M1- (Machine Cycle One Signal) = 0
IORQ- = 0
dvojno čakalno periodo T_W

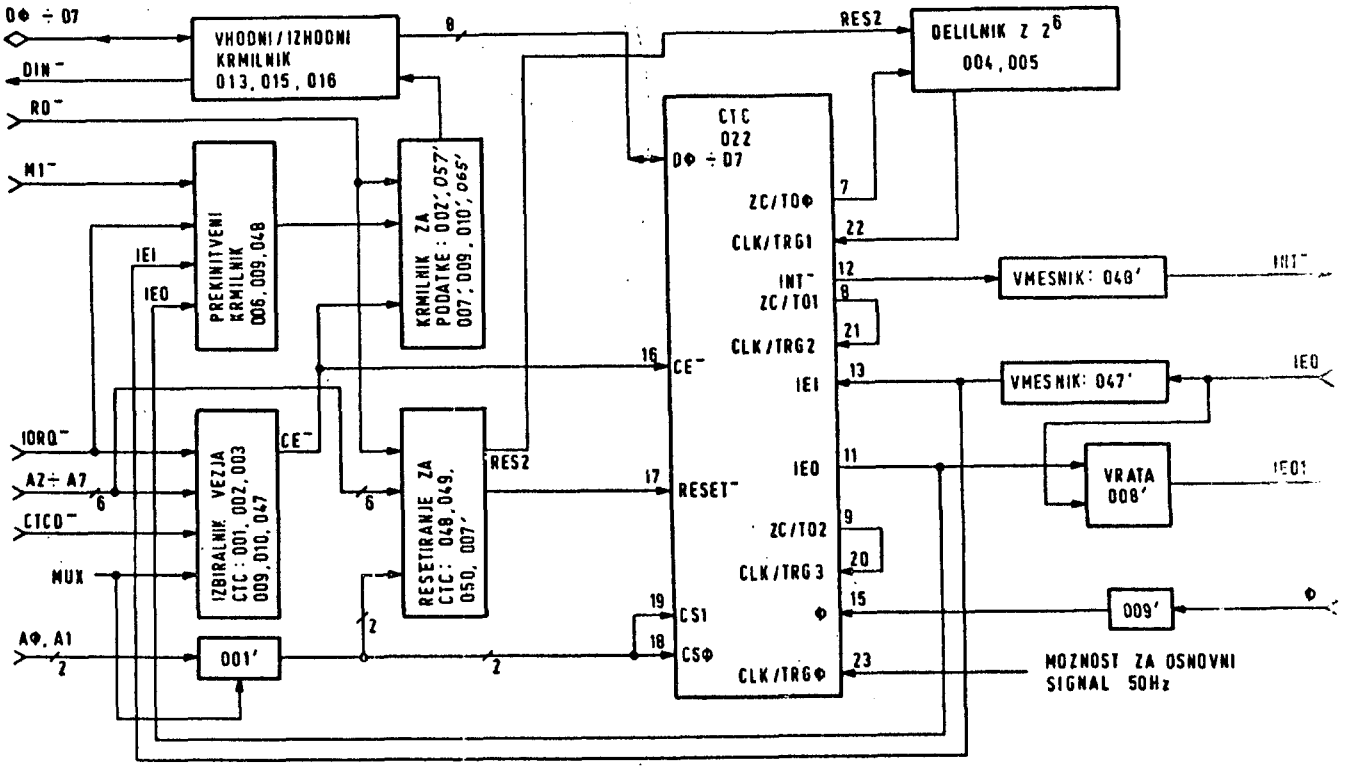
Medtem mora prekinitvena logika elementa CTC določiti kanal z najvišjo prednostjo, ki je prekinitev zahteval (lahko je več kanalov sprožilo prekinitev). Linija prednostne prekinitvene verige IEI (Interrupt Enable In) in IEO (Interrupt Enable Out) se stabilizirata v času, ko je M1- še aktiven. Če je IEI = 1 in IEO = = 0, postavi po prednosti prvi kanal vezja CTC vsebino svojega registra za prekinitveni vektor na podatkovno vodilo.

4° Vrnitev iz prekinitvenega cikla: na koncu prekinitvene servisne subrutine se uporablja ukaz RETI (Return from Interrupt), ki signalizira konec prekinitvene subrutine in se tako uporablja za inicializacijo linije IEO na vrenost 1, s čemer se doseže pravilno delovanje poljubno dolge prednostne verige (daisy chain) perifernih integriranih vezij družine Z-80 (PIO, SIO, CTC, DMA). Vezje CTC, ki je prekinitveno aktivirano, samo dekodira ukazna zloga ukaza RETI (tj. heksadecimalno zaporedje ED 4D), ki se pojavita na podatkovnem vodilu eden za drugim (v presledku nekaj taktnih period) ob vsakokratni prisotnosti signalov RD- = 0 in M1- = 0.

Z upoštevanjem točk 1° do 4° lahko narišemo krmilnik ure z vezjem CTC shematično, kot kaže slika 3. krmilnik ure je optimiziran tako, da ima skupne dele s krmilnikoma za RAM in ROM. Posebej pa velja omeniti nekatere njegove značilnosti.

Naslovi vrat kanalov vezja CTC so heksadecimalna števila

C8 (kanal 0)
C9 (kanal 1)
CA (kanal 2)
CB (kanal 3)



Slika 3. Shema krmilnika avtonomne ure realnega časa s časovnikom/števnikom CTC; številke v pravokotnikih označujejo integrirana vezja na sliki 5, s črtico označene številke pa dele teh vezij. Podatek o času nastane kot derivacija signala Ø v registrih kanalov vezja CTC. Čas je mogoče programsko nastavljati (začetni čas), odčitavati (v poljubnem trenutku), korigirati prekinitveno (v primeru prehitevanja ali zaostajanja ure) ter povzročati prekinitve (npr. vsako minuto, uro, dan itn.); uro je mogoče tudi ustaviti oziroma prekiniti njen tek (s programsko ustavitvijo oziroma resetiranjem).

Resetiranje vezja CTC se doseže z naslovitvijo vrat

CF

in sicer samo z ukazom

OUT (ØCFH),A

ker se v vezju za resetiranje CTC (slika 3) uporablja signal RD in ne RD-. Tako je ukaz IN A,(ØCFH) rezerviran za kako drugo situacijo, če se bo pokazala potreba.

Med kanaloma Ø in 1 vezja CTC imamo zunanji števnik z delilno konstanto 64, tako da že v kanalu 1 dobimo sekunde, v kanalu 2 minute ter v kanalu 3 ure. Kanal Ø deluje v časovniškem režimu z delilnim faktorjem 256×150 (s programsko nastavitvijo), ostali kanali pa v števniškem režimu z delilnimi faktorji 60, 60 in 24; vse delilne faktorje nastavimo programsko, kot je opisano v poglavju 6. S časovniškim režimom kanala Ø se je vezje zreduciralo za 8 bistabilnih elementov (npr. za dve integrirani vezji tipa 74LS93). Slabost vstavitve delilnika s faktorjem 64 med kanala Ø in 1 je v tem, da ne moremo pri danem konceptu ure prožiti prekinitve vsako sekundo. Če pa se uram odpovemo, lahko prožimo prekinitve tudi vsako sekundo, dve sekundi, tri sekunde itd.

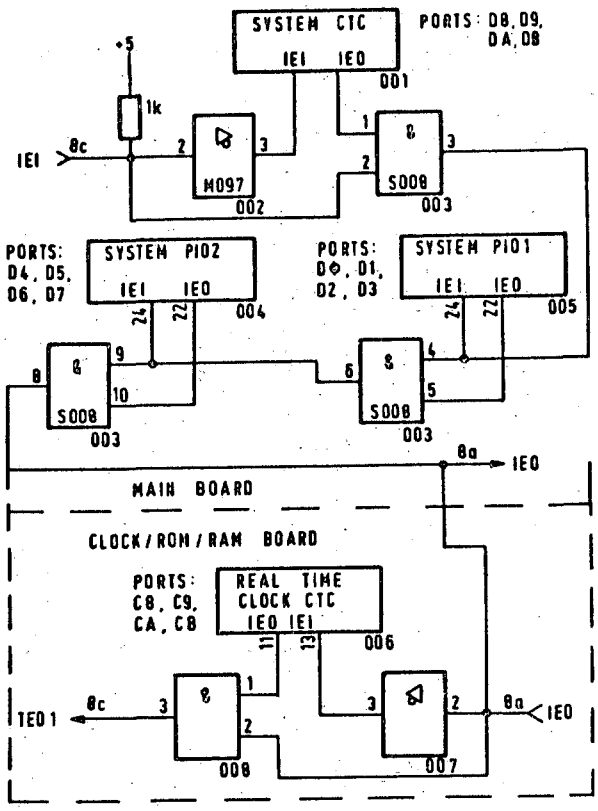
Kot je razvidno iz slike 3, so kanali Ø, 1, 2, 3 že ustrezno medseboj povezani. Prekinitveni krmilnik upošteva prekinitveno prednostno verigo (daisy chain). Uro je mogoče tudi izločiti (izključiti kot vezje), in sicer s signalom CTCD- (CTC Disable) in naslovi njenih vrat se tako lahko uporabijo tudi za druge namene.

Do drugačne realizacije ure realnega časa pridemo tako, da vzamemo za osnovni signal ure 50 Hz iz omrežja. Ustrezno oblikovane impulze (nivoja med Ø in 5V) pripeljemo na vhodno sponko CTC, imenovano CLK/TRGØ (nožica 23). Na sliki 3 postane s tem delilnik s faktorjem 64 odvečen in nožici 7 in 22 enostavno sklenemo. Seveda mora biti izvor signala s 50 Hz dobro izoliran od omrežja pa tudi ustrezno oblikovan (signalne fronte, napetostna nivoja).

Na sliki 4 je narisana prekinitvena prednostna veriga sistema (na glavni in dodatni plošči). Prekinitveno prednostno zaporedje perifernih kanalov je tedaj

CHAN Ø CHAN 1 CHAN 2 CHAN 3	CTC na glavni plošči
PORT A PORT B	PIO 1 na glavni plošči: prednost med A in B ima prej aktivirani
PORT A PORT B	PIO 2 na glavni plošči: prednost med A in B ima prej aktivirani
CHAN Ø CHAN 1 CHAN 2 CHAN 3	CTC na dodatni plošči

Preko sponke IEO1 lahko prednostno verigo nadaljujemo (z nižjo prednostjo), podobno pa velja tudi za sponko IEI (z višjo prednostjo) na sliki 4. Vmesno prednost pa bi dobili z vstavitvijo dodatnih prednostnih kanalov ali vrat v verigo med sponki IEO na glavni in dodatni plošči.



Slika 4. Prekinitvena prednostna veriga, ki jo oblikujejo periferna integrirana vezja glavne plošče (sistemski CTC in dve paralelni V/I vezji tipa PIO) ter vezje ure realnega časa dodatne plošče (vezje CTC). Prioriteta je tale: sistemski CTC, PIO1, PIO2 in urni CTC. Verigo je moč podaljšati na njenem začetku, koncu in v njeni sredini (med glavno in dodatno ploščo).

5. Skupno vezje za razširitev pomnilnika ROM in RAM ter za uro realnega časa

Slike 1, 2, 3 in 4 shematično prikazujejo štiri krmilnike s pomnilniki tipa ROM, RAM in vezjem CTC. Logično vezje na sliki 5 prikazuje združitev teh krmilnikov, ko je dosežena tudi določena optimizacija (minimizacija).

Vezje na sliki 5 uporablja dokaj standardne (cenene in navadne) tipe integriranih vezij, kot kaže tabela materiala na sliki 6. Samo vezje je bilo realizirano na standardni dvojni evropski univerzalni plošči (proizvajalec TRS, Zagreb) s tehniko ožičevanja in s spajkanjem.

Za popolnejše razumevanje delovanja vezja na sliki 5 je priporočljivo upoštevanje shem na slikah 1, 2, 3 in 4, ki dajejo shematično povezavo elementov v podenote (krmilnike). Čeprav se izdatno uporablja tehnika "zaporednega" povezovanja logičnih elementov, ko vsak element nekaj prispeva k zakasnitvi signala, vsota teh zakasnitev ni nikjer kritična. Uporabili bi lahko tudi integrirana vezja tipa N (normal), ki imajo včasih nekoliko večje signalne zakasnitve kot vezja tipa LS. Časi (zakasnitve), ki so dopustni pri procesorju Z-80 (takt s frekvenco 2,5 MHz) za posamezne akcije, so tile:

- prenos operacijskega koda iz pomnilnika: od pojavitve signala MRQ- do včitanja podatkov v CPU imamo vsaj 500 ns;

-čitanje pomnilnika (RAM in ROM): od pojavitve signala MRQ- do včitanja podatkov v CPU imamo vsaj 600 ns;

- vpis podatkov v pomnilnik (RAM): od pojavitve signala MRQ- do vpisa podatkov v pomnilnik imamo vsaj 600 ns;

- vhodni in izhodni cikel (ukaza IN in OUT): od pojavitve signala IORQ- do včitanja iz periferije v akumulator in obratno imamo vsaj 800 ns; podobno velja tudi za pojavitev naslova vrat na linijah AØ,...,A7;

- prekinitvena zahteva in njena potrditev: od pojavitve signala IORQ- do včitanja 8-bitnega polvektorja iz podatkovnega vodila v CPU imamo vsaj 400 ns.

V primeru vezja na sliki 5 pa velja tole:

- prenos operacijskega koda oziroma podatkov iz pomnilnika ROM: zaporedje logičnih elementov D je 117,213,133,116,115,111,128,131,127, 126,140,141,118, kar je 13 zakasnilnih enot; celo z normalnimi tipi integriranih vezij, kjer je zakasnitev enote do 20 ns, ne dosežemo meje 500 ns;

- prenos operacijskega koda oziroma podatkov iz pomnilnika RAM: zaporedje elementov D je 117,210,238,235,236,237,285,286,234, kar je 9 zakasnilnih enot (torej ni kritično);

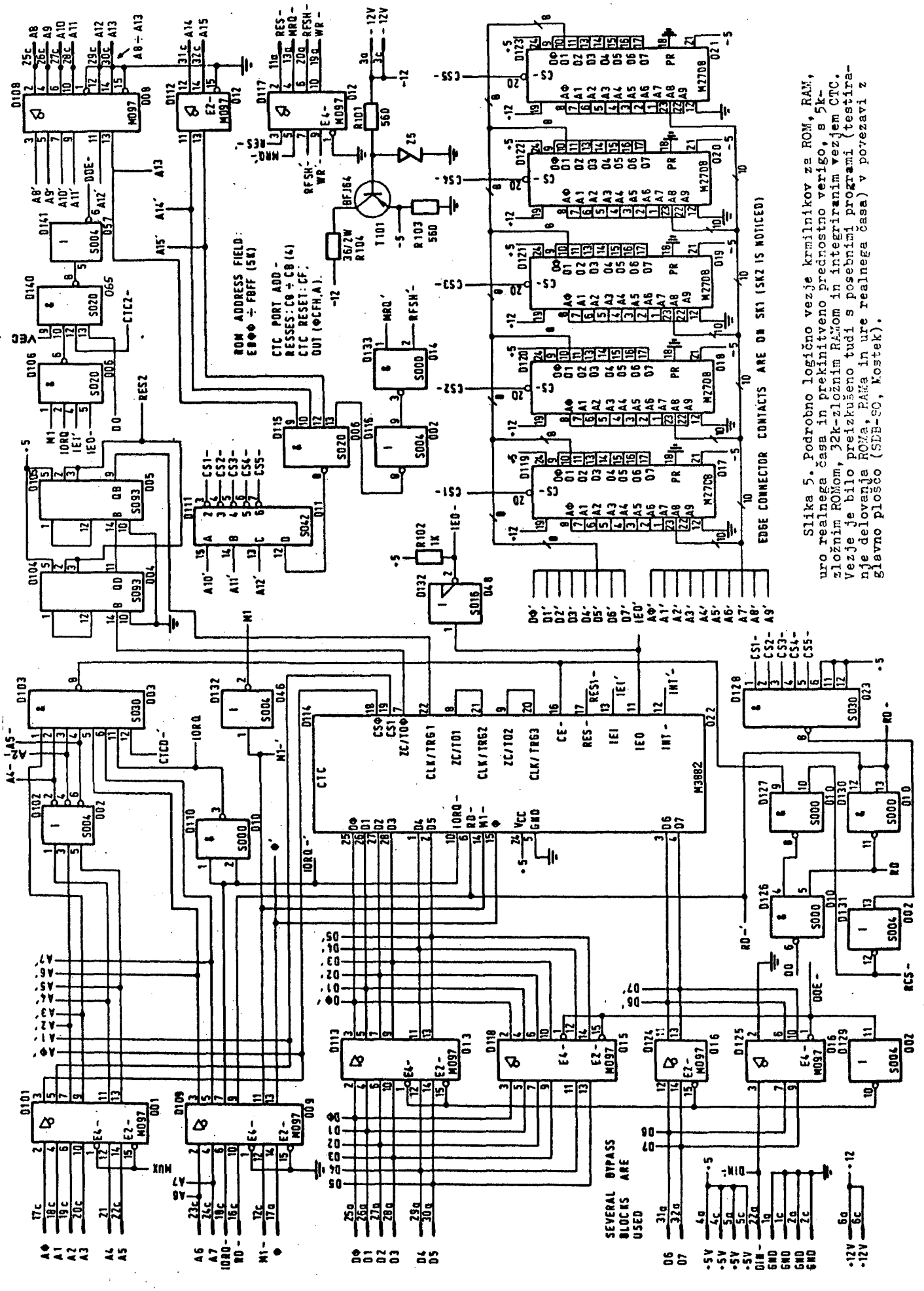
- vpis podatkov v pomnilnik RAM: zaporedje elementov D je npr. 117,210,211,228,229,230, 231,232,233,216, kar je 10 zakasnilnih enot in od teh imajo le 4 večjo zakasnitev (74LØ4);

- vhodni in izhodni cikel za CTC: vhodno vezje je realizirano z zaporedjem elementov D, in sicer 109,110,103,127,126,140,141,129,113, kar je le 9 zakasnilnih enot in podobno velja tudi za izhodno vezje;

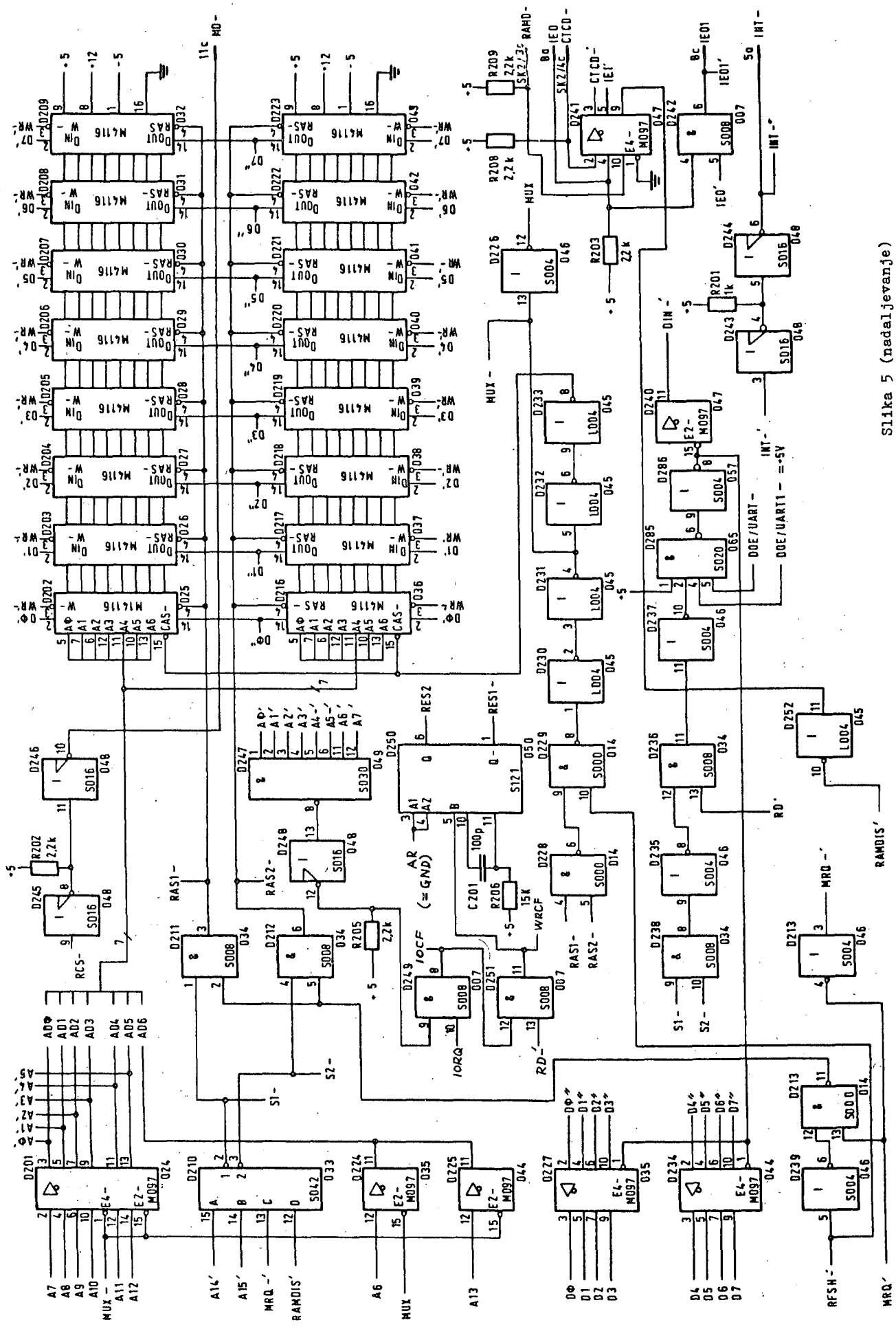
- za potrditev prekinitve imamo zaporedje D elementov 106,140,141,118, to je le 4 zakasnilne enote.

št.	tip elementa	število	številka v vezju
1	74LSØØ	2	010,014
2	74LØ4	1	045
3	74LSØ4	3	002,046,057
4	74LSØ8	2	007,034
5	74LS16	1	048
6	74LS2Ø	2	006,065
7	74LS3Ø	3	003,023,049
8	74LS42	2	011,033
9	74LS93	2	004,005
10	74LS121	1	050
11	2708	5	017,018,019,020, 021
12	4116	16	025÷032,036÷043
13	3881(CTC)	1	022
14	8T97	11	001,008,009,012, 013,015,016,024, 035,044,047
15	BFJ 64	1	T101
16	360hm/2W	1	R104
17	560 Ohm	1	R101
18	1 k	2	R102,201
19	2,2 k	5	R202,203,205,208, 209
20	Z5	1	Zenerjeva dioda
21	15 k	1	R206
22	100 pF	1	C201
23	47 nF	36	blokiranje
24	200 uF	6	blokiranje

Slika 6. Tabela materiala za vezje na sliki 5



Slika 5. Podrobno logično vezje krmilnikov za ROM, RAM, uro realnega časa in prekinitveno prednostno verigo, s 5k-ložnim ROMom, 32k-ložnim RAMom in integriranim vezjem CTC. Vezje je bilo preizkušeno tudi s posebnimi programi (testiranje delovanja ROMa, RAMa in ure realnega časa) v povezavi z glavno ploščo (SBB-80, Kostek).



Slika 5 (nadaljevanje)

Razen standardnih signalov imamo na sliki 5 še nekatere posebne signale, kot so signali onesposobitve določenih podenot plošče zaradi zunanjih zahtev (RAMD- rabi za izključitev 32k-složnega pomnilnika RAM, CTCD- se lahko uporabi za izključitev vezja CTC, tj. D114); nadalje imamo signal DIN-, s katerim odpiramo podatkovna vrata na glavno ploščo in signal MD- (sponka 11c), s katerim onesposobimo pomnilnika ROM in RAM na glavni plošči.

6. Preizkus vezja ure realnega časa

Vezje na sliki 5 vsebuje uro realnega časa, ki ima tele osnovne karakteristike:

1. Ura realnega časa je programsko nastavljiva v poljubnem trenutku (na poljuben čas), ko nastavimo sekunde, minute in ure ter uro tudi poženemo.

2. Ura realnega časa je programsko čitljiva v poljubnem trenutku, ko lahko odčitamo npr. ure, minute in sekunde.

3. Ura realnega časa lahko proži prekinitve, ko se s pomočjo subrutin opravljajo časovno odvisne akcije.

4. Uro je mogoče tudi programsko (in samo programsko) resetirati (naslov vrat je ØCFH), in sicer z uporabo ukaza OUT (ØCFH),A.

5. Naslovi kanalov v vezju CTC so: C8, C9, CA, CB. Med kanaloma CHANØ in CHAN1 se nahaja še števnik (dve vezji 74LS93), ki šteje po modulu 64.

Kanal CHANØ z naslovom vrat C8 je preddelilnik, ki deluje v časovniškem režimu; uporablja predštevnik z delilnim faktorjem 256 ter števnik z delilnim faktorjem 150; kanal ima tako skupni delilni faktor 38400. Temu kanalnemu delilniku sledi 6 zunanjih bistabilnih členov (74LS93) s faktorjem 64. Tako dobimo pred vhodom v sekundni kanal (CHAN1, naslov vrat C9) že delilni faktor 2437600 (namesto točne vrednosti 2458000, ki je tudi frekvenca kristala).

Delovanje vezja ure realnega časa preizkusimo najprej "ročno", in sicer tako, da vstavimo v vsak kanal vezja CTC ustrezni krmilni zlog in časovno konstanto; pri tem deluje CHANØ časovniško, kanali CHAN1, CHAN2 in CHAN3 pa delujejo števniško. Imamo tole tabelo vstavljenih vrednosti:

naslov vrat	krmilni zlog	časovna konstanta	decimalna vrednost
ØC8H	35H	96H	150
ØC9H	5DH	3CH	60 (sek)
ØCAH	5DH	3CH	60 (min)
ØCBH	5DH	18H	24 (hr)

S tako, celovito vstavitvijo ura že teče in z direktivo sistema "P" (port) opazujemo njeno delovanje, saj naredi ura pri opazovanju vrat ØC9H vsako sekundo korak navzdol (uporabljamo izmenoma "CR" in "^"). V tej fazi lahko opazujemo tudi točnost ure, tako da jo pustimo teči nekaj ur in ugotavljamo razliko pri sekundah.

Nadalje moramo preizkusiti še delovanje prekinitvenega mehanizma. Preizkus opravimo takole: na naslov 1000H postavimo ukaz HALT (tj. 76H); prekinitev realiziramo s kanalom CHAN1, kjer imamo sekundni števnik; prekinitev se bo tako pojavila vsako minuto. Za lokacijo prekinitvenega skoka izberemo naslova 2H in 3H (kar je v skladu s kanalom 1) ter zapišemo na ta naslova naslov ukaza HALT, tj. (2) = 00 in (3) = = 10 (v obratnem vrstnem redu, tj. za 1000H). Inicializiramo še (I) = 00 in vstavimo v CHANØ vezja CTC polvektor ØØ, npr. z direktivo

```
.P C8↵
C8 77 00↵
```

Vstavimo tudi novi kontrolni zlog za prekinitev v kanal CHAN1, npr. zlog ØDDH in časovno konstanto 3CH. Ko bo kanal CHAN1 dosegel ničlo, bo CPU skočil v stanje HALT: prižgala se bo lučka na liniji HALT-. To je znak, da se je prekinitev sprožila.

Nazadnje preizkusimo še resetiranje elementa CTC z direktivo

```
.P CF↵
CF FF 00↵
```

Rezultat resetiranja je, da dobimo pri odčitanju vrat C8, C9, CA in CB vsebine 96, 3C, 3C in 18 (vstavljene časovne konstante).

Nadaljni preizkus delovanja integriranega vezja CTC obsega dva programa, in sicer nastavitveni program Y24 in prekinitveno servisno subrutino Y25; listi teh programov sta prikazani na sliki 7 in sliki 8.

Začetni čas vstavimo ročno na lokacije HOUR, MIN in SEC; te lokacije se določijo absolutno z naložitvijo (premestljivo) modula Y24. Psevdo kod modula izgleda takole:

```
SUBROUTINE CLKLD1
  save registers AF, BC, DE, HL;
  set interrupt mode 2;
  reset real time clock;
  load control bytes and time values from
    HOUR,MIN and SEC locations into CTC
    channels 3, 2 and 1;
  load control byte and time constant into
    CTC channel Ø;
  load interrupt vector into CTC (LSB) and
    into register I (MSB);
  load the address of interrupt service sub-
    routine to vector location;
  load hour, minute and second time constants
    (24, 60, 60) to CTC channels 3, 2 and 1;
  reload registers AF, BC, DE, HL from save
    area;
  return to caller
ENDSUBROUTINE
```

Prekinitveno servisno subrutino moramo naložiti pred začetkom izvajanja subrutine CLKLD1. Iz liste za Y24 na sliki 7 je razvidno, da imamo v sekundnem kanalu (CHAN1) krmilni zlog ØC5H, kar pomeni sprožitev prekinitve v tem kanalu. Torej se nam bo servisna subrutina CLK1 (modul Y25 na sliki 8) aktivirala vsako minuto (ko gre sekundni števnik skozi ničlo). Psevdo kod subrutine CLK1 je tedaj:

```
SUBROUTINE CLK1
  disable interrupt;
  set interrupt mode 2;
  save registers AF, BC, DE, HL;
  load time items from CTC channels 3, 2 and 1
    to time area HOUR1;
  print time message "CR""LF"HOUR_MIN_SEC ;
  reload registers AF, BC, DE, HL from
    save area;
  enable interrupt;
  return from interrupt
ENDSUBROUTINE
```

K moduloma Y24 in Y25 velja pripomniti še tole: števniki v vezju CTC odštevajo navzdol, in sicer binarno. Zaradi tega moramo v modulu Y24 pretvoriti podatke iz časovnega območja HOUR, MIN, SEC iz decimalne (npr. 16 59 33) v binarno obliko (tj. 10 3B 21) z uporabo segmenta BCDBIN. Tej konverziji sledi zaradi odštevanja CTC števnikov inverzija glede na vrednosti 24, 60, 60 (binarno 18, 3C, 3C), ko imamo za naš primer čas Ø8 Ø1 1B. Pri izpisovanju časa (modul Y25, slika 8) je postopek obraten: najprej imamo inverzijo, tej pa sledi konverzija iz binarne v decimalno vrednost (segment BINBCD v modulu Y25).

Nazadnje poudarimo še tole: nastavitveni modul Y24 je pisan seveda za specifičen primer.

```
Y24     CLOCK SETTING
   ADDR  OBJECT    ST #

                  0002          NAME Y24
                  0003 ;        REAL TIME CLOCK SETTING AND
                  0004 ;          STARTING SUBROUTINE
                  0005 ;*****************************************
                  0006 ;    PROGRAMMED BY ANTON P. ZELEZNIKAR
                  0007 ;      FOR MOSTEK SDB-80 (Z-80) WITH ADDI-
                  0008 ;      TIONAL CTC MODULE ...
                  0009 ;    DATE: AUGUST 24, 1978
                  0010 ;*****************************************
                  0011          GLOBAL HOUR
                  0012          GLOBAL MIN
                  0013          GLOBAL SEC
                  0014          GLOBAL CLK1
                  0015 ;******
                  0016 ;    SUBROUTINE "CLKLD1" LOADS REAL TIME
                  0017 ;      FROM LOCATIONS "HOUR", "MIN", AND
                  0018 ;      "SEC" INTO CTC CLOCK AND STARTS THE
                  0019 ;      AUTONOMOUSLY RUNNING REAL TIME
                  0020 ;      CLOCK (PORT ADDRESSES: C8, C9, CA,
                  0021 ;      CB; RESET PORT ADDRESS: CF, USING
                  0022 ;      THE INSTRUCTION OUT (0CFH),A ). THE
                  0023 ;      INTERRUPT VECTOR IS LOADED AUTOMA-
                  0024 ;      TICALLY (DEPENDING ON INTERRUPT
                  0025 ;      SUBROUTINE "CLK1" LOCATION).
                  0026 ;*****************************************
*0000  F5         0027 CLKLD1:PUSH AF           ;SAVE (AF)
*0001  C5         0028        PUSH BC           ;SAVE (BC)
*0002  D5         0029        PUSH DE           ;SAVE (DE)
*0003  E5         0030        PUSH HL           ;SAVE (HL)
*0004  ED5E       0031        IM 2              ;SET MODE 2
*0006  D3CF       0032        OUT (0CFH),A      ;RESET CLOCK
*>0008            0033 SETIN:;          SET USER'S INITIAL TIME
*0008  217600*     0034        LD HL,HOUR        ;SET HL TO HOUR
*000B  0ECB       0035        LD C,0CBH         ;SET PORT COUNTER
*000D  3E18       0036        LD A,24           ;SET HOUR REFERENCE
*000F  0603       0037        LD B,3            ;SET LOOP PARAMETER
*0011  165D       0038        LD D,05DH         ;SET CONTROL BYTE
*0013  ED51       0039 CONT:  OUT (C),D         ;LOAD CONTROL BYTE
*>0015            0040 BCDBIN:;   DECIMAL TO BINARY CONVERSION
*0015  F5         0041        PUSH AF           ;SAVE ACC
*0016  7E         0042        LD A,(HL)         ;TIME ITEM IN ACC
*0017  E60F       0043        AND 0FH           ;TAKE LOWER 4 BITS
*0019  CB66       0044        BIT 4,(HL)        ;TEST BIT 4
*001B  2802       0045        JR Z,FIVE-$       ;IF Z GO TO FIVE
*001D  C60A       0046        ADD A,10          ;ELSE ADD 10
*001F  CB6E       0047 FIVE:  BIT 5,(HL)        ;TEST BIT 5
*0021  2802       0048        JR Z,SIX-$        ;IF Z GO TO SIX
*0023  C614       0049        ADD A,20          ;ELSE ADD 20
*0025  CB76       0050 SIX:   BIT 6,(HL)        ;TEST BIT 6
*0027  2802       0051        JR Z,CONV-$       ;IF Z GO TO CONV
*0029  C628       0052        ADD A,40          ;ELSE ADD 40
*002B  5F         0053 CONV:  LD E,A            ;SAVE RESULT
*002C  F1         0054        POP AF            ;RELOAD ACC
                  0055 ; END OF DECIMAL TO BINARY CONVERSION
*002D  B7         0056        OR A              ;CLEAR CARRY FLAG
*002E  9B         0057        SBC A,E           ;CONVERT TIME
*002F  ED79       0058        OUT (C),A         ;LOAD TIME TO CTC
*0031  23         0059        INC HL            ;INCREMENT (HL)
```

```
Y24     CLOCK SETTING
   ADDR  OBJECT    ST #

*0032  0D         0060        DEC C             ;DECREMENT (C)
*0033  3E02       0061        LD A,2            ;COMPARISION
*0035  B8         0062        CP B              ;IF (B)=2
*0036  3E3C       0063        LD A,60           ;LOAD NEXT REFER.
*0038  2802       0064        JR Z,SECREF-$     ;THEN GO TO SECREF
*003A  1802       0065        JR CONT1-$        ;ELSE GO TO CONT1
*003C  16C5       0066 SECREF:LD D,0C5H         ;SECOND CONTROL
*003E  10D3       0067 CONT1: DJNZ CONT-$       ;IF B .NE. 0 GO TO
                  0068 ;                        ; CONT...
*0040  3E35       0069        LD A,35H          ;LOAD CONTROL BYTE
*0042  ED79       0070        OUT (C),A         ; TO CHANNEL C8...
*0044  3E96       0071        LD A,150          ;LOAD DIVIDER TO
*0046  ED79       0072        OUT (C),A         ; CHANNEL C8...
*0048  21FABF*    0073        LD HL,VEC         ;LOAD VEC TO HL
*004B  ED69       0074        OUT (C),L         ;LOAD LSB OF INT.
                  0075 ;                        ; VEC. TO PORT C8
*004D  7C         0076        LD A,H            ;LOAD MSB OF INT.
*004E  ED47       0077        LD I,A            ; VEC. TO REG I
*0050  11FFFF     0078        LD DE,CLK1        ;LOAD ADDRESS OF
*0053  73         0079        LD (HL),E         ; INTER. SERVICE
*0054  23         0080        INC HL            ; SUBROUTINE TO
*0055  72         0081        LD (HL),D         ; VEC LOCATION
*>0056            0082 SETCON:;           SET TIME CONSTANTS
*0056  0ECB       0083        LD C,0CBH         ;SET PORT COUNTER
*0058  3E18       0084        LD A,24           ;SET HOUR CONSTANT
*005A  0603       0085        LD B,3            ;SET LOOP PARAMETER
*005C  165D       0086        LD D,05DH         ;SET CONTROL BYTE
*005E  ED51       0087 CONT2: OUT (C),D         ;LOAD CONTROL BYTE
*0060  ED79       0088        OUT (C),A         ;LOAD TIME CONSTANT
*0062  0D         0089        DEC C             ;SET NEXT CHANNEL
*0063  3E02       0090        LD A,2            ;COMPARE (B)
*0065  B8         0091        CP B              ;IF B=2
*0066  3E3C       0092        LD A,60           ;LOAD REFERENCE
*0068  2802       0093        JR Z,SEC1-$       ;THEN GO TO SEC1
*006A  1802       0094        JR CONT3-$        ;ELSE GO TO CONT3
*006C  16C5       0095 SEC1:  LD D,0C5H         ;LOAD CONTROL BYTE
*006E  10EE       0096 CONT3: DJNZ CONT2-$      ;IF (B) .NE. 0 GO
                  0097 ;                        ; TO CONT2 ...
*0070  E1         0098        POP HL            ;RELOAD (HL)
*0071  D1         0099        POP DE            ;RELOAD (DE)
*0072  C1         0100        POP BC            ;RELOAD (BC)
*0073  F1         0101        POP AF            ;RELOAD (AF)
*0074  FB         0102        EI                ;ENABLE INTERRUPT
*0075  C9         0103        RET               ;RETURN TO CALLER
                  0104 ;*                       **********
*0076  00         0105 HOUR   DEFB 0            ;HOUR LOCATION
*0077  00         0106 MIN    DEFB 0            ;MIN LOCATION
*0078  00         0107 SEC    DEFB 0            ;SEC LOCATION
                  0108        ORG 0BFFAH        ;VECTOR AREA
                  0109 VEC    DEFS 2            ;VECTOR LOCATION
                  0110 ;******
                  0111        END
```

```
BCDBIN      0015 CLK1   [EXT] 0051 CLKLD1       0000 CONT       0013
CONT1       003E CONT2        005E CONT3        006E CONV       002B
FIVE        001F HOUR   [INT] 0076 MIN    [INT] 0077 SEC    [INT] 0078
SEC1        006C SECREF       003C SETCON       0056 SETINI     0008
SIX         0025 VEC          BFFA
```

Slika 7. Nastavitvena in startna subrutina ure realnega časa; z njo nastavimo začetni čas iz lokacij HOUR, MIN in SEC ter uro pošenemo v tek.

```
Y25    CLOCK INTERRUPT
  ADDR  OBJECT    ST #

                  0002        NAME Y25
                  0003 ;      REAL TIME CLOCK INTERRUPT
                  0004 ;        SERVICE SUBROUTINE
                  0005 ;*****************************************
                  0006 ;    PROGRAMMED BY ANTON P. ZELEZNIKAR
                  0007 ;      FOR MOSTEK SDB-80 (Z-80) USING
                  0008 ;      ADDITIONAL CTC MODULE ...
                  0009 ;    DATE: AUGUST 26, 1978
                  0010 ;*****************************************
                  0011        GLOBAL CLK1
                  0012        GLOBAL HOUR1
                  0013 ;******
                  0014 ;    INTERRUPT SUBROUTINE "CLK1" TAKES
                  0015 ;      TIME ITEMS (HOUR, MIN, SEC) FROM
                  0016 ;      CTC CHANNELS AND LOADS THEM TO
                  0017 ;      HOUR1 TIME AREA. THIS SUBROUTINE
                  0018 ;      PRINTS HOUR-MIN-SEC MESSAGE WHEN
                  0019 ;      CALLED.
                  0020 ;*****************************************
 '0000  F3        0021 CLK1:  DI              ;DISABLE INTERRUPT
 '0001  ED5E      0022        IM 2            ;SET MODE 2
 '0003  F5        0023        PUSH AF         ;SAVE (AF)
 '0004  E5        0024        PUSH HL         ;SAVE (HL)
 '0005  C5        0025        PUSH BC         ;SAVE (BC)
 '0006  D5        0026        PUSH DE         ;SAVE (DE)
 '0007  214F00'   0027        LD HL,HOUR1     ;TAKE TIME AREA
 '000A  0ECB      0028        LD C,0CBH       ;SET PORT ADDRESS
 '000C  3E18      0029        LD A,24         ;SET HOUR REFERENCE
 '000E  0603      0030        LD B,3          ;SET LOOP PARAMETER
 '0010  ED50      0031 CONT5: IN D,(C)        ;INPUT PORT CONTENT
 '0012  B7        0032        OR A            ;CLEAR CARRY FLAG
 '0013  9A        0033        SBC A,D         ;CONVERT TIME
 '>0014           0034 BINBCD:;   BINARY TO DECIMAL CONVERSION
 '0014  57        0035        LD D,A          ;LOAD ACC TO D
 '0015  E607      0036        AND 07H         ;TAKE LOWER 3 BITS
 '0017  CB5A      0037        BIT 3,D         ;TEST BIT 3 OF D
 '0019  2803      0038        JR Z,FOUR-$     ;IF Z GO TO FOUR
 '001B  C608      0039        ADD A,8         ;ELSE ADD 8
 '001D  27        0040        DAA             ;DECIMAL ADJUST
 '001E  CB62      0041 FOUR:  BIT 4,D         ;TEST BIT 4 OF D
 '0020  2803      0042        JR Z,FIVE-$     ;IF Z GO TO FIVE
 '0022  C616      0043        ADD A,16H       ;ELSE ADD 16H
 '0024  27        0044        DAA             ;DECIMAL ADJUST
 '0025  CB6A      0045 FIVE:  BIT 5,D         ;TEST BIT 5 OF D
 '0027  2803      0046        JR Z,SIX-$      ;IF Z GO TO SIX
 '0029  C632      0047        ADD A,32H       ;ELSE ADD 32H
 '002B  27        0048        DAA             ;DECIMAL ADJUST
 '>002C           0049 SIX:;  END OF BINARY TO DECIMAL CONVER.
 '002C  77        0050        LD (HL),A       ;LOAD TIME TO AREA
 '002D  23        0051        INC HL          ;INCREMENT (HL)
 '002E  0D        0052        DEC C           ;DECREMENT PORT AD.
 '002F  3E3C      0053        LD A,60         ;LOAD MIN/SEC REF
 '0031  10DD      0054        DJNZ CONT5-$    ;IF NZ GO TO CONT5
 '0033  214F00'   0055        LD HL,HOUR1     ;TAKE TIME AREA
 '0036  1E00      0056        LD E,0          ;LOAD I/O CONTROL
 '0038  CD9CE5    0057        CALL CRLF       ;WRITE CR & LF
 '003B  0603      0058        LD B,3          ;SET LOOP PARAMETER
 '003D  7E        0059 TIME1: LD A,(HL)       ;LOAD TIME TO ACC
```

```
Y25    CLOCK INTERRUPT
  ADDR  OBJECT    ST #

 '003E  57        0060        LD D,A          ; AND TO REG D
 '003F  CD8BE5    0061        CALL PACC       ;WRITE TIME
 '0042  CDA5E5    0062        CALL SPACE      ;WRITE SPACE
 '0045  23        0063        INC HL          ;INCREMENT (HL)
 '0046  10F5      0064        DJNZ TIME1-$    ;IF NZ GO TO TIME1
 '0048  D1        0065        POP DE          ;RELOAD (DE)
 '0049  C1        0066        POP BC          ;RELOAD (BC)
 '004A  E1        0067        POP HL          ;RELOAD (HL)
 '004B  F1        0068        POP AF          ;RELOAD (AF)
 '004C  FB        0069        EI              ;ENABLE INTERRUPT
 '004D  ED4D      0070        RETI            ;RETURN FROM INT
                  0071 ;******
 '004F  303030    0072 HOUR1  DEFM '000'      ;TIME SAVE AREA
                  0073 ;******
  >E59C           0074 CRLF   EQU  0E59CH     ;WRITE CR & LF
  >E58B           0075 PACC   EQU  0E58BH     ;WRITE 2 HEX DIG
  >E5A5           0076 SPACE  EQU  0E5A5H     ;WRITE SPACE
                  0077 ;******
                  0078        END
```

```
BINBCD      0014 CLK1   [INT] 0000 CONT5       0010 CRLF        E59C
FIVE        0025 FOUR         001E HOUR1  [INT] 004F PACC      E58B
SIX         002C SPACE        E5A5 TIME1        003D
```

```
.L 0
BEG ADDR 0000
EXECUTE  0000
END ADDR 0078
UNDEF SYM 01
*L
BEG ADDR 0079
END ADDR 00CA
UNDEF SYM 00
*T
SYMBOL TABLE (UNDEF=****)

CLK1    0079    HOUR    0076    HOUR1   00C8    MIN     0077
SEC     0078
*.
```

Slika 8. Prekinitvena servisna subrutina CLK1 se aktivira skladno z subrutino CLKLD1 iz slike 7 vsako minuto ter odtisne čas v urah, minutah in sekundah. Ta subrutina rabi tako skupaj s subrutino CLKLD1 za osnovni preizkus delovanja ure realnega časa, ki je predstavljena z ustreznim vezjem na sliki 5. Na koncu liste za CLK1 imamo simbolno tabelo, nazadnje pa je prikazana naložitev obeh subrutin in sicer CLKLD1 z začetkom na lokaciji ØØØØH in CLK1 z začetkom na lokaciji ØØ79; povezavo med obema rutinama je opravil nalagalnik za premeščanje in povezovanje.

```
Y26    MEMORY TEST
   ADDR  OBJECT    ST #

                   0002         NAME Y26
                   0003 ;       MEMORY TEST PROGRAM
                   0004 ;*****************************************
                   0005 ;    PROGRAMMED BY ANTON P. ZELEZNIKAR
                   0006 ;      FOR MOSTEK SDB-80 USING ADDITIONAL
                   0007 ;      RAM/ROM/REAL TIME CLOCK MODULE ...
                   0008 ;    DATE: SEPTEMBER 23, 1978
                   0009 ;*****************************************
                   0010         GLOBAL AEBEG
                   0011         GLOBAL ERMESS
                   0012         GLOBAL MESS
                   0013         GLOBAL ANDMESS
                   0014         GLOBAL FAILME
                   0015         GLOBAL RIGHT
                   0016 ;******
                   0017 ;    PROGRAM "MEMORY TEST" TESTS A MEMORY
                   0018 ;      INTERVAL ((AEBEG).(AEBEG+1),
                   0019 ;      (AEBEG+2).(AEBEG+3)) FOR CORRECT-
                   0020 ;      NESS IN STORING, READING AND HOL-
                   0021 ;      DING OF DATA AT EACH MEMORY ADDRESS
                   0022 ;      OF THE GIVEN INTERVAL FOR ALL
                   0023 ;      POSSIBLE DATA COMBINATIONS.
                   0024 ;*****************************************
'0000  DD216500'   0025 INIT:   LD IX,AEBEG     ;SET IX TO AREA BEG
'0004  0600        0026         LD B,0          ;RESET REGISTER B
'0006  DD7004      0027         LD (IX+4),B     ;RESET ERROR FLAG
'0009  DD7005      0028         LD (IX+5),B     ;RESET LINE COUNT
'000C  DD7006      0029         LD (IX+6),B     ;RESET HEADING FLAG
'000F  ED5B6700'   0030         LD DE,(AEEND)   ;SET DE TO AREA END
                   0031 ;  ENDINIT
                   0032 ;  DOUNTIL LOOP
'0013  2A6500'     0033 LOOP:   LD HL,(AEBEG)   ;SET HL TO AREA BEG
'0016  78          0034         LD A,B          ;LOAD (B) TO ACC
'0017  77          0035 STORE:  LD (HL),A       ;STORE (ACC)
'0018  B7          0036         OR A            ;RESET CARRY FLAG
'0019  E5          0037         PUSH HL         ;SAVE (HL)
'001A  ED52        0038         SBC HL,DE       ;(HL)-(DE) NEG ?
'001C  E1          0039         POP HL          ;RELOAD (HL)
'001D  23          0040         INC HL          ;INCREMENT (HL)
'001E  2F          0041         CPL             ;COMPLEMENT (ACC)
'001F  38F6        0042         JR C,STORE-$    ;IF NEG REPEAT
                   0043 ;  ENDSTORE
'0021  2A6500'     0044         LD HL,(AEBEG)   ;SET HL TO AREA BEG
'0024  78          0045         LD A,B          ;LOAD (B) TO ACC
'0025  BE          0046 COMPAR: CP (HL)         ;COMPARE ((HL))
'0026  C4FFFF      0047         CALL NZ,ERMESS  ;IF NZ CALL ERMESS
'0029  B7          0048         OR A            ;RESET CARRY FLAG
'002A  E5          0049         PUSH HL         ;SAVE (HL)
'002B  ED52        0050         SBC HL,DE       ;(HL)-(DE) NEG ?
'002D  E1          0051         POP HL          ;RELOAD (HL)
'002E  23          0052         INC HL          ;INCREMENT (HL)
'002F  2F          0053         CPL             ;COMPLEMENT (ACC)
'0030  38F3        0054         JR C,COMPAR-$   ;IF NEG REPEAT
                   0055 ;  ENDCOMPAR
'0032  10DF        0056         DJNZ LOOP-$     ;IF NZ GO TO LOOP
                   0057 ;  ENDDOUNTILLOOP
'0034  21FFFF      0058 SUCC:   LD HL,MESS      ;COMMON MESSAGE
'0037  1E01        0059         LD E,1          ;SET I/O CONTROL
```

```
Y26    MEMORY TEST
   ADDR  OBJECT    ST #

'0039  CDC7E3      0060         CALL PTXT       ;PRINT MESSAGE
'003C  2A6500'     0061         LD HL,(AEBEG)   ;PRINT THE ADDRESS
'003F  CD04E6      0062         CALL PADD0      ; INTERVAL TESTED
'0042  21FFFF      0063         LD HL,ANDMES    ;
'0045  CDC7E3      0064         CALL PTXT       ;
'0048  2A6700'     0065         LD HL,(AEEND)   ;
'004B  CD04E6      0066         CALL PADD0      ;
'004E  AF          0067         XOR A           ;RESET ACC
'004F  DDBE04      0068         CP (IX+4)       ;IF NO FAILURE
'0052  2808        0069         JR Z,OKMESS-$   ; PRINT SUCCESS
'0054  21FFFF      0070         LD HL,FAILME    ; OTHERWISE PRINT
'0057  CDC7E3      0071         CALL PTXT       ; ERROR STATEMENT
'005A  1806        0072         JR END-$        ;
'005C  21FFFF      0073 OKMESS: LD HL,RIGHT     ;SUCCESS MESSAGE
'005F  CDC7E3      0074         CALL PTXT       ;
'0062  C31DE1      0075 END:    JP MONITR       ;PROGRAM END
                   0076 ;  ENDSUCC
                   0077 ;******
'0065  0000        0078 AEBEG   DEFW 0          ;AREA BEGIN ADDRESS
'0067  30303435    0079 AEEND   DEFM '00456'    ;AREA END ADDRESS
                   0080 ;                       ; AND FLAG LOCATIONS
                   0081 ;*****************************************
 >E3C7             0082 PTXT    EQU 0E3C7H      ;PRINT TEXT
 >E604             0083 PADD0   EQU 0E604H      ;PRINT ADDRESS
 >E11D             0084 MONITR EQU 0E11DH       ;MONITOR REENTRY
                   0085 ;******
                   0086         END
```

```
Y26    MEMORY TEST
   ADDR  OBJECT    ST #

AEBEG   [INT] 0065 AEEND          0067 ANDMES [EXT] 0043 COMPAR        0025
END           0062 ERMESS [EXT]   0027 FAILME [EXT] 0055 INIT        0000
LOOP          0013 MESS   [EXT]   0035 MONITR       E11D OKMESS      005C
PADD0         E604 PTXT           E3C7 RIGHT  [EXT] 005D STORE       0017
SUCC          0034
```

Slika 9. Program RAMTE1 (modula Y26 in Y27) rabi za preizkus pomnilniškega segmenta, ko se testirajo vse celice (lokacije, besede zlogi) segmenta z vsemi možnimi vsebinskimi kombinacijami (256 različic). Program sestavljata segmenta Store in Compar v dountil zanki ter sporočevalna segmenta za uspeh Succ in napako (subrutina Ermess); subrutina Ermess s sporočilnimi teksti je programirana kot poseben modul Y27 (nadaljevanje slike 9).

```
Y27     MEMORY TEST MESSAGES
   ADDR  OBJECT    ST #

                  0002         NAME Y27
                  0003 ;       MESSAGES FOR MEMORY TEST PROGRAM
                  0004 ;         AND ERROR MESSAGE SUBROUTINE
                  0005 ;*************************************
                  0006 ;    PROGRAMMED BY ANTON P. ZELEZNIKAR
                  0007 ;      FOR MOSTEK SDB-80
                  0008 ;    DATE: SEPTEMBER 24, 1978
                  0009 ;*************************************
                  0010         GLOBAL ERMESS
                  0011         GLOBAL MESS
                  0012         GLOBAL ANDMESS
                  0013         GLOBAL OKMESS
                  0014         GLOBAL FAILME
                  0015         GLOBAL RIGHT
                  0016 ;******
                  0017 ;    THIS PROGRAM INCLUDES ERROR MESSAGE
                  0018 ;      SUBROUTINE FOR Y26 PROGRAM AND THE
                  0019 ;      MESSAGES FROM GLOBAL LIST.
                  0020 ;******
 '0000  DD3604FF  0021 ERMESS:LD (IX+4),0FFH ;SET ERROR FLAG
 '0004  D5        0022         PUSH DE       ;SAVE (DE)
 '0005  4F        0023         LD C,A        ;SAVE (ACC)
 '0006  AF        0024         XOR A         ;RESET ACC
 '0007  DDBE06    0025         CP (IX+6)     ;TEST HEADING FLAG
 '000A  200E      0026         JR NZ,TYPE-$  ;IF NZ TYPE ERROR D.
 '000C  E5        0027 HEAD:   PUSH HL       ;SAVE (HL)
 '000D  214200'   0028         LD HL,HEADMS  ;PRINT HEADING ROW
 '0010  1E01      0029         LD E,1        ;SET I/O CONTROL
 '0012  CDC7E3    0030         CALL PTXT     ;
 '0015  E1        0031         POP HL        ;RELOAD (HL)
 '0016  DD3606FF  0032         LD (IX+6),0FFH ;SET HEADING FLAG
 '001A  DD3405    0033 TYPE:   INC (IX+5)    ;INCR LINE COUNT
 '001D  CD04E6    0034         CALL PADD0    ;PRINT ERROR ADDRESS
 '0020  7E        0035         LD A,(HL)     ;PRINT MEMORY
 '0021  CD8BE5    0036         CALL PACC     ; CONTENT "MY"
 '0024  CDA5E5    0037         CALL SPACE    ;PRINT SPACE
 '0027  79        0038         LD A,C        ;PRINT (ACC)
 '0028  CD8BE5    0039         CALL PACC     ;
 '002B  CDA5E5    0040         CALL SPACE    ;PRINT SPACE
 '002E  CDA5E5    0041         CALL SPACE    ;PRINT SPACE
 '0031  3E06      0042         LD A,6        ;TEST LINE COUNT
 '0033  DDBE05    0043         CP (IX+5)     ; FOR END OF LINE
 '0036  2007      0044         JR NZ,EERM-$  ;IF NZ GO TO EERM
 '0038  DD360500  0045         LD (IX+5),0   ; ELSE CLEAR LINE C.
 '003C  CD9CE5    0046         CALL CRLF     ;PRINT CR & LF
 '003F  D1        0047 EERM:   POP DE        ;RELOAD (DE)
 '0040  79        0048         LD A,C        ;RELOAD (ACC)
 '0041  C9        0049         RET           ;RETURN TO CALLER
                  0050 ;*************************************
  >E3C7           0051 PTXT    EQU 0E3C7H    ;PRINT TEXT
  >E604           0052 PADD0   EQU 0E604H    ;PRINT ADDRESS
  >E58B           0053 PACC    EQU 0E58BH    ;PRINT (ACC)
  >E5A5           0054 SPACE   EQU 0E5A5H    ;PRINT SPACE
  >E59C           0055 CRLF    EQU 0E59CH    ;PRINT CR & LF
                  0056 ;*************************************
 '0042  0D0A      0057 HEADMS DEFW 0A0DH     ;CR & LF
 '0044  41444452  0058         DEFM 'ADDR MY AC  ADDR MY AC  '
 '005C  41444452  0059         DEFM 'ADDR MY AC  ADDR MY AC  '
```

```
Y27     MEMORY TEST MESSAGES
   ADDR  OBJECT    ST #

 '0074  41444452  0060         DEFM 'ADDR MY AC  ADDR MY AC'
 '008A  0D0A      0061         DEFW 0A0DH    ;CR & LF
 '008C  03        0062         DEFB 03       ;END OF MESSAGE
 '008D  0D0A      0063 MESS    DEFW 0A0DH    ;CR & LF
 '008F  54455354  0064         DEFM 'TESTED MEMORY INTERVAL'
 '00A5  20424554  0065         DEFM ' BETWEEN ADDRESSES'
 '00B7  0D0A      0066         DEFW 0A0DH    ;CR & LF
 '00B9  20202020  0067         DEFM '        '
 '00C3  03        0068         DEFB 03       ;END OF MESSAGE
 '00C4  2020414E  0069 ANDMES  DEFM '  AND   '
 '00CC  03        0070         DEFB 03       ;END OF MESSAGE
 '00CD  0D0A      0071 FAILME  DEFW 0A0DH    ;CR & LF
 '00CF  49532044  0072         DEFM 'IS DEFECTIVE. GO ANALYZE '
 '00E8  54484520  0073         DEFM 'THE ERROR MESSAGE!'
 '00FA  03        0074         DEFB 03       ;END OF MESSAGE
 '00FB  0D0A      0075 RIGHT   DEFW 0A0DH    ;CR & LF
 '00FD  49532046  0076         DEFM 'IS FUNCTIONING PROPERLY. THE'
 '0119  20464149  0077         DEFM ' FAILURE, IF ANY,'
 '012A  0D0A      0078         DEFW 0A0DH    ;CR & LF
 '012C  4953204F  0079         DEFM 'IS OUTSIDE THIS MEMORY BLOCK.'
 '0149  03        0080         DEFB 03       ;END OF MESSAGE
                  0081 ;*************************************
                  0082         END
```



```
Y27     MEMORY TEST MESSAGES
   ADDR  OBJECT    ST #

ANDMES [INT] 00C4 CRLF        E59C EERM        003F ERMESS [INT] 0000
FAILME [INT] 00CD HEAD        000C HEADMS      0042 MESS   [INT] 008D
OKMESS [EXT] 0000 PACC        E58B PADD0       E604 PTXT         E3C7
RIGHT  [INT] 00FB SPACE       E5A5 TYPE        001A
```

Slika 9 (nadaljevanje)

```
.L 0
BEG ADDR 0000
EXECUTE  0000
END ADDR 006B
UNDEF SYM 05
*L
BEG ADDR 006C
END ADDR 01B5
UNDEF SYM 00
*T
SYMBOL TABLE (UNDEF=****)

AEBEG   0065    ANDMES 0130    ERMESS 006C    FAILME 0139
MESS    00F9    RIGHT  0167
*
```

Slika 10. Naložitev modulov Y26 in Y27 iz slike 9. Na naslove AEBEG, ...,AEBEG+3 naložimo pred uporabo programa začetek in konec pomnilniškega segmenta, ki ga želimo preizkusiti. Program poženemo nato z direktivo .E ØØ

Njegova struktura ostane sicer nespremenjena, spremenijo pa se lahko vrednosti krmilnih zlogov in konstant, pač v odvisnosti od vsakokratnih zahtev. Če nimamo prekinitvenega krmilnega zloga v nobenem kanalu, potem tudi prekinitvene servisne subrutine ne potrebujemo. Za izpisovanje časa imamo tedaj neprekinitveni programski segment, ki pa mora imeti vgrajeno zaščito pred prenosom (carry) v CTC kanalih, ko podatke izpisujemo. Seveda je tudi časovna sporočila mogoče vključevati v sporočilni generator z vrsto, kot je bilo opisano v članku (4).

7. Preizkus delovanja pomnilnika EPROM

Naslovni interval

(E8ØØ, FBFF)

dodatnega pomnilnika tipa EPROM se prekriva z dvema naslovnima intervaloma na glavni plošči, in sicer

(E8ØØ, EFFF) in (F8ØØ, FFFF)

Prvi interval je ostanek naslovnega prostora za ROM operacijskega sistema na glavni plošči, drugi interval pa je samo delno zaseden z beležko (sistemski RAM), in sicer v podintervalu

(FFØØ, FFFF)

Beležka se tako na glavni plošči pojavlja v več naslovnih kopijah.

Opisani problem naslovnega prekrivanja rešimo enostavno s signalom MD- (Memory Disable), ki se generira ob vsakem vpisu in izpisu, kadar se le-ta nanašata na naslovni interval (E8ØØ, FBFF); s tem signalom se onesposobi enak naslovni interval na glavni plošči. Prva kontrola pokaže, da so kopije beležke odsotne, ko smo novo ploščo vstavili v sistem.

Delovanje dodatnega pomnilnika tipa EPROM preizkusimo najhitreje tako, da vstavimo v podnožja integrirana vezja z znanimi podatki (vsebinami) ali programi; potem te vsebine izlistamo ali pa izvajamo programe ter preverjamo dobljene rezultate.

Kot je bilo že poudarjeno, časovno delovanje krmilnika za ROM in integriranih vezij tipa 2708 ni kritično; kadar pa se pojavijo nepredvideni učinki, je najenostavneje preveriti vsebino, vpisano v EPROM elemente. V skrajnem primeru moramo pregledati tudi povezave in napetosti na nožicah (-5, Ø, +5, 12V).

8. Preizkus delovanja pomnilnika RAM

Ko smo vstavili integrirana vezja tipa 4116 v modul, preizkusimo naključno delovanje nekaterih pomnilniških lokacij (npr. z direktivo "M", ko imamo vpis in izpis). To seveda še ne pomeni, da je delovanje pomnilnika RAM v celoti preizkušeno in zanesljivo, kadar delujejo posamezne celice brez napake. V segmente RAMa naložimo tudi krajše programe, jih izvajamo ter opazujemo rezultate.

Novi pomnilniški segment (32k zlogov) preizkusimo s programom za diverzno testiranje pomnilnika; ta program ima ime RAMTE1 (RAM test 1); podoben program je bil opisan v članku (2).

S programom RAMTE1 preizkusimo vsako celico novega pomnilnika na vse možne kombinacije; lista tega programa z navodili za njegovo uporabo je prikazana na sliki 9; ta program je premestljiv in ga po potrebi lahko pomikamo po pomnilniku. Testiranje segmenta 32k zlogov traja približno 8 minut. Začetek testirnega območja naložimo na lokacijo AEBEG, npr. (AEBEG)= ØØ in (AEBEG+1) = 4Ø, konec testirnega območja pa v (AEBEG+2) = FF in (AEBEG+3) = BF. Modula Y26 in Y27 povežemo z nalagalnikom, naložimo pa ju eden za drugim, tako kot kaže slika 10.

9. Sklep

V članku smo pokazali, kako je mogoče z enim samim dodatnim modulom (dodatno ploščo) razširiti mikroračunalniško konfiguracijo. Namen te razširitve je bil, da se sistem osposobi za tekstovno komuniciranje tako, kot je bilo opisano v članku (4) ter da se povečajo možnosti za razvoj programske opreme. V enem od naslednjih člankov bo opisana še dodatna vhodna/izhodna enota ter programska oprema, ki omogoča uporabo kasetne periferije s hitrostjo 9600 Baud ter dodatnega, linijskega teleprinterja za Baudotov kod.

Dodatni pomnilnik tipa EPROM (5k zlogov) je predviden za razširitev operacijskega sistema; jedro te razširitve je generator teksta, tekstovni razpoznavnik ter subrutine za mehanizem "vrsta". Ura realnega časa omogoča oddajo in referenco časa v poljubnem trenutku, razširjeni pomnilnik RAM pa se uporablja za nekaj "vrst" ter za začasno shranjevanje po liniji sprejetih podatkov.

Literatura

(1) A.P.Železnikar, I.Ozimek, M.Kovačević, D. Novak, Programiranje mikro računalnikov s procesorjem Z 80, Informatika 1 (1977), 5-12,št.2.

(2) A.P.Železnikar, M.Kovačević, D.Novak, Razvoj dinamičnih pomnilnikov za mikro računalnike, Informatika 1 (1977), 9-21,št.4.

(3) A.P.Železnikar, Uporaba časovnikov in števnikov v mikroprocesorskih sistemih s procesorjem Z-80, Informatika 2 (1978), 25-33, št.1.

(4) A.P.Železnikar, Vrsta s sporočili za komuniciranje z uporabo mikro računalnika, Informatika 2 (1978), v tisku,št.2.

Opomba. Signali VEC, CTC2- (element D14o), DOE/UART- in DOE/UART1- (element D285) se nanašajo na V/I dodatek z UARTji, ki je tudi na tej plošči (modulu), vendar bo opisan drugič; za pravilno delovanje si mislimo te signale =5V.

Informatica št. 3 letnik 1978

selekcija v množičnem komuniciranju

v.batagelj
a.ferligoj
s.splichal

UDK 301:51:681.3

FNT - VTO matematika in mehanika
FSPN
Univerza v Ljubljani

Članek obravnava možnosti pojasnjevanja selekcije v množičnih komunikacijskih procesih s hierarhičnim določanjem skupin spremenljivk ter poskus avtomatične selekcije s klasifikacijo sporočil v disjunktne skupine.

SELECTION IN MASS COMMUNICATION - The article deals with possibilities of explanation of selection in mass communication processes based on hierarchical clustering of variables and an attempt of automatical selection based on the classification of messages into disjunctive groups.

Selektivno vedénje je eden izmed bistvenih momentov komunikacijskih procesov; nekatere definicije (Luhmann, 1975) celo označujejo komuniciranje kot zapovrstnost selektivnih reakcij na (predhodne) selekcije. V najbolj razvitem komunikacijskem sistemu - množičnem komuniciranju - so selekcijski procesi odvisni od dveh temeljnih skupin dejavnikov, določenih z "dvojno naravo" množičnih medijev, tj. njihovo pripadnostjo materialni bazi po eni strani ter hkratno zasidranostjo v družbeni nadstavbi po drugi (idejni) strani. Razvoj (ali nastanek novih) medijev je odvisen od razvoja proizvajalnih sil; zgodovinska analiza kaže, da gre razvoj komunikacijske tehnologije v smeri vse večje prepustnosti komunikacijskih kanalov (povečevanja sprejemnih in oddajnih kapacitet). Kot abstraktna možnost torej komunikacijska tehnologija zmanjšuje intenzivnost selekcije, saj omogoča (kot nujen, ne pa zadosten pogoj) izmenjavo vse večje količine informacij. V krajših časovnih razdobjih (ob dani komunikacijski tehnologiji) pa je tehnična prepustnost komunikacijskih kanalov konstantna, zato nas v proučevanju selektivnega vedenja komunikatorjev bolj zanima kvalitativna, vsebinska stran selekcije.

Schramm je v svoji raziskavi leta 1957 ugotovil, da ponudba množičnih medijev pomeni le 3 odstotke prvotne množice sporočil, ki jo oblikujejo dopisniki informacijskih agencij. Vendar pa v svoji študiji procesa selekcije v množičnem komuniciranju ne proučuje tudi faktorjev, ki določajo selektivno vedenje na različnih ravneh komunikacijskega procesa. Po drugi strani najdemo - zlasti v novejšem obdobju - vrsto raziskav ene same ravni selekcije v množičnem komuniciranju (v komunikacijskih institucijah), usmerjenih v spoznavanje in pojasnjevanje faktorjev in kriterijev selekcije.

Velik del empiričnih raziskav na tem področju ne razlikuje med faktorji in kriteriji selekcije. Gre za analogijo med "objektivnim" in "subjektivnim" v politični socializaciji: s (subjektivnimi) kriteriji selekcije tako lahko poimenujemo v posamezniku nakopičene izkušnje, na katerih temelji posameznikov (komunikatorjev) odnos do družbenih procesov in odnosov, medtem ko se v (objektivnih) faktorjih izraža objektivno dana, zgodovinsko določena družbena stvarnost, v kateri poteka komunikacijski proces. V podružbljenem množičnem komuniciranju imajo faktorji selekcije še poseben pomen, saj ne le prispevajo k oblikovanju subjektivnih kriterijev selekcije, marveč nanjo tudi neposredno vplivajo. Značilnosti izhodne množice sporočil so torej odvisne od značilnosti vhodne množice sporočil v komunikacijskem procesu, komunikatorjevih kriterijev selekcije ter objektivnih faktorjev, ki vplivajo na oblikovanje kriterijev selekcije in na selekcijo samo. Od prepustnosti komunikacijskih kanalov je seveda

odvisno, ali se izhodna množica sporočil aktualizira v medijski vsebini ali ne.

Razlikovanje med kriteriji in faktorji selekcije ima tudi pomembne metodološke implikacije: medtem ko merimo (ugotavljamo) značilnosti selekcijskih kriterijev z istimi spremenljivkami kot značilnosti (vhodnih in izhodnih) sporočil, to ne velja za faktorje selekcije. Z drugimi besedami: kriteriji selekcije na višjem nivoju (v času t-1) postanejo značilnost sporočil na nižjem nivoju (v času t) vsaj implicitno, medtem ko faktorji selekcije na prvem nivoju ne delujejo več neposredno na selekcijo na drugem nivoju neodvisno od faktorjev drugega nivoja. (Če proučujemo selekcijo na različnih ravneh, a npr. znotraj istega političnega sistema, je seveda odveč opozarjati, da gre za delovanje številnih identičnih faktorjev na obeh ravneh, vendar neposredno na vsaki ravni posebej.) Končno to pomeni, da je mogoče raziskovati uveljavljanje kriterijev selekcije v komunikacijskih procesih dovolj veljavno z metodami analize sporočil, kar nikakor ne velja za proučevanje faktorjev selekcije. V nadaljevanju se bomo omejili na prvi del problema, tj. raziskovanje selekcijskih kriterijev.

Cilj empiričnega raziskovanja, ki je bilo izvedeno na pobudo UNESCO, je bil preskus veljavnosti hipotez o kriterijih selekcije v množičnih medijih, ki so jih postavili in praktično potrdili raziskovalci v ZDA, ZRN, Veliki Britaniji, na Švedskem in Finskem. Raziskovanje kriterijev selekcije (Galtung in Ruge, 1965, Ostgaard, 1965, Rosengren, 1970, Schulz, 1976, Harris, 1976) je bilo doslej omejeno skoraj izključno na analizo ponudbe množičnih medijev, na podlagi take usmerjenosti pa nikakor ni mogoče zanesljivo sklepati o dejanskih kriterijih selekcije. Mnogi med njimi so poskušali svoje ugotovitve posplošiti, ne da bi upoštevali delovanje faktorjev selekcije, in dokazati splošno veljavnost posebnih kriterijev selekcije v množičnem komuniciranju.

Predmet naše analize so bila sporočila, ki jih je iz lastnih in tujih virov (tiskovnih agencij) sprejel Tanjug v času med 19. in 26. septembrom 1977. S pomočjo postavljenih kriterijev selekcije (tabela 1) smo hoteli:

1. ugotoviti, ali je z njimi mogoče pojasniti selekcijo vhodnih sporočil v Tanjugu in
2. določiti kriterije, s katerimi bi lahko vsaj delno) sporočila selekcionirali avtomatično.

Za analizo smo izbrali nalbolj pogosto verificirane hipoteze s kriterijskimi spremenljivkami, ki jih povzemamo v tabeli 1.

TABELA 1: Raziskovalne hipoteze

KRITERIJSKE SPREMENLJIVKE	HIPOTETIČNA VERJETNOST, DA SPOROČILO PRESTANE SELEKCIJO	
	VELIKA	MAJHNA
Viri selektorja	lastni	tuji
Etnocentrizem	močan	šibak
Personifikacija	visoka	nizka
Konfliktnost dogajanja	velika	majhna
Ekonomska moč subjekta dogajanja	velika	majhna
Število vključenih subjektov (le za mednarodne odnose)	veliko	majhno

Kriterijske spremenljivke in spremenljivka o selekciji so merjene z nominalnimi ali kvečjemu ordinalnimi lestvicami. Zato za preverjanje hipotez o odvisnosti kriterijskih spremenljivk na selekcijo sporočil ne moremo uporabiti klasičnih postopkov multivariatne analize (kot npr. multiplo regresijo). Metodologi si v zadnjih desetletjih prizadevajo, da bi izdelali multivariatne metode tudi za nominalne oziroma ordinalne spremenljivke. Nekaj takih metod je ob določenih pogojih glede na spremenljivke že izdelanih kot npr. več metod analize skupin (Sokal in Sneath, 1963, Jardine in Sibson, 1968, Hartigan, 1975, Everitt, 1974), log-linearni modeli (Goodman, 1971) in latentni strukturalni modeli (Mooijaart, 1978). V raziskovanju selekcijskih kriterijev jugoslovanske agencije Tanjug smo se odločili za metode za analizo skupin. K tej odločitvi je precej pripomoglo tudi dejstvo, da so bili dostopni le programi za analizo skupin (CLUSE); vendar pa smo jih nekaj izdelali na novo (LEADER).

Raziskovanje selekcijskih kriterijev je potekalo v treh korakih:

1. dihotomizacija (nominalnih) kriterijskih spremenljivk,
2. proučevanje povezanosti dihotomnih kriterijskih spremenljivk (dks) in spremenljivke o selekciji,
3. klasifikacija sporočil glede na določene vrednosti dks in primerjava med avtomatsko klasifikacijo in selekcijo v agenciji Tanjug.

Skoraj vsaka kriterijska spremenljivka je merjena z nominalno lestvico. Take spremenljivke smo dihotomizirali tako, da je vsaka vrednost kriterijske nominalne spremenljivke nova (dummy) spremenljivka (1 - ima dano vrednost, 0 - nima dane vrednosti). Ta postopek dihotomizacije no-

minalnih spremenljivk se pogosto uporablja, vendar je potrebna opreznost pri statistični analizi takih spremenljivk.

TABELA 2: Dihotomizacija kriterijskih spremenljivk

KRITERIJSKE SPREMENLJIVKE	DKS
Viri selektorja	Tanjug
Etnocentrizem	Evropa
Personifikacija	delovanje posameznikov
Konfliktnost dogajanja	konfliktni dogodki niti konflikt niti kooperacija kooperacija
Ekonomska moč subjekta	razviti subjekti MO
Število vključenih subjektov mednarodnih odnosov	notranja politika bilateralni odnosi multilateralni odnosi mednarodne organizacije
Področje dogajanja v mednarodnih odnosih	politika gospodarstvo vojna in mir kultura "human interest"
Prepustnost	prepustnost

Za i-to in j-to dks lahko zapišemo naslednjo kontingenčno tabelo:

i \ j	1	0
I	a_{ij}	b_{ij}
0	c_{ij}	d_{ij}

Povezanosti (podobnosti) med dvema dks smo merili s Sokal-Michner-jevim koeficientom asociacije:

$$s_{ij} = \frac{a_{ij} + d_{ij}}{a_{ij} + b_{ij} + c_{ij} + d_{ij}}$$

Za proučevanje povezanosti dks smo uporabili metodo hierarhičnega določanja skupin spremenljivk (Sokal in Sneath, 1963, Everitt, 1974), ki temelji na postopnem združevanju skupin spremenljivk v novo skupino. Združevanje poteka takole: začetne skupine so kar posamezne spremenljivke. Med njimi poiščemo najbližji (najpodobnejši) skupini, ki ju nadomestimo z novo skupino - njunim predtsavnikom. Nato določimo "razdalje" med novo skupino in preostalimi skupinami. Zopet poiščemo najbližji skupini itd. Postopek ponavljamo dokler se vse skupine ne zlijejo v eno samo skupino. Potek združevanja grafično ponazorimo z drevesom - dendrogramom. Metode hierarhičnega določanja skupin se ločijo po tem, kako določimo "razdaljo" med novo skupino in preostalimi skupinami. Program CLUSE temelji na Lance-Williamsovem obrazcu:

$$D(k,\{i,j\}) = \alpha_1 D(k,i) + \alpha_2 D(k,j) + \beta D(i,j) + \gamma \left| D(k,i) - D(k,j) \right|$$

kjer je D(m,n) "razdalja" (različnost) med skupinama m in n. Z $\{i,j\}$ je označena skupina, ki jo dobimo z združitvijo i-te in j-te skupine. S primerno izbiro koeficientov v gornjem obrazcu dobimo večino znanih metod hierarhičnega določanja skupin.

V primeru raziskovanja kriterijskih spremenljivk smo se odločili za Wardovo metodo, katero več avtorjev (npr. Mojena, 1976) na podlagi empiričnih primerjav metod priporoča kot najprimernejšo. Za Wardovo metodo so

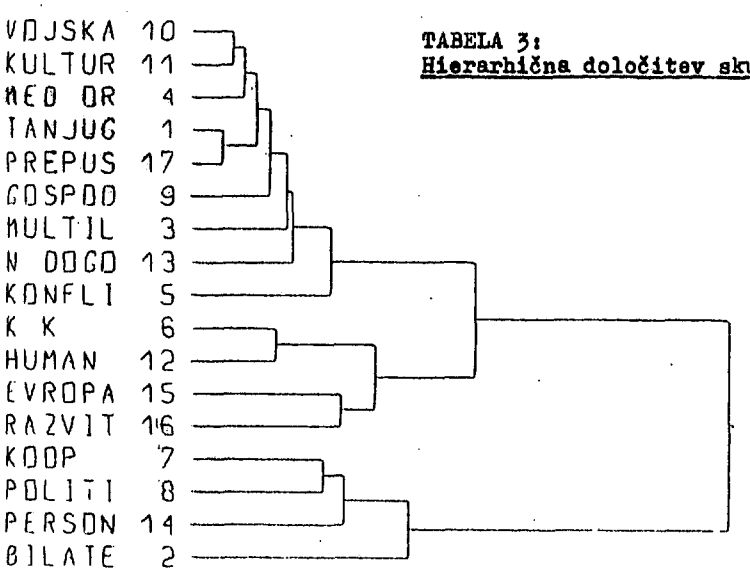


TABELA 3:
Hierarhična določitev skupin dks

$$\alpha_1 = n(k,i)/n(i,j,k)$$
$$\alpha_2 = n(k,j)/n(i,j,k)$$
$$\beta = -n(k)/n(i,j,k)$$
$$\gamma = 0$$

kjer n(i) pomeni število spremenljivk v skupini i in podobno n(i,j)=n(i)+n(j) in n(i,j,k)=n(i)+ +n(j)+n(k).

Različnost med samimi spremenljivkami smo merili takole:

$$D(i,j) = 1 - S_{ij}$$

kjer je S_{ij} Sokal-Michenerjev koeficient asociacije.

Rezultat združevanja je podan v dendrogramu v tabeli 3. Zgornja skupina spremenljivk vključuje spremenljivko o selekciji (PREPUS) in jo tedaj lahko označimo kot tisto skupino kriterijev, ki v Tanjugu povečujejo verjetnost objave sporočila. Torej: čim večje število značilnosti (dks) iz zgornje skupine je prisotnih v posamičnem sporočilu, tem večja je verjetnost, da ga bodo Tanjugovi selektorji objavili. Med kriteriji je najpomembnejši zaupanje lastnim informacijskim virom (TANJUG), saj je "najbliže" objavi (PREPUS). Označitev srednje in spodnje skupine dks je lahko dvojna: lahko gre za tiste dks, ki so irelevantne s stališča selekcije, ali pa za dks, ki povečujejo intenzivnost selekcije oz. zmanjšujejo verjetnost objave sporočila. Za obe skupini pa velja, da se posamične dks med seboj izključujejo, da torej posamično sporočilo ne more imeti vseh značilnosti dane skupine. To je pomembno, če hočemo simulirati selekcijski proces v Tanjugu, saj je treba poiskati (ali določiti) take posebne skupine sporočil, znotraj katerih ni izključujočih se dks, ne pa le dve splošni skupini, od katerih bi ena predstavljala hipotetično skupino neobjavljenih sporočil.

Predikcijsko vrednost uporabljenih dks smo preskusili z avtomatsko klasifikacijo sporočil glede na določene vrednosti dks in jo nato primerjali z dejansko selekcijo v Tanjugu. Za klasifikacijo sporočil smo uporabili metodo voditeljev (Hartigan, 1975), ki razvrsti enote - sporočila v disjunktne skupine enot s tipičnimi predstavniki skupin - voditelji. Pri metodi voditeljev so lahko voditelji podani (klasifikacija) ali pa jih določi postopek sam (določanje skupin). Pri klasifikaciji vsako enoto priredimo skupini, ki je določena z enoti najbližjim (najbolj podobnim) voditeljem. Pri določanju skupin je postopek podoben, le da med postopkom določamo nove voditelje. Za nove voditelje postavimo ali enote, ki so od danih voditeljev preveč oddaljene, ali pa enoto, ki je od vseh voditeljev najoddaljenejša.

Za merjenje razdalje (različnosti) med dvema sporočiloma smo zopet izbrali Sokal-Michnerjev koeficient asociacije.

Rezultat klasifikacije sporočil z metodo voditeljev je podan v tabeli 4, v kateri so sporočila razvrščena v pet disjunktnih skupin. To število se je namreč izkazalo kot optimalno s stališča predikcijske vrednosti uporabljenih dks, tj. z določitvijo petih skupin je bil povprečni odstotek pravilno uvrščenih sporočil glede na dejanske odločitve selektorjev o selekciji največji.

TABELA 4: Klasifikacija sporočil po metodi voditeljev

	vrednosti dks za voditelje				
dks	1	2	3	4	5
Tanjug	0	1	0	1	1
bilateralni odnosi	1	0	0	1	0
multilateralni odnosi	0	0	1	0	1
mednarodne organizacije	0	1	0	0	0
konfliktni dogodki	0	0	1	1	0
neutralni dogodki	1	0	0	0	0
kooperacija	0	1	0	0	1
politika	0	0	1	0	1
gospodarstvo	0	1	0	0	0
vojna in mir	0	0	0	0	0
kultura	0	0	0	0	0
"human interest"	0	0	0	0	0
notranja politika	0	0	0	1	0
personifikacija	1	0	0	1	1
Evropa	0	1	1	1	0
razviti subjekt	0	1	1	1	0
prepustnost	0	1	0	1	1
% vseh vhodnih sporočil po dobljenih skupinah	66.4	5.6	11.9	6.3	9.8
% pravilno klasificiranih sporočil glede na dejanske uredniške odločitve Tanjuga x/	97.4	24.8	95.3	38.0	31.4

x/ Sporočilo je pravilno klasificirano, če ima enako vrednost prepustnosti kot voditelj, k kateremu je pridruženo.

Če analizo omejimo zgolj na obe skupini avtomatično izločenih sporočil (1. in 3. skupina v tabeli 4), je predikcijska vrednost dks tako velika, da bi te dks lahko že praktično uporabljali. S prvo skupino smo avtomatično izločili 66.4 odstotka izmed 5516 prispelih sporočil z le 2.6 odstotno napako. S tretjo skupino smo

dodatno izločili nadaljnjih 11.9 odstotka prispelih sporočil s 4,7 odstotno napako. Na temelju uporabljenih kriterijev smo torej skupaj izločili 78.3 odstotka vhodnih sporočil s povprečno 2.9 odstotno napako.

Za ostanek sporočil uporabljene dks niso dovolj izčrpne, tako da z njimi ni mogoče v celoti pojasniti odločitev selektorjev, ne smemo pa zavreči tudi drugega možnega vzroka neskladnosti med avtomatično in dejansko selekcijo, da namreč selektorji niso vedno konsistentni v svojih odločitvah.

Z napovedovanjem celote uredniških odločitev v Tanjugu na temelju hipotetičnih kriterijskih spremenljivk bi namreč četrtino (24.4 %) sporočil, ki so jih Tanjugovi uredniki objavili, izključili iz komunikacijskega procesa. Po drugi strani pa bi podvojili število sporočil, ki prestanejo selekcijo, z vključitvijo takih sporočil, ki so jih Tanjugovi uredniki dejansko izločili, tako da bi se "avtomatična prepustnost sporočil v Tanjugu povečala na 21.7 odstotka, medtem ko dejanska znaša 9.8 odstotka.

V zaključku moramo torej znova opozoriti na problem, ki smo ga omenili že na začetku - na vlogo faktorjev, ki v času spreminjajo kriterije selekcije.

LITERATURA:

1 V. Batagelj: CLUSE, priročnik, Ljubljana, 1977

2 V. Batagelj: LEADER, priročnik, Ljubljana, 1978

3 B. Everitt: Cluster Analysis, SSRC, London, 1974

4 W. Galtung, M. Ruge: The Structure of Foreign News, Journal of Peace Research, 1965, Vol. 2

5 L.A. Goodman: The Analysis of Multidimensional Contingency Tables: Stepwise Procedures and Direct Estimation Methods for Building Models for Multiple Classification, Technometrics 13 (1971)1

6 Ph. Harris: Selective Images, IAMCR Conference, Leicester 1976

7 J.A. Hartigan: Clustering Algorithms, John Wiley and Sons, New York, 1975

8 N. Jardin, R. Sibson: Mathematical Taxonomy, John Wiley and Sons, New York, 1971

9 N. Luhmann v: O. Schatz (ed.), Die elektronishe Revolution, Graz,1975

10 R. Mojena: Hierarchical Grouping Methods and Stopping Rules:An Evaluation, The Computer Journal, 20(1976)4

11 A. Mooijaart: Latent Structure Models, disertacija, Leiden, 1978

12 E. Ostgaard: Factors Influencing the Flow of News, Journal of Peace Research, 1965, Vol. 2

13 K. Rosengren: International News: Intra and Extra Media Data, Acta Sociologica, 1970, Vol. 13

14 W. Schramm: L'information et le development national, Paris, 1965

15 W. Schultz: Die Konstruktion von Realität in den Nachrichtenmedien, München, 1976

16 R.R. Sokal, P.H.A. Sneath: Principles of Numerical Taxonomy, Freeman, London, 1963

17 C. Trampuš, A. Ferligoj: Nekateri vidiki uporabe računalnikov v sociologiji in politologiji, Informatica 1(1977)1

primjena mikroprocesorskih komponenata z80 u proizvodnji računarske opreme

m.žagar

UDK 681.3 - 181.4 Z 80

Zavod za RST
Elektrotehnički fakultet
Unska 17
41000 ZAGREB

U članku je dan pregled i najbitnije karakteristike mikroprocesorskih komponenata iz serije Z80 te primjeri programiranja tih komponenata za izvršavanje odredjenih zadataka. Razmotrene su mogućnosti upotrebe mikroprocesora Z80 kao procesorskog elementa većeg računala, a isto tako i upotreba Z80 mikroprocesorskih komponenata u upravljanju periferijskih jedinica računala.

Z80 MICROCOMPUTER COMPONENTS IN PRODUCTION OF COMPUTER EQUIPMENT: In the article a review of Z80 microcomputer components is given and the most important characteristics are described. The examples of programming Z80 components for special purposes are presented. Possibilities of using Z80 microcomputer components in production of computer equipment are discussed.

1. UVOD

Z80 serija mikroprocesorskih komponenata uključuje sve dijelove potrebne za izgradnju mikroračunala vrlo dobrih karakteristika. Pri tome je potrebna minimalna klasična hardverska logika te standardni, jeftini memorijski elementi.

Sve komponente izradjene su u NMOS tehnologiji, koriste samo jedan izvor napajanja (+5V), jednofazni generator takta (takodjer +5V), svi pinovi su TTL kompatibilni što su vrlo bitne karakteristike koje pojednostavljuju projektiranje i izgradnju uredjaja pomoću tih komponenata.

Zbog svoje koncepcije i vrlo "jakih" komponenata čija se svojstva mogu isprogramirati, mikroprocesorski sistem Z80 pogodan je za primjenu u raznim područjima obrade podataka.

Kombiniranjem samog Z80-CPU (Central Processing Unit) mikroprocesora i komponenata kao što su: Z80-PIO (Parallel Input/Output) paralelni ulazno/izlazni kontroler, Z80-SIO (Serial Input/Output) serijska ulazno/izlazna jedinica, Z80-CTC (Counter Timer Circuit) jedinica za generiranje različitih vremenskih signala i mjerenje vremena, te Z80-DMA (Direct Memory Access) jedinica za razmjenu podataka izmedju vanjskog svijeta i memorije bez prisustva procesora, može se izgraditi procesorski dio većeg računala, a isto tako i upravljački dio u periferijskim jedinicama računala.

Za potpuno razumijevanje rada Z80 mikroprocesorskih komponenata potrebno je proučiti tehničke karakteristike koje daje proizvodjač. U nastavku su dana samo najbitnija svojstva pojedinih komponenata te primjerima objašnjen način njihovog programiranja za obavljanje unaprijed odredjenih zadataka.

Razmotrena je takodjer mogućnost upotrebe Z80 serije mikroprocesorskih komponenata u upravljanju različitih procesa, a prvenstveno u proizvodnji računarske opreme.

2.1 Z80-CPU (Central Processing Unit)

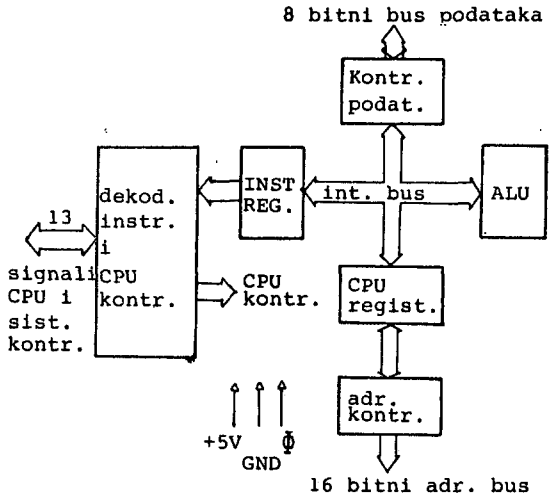
Z80-CPU mikroprocesor pripada tzv. "trećoj" generaciji mikroprocesora. To je jednočipni procesor (Blok dijagram dan je na sl. 1) čiji se instrukcioni set sastoji iz 158 instrukcija (uključujući 78 softverski kompatibilnih instrukcija od mikroprocesora Intel 8080).

Nove instrukcije uključuju 4, 8 i 16 bitne operacije uz desetak različitih načina adresiranja. Brzina izvršavanja instrukcija (kod standardne verzije) iznosi 1,6 mikrosek. za najbrže instrukcije. Rad s mikroprocesorom Z80 zahtijeva 25% do 50% manje memorije i ima oko 50% veću mogućnost obrade podataka nego I8080.

Na slici 2 prikazano je 208 bitova RAM memorije organizirane u 8 i 16 bitne registre koji su dostupni korisniku. Ti registri uključuju dvije grupe po 6 registara koji se mogu koristiti kao 8-bitni registri i kao 16-bitni registarski parovi. Postoje takodjer dvije grupe akumulatora i "flag" registara.

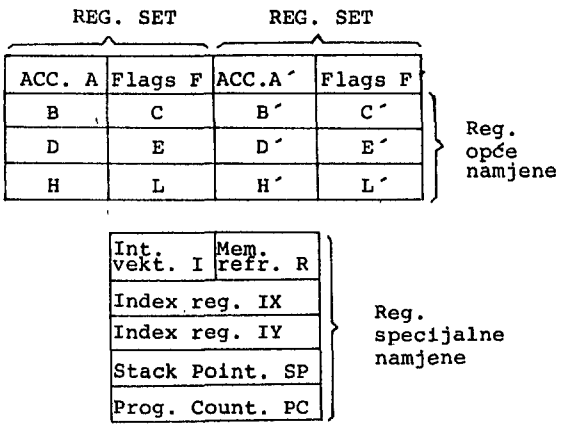
Programer ima pristup u bilo koju grupu registara preko jednostavnih instrukcija za izmjenu registara (EX). Takva organizacija omogućava vrlo brzu obradu zahtjeva za prekid jer se ne moraju privremeno spremati sadržaji registara već se samo zamijeni grupa registara s kojima se radi. Mikroprocesorski čip takodjer sadrži 16-bitni "Stack Pointer" koji mu omogućava neograničeni broj prekida i potprogramskih nivoa

te jednostavnu obradu podataka u velikom broju slučajeva.



Z80-CPU BLOK DIJAGRAM

Slika 1



REGISTRI Z80-CPU

Slika 2

Dva 16-bitna "index" registra dopuštaju operacije nad tabelama podataka i jednostavnu upotrebu relokatibilnog koda. Registar za osvježavanje dinamičkih memorija (Refresh Reg.)-R radi potpuno nezavisno i omogućava upotrebu jeftinih dinamičkih RAM memorija bez dodatnog hardvera. "Interrupt Register" (I) služi za formiranje gornjih 8 bitova adresnog vektora (donjih 8 bitova generira jedinica koja traži prekid). Na taj način dobivamo jednostavnu i moćnu obradu zahtjeva za prekid. Sadržaj lokacija na koje pokazuje vektor zahtjeva za prekid predstavlja početak programskog odsječka koji obrađuje zahtjev za prekid.

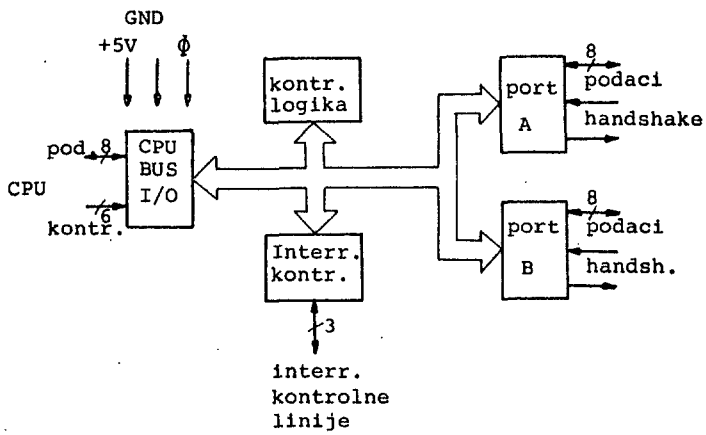
Osim ovog načina zahtjeva za prekid , postoje još dva standardna načina kao kod I8080 i jedan bezuvjetni (nemaskirani) zahtjev za prekid.

Instrukcioni set obuhvaća 8 bitne i 16 bitne "LOAD" instrukcije, izmjene sadržaja registara, premještanje memorijskih blokova, pretraživanje memorijskih blokova, 8-bitne aritmetičko logičke operacije, 16-bitne aritmetičke operacije, operacije nad sadržajem akumulatora i "Flag" registra, različite instrukcije za rotiranje i pomicanje sadržaja, operacije nad pojedinim bitovima bilo koje memorijske lokacije, grupu vrlo jakih ulazno/izlaznih instrukcija, instrukcije za skok na odredjenu lokaciju memorije, instrukcije za poziv potprograma i povratak iz njih te instrukcije opće namjene.

Sva prethodno navedena svojstva omogućavaju mikroprocesoru Z80 široku primjenu od najjednostavnijih zadataka do vrlo kompliciranih kao što su operacije nad blokovima memorije, blokovski ulazno/izlazni prijenos podataka i dr.

2.2 Z80-PIO (Parallel Input/Output)

Z80-PIO paralelni ulazno/izlazni kontroler je programabilni čip s dva U/I porta koji omogućava TTL kompatibilno paralelno povezivanje izmedju vanjskog svijeta i mikroprocesora Z80 (sl. 3). Z80-PIO može biti isprogramiran za vršenje različitih funkcija, a prvenstveno je namijenjen za paralelnu komunikaciju sa standardnim periferijskim jedinicama računala kao što su: čitači/bušači papirnih traka (kartica), štampači, tastature i dr.



Z80-PIO BLOK DIJAGRAM

Slika 3

Uz prethodno opisane karakteristike Z80-PIO je smješten u 40-pinsko DIP kućište, a najbitniji dio su mu dva nezavisna 8-bitna dvosmjerna porta s "handshake" kontrolom prijenosa podataka. Kontrola prijenosa podataka upravljana je zahtjevima za prekid.

Portovi mogu biti isprogramirani da rade u 4 različita moda:

- ulazni mod (Byte Input), port A i port B mogu se isprogramirati da primaju podatke iz vanjskog svijeta uz "handshake" kontrolu,

- izlazni mod (Byte Output), port A i B mogu se isprogramirati da šalju podatke na izlazne linije (uz "handshake" kontrolu),

- dvosmjerni (Bidirectional) mod moguć je samo za port A jer se koriste kontrolne linije od porta A i B. U tom slučaju port B može se koristiti samo u "BIT" modu. Preko 8-bitnog porta A moguća je dvosmjerna komunikacija uz "handshake" kontrolu u oba smjera.

- Bit mod omogućava bilo koju kombinaciju ulaznih i izlaznih bitova unutar jednog porta.

U ovom modu ne koriste se "handshake" signali. Mogu se odrediti samo neke linije koje omogućavaju zahtjev za prekid. Programski se može odrediti da se prekid dogodi uz odredjenu kombinaciju stanja na tim linijama.

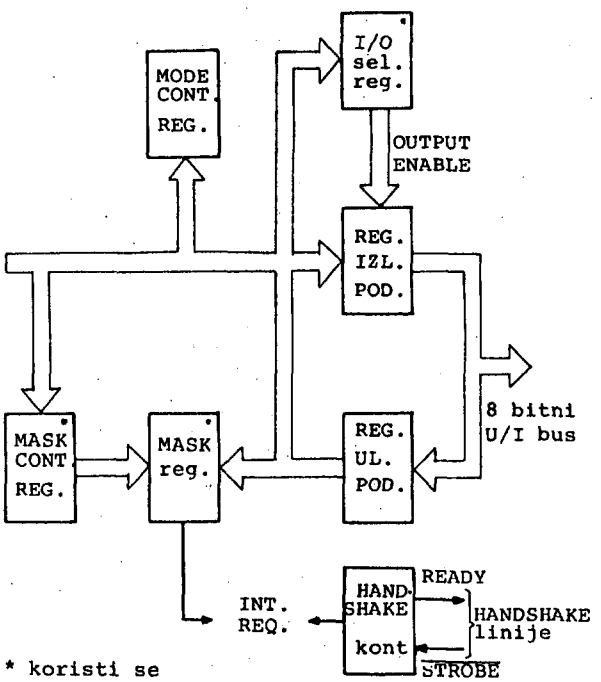
Z80-PIO komponenta (sl.3) sastoji se iz bus interfejsa, kontrolne logike, ulazno/izlazne logike za port A, U/I logike za port B i logike zahtjeva za prekid.

Tipična primjena jedne takve PIO komponente može biti: port A koristi se za prijenos podataka, a port B za odredjivanje statusa i kontrolu vanjskog uredjaja.

Na slici 4 prikazana je kontrolna logika jednog U/I porta. Sastoji se iz 6 registara i "handshake" kontrolne logike. Registri su: 8-bitni ulazni registar, 8-bitni izlazni registar, 8-bitni registar za selektiranje ulazno/izlaznih linija, 8-bitni registar u koji se upisuje "maska" tj. odredjuju bitovi koji mogu izazvati zahtjev za prekid. Zatim još postoje 2-bitni registar za odredjivanje moda rada i 2-bitni registar (u bit modu) odredjuje aktivno stanje koje izaziva zahtjev za prekid te da li će se on dogoditi na aktivno stanje bilo kojeg bita ili svih bitova pod kontrolom "mask" registra.

Kontrolna logika zahtjeva za prekid ima "Daisy Chain" organizaciju (kao i sve ostale komponente Z80) što znači da je prioritet bilo koje jedinice odredjen njezinom pozicijom u lancu povezivanja s računalom. Unutar same Z80-PIO komponente port A ima veći prioritet od porta B.

Kod vektorskog zahtjeva za prekid, svaki port sadrži donji dio int. vektora koji uz pomoć I registra u CPU formira pokazivač lokacije memorije u kojoj se nalazi adresa početka programa za obradu zahtjeva za prekid. Budući da se za jednu 16-bitnu adresu koriste dvije lokacije memorije, početak adrese mora biti uvijek na parnoj lokaciji pa je stoga najniži bit donjeg dijela int. vektora jednak 0.



* koristi se samo u BIT modu

BLOK DIJAGRAM Z80-PIO U/I PORTA

Slika 4

Programiranje Z80-PIO porta

Programiranje se vrši tako da se u kontrolni dio porta upisuju odredjeni bajtovi,npr. ako je najniži dio bajta koji se upisuje u kontrolni dio porta jednak 0, to znači da je preostalih 7 bitova donji dio int. vektora (budući da najniži bit mora biti 0 zbog parnih lokacija u memoriji gdje počinje adresa obrade prekida). Ta 0 ujedno označava da se u kontrolni dio porta upisuje donji dio int. vektora.

D7	D6	D5	D4	D3	D2	D1	D0
V7	V6	V5	V4	V3	V2	V1	Ø

Odredjivanje moda rada vrši se slijedećom kombinacijom kontrolnog bajta:

D7	D6	D5	D4	D3	D2	D1	D0
M1	M0	X	X	1	1	1	1

MOD — NE KORISTI SE — "MOD" BAJT

(0) 0 0 IZLAZNI MOD
(1) 0 1 ULAZNI -u-
(2) 1 0 DVOSMJERNI (BIDIR.)
(3) 1 1 BIT MOD

Ako je odredjen "bit mod" , potrebni su još slijedeći kontrolni bajtovi:

D7	D6	D5	D4	D3	D2	D1	D0
U/I$_7$	U/I$_6$	U/I$_5$	U/I$_4$	U/I$_3$	U/I$_2$	U/I$_1$	U/I$_0$

U/I = 1 BIT JE ULAZNI
U/I = 0 BIT JE IZLAZNI

zatim:

D7	D6	D5	D4	D3	D2	D1	D0
EI	AND/OR	HIGH/LOW	MASK FOLLOWS	0	1	1	1

KORISTI SE U MODU 3 — INTERR. KONTR. BAJT

DOZVOLJAVANJE ZAHTJEVA ZA PREKID

Ako je (D4 = 1), slijedeći kontrolni bajt je "maska". Samo one linije porta čija je maska 0 bit će uzete u obzir kod generiranja zahtjeva za prekid:

D7	D6	D5	D4	D3	D2	D1	D0
MB$_7$	MB$_6$	MB$_5$	MB$_4$	MB$_3$	MB$_2$	MB$_1$	MB$_0$

Zahtjev za prekid može biti dozvoljen ili zabranjen a da se ne mijenja preostali dio kontrole zahtjeva za prekid. To se postiže slijedećim kontrolnim bajtom:

INT. ENABLE	X	X	X	0	0	1	1

DOZVOLA ZAHTJEVA ZA PREKID

2.3 Z80-SIO (Serial Input/Output)

Serijski ulazno/izlazni kontroler je programabilni dvokanalni čip koji ima mogućnost formatiranja podataka za serijsku komunikaciju.

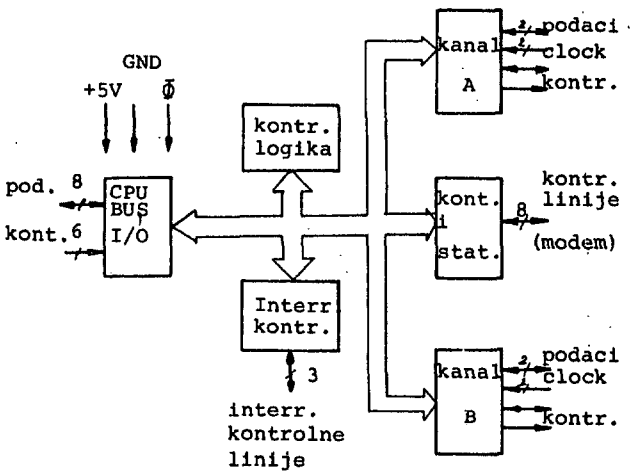
Omogućava podršku za asinhrone, sinhrone i sinhrone (bit orijentirane) protokole kao što su: IBM BiSync, HDLC, SDLC i bilo koji drugi serijski protokol. Automatski generira CRC (Cyclic Redundancy Check) karakter u sinhronoj komunikaciji. Takodjer može biti isprogramiran za bilo koju tradicionalnu asinhronu komunikaciju.

Z80-SIO je smješten u 40-pinsko DIP kućište i ima slijedeće karakteristike: dva potpuno nezavisna dvosmjerna kanala s brzinama prijenosa od 0 do 550 Kbita/s. , prijemni registri podataka su "četverostruko baferirani", a predajni "dvostruko baferirani", kod asinhrone komunikacije mogući su 5,6,7 ili 8 bitni karakteri, 1 i 2 stop bita, parni ili neparni paritet, taktni impulsi x1, x16, x32 i x64, generiranje i otkrivanje prekida te otkrivanje pogrešaka (Parity, Overrun, Framing error).

Kod binarne sinhrone komunikacije moguća je vanjska i unutarnja sinhronizacija, jedan ili dva sinhro karaktera u odvojenim registrima, automatsko umetanje "sinhro" karaktera te CRC generiranje i provjera. Kod HDLC ili IBM SDLC operacija moguće je automatsko umetanje i izbacivanje 0, umetanje oznake početka i kraja poruke, automatsko prepoznavanje adrese bloka, CRC generiranje i provjera i dr. (CRC-16 i CRC-CCITT su uključeni).

Uz sve prethodno opisane karakteristike postoji 8 ulazno/izlaznih linija za modem kontrolu.

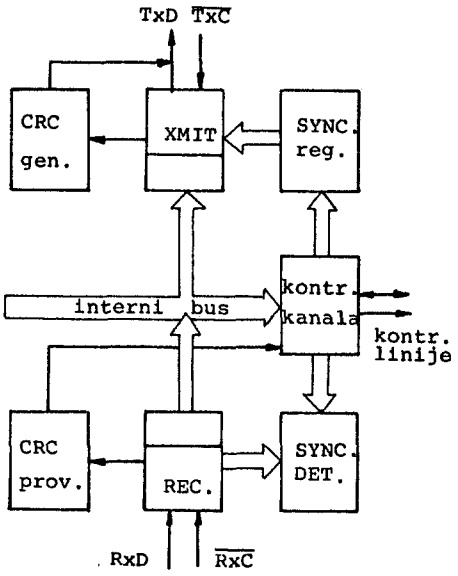
Blok dijagram Z80-SIO dan je na slici 5. SIO se sastoji iz bus interfejsa, interne kontrole, logike zahtjeva za prekid i dva dvosmjerna kanala. Kontrola zahtjeva za prekid odredjuje prioritet pojedinih dijelova (port A ima veći prioritet od porta B). Unutar porta redosljed prioriteta je: prijemni dio, predajni dio, vanjski signali i status.



Z80-SIO BLOK DIJAGRAM

Slika 5

Blok dijagram jednog kanala dan je na sl. 6. Svaki kanal ima pet 8-bitnih kontrolnih registara, dva 8-bitna status registra i dva reg. za sinhro karaktere. Donji dio vektora zahtjeva za prekid upisuje se u 8-bitni registar u kanalu B i isto tako može biti pročitan preko kanala B. Prijemni dio ima tri 8-bitna "buffer" registra u FIFO organizaciji i jedan 8-bitni posmačni registar. Predajni dio ima jedan "buffer" reg. i jedan 8-bitni posmačni registar. CRC generatori su 16-bitni posmačni registri programabilni za dva različita CRC polinoma.



BLOK DIJAGRAM Z80-SIO U/I PORTA

Slika 6

Zbog velikog broja kontrolnih i status registara te različitih mogućnosti programiranja rada ovog čipa, nećemo ulaziti u detalje programiranja i značenja odredjenih kombinacija bitova. Za potpuno razumijevanje i korištenje Z80-SIO komponente potrebno je proučiti tehničku dokumentaciju proizvodjača.

2.4 Z80-DMA (Direct Memory Access)

Z80-DMA komponenta je smještena u 40-pinsko DIP kućište. Omogućava generiranje svih adresnih, vremenskih i kontrolnih signala potrebnih za prijenos blokova podataka izmedju dva porta unutar Z80 sistema.

Ti portovi mogu biti povezani s glavnom memorijom i bilo kojom periferijskom jedinicom. DMA komponenta može pretraživati blok podataka za odredjenu kombinaciju bitova sa ili bez istovremenog prijenosa podataka.

Postoje 3 načina rada:

- prijenos bloka podataka
- pretraživanje bloka podataka
- pretraživanje bloka podataka i prijenos

Najbitnije karakteristike su: dvostruko generiranje adresa za vrijeme prijenosa (jedne za prijem podataka, druge za predaju), programabilni prijenos podataka i pretraživanje, automatsko

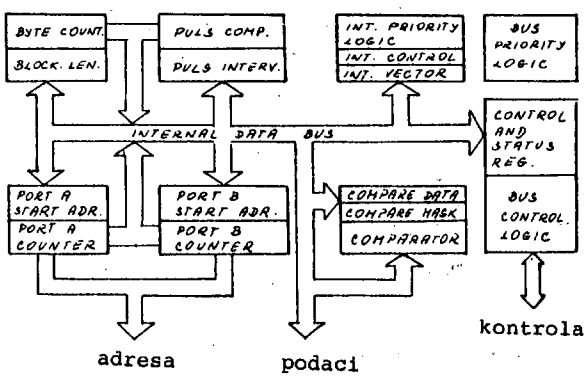
povećanje i smanjenje adrese portova od zadane startne adrese, taktni impulsi programabilni za brzinu bilo kojeg porta, zahtjevi za prekid u slučaju kraja bloka, pronadjenog uzorka i dr.

DMA može takodjer signalizirati kada je prenesen odredjeni broj bajtova bez zaustavljanja DMA prijenosa. Postoji mogućnost dobivanja statusa DMA jedinice na zahtjev programa, a isto tako stanje port brojača podataka i adresa.

Brzina prijenosa kreće se do 1,25 Mbajta/s kod pretraživanja ili prijenosa podataka.

Blok dijagram Z80-DMA dan je na slici 7. Sastoji se iz bus interfejsa, kontrolne logike i registara, adresnog dijela, brojača bajtova, dijela za generiranje taktnih impulsa potrebnih za prijenos podataka i dr.

Postoji niz registara kao što su: kontrolni registri, registri za odredjivanje vremenskih parametara portova, registar vektorskog zahtjeva za prekid, registar dužine bloka podataka, registar broja prenešenih podataka, adresni registri za port A i B, adresni brojači za port A i B, status registri i dr.



BLOK DIJAGRAM Z80-DMA

Slika 7

DMA čip moguće je isprogramirati za 4 različita moda rada:

- "Byte at a time" , kontrola se vraća CPU nakon svakog jednobajtnog ciklusa

- "Burst", operacija se nastavlja tako dugo dok je DMA "Ready" signal aktivan. Kontrola se vraća CPU kada "Ready" signal prestaje biti aktivan(tj. kada perif. jedinica više nije spremna za komunikaciju) ili je kraj bloka podataka koji se prenose (pretražuju)

- "Continuous",prijenos ili pretraživanje bloka podataka se dovrši prije nego kontrolu preuzme CPU

- "Transparent", DMA prijenos obavlja se za vrijeme osvježavanja dinamičke memorije (bez usporavanja rada CPU).

Za vrijeme prijenosa podataka, čitaju se podaci s jednog porta DMA i šalju na drugi port DMA. Portovi mogu biti isprogramirani da povezuju memoriju ili periferijske jedinice. Tako blokovi podataka mogu biti prenešeni iz jedne perif. jedinice u drugu, iz jednog dijela memorije u drugi ili iz periferijske jedinice u memoriju.

Tokom pretraživanja bloka podataka, podaci se samo čitaju i usporedjuju sa sadržajem dvaju registara DMA od kojih jedan usporedjuje bajtove, a drugi samo unaprijed odredjene bitove unutar jednog bajta. Kada se pronadje odgovarajuća kombinacija, postavi se status bit ili generira zahtjev za prekid.

U kombiniranom pretraživanju i prijenosu podataka, prenosi se blok podataka dok se ne otkrije odgovarajuća kombinacija. Iza toga prijenos se može nastaviti ili prekinuti.

Postoji više različitih mogućnosti rada Z80-DMA komponente koje zbog ograničenog prostora nisu ovdje objašnjene.

Budući da su instrukcijskim setom Z80 omogućene operacije nad blokovima podataka uz veliku brzinu izvršavanja operacija, u većini jednostavnijih primjena mikroprocesora Z80 nije potrebna DMA komponenta medjutim zbog svoje programabilnosti i velikih mogućnosti rada, korisno ju je upotrijebiti tamo gdje je potreban brzi prijenos i pretraživanje blokova podataka.

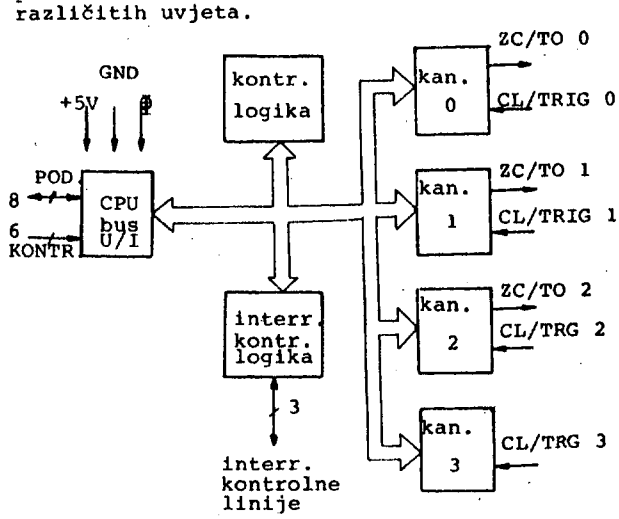
Z80-CTC (Counter Timer Circuit)

Z80-CTC je programabilna 4-kanalna jedinica koja omogućava brojenje i generiranje vremenskih impulsa različite dužine trajanja. Svaki kanal može biti isprogramiran da radi kao brojač ili generator vremenskih impulsa uz različite mogućnosti generiranja zahtjeva za prekid.

Z80-CTC gradja dana je na slici 8. Svaki kanal (slika 9) sastoji se iz 2 registra (kontrola kanala i registar za vremensku konstantu), 2 brojača i kontrolne logike.

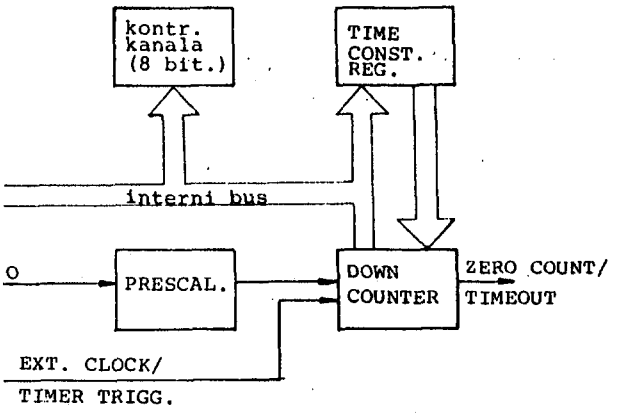
"Time Constant Register" (8-bitni) puni se preko CPU a služi za inicijalizaciju i ponovno upisivanje u "Down Counter" registar kad ovaj dodje do 0.

"Channel Control Register" (8-bitni) puni se preko CPU i služi za odredjivanje moda rada i različitih uvjeta.



Z80- CTC BLOK DIJAGRAM

Slika 8



BLOK DIJAGRAM KANALA
Z80-CTC

Slika 9

"Down Counter" se puni preko "Time Constant" registra pod programskom kontrolom ili automatski u trenutku kada sadržaj "Down Countera" postane 0. U bilo kojem trenutku CPU može pročitati sadržaj tog registra. Sadržaj registra umanjuje se za 1 ili preko "Prescaler"-a u "TIMER" modu ili impulsa na liniji CLK/TRIG u "COUNTER" modu.

"Prescaler" (8-bitni) registar dijeli sistemski takt sa 16 ili 256 i umanjuje sadržaj "Down Counter"-a (unotrebljava se u TIMER modu).

Primjeri odredjivanja moda rada i programiranja Z80-CTC-a dani su u nastavku.

3. PRIMJERI PROGRAMIRANJA Z80 MIKROPROCESORSKIH KOMPONENATA ZA IZVRŠAVANJE RAZLIČITIH ZADATAKA

Na slici 10 dani su primjeri inicijalizacije Z80 mikroprocesorskih komponenata za izvršavanje odredjenih zadataka.

3.1 Programiranje Z80-PIO (PORT A)

Kombinacija bajtova u tabeli PTABA (sl. 10) omogućava programiranje PIO porta A za "INPUT" mod rada:

12) svih 8 bitova služi za paralelni ulaz podataka, a unošenje podataka kontrolira se pomoću linija STROBE i READY

11) donji dio int. vektora je 20 H. (gornji dio int. vektora nalazi se u I registru CPU)

13) slanjem ovog bajta u kontrolni dio porta A omogućava se generiranje zahtjeva za prekid.

3.2 Programiranje Z80-PIO (PORT B)

Kombinacija bajtova u tabeli PTABB koja se upiše u kontrolni dio porta B, omogućava portu B da radi u "BIT" modu:

23) Bitovi D0 do D5 su ulazni, a bitovi D6 i D7 su izlazni

24) zahtjev za prekid dogodit će se ako se promijeni bilo koja od nemaskiranih linija u stanje "1"

25) zahtjev za prekid mogu izazvati samo linije D0 do D5.

3.3 Programiranje CTC kanala 2

Nakon što se u CTC kanal 2 upiše bajtove iz tabele CTAB2, kanal 2 raditi će u "TIMER" modu što znači da će generirati impulse i zahtjeve za prekid u odredjenim vremenskim intervalima (period $T=t_c \cdot P \cdot TC$. U ovom slučaju $T= 0,4 \cdot \cdot 256 \cdot 255$ mikrosek. , približno 26msek.). (t_c je sistemski takt, P je vrijednost "Prescaler"-a, a TC je vrijednost "Time Constant" reg.).

3.4 Programiranje CTC kanala 0

Kombinacija bajtova iz tabele CTAB0 omogućava kanalu 0 da radi u "COUNTER" modu. Zahtjev za prekid dogodit će se u ovom slučaju nakon što brojač izbroji 11_{10} vanjskih impulsa. Brojač se automatski inicijalizira kad dodje do 0.

3.5 , 3.6 Programiranje Z80-SIO

Programiranje SIO porta B

Bajtovi iz tabele STABB omogućavaju SIO portu B primanje i slanje podataka u asinhronoj komunikaciji:

62) donji dio int. vektora

64) paran paritet, jedan stop bit, x16 takt, asinhrona komunikacija

66) 7-bitni karakter koji se šalje, slanje dozvoljeno

68) 7-bitni karakter koji se prima, primanje dozvoljeno

6A) zahtjev za prekid generira svaki karakter.

Programiranje SIO porta A

Upisivanjem 12 bajtova iz tabele STABA u kontrolni dio SIO porta A, vrši se programiranje tog porta za "SDLC" komunikaciju.

3.7 Programiranje Z80-DMA

Slijedeći primjer pokazuje kako DMA čip može biti isprogramiran za prijenos podataka iz periferijske jedinice priključene na DMA port A u memoriju (priklj. na port B).

Tabela DMATAB sadrži bajtove koji to omogućavaju, a programiranje DMA čipa vrši se izlaznom instrukcijom "OTIR" koja prenosi bajtove iz tabele DMATAB u kontrolni dio DMA čipa.

```
INKO   INICIJALIZACIJA Z80 KOMPONENATA
  ADDR  OBJECT    ST #

                  0002  NAME INKO
                  0003 ;*************************************
                  0004 ;ID:     M. ZAGAR    V1.1    251178    *
                  0005 ;*************************************
                  0006 ;
                  0007          ORG     0100H
                  0008 ;*************************************
                  0009          GLOBAL  KPINIC
                  0010 ;*************************************
  >0000           0011 INTUP    EQU     0H
  >1000           0012 STP      EQU     1000H
                  0013 ;*
  >0004           0014 PDATA    EQU     04H     ;PIO (PORT A) PODACI
  >0005           0015 PADPA    EQU     05H     ;PIO (PORT A) KONTROLA
  >0006           0016 PDATE    EQU     06H     ;PIO (PORT B) PODACI
  >0007           0017 PADPB    EQU     07H     ;PIO (PORT B) KONTROLA
                  0018 ;*
  >0008           0019 CADR0    EQU     08H     ;CTC CHAN.0
  >0009           0020 CADR1    EQU     09H     ;CTC CHAN.1
  >000A           0021 CADR2    EQU     0AH     ;CTC CHAN.2
  >000B           0022 CADR3    EQU     0BH     ;CTC CHAN 3
                  0023 ;*
  >000C           0024 SDATA    EQU     0CH     ;SIO (PORT A) PODACI
  >000D           0025 SADPA    EQU     0DH     ;SIO (PORT A) KONTROLA
  >000E           0026 SDATB    EQU     0EH     ;SIO (PORT B) PODACI
  >000F           0027 SADPB    EQU     0FH     ;SIO (PORT B) KONTROLA
                  0028 ;*
  >0010           0029 DMAADR   EQU     10H     ;ADRESA DMA (KONTROLA)
                  0030 ;*************************************
                  0031 ;ADRESE INT. VEKTORA
                  0032 ; PIO INT. ADR. (A) = 0020H
                  0033 ; PIO INT. ADR. (B) = 0022H
                  0034 ; CTC0 INT.ADR.     = 0030H
                  0035 ; CTC1 INT.ADR.     = 0032H
                  0036 ; CTC2 INT.ADR.     = 0034H
                  0037 ; CTC3 INT.ADR.     = 0036H
                  0038 ; SIO  INT.ADR. (B) = 004CH
                  0039 ;*************************************
                  0040 ;
                  0041 ; POCETAK INICIJALIZACIJE
 '0100  F3        0042 POC:     DI              ;ONEMOGUCI PREKID
 '0101  3E00      0043          LD A,INTUP      ;GORNJI DIO INT. VEKT.
 '0103  ED47      0044          LD I,A          ;SPREMI U I REG.
 '0105  310010    0045          LD SP,STP       ;INIC. "STACK POINTER"
 '0108  ED5E      0046          IM 2            ;VEKT. ZAHTJ. ZA PREK.
                  0047 ;
                  0048 ; PROGRAMIRANJE PIO (PORT A)
                  0049 ;
 '010A  215C01'   0050          LD HL,PTAEA     ;POC. TAB. ZA INIC.
 '010D  0603      0051          LD B,3          ;BROJ BAJTOVA
 '010F  0E05      0052          LD C,PADPA      ;ADR. PORTA (KONTR.)
 '0111  EDB3      0053          OTIR            ;PROGRAMIRANJE PORTA
                  0054 ;
                  0055 ; PROGRAMIRANJE PIO (PORT B)
                  0056 ;
 '0113  215301'   0057          LD HL,PTAEB     ;TAB. INIC. PIO-B
 '0116  0606      0058          LD B,6          ;BROJ BAJTOVA
 '0118  0E07      0059          LD C,PADPB      ;ADR. PORTA (KONTR.)
```

```
INKO   INICIJALIZACIJA Z80 KOMPONENATA
  ADDR  OBJECT    ST #

 '011A  EDB3      0060          OTIR            ;PROGRAMIRANJE PORTA
                  0061 ;
                  0062 ; PROGRAMIRANJE CTC (KANAL 2)
                  0063 ;
 '011C  215901'   0064          LD HL,CTAE2     ;TAB. INIC. CTC-2
 '011F  0603      0065          LD B,3          ;BROJ BAJTOVA
 '0121  0E08      0066          LD C,CADR0      ;DONJI DIO INT. VEKT.
 '0123  EDA3      0067          OUTI            ;UPISUJE SE U KANAL 0
 '0125  0E0A      0068          LD C,CADR2      ;OSTALE KOMANDE
 '0127  EDB3      0069          OTIR            ;UPISUJU SE U KANAL 2
                  0070 ;
                  0071 ; PROGRAMIRANJE CTC (KANAL 0)
                  0072 ;
 '0129  215C01'   0073          LD HL,CTAE0     ;TAB. INIC. CTC-0
 '012C  0603      0074          LD B,3          ;BROJ BAJTOVA
 '012E  0E08      0075          LD C,CADR0      ;ADRESA KANALA
 '0130  EDB3      0076          OTIR            ;PROGRAMIRANJE KANALA
                  0077 ;
                  0078 ; PROGRAMIRANJE SIO (PORT B)
                  0079 ;
 '0132  216B01'   0080          LD HL,STAEB     ;TAB. INIC. SIO-B
 '0135  060A      0081          LD B,0AH        ;BROJ BAJTOVA
 '0137  0E0F      0082          LD C,SADPB      ;ADRESA PORTA B
 '0139  EDB3      0083          OTIR            ;PROGRAMIRANJE PORTA B
                  0084 ;
                  0085 ; PROGRAMIRANJE SIO (PORT A)
                  0086 ;
 '013B  215F01'   0087          LD HL,STAEA     ;TAB. INIC. SIO-A
 '013E  060C      0088          LD B,0CH        ;BROJ BAJTOVA
 '0140  0E0D      0089          LD C,SADPA      ;ADRESA PORTA A
 '0142  EDB3      0090          OTIR            ;PROGRAMIRANJE PORTA A
                  0091 ;
                  0092 ; PROGRAMIRANJE DMA
                  0093 ;
 '0144  217501'   0094          LD HL,DMATAB    ;TAB.INIC. DMA
 '0147  060C      0095          LD B,0CH        ;BROJ KONTR. BAJTOVA
 '0149  0E10      0096          LD C,DMAADR     ;ADR. DMA JEDINICE
 '014B  EDB3      0097          OTIR            ;PROG. DMA JEDINICE
                  0098 ;
                  0099 ; KRAJ PROGRAMIRANJA Z80 KOMPONENATA
                  0100 ;
 '014D  C3FFFF    0101 KRAJ     JP KRINIC       ;NAST. PROG. NA "KRINIC"
                  0102 ;
                  0103 ;*************************************
                  0104 ; TABELA PODATAKA POTREBNIH ZA INICIJALIZACIJU
                  0105 ; Z80 KOMPONENATA
                  0106 ;*********************************************
                  0107 ;
                  0108 ; KONTROLNI BAJTOVI ZA PIO PORT A  "INPUT MODE"
 '0150  20        0109 PTAEA:   DEFB    020H    ;11) INTERRUPT VECTOR
 '0151  4F        0110          DEFB    04FH    ;12) INPUT MODE
 '0152  83        0111          DEFB    083H    ;13) INTERRUPT CONTROL
                  0112 ;*
                  0113 ; PIO PORT B  "BIT MODE"
 '0153  22        0114 PTAEB:   DEFB    022H    ;21) INTERRUPT VECTOR LO
 '0154  CF        0115          DEFB    0CFH    ;22) BIT MODE
 '0155  3F        0116          DEFB    03FH    ;23) INPUT-OUTPUT LINES
 '0156  37        0117          DEFB    037H    ;24) INTERRUPT CONTROL
```

```
INKO   INICIJALIZACIJA Z80 KOMPONENATA
  ADDP  OBJECT    ST #

'0157  C0        0118          DEFB    0C0H    ;25) MASK
'0158  83        0119          DEFB    083H    ;26) INTERPUPT ENAELE
                 0120 ;*
                 0121 ; CTC CHAN. 2  "TIMER MODE"
'0159  30        0122 CTAB2:   DEFB    030H    ;31) INTERRUPT VECTOP
'015A  A5        0123          DEFB    0A5H    ;32) TIMEP MODE
'015B  FF        0124          DEFB    0FFH    ;33) TIME CONSTANT (255D
                 0125 ;*
                 0126 ; CTC CHAN.0  "COUNTEP MODE"
'015C  30        0127 CTAB0:   DEFB    030H    ;41) INTERPUPT VECTOP
'015D  C5        0128          DEFB    0C5H    ;42) COUNTEP MODE
'015E  0B        0129          DEFB    00BH    ;43) DOWN COUNTEP CONST.
                 0130 ;*
                 0131 ; SIO POPT A  "SDLC TRANSFERS"
'015F  04        0132 STABA:   DEFB    004H    ;51) POINT. TO REG. 4A
'0160  20        0133          DEFB    020H    ;52) CONTROL EYTE
'0161  46        0134          DEFB    046H    ;53) REG. 6A
'0162  FF        0135          DEFB    0FFH    ;54) CONTPOL EYTE
'0163  87        0136          DEFB    087H    ;55) PEG. 7A
'0164  7E        0137          DEFB    07EH    ;56) CONTPOL EYTE
'0165  01        0138          DEFB    001H    ;57) PEG. 1A
'0166  17        0139          DEFB    017H    ;58) CONTPOL EYTE
'0167  15        0140          DEFB    015H    ;59) REG. 5A
'0168  E8        0141          DEFB    0E8H    ;5A) CONTPOL EYTE
'0169  03        0142          DEFB    003H    ;5B) REG. 3A
'016A  ED        0143          DEFB    0EDH    ;5C) CONTPOL EYTE
                 0144 ;*
                 0145 ; SIO POPT B  "ASYNCHRONOUS COMMUNICATION"
'016B  02        0146 STABB:   DEFB    002H    ;61) POINT TO PEG. 2B
'016C  40        0147          DEFB    040H    ;62) CONTPOL EYTE
'016D  04        0148          DEFB    004H    ;63) PEG. 4B
'016E  47        0149          DEFB    047H    ;64) CONTPOL EYTE
'016F  05        0150          DEFB    005H    ;65) REG. 5B
'0170  4A        0151          DEFB    04AH    ;66) CONTPOL EYTE
'0171  03        0152          DEFB    003H    ;67) REG. 3B
'0172  61        0153          DEFB    061H    ;68) CONTPOL EYTE
'0173  01        0154          DEFB    001H    ;69) PEG. 1B
'0174  17        0155          DEFB    017H    ;6A) CONTPOL EYTE
                 015E ;*
                 0157 ; DMA "TPANSFEP DATA FPOM PEPIPHEPAL (POPT A)
                 0158 ; TO MEMOPY (POPT B)
'0175  6D        0159 DMATAB:  DEFB    06DH    ;71) COMMAND EYTE 1A
'0176  05        0160          DEFB    005H    ;72) P. A ADDP. L. 8PITS
'0177  00        0161          DEFB    000H    ;73) BLOCK LENG. L.8PITS
'0178  10        0162          DEFB    010H    ;74) BLOCK LENG. H.8PITS
'0179  28        0163          DEFB    028H    ;75) COMMAND EYTE 1P
'017A  14        0164          DEFB    014H    ;76) COMMAND EYTE 1E
'017B  CD        0165          DEFB    0CDH    ;77) COMMAND EYTE 2B
'017C  50        0166          DEFB    050H    ;78) P.B. ADDP. L. 8PITS
'017D  10        0167          DEFB    010H    ;79) P.B. ADDP. H. 8PITS
'017E  8A        0168          DEFB    08AH    ;7A) COMMAND EYTE 2C
'017F  CF        0169          DEFB    0CFH    ;7B) COMMAND EYTE 2D
'0180  87        0170          DEFB    087H    ;7C) COMMAND EYTE 2D
                 0171 ;*
                 0172 ; KPAJ TABELE
                 0173 ;*****************************************
                 0174 ;
                 0175          END

EPROPS=0000
EPROPS=0000

SINKO  05016A
$KPINIC03014EED
:20010000F33E00ED47310010EDSE2150010E030E05EDB32153010E0ECEC7EDB32159010
:20012000030E08EDA30E0AEDB3215C010E030E08EDB321EE01E60A0E0FEDB3215FC1E60
:20014000E0EDEDB32175010E0C0E10EDB3C3FFFF204F8322CF3F37C08330A5FF3CC5C98
:20016000204EFF877EC11715E803ED0240044705 4A03E101176DD500102814CD50108A8
:0101800087F7
$0E000004010E0114011D012A0133013C0145CD
:00010001FE


INKO   INICIJALIZACIJA Z80 KOMPONENATA
  ADDR  OBJECT    ST #


CADR0        0008 CADR1        0009 CADR2        000A CADR3        000B
CTAB0        015C CTAB2        0159 DMAADR       0010 DMATAB       0175
INTUP        0000 KPAJ         014D KPINIC [EXT] 014E PADRA        0005
PADRB        0007 PDATA        0004 PDATE        0006 POC          0100
PTABA        0150 PTABB        0153 SADRA        000D SADRB        000F
SDATA        000C SDATE        000E STABA        015F STABB        016B
STP          1000
ERPOPS=0000
```

PROGRAM ZA INICIJALIZACIJU Z80 KOMPONENATA

Slika 10

(Instrukcija "OTIR" izvrši prijenos jednog bajta podataka iz memorijske lokacije adresirane indirektno preko "HL" registarskog para na izlaz čija se adresa nalazi u "C" registru. Zatim uveća sadržaj HL reg. para za 1, smanji sadržaj "B" registra za 1. Ako sadržaj "B" reg. nije "0" , ponovi sve korake izvršavanja. (Ako sadržaj "B" reg. je "0", izvršava se slijedeća instrukcija po redu).

72) adresa periferijske jedinice je "5"

73) i 74) dužina bloka koji se prenosi je 1000H bajtova

75) port A ima adresu koja se ne mijenja

76) port B ima adresu koja se uvećava za 1

77) "BURST" mod rada što znači da se DMA prijenos vrši tako dugo dok je port A spreman za slanje ili je kraj bloka koji se prenosi

78) i 79) početna adresa porta B je 1050 H

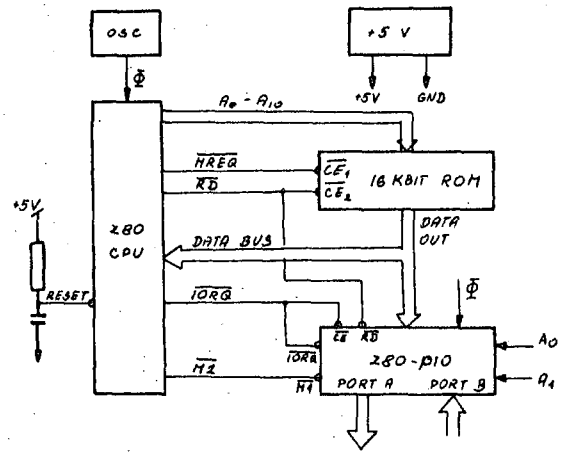
7A), 7B) i 7C) priprema DMA komunikacije i njezino omogućavanje.

Ukratko, nakon prethodno upisanih kontrolnih bajtova u Z80-DMA, omogućen je DMA prijenos bloka od 1000H bajtova iz periferijske jedinice (adresa 5H) u memoriju s početnom adresom 1050H. Adresa se automatski uvećava za 1 nakon upisa svakog bajta.

4. PRIMJENA MIKROPROCESORSKIH KOMPONENATA Z80

Mikroprocesor Z80-CPU i dodatne komponente su zbog svojih karakteristika vrlo pogodne za rješavanje različitih problema.

U najjednostavnijem slučaju moguća je minimalna konfiguracija od samo 2 čipa (Z80-CPU i Z80-PIO) plus dodatna memorija (slika 11).



Z80 MINIMALNA KONFIGURACIJA

Slika 11

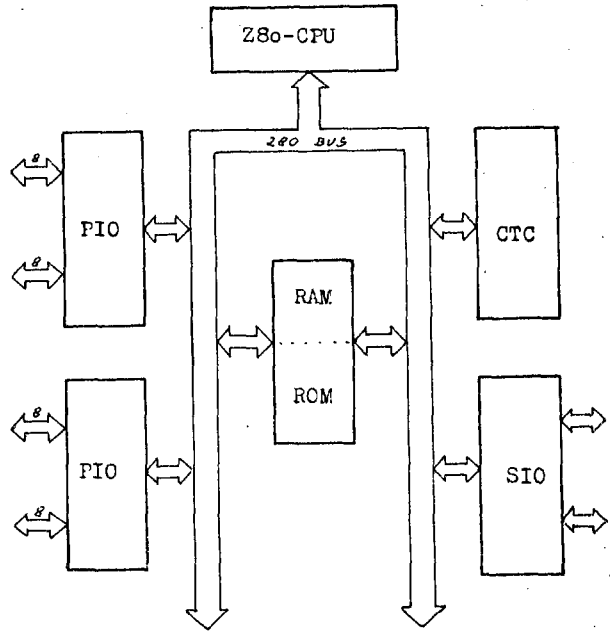
Ta je konfiguracija dovoljna za rješavanje niza problema upravljanja jednostavnijih procesa (dostupno je 16 ulazno/izlaznih linija koje se mogu isprogramirati po želji). Može se koristiti bilo koji tip memorija danas raspoloživih na tržištu (statičke, dinamičke) jer je zbog koncepcije CPU (koji ima 16 linija za adresiranje i automatski "refresh" tj. osvježavanje dinamičkih memorija) to vrlo jednostavno.

Proširivanjem sistema (tako da dodajemo nove komponente iz serije Z80) može se dobiti sistem za upravljanje većine složenih procesa. Maksimalna konfiguracija sistema može imati do 64 Kbajta memorije i preko 1000 ulazno/izlaznih linija uz vrlo snažan instrukcioni set koji omogućava brzo izvršavanje operacija uz mali kapacitet memorije.

Kao što je već prethodno bilo opisano, serija Z80 sastoji se iz procesorskog elementa (mikroprocesor), čipa za paralelnu komunikaciju, čipa za serijsku komunikaciju, DMA čipa i čipa za generiranje vremenskih impulsa i brojenje. Takva koncepcija daje mu velike mogućnosti u proizvodnji računarske opreme.

Serija Z80 može se upotrijebiti kao procesorski dio većeg računala. To omogućavaju vrlo jake komponente za serijsku i paralelnu komunikaciju s okolinom (uz to i programabilne dakle prilagodljive različitim zahtjevima), zatim jednostavno generiranje različitih vremenskih signala te mogućnost DMA komunikacije sa sistemskom memorijom. Instrukcioni set omogućava 4,8 i 16 bitne operacije, vrlo snažne ulazno/izlazne i memorijske instrukcije koje omogućavaju operacije nad blokovima podataka, različiti načini adresiranja memorije i dr.

S druge strane ista grupa Z80 komponenata može se koristiti u upravljanju periferijskih jedinica računala.



BLOK DIJAGRAM SISTEMA Z80

Slika 12

Na slici 12 prikazana je jedna takva konfiguracija koja omogućava primjenu u upravljanju različitih periferijskih jedinica kao što su: jedinice magnetskih kazeta, "floppy" diskova, štampača, video-terminala, čitača/bušača papirnih traka (kartica) dakle svih standardnih periferijskih jedinica računala. Time se glavno računalo oslobadja niza nepotrebnih poslova jer se korištenjem mikroprocesorskog sistema za upravljanje periferijske jedinice dobije inteligentni uredjaj koji može obavljati niz poslova bez intervencije glavnog računala, npr. priprema i unošenje podataka, pohranjivanje podataka, njihovo ispravljanje te vrlo brza komunikacija s glavnim računalom kada je to potrebno.

Hardver koji se koristi ostaje uglavnom isti. Mijenja se samo upravljački dio programa koji omogućava da jednom razradjenu hardversku konfiguraciju koristimo npr. za upravljanje jedinice magnetskih kazeta ali i za upravljanje neke druge periferijske jedinice računala.

5. ZAKLJUČAK

Prethodno navedene mogućnosti upotrebe mikroprocesora Z80 kao procesorskog elementa većeg računala, a isto tako i upotreba Z80 komponenata u upravljanju periferijskih jedinica računala, zahtijevaju posebnu pažnju.

Sa stanovišta proizvodnje računarske opreme (pogotovo u domaćim uvjetima) postoji za takav način rješavanja problema više razloga.

Medju njima su najvažniji:

- kompatibilnost izmedju različitih uredjaja s ugradjenim Z80 komponentama
- manji broj različitih komponenata koje se koriste, a time i jednostavnija nabava te jednostavniji razvoj i izgradnja odredjenih jedinica računala
- mogućnost programiranja Z80 komponenata za izvršavanje različitih zadataka
- kompatibilnost s već postojećim uredjajima s ugradjenim komponentama iz serije INTEL
- mogućnost razvoja i korištenja novijih komponenata koje se pojavljuju na tržištu (Z8000 - 16 bitni mikroprocesor koji proizvodi firma Zilog, ima bolje karakteristike od DEC PDP 11/45 i prevodioce za PLZ,BASIC,COBOL,FORTRAN i dr.
  Uz upotrebu posebnog prevodioca moguće je pretvoriti "kod" mikroprocesora Z80 (8-bitni) u "kod" procesora Z8000 (16-bitni) jer su u mikroprocesoru Z8000 zadržane osnovne karakteristike mikroprocesora Z80 i dodana nova svojstva (adresiranje 8 Mbajta memorije i dr.) koja mu daju veće mogućnosti obrade podataka).

U cijeni cijelog uredjaja, cijena ugradjenih mikroprocesorskih komponenata predstavlja jedan manji dio, a uz niz prednosti koje se dobiju razvojem univerzalnog mikroprocesorskog sistema za razne primjene gdje se mijenja samo softverski dio, ta cijena samih komponenata još manje dolazi do izražaja.

Prema tome, primjena mikroprocesorskih komponenata Z80 u proizvodnji računarske opreme ima puno opravdanje.

LITERATURA:

1) MOSTEK: Z80 Technical Manual, MK3880 Central Processing Unit , 1976.

2) Mostek: Z80 Technical Manual, MK3881 Parallel I/O Controlier, 1976.

3) ZILOG: Z80-SIO, Product Specification, October 1977.

4) ZILOG: Z80-CTC, Product Specification, 1977.

5) ZILOG: Z80-DMA, Product Specification, 1977.

6) M.Žagar: Upravljanje kompjuterskih periferijskih jedinica s mikroprocesorima, Magistarski rad, 1978.

7) M.Žagar: Prednosti upotrebe mag. kazete tipa DC300A u odnosu na klasične medije za zapis podataka, INFORMATICA 78, Bled 1978.

računalniško generiranje popolnih strategij za šahovske končnice

m.gams
i.bratko

UDK 681.3:794.1

Institut J.Stefan in Fakulteta za elektrotehniko,
Univerza v Ljubljani, Ljubljana

V članku je opisan algoritem, ki v smeri od zadaj naprej zgradi prostor vseh možnih pozicij dane igre in za vsako pozicijo določi njeno vrednost. S tem algoritmom smo izračunali popolno strategijo za šahovsko končnico kralj in trdnjava proti kralju in skakaču, kar je zaradi kompleksnosti izračuna zahtevna programska naloga. Rezultati so zanimivi tudi s stališča same šahovske igre, ker je optimalna igra v tej končnici v splošnem že preveč težavna za šahovske mojstre. Ta eksperiment vodi k zaključku, da je na tak način možno v celoti reševati končnice z največ s pet do šest figurami, medtem ko postane od tu dalje kompleksnost izračuna že praktično nesprejemljiva in je potrebna uporaba metod umetne inteligence.

COMPUTER GENERATION OF COMPLETE STRATEGIES FOR CHESS END-GAMES. The article presents a backward-chaining algorithm for the generation of the space of all possible positions for a given game and for each position its value is computed. Using this algorithm, a complete strategy for the king and rook vs. king and knight chess ending was generated, which proved to be a programming task of considerable difficulty. The results are of interest also from the chess game point of view since optimal play in this ending is beyond chess-master's skill. A conclusion of the experiment is that endings with at most 5 or 6 pieces can still be completely solved in this way, while for more complex endings the computation becomes infeasable and from here on artificial intelligence methods are to be employed.

UVOD

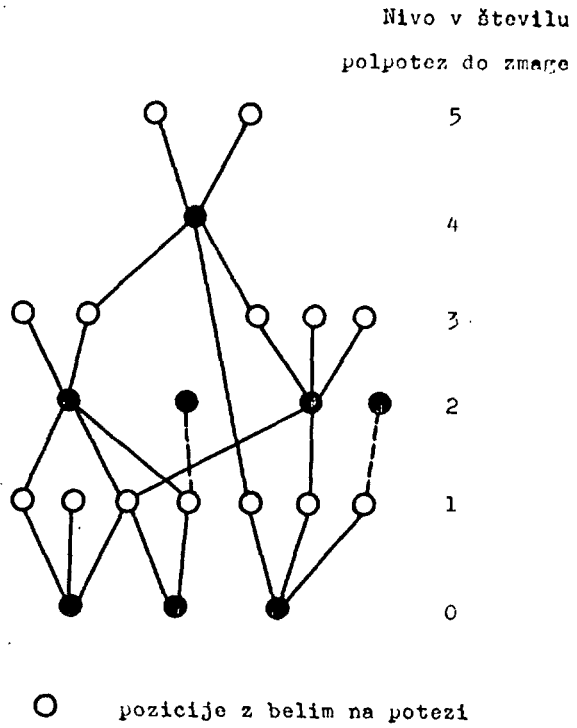
Znani so razmeroma enostavni algoritmi, ki bi načeloma popolnoma pravilno igrali šah. Mednje sodi npr. algoritem, ki preišče vsa možna nadaljevanja iz dane pozicije, dokler ne naleti na končne pozicije, tj. take, ki so dobljene, izgubljene ali remi po pravilih igre (npr. eden izmed kraljev v matu). Imenujmo tak algoritem "algoritem A". Hitro postane očitno, da algoritem A praktično ni izvedljiv zaradi svoje *časovne kompleksnosti*.

V povprečju je v vsaki šahovski poziciji možnih okrog 35 potez. Tako mora algoritem A v dani začetni poziciji, kjer je npr. na potezi beli, pregledati 35 potez belega; za vsako od nastalih pozicij mora upoštevati po 35 možnih odgovorov črnega itd. Tako mora za preiskovanje samo do globine ene cele poteze (tj. dveh polpotez, belega in črnega) pregledati okrog 1000 pozicij, za vsako naslednjo potezo v globino pa število pozicij naraste še za faktor 1000. Če optimistično ocenimo, da lahko računalnik generira po eno legalno potezo v eni mikrosekundi, potem bi potrebovali za izračun najboljše poteze v začetni šahovski poziciji po algoritmu A čas v velikostnem razredu, ki presega 10^{100} let (starost našega vesolja je približno 10^{10} let).

Algoritem A preiskuje možna nadaljevanja v smeri naprej od dane začetne pozicije proti končnim pozicijam. Drugi algoritem, ki tudi v načelu rešuje šahovsko igro, preiskuje prostor možnih pozicij v smeri nazaj. Imenujmo ga algoritem B. Algoritem B generira popolno strategijo npr. za belega tako, da generira najprej vse pozicije, ki so po definiciji dobljene za belega, npr. črni kralj v matu. Te pozicije so dobljene v 0 polpotezah (slika 1). Iz vseh teh pozicij poišče vzvratne poteze belega in tako dobi množico vseh pozicij, ki so dobljene za belega v eni polpotezi. Zatem poišče vse pozicije s črnim na potezi, v katerih vse možne poteze črnega vodijo v pozicije, dobljene za belega v eni polpotezi. Tako imamo množico vseh pozicij, dobljenih za belega v dveh polpotezah. Izhajajoč iz te množice lahko prejšnji postopek ponovimo in dobimo pozicije, dobljene v 3 polpotezah, 4 polpotezah itn. Kot končni rezultat algoritma B dobimo množico vseh dobljenih pozicij, pri čemer za vsako pozicijo vemo tudi število polpotez, potrebnih do zmage ob najboljši obrambi nasprotnika. Tako klasificirana množica možnih pozicij nam omogoča ne samo pravilno igro temveč tudi *optimalno igro, optimalno v tem smislu, da dobljeno pozicijo* privedemo do zmage v minimalnem številu potez.

Ker je *vseh možnih* šahovskih pozicij okrog 10^{44} (npr. Berliner 1978) in ker moramo pri izvajanju algoritma B hraniti vsaj vse dobljene pozicije, je jasno, da tudi algoritem B za celotno šahovsko igro ni izvedljiv. Možno pa je z algoritmom B rešiti končnice z manjšim številom figur. Tako je npr. v končnici s *tremi* figurami, npr. kralj in kmet proti kralju (kratko: končnica KPK), možnih manj kot 2×64^3 pozicij (ki se med seboj razlikujejo po položaju treh figur na šahovnici in po tem, kdo je na potezi). Zaradi simetrije lahko to število v končnici KPK reduciramo še za faktor 2 in tako dobimo problemski prostor z okrog 200.000 legalnimi pozicijami, ki ga je mogoče praktično obvladati z algoritmom B. Znana tovrstna rešitev te končnice je opisana v Clarke (1977). Če dodamo eno figuro, se velikost problemskega prostora poveča približno za faktor 64, kompleksnost algoritma B pa raste z večanjem števila figur še hitreje.

Kompleksnost algoritma A raste eksponencialno z globino iskanja v smeri naprej. Kompleksnost algoritma B pa raste eksponencialno s številom figur, prisotnih v končnici, ki jo *rešujemo. Jasno je, da postane zaradi eksponencialne rasti* tako kompleksnost algoritma A kot algoritma B praktično nesprejemljiva že za razmeroma majhno globino iskanja oziroma za razmeroma majhno število figur v končnici. Od teh dveh mejnih vrednosti naprej šahovske igre ne moremo več programirati s premočrtnimi eksaktnimi algoritmi, kot sta algoritma A in B, temveč z uporabo hevrističnih metod oziroma metod umetne inteligence.



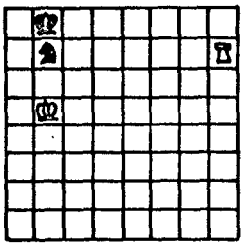
Sl. 1: Grafični prikaz delovanja algoritma B. Algoritem B generira pozicije od spodnjih nivojev navzgor

Postavljata se vprašanji:

(a) Do kakšne globine iskanja je algoritem A še praktično izvedljiv?
(b) Za kolikšno število figur v končnici je algoritem B še praktično izvedljiv?

Na vprašanje (a) dajejo odgovor izkušnje s turnirskimi šahovskimi programi. Trenutno najmočnejši in najhitrejši šahovski program, CHESS 4.7, preišče (če igra pod turnirskimi igralnimi pogoji, tj. po nekaj minut razmišljanja na potezo) vse variante do globine 8 ali 9 polpotez, odvisno od tipa pozicije, pri čemer teče na hitrem računalniku CYBER 176. Pri tem je algoritem A implementiran z znanim alfa-beta postopkom (npr. Knuth, Moore, 1975), ki je hitrejši od algoritma A, daje pa enak rezultat kot A. Izkušnje kažejo, da bi bilo tudi s precej hitrejšim računalnikom težko doseči bistveno večjo globino iskanja, npr. globino preko 10 polpotez, tj. 5 polnih potez.

Eksperiment, opisan v preostalem delu tega članka, pa daje odgovor na vprašanje b. V tem poskusu smo izračunali z algoritmom B popolno strategijo za končnico štirih figur: kralj in trdnjava proti kralju in skakaču (končnica KTKS). Rezultati tega izračuna, ki so zanimivi tudi s stališča same šahovske igre, predstavljajo pomembno orodje pri eksperimentiranju z metodami umetne inteligence, apliciranimi na to eksperimentalno okolje. Da mora biti pravilna strategija za to končnico generirana s pomočjo računalnika, potrjujejo rezultati psiholoških eksperimentov v Kopec, Niblett (1978), ki kažejo na to, da je popolnoma pravilna igra v tej na videz enostavni končnici že izven dometa šahovskih mojstrov. Celo obsežna obravnava te končnice v klasični Fineovi knjigi o šahovskih končnicah je polna netočnosti. Primer je na sliki 2.



Sl. 2: R. Fine (Basic Chess Endings, 1964), v skladu z analizami Bergerja, ocenjuje to pozicijo kot remi. Računalniško generirana strategija za končnico KTKS vsebuje za to pozicijo 15 potezno zmagovito varianto, ki se začenja z 1.Kc6.

Naš poskus kaže, skupaj z nekaterimi podobnimi znanimi rezultati, da je za končnice 4 figur še mogoče graditi popolne strategije z algoritmom B, da pa je sama programska izvedba tega algoritma zaradi časovnih omejitev zelo zahtevna. Če namreč želimo, da je čas izvajanja programa za rešitev celotne končnice v razumnih mejah, npr. nekaj ur, potem je treba najti učinkovito metodo za predstavitev pozicij v računalniku ter za organizacijo podatkovnih struktur na zunanjem pomnilniku. Npr. premočrtna, neustrezno strukturirana implementacija podatkovnih množic z datotekami z naključnim dostopom lahko poveča časovno kompleksnost programa za nekaj velikostnih razredov.

Če število figur v končnici povečamo samo za 1, postane izračun popolne strategije še neprimerno težji. Rešitev končnice kralj, trdnjava in kmet proti kralju in trdnjavi (Arlazarov, Futer, 1977) se lahko po programski zahtevnosti primerja z izdelavo prevajalnikov za programirne jezike. Končnice s 6 figurami pa so (razen v trivialnih primerih) verjetno že na meji dosegljivega pri današnjem stanju tehnologije.

ALGORITEM B

Spodnji algoritem zgradi množico pozicij, ki so dobljene za "nas" proti "njim" (ne za belega proti črnemu). V algoritmu so uporabljene naslednje kratice:

ttm - them to move, oni na potezi (nasprotnik na potezi, npr. črni na potezi)
utm - us to move, mi na potezi (npr. beli na potezi)
L - število polpotez do zmage ob optimalni igri obeh igralcev
n(L) - število pozicij z L polpotezami do zmage ob optimalni igri
poz(p) - število polpotez do zmage za pozicijo p
goal(p) - "končni predikat" za razpoznavanje pozicij, ki so po definiciji dobljene, npr. mat.

1. Inicializacija.
2. Za vse ttm pozicije označi vse nelegalne poteze.
3. Generiranje pozicij nivoja 0: vse ttm pozicije p, ki izpolnjujejo končni predikat goal(p), označi s "satisfied" in poz(p) = 0, n(0) = število pozicij s poz(p) = 0, L = 0.
4. Če je n(L) ≠ 0 in so še možne inverzne poteze nazaj iz pozicij nivoja L, nadaljuj, drugače končaj.
5. L = L+1, n(L) = 0.
6. Če je L sod, pojdi na 9, drugače nadaljuj s 7.
7. Generiraj utm nivo L iz ttm nivoja L-1: Za vsako ttm pozicijo p, označeno s "satisfied", naredi:

poišči vse predhodnike p (generiraj inverzne "us-move" poteze iz p) in za vsakega predhodnika q (na utm nivoju), če q ni označen s "satisfied", naredi:
1. označi q s "satisfied"
2. poz(q) = L
3. n(L) = n(L) + 1

8. Pojdi na 4.
9. Generiraj ttm nivo L iz utm nivoja L-1. Za vsako utm pozicijo p, označeno s "satisfied" in poz(p) = L-1 naredi:
poišči vse predhodnike p (generiraj inverzne "them - move" poteze iz p) in za vsakega naslednika q (na ttm nivoju), če q ni označeno s "satisfied", naredi:
1. označi ustrezno potezo iz q z "badmove"
2. če so vse poteze "badmove", potem
1. označi q s "satisfied"
2. poz(q) = L
3. n(L) = n(L+1)
10. Pojdi na 4.

Za izvedbo algoritma B potrebujemo naslednje osnovne podatkovne strukture:

UTMP: vektor utm pozicij dolžine N (= število vseh možnih pozicij). Za vsako pozicijo p: poz(p) (če je poz(p) večje kot $L_{max} = \max(L)$, potem p ni označeno s "satisfied"). Indeks pozicije v vektorju je koda pozicije p.

TTMP: vektor ttm pozicij dolžine N. Za vsako pozicijo p:
1 bit za oznako "satisfied"
M bitov za M možnih "them move" potez
(če je pozicija p označena s "satisfied", potem je v teh M bitih shranjeno poz(p)).

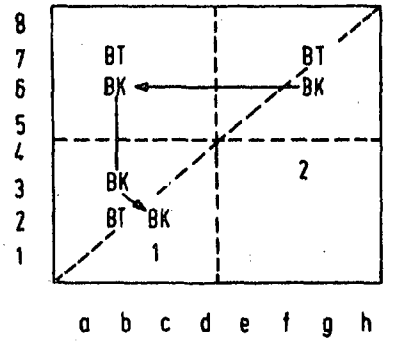
Za hitrejšo izvedbo algoritma običajno potrebujemo še:

UTMNEW: vektor bitov dolžine N. UTMNEW(koda(p)) = 1, če je bila utm pozicija p označena na predhodnem nivoju, to je poz(p) = L-1

TTMNEW: isto kot UTMNEW za ttm pozicije.

OCENITEV ŠTEVILA POZICIJ ZA KONČNICO KRALJ + TRDNJAVA : KRALJ + SKAKAČ

Ker imamo na šahovski deski (64 polj) 4 figure, je vseh možnosti 64 . 63 . 62 . 61, to je 15249024 pozicij. V tej oceni so zajete tudi pozicije, ki niso možne. Z upoštevanjem simetrije lahko v prvem zamahu zmanjšamo število pozicij na 10 . 63 . 62 . 61 = 2382660. Šahovsko desko razrežemo na 8 trikotnikov in izbrano figuro s transformacijami prevrtimo v izbrani trikotnik velikosti 10 polj, kot je to razvidno s slike 3. Pri tem je BT bela trdnjava, BK beli kralj, CK črni kralj in CS črni skakač.



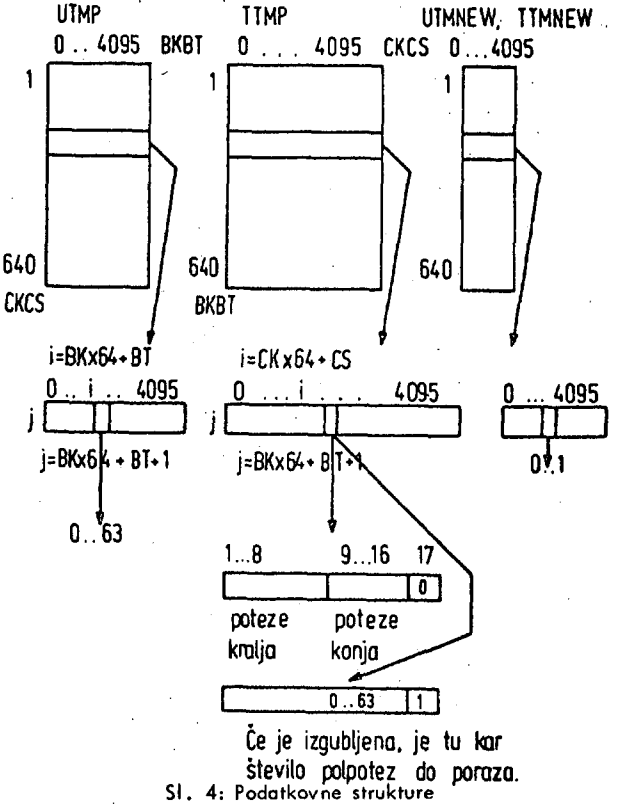
Sl. 3: Zmanjševanje števila pozicij z upoštevanjem simetrije. Preslikavanja BK (belega kralja) v trikotnik 1. BT (bela trdnjava) je zraven, da lažje vidimo, kako potekajo preslikave preko osi simetrije

Figure so urejene po prioriteti, recimo BK, BT, CK, CS. Če je beli kralj na diagonali (a1 - d4), lahko po prioriteti drugo figuro preslikamo v trikotnik 1+2. Če so vse prioritetnejše figure na diagonali, lahko zavrtimo pozicijo preko diagonale (a1 - h8) tako, da po prioriteti najpomembnejša figura, ki ni na diagonali, pristane v trikotniku 1+2. S tem lahko še bolj zmanjšamo število pozicij na eno osmino od 15249024 = 1906128. Seveda je treba razlikovati še v tem, kdo je na potezi, tako da je dejansko število pozicij dvakrat večje.

IMPLEMENTACIJA ALGORITMA B

Pri izvajanju algoritma B je treba za vsako možno pozicijo hraniti določeno množino informacij (po 6 bitov za utm pozicije oziroma po 17 bitov za ttm pozicije, kot je razvidno iz nadaljevanja). Tako je celotni potrebni pomnilni prostor pri obdelavi končnice KTKS velikosti okrog 60 milijonov bitov.

V grobem lahko časovno kompleksnost algoritma B ocenimo takole: za vsako dobljeno pozicijo moramo generirati vse možne inverzne poteze, tj. za končnico KTKS v velikostnem razredu 30 milijonov. Tolikokrat je potem potrebno tudi ažurirati podatke v datotekah skupne velikosti 60 milijonov bitov. Zato premočrtna organizacija teh podatkov, tako da bi bilo pri izvedbi algoritma B potrebnih 30 milijonov naključnih dostopov, praktično ni sprejemljiva. V naslednjih odstavkih so opisane podatkovne strukture, ki omogočajo, da ažuriramo te podatke povečini sekvenčno.



Sl. 4: Podatkovne strukture

Na sliki 4 so osnovne podatkovne strukture. To so datoteke dolžine 640 okenc, okence pa je polje 4096 elementov. Indeks okenca dobimo iz položaja dveh najbolj prioritetnih figur. Položaj figure je dan z zaporedno številko polja na katerem stoji figura, pri čemer so polja oštevilčena od 0 do 63. Kratice BK, BT, CK in CS v nadaljevanju pomenijo po vrsti položaj belega kralja, bele trdnjave, črnega kralja in črnega skakača. Npr. UTMP je datoteka, katere okence i = CK x 64 + CS + 1 ($0 \leq CK \leq 9$, v trikotniku 1). Indeks polja v okencu je določen s figurama manj prioritetne barve, za

UTMP je to BK x 64 + BT. Element polja je za UTMP kar število polpotez do zmage ali 63, če beli iz te pozicije ne more izsiliti zmage. Za TTMP je element polja v primeru, če je pozicija izgubljena (bit št. 17 je v tem primeru 1), število polpotez do poraza.

Dokler z algoritmom še nismo ugotovili, da je pozicija izgubljena, je 17. bit nič, element polja pa sestavlja še 16 bitov, prvih 8 za poteze kralja, drugih 8 za poteze konja. Če je i-ta poteza nič, vodi v za črnega neizgubljeno pozicijo. Če je 1, je ali nelegalna ali vodi v pozicijo, kjer lahko beli z optimalno igro izsili zmago. Element polja v okencu za UTMNEW in TTMNEW je kar en bit in je 1 v primeru, če je ta pozicija dobljena za belega ali izgubljena za črnega v L polpotezah za tekoči L in nič drugače (L je oznaka nivoja).

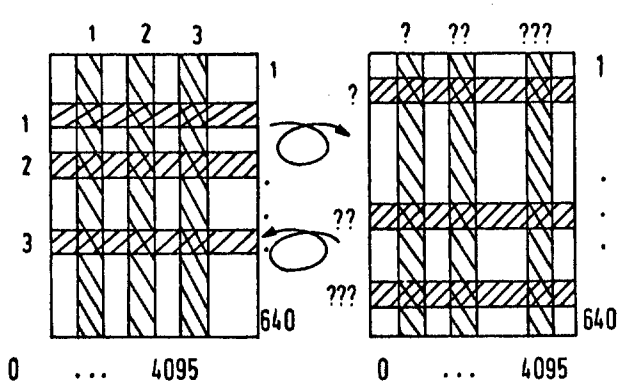
Datoteka UTMP zavzema 2 621 440 x 6 bitov = 15 728 640 bitov. Datoteka TTMP ima 2 621 440 x 17 bitov = 44 564 480 bitov, obe datoteki skupaj približno 60 milijonov bitov. Hitrost računalnika CYBER 172 je okrog milijon tristo tisoč operacij na sekundo. Če za pregled vsake poteze v vsaki poziciji porabimo čas stotih operacij, bo to približno 16 x 2 milijona x 100 operacij, to je približno ena ura računalniškega časa. V resnici je program, napisan v Pascalu, porabil nekaj več kot 3 ure.

Tako urejene podatkovne strukture imajo več dobrih lastnosti, predvsem pa:

1. V primarnem pomnilniku imamo samo eno okence vsake datoteke.
2. Če v neki poziciji delamo poteze s figurami manj prioritetne barve, so vse tako dobljene pozicije še vedno v okencu, saj nam indeks okenca določata bolj prioritetni figuri. Kolikor nam je znano, drugi avtorji ne uporabljajo take ureditve, najbrž zato, ker niso uspeli rešiti problema prehoda iz UTMP v TTMP in obratno. To omogočata pomožni datoteki UTMNEW in TTMNEW.

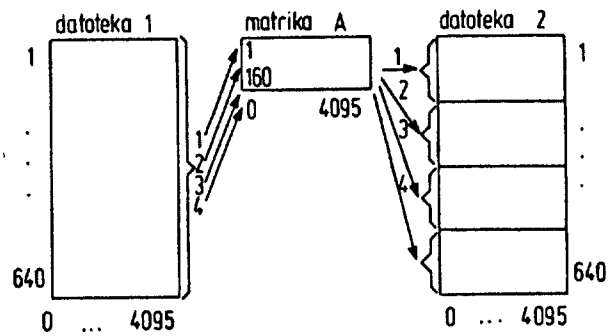
Ko naredimo vse poteze z manj prioritetno barvo in tako zgradimo nivo L, za vsako pozicijo nivoja L sproti vpišemo "1" v malo datoteko UTMNEW ali TTMNEW. Nato pozicije nivoja L prevrtimo iz ene v drugo malo datoteko.

Kot je razvidno s slike 5, se določene vrstice male datoteke preslikajo v določene stolpce druge male datoteke, določeni stolpci pa v vrstice, vendar se vrstni red znotraj stolpcev oziroma vrstic ne ohranja. Indeks vrstice je določen z bolj prioritetnima figurama iste barve, recimo CK * 64 + CS + 1, indeks stolpca pa z manj prioritetnima figurama, torej BK * 64 + BT.



Sl. 5: Vrtenje malih datotek

Vidimo, da so vse datoteke urejene tako, da jih procesiramo sekvenčno, torej bistveno hitreje, kot če bi jih morali naključno. Ostal pa nam je problem vrtenja malih datotek pri prehodu iz nivoja L na nivo L+1. Ker sta datoteki preveliki za primarni pomnilnik, si pomagamo z vmesno matriko A, kot je to razvidno s slike 6. Velikost te matrike je četrtina velikosti male datoteke in gre v celoti v primarni pomnilnik.



Sl. 6: Programska realizacija vrtenja malih datotek

Matrika bitov A je v primarnem pomnilniku, zato je procesiranje z naključnim dostopom hitro, medtem ko sta obe mali datoteki, 1 in 2, v sekundarnem pomnilniku in ju procesiramo sekvenčno. Matrika A ustreza četrtini male datoteke, zato moramo štirikrat sekvenčno "prečesati" malo datoteko 1. Pri tem z masko gledamo samo stolpce, ki vsebujejo pozicije, ki se lahko preslikajo v matriko A. Če v okencu male datoteke opazimo 1, potem zamenjamo prioriteto figur in enko z novim indeksom prepišemo v matriko A. Ko smo s prvo masko preleteli malo datoteko 1, se v matriki A nahajajo vsi stolpci datoteke 1, ki se bodo prepisali v prvo četrtino male datoteke 2. Matriko A po vrsticah prenesemo v prvo četrtino datoteke 2 in nadaljujemo postopek z masko 2.

Za uspešno realizacijo projekta je bil poudarek tudi na učinkovitosti podprogramov, ki se pogosto izvajajo. Poglejmo si primer enega izmed njih. Pri tem je npr. BKX koordinata X polja, ki ga zaseda beli kralj.

```
Function  NICSAMB:boolean;
(* funkcija je true, če črni ne daje belemu šaha s konjem *)
begin
     p:= true;
     j:= abs(BKX - CSX);
        if  j < 3  then
            if  j ≠ 0  then
                if  j = 1 then  p:= abs(BKY-CSY) ≠ 2
                          else  p:= abs (BKY-CSY) ≠ j;
     NICSAMB:= p
end; (* NICSAMB *)
```

Zato, da je gornja funkcija čimbolj učinkovita, so bile pri vrstnem redu testov upoštevane verjetnosti, da so pripadajoči pogoji izpolnjeni.

REZULTATI

Po končanem generiranju celotnega problemskega prostora za končnico KTKS smo dobili naslednje statistične rezultate. Vseh legalnih pozicij za belega na potezi je 1 347 906, za črnega pa 1 567 222. Narejena je bila tudi statistika po številu polpotez do zmage belega ob optimalni igri obeh nasprotnikov.

Beli na potezi		Črni na potezi	
št. polpotez	št. pozicij	št. polpotez	št. pozicij
1	378518	0	33156
3	95450	2	54539
5	46269	4	14839
7	30749	6	15242
9	20055	8	10563
11	15071	10	9294
13	11740	12	8493
15	9495	14	7721
17	8562	16	7664
19	7415	18	7917
21	6308	20	7014
23	5356	22	6085
25	4133	24	5129
27	3356	26	3776
29	2290	28	2921
31	1621	30	2025
33	1333	32	1580
35	1046	34	1390
37	727	36	1006
39	556	38	707
41	458	40	544
43	373	42	396
45	302	44	286
47	178	46	178
49	111	48	135
51	18	50	41
53	2	52	13
		54	1

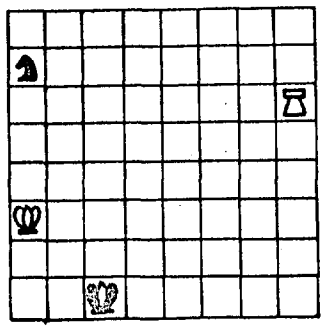
Skupaj	
Beli	Črni
651 492	221 191

Statistika je presenetljiva. Za bele z večjim številom polpotez do zmage število dobljenih pozicij lepo pada, zato pa ima statistika črnih dva skoka. Za 2 polpotezi do poraza je število pozicij večje kot za po definiciji izgubljene pozicije. Podoben skok opazimo pri 4-6 in 16-18. Druga presenetljiva ugotovitev je, da je pri majhnem številu polpotez do zmage belega število belih pozicij z L polpotezami večje kot število črnih z L-1 polpotezami, kar je razumljivo, vendar se razmerje spremeni v korist črnih od L = 18 - 19 naprej in ostane tako do konca razen za L = 44 - 45, kjer je belih več kot črnih, in za L = 46-47, kjer je število pozicij za bele in črne enako.

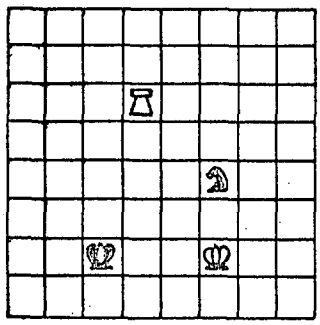
Pri statistiki so rezultati dobljeni z upoštevanjem skrčitve števila pozicij, zato je resnično število pozicij osemkrat večje.

Vidimo, da obstaja ena sama pozicija, kjer se črni na potezi lahko upira še 27 potez (= 54 polpotez). Poglejmo si optimalno odigrano partijo iz te pozicije. Pri tem se v oklepajih pojavljajo ekvivalentne variante (poteze, ki vodijo do zmage belega v istem številu polpotez). Pravilna igra je tu tako zahtevna, da že sodi v problemski šah. Poskusi Kopeca in Nibletta (1978) so pokazali, da šahovski mojstri praktično nimajo nobenih možnosti, da bi teoretično dobljeno izhodiščno pozicijo privedli do zmage pod turnirskimi igralnimi pogoji.

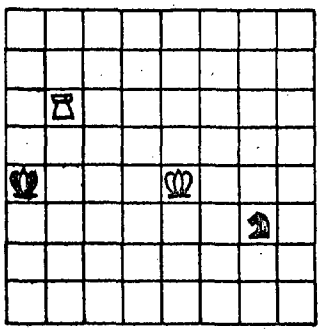
Začetni položaj je na spodnjem diagramu.



1. Sb5
Obe črni figuri sta dokaj blizu središča, pozicija izgleda prej remi kot poraz za črnega.
2. Kb4 Sd4, 3. Kc3 Se2, 4. Kd3 Sf4,
5. Ke3 Sg2!, 6. Kf2 Sf4,
Dvoboj belega kralja in črnega konja je končan, črni konj ni uspel priti pod zaščito svojega kralja.
7. Td6 Kc2



Kralj skuša priti v aktivnejši položaj, toda bela trdnjava zdaj ločuje obe črni figuri.
8. Ke3 Sg2, 9. Kf3 Sh4, 10. Ke2! Sf5,
11. Tc6 Kb3, 12. Kd3 Kb4
Po manevriranju ostaneta črni figuri še naprej ločeni, vendar sta še blizu središča.
13. Te6 Kb3!, 14. Tb6 Ka4, 15. Ke4 Sg3
(15. Kc4 ..)



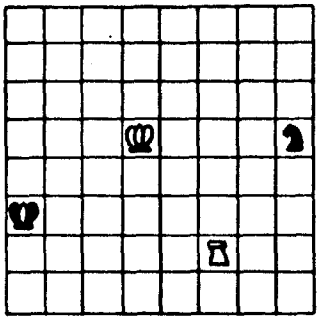
Beli sili črnega v čedalje slabše pozicije, črni figuri sta še bolj ločeni in že odrinjeni iz središča.
16. Kd5 Se2 (16. Kd4 ..) (16. .. Ka3),
17. Kc4 Ka3, 18. Tb1 Sf4 (18. .. Tb3 ..),
19. Tg1 Se6 (19. Te1 ..), 20. Tg6 Sf4,
21. Tf6 Sh5 (21. Tb3 ..), 22. Tf2 Sg7 (22. Tf3..),
23. Kd5 Sh5 (23. .. Kb3, b4, a4).

Če bi poskušali s kakšnim preprostim algoritmom tipa A za preiskovanje v smeri naprej optimalno odigrati takšno partijo, bi za to potrebovali približno 1000 krat toliko časa, kolikor je stara Zemlja.

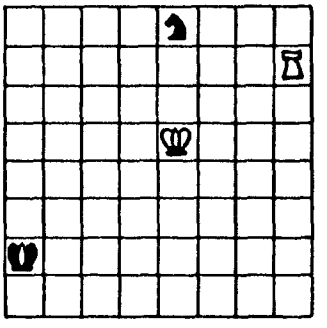
V eni potezi dobimo do 352 novih pozicij iz ene stare. Če namesto 352 vzamemo faktor 10 in za generacijo ene pozicije porabimo 10^{-6} sekunde, bo to

$$\frac{10^{27} \cdot 10^{-6}\,s}{10^{18}\,s} = 1000$$

Pri tem je 10^{18} s starost Zemlje.



Obe beli figuri se odpravljata na lov na črnega konja.
24. Ke5 Sg7,
25. Th2 Se8 (25. Tf7, f8 ..) (25. .. Ka4, b3, b4)
26. Th7 Ka2 (26. .. Kb2, b3, b4, a4)



Črni konj je očitno izgubljen.
27. Te7 Sg7 (27. .. Ka1, a3, b1, b2, b3, Sc7, d6, f6)
28. Tg7:

LITERATURA:

1. Arlazarov, V.L., Futer, A.V., Computer analysis of a rook end-game, v Machine Intelligence 9 (Ed. D.Michie), v tisku.
2. Berliner, H.J., Computer Chess, Nature, Vol. 274, 24. Aug. 1978, 745-748.
3. Clarke, M.R.B., A quantitative study of king and against king, v Advances in Computer Chess 1, 108-118 (Ed. M.R.B. Clarke), Edinburgh: University Press, 1977.
4. Knuth, D.E., Moore, R.W., An analysis of alpha-beta pruning, Artificial Intelligence, 6, 293-326, 1975.
5. Kopec, D., Niblett, T., How difficult is the king and rook vs. king and knight ending?, v Advances in Computer Chess 2, (ed. M.R.B. Clarke), v tisku.

some views on redundancy in natural language processing

m.boot

UDK 681.323:801

Institute of Apllied and Computational Linguistics,
State University Utrecht, 3581 NC Utrecht, Holland.

The paper focuses on the function of redundancy in language communication. In pure as well as applied linguistics redundancy is considered to be superfluous information. In the paper, on the other hand, it will be argued that even theoretically redundancy cannot be derived from natural language. On the level of language communication redundancy turns out to be the error controlled regulation of the system. Language needs such a mechanism because it is necessarrily ambiguous on every level of communication. Out of these considerations a model for automated LP was developed, that proves to be more successfull than other systems developed in the past for automated LP.

NEKAJ POGLEDOV NA REDUNDANTNOST PRI OBRAVNAVANJU NARAVNEGA JEZIKA - V članku je obravnavana redundantnost pri jezikovnem komuniciranju. V teoretični in aplikativni lingvistiki je redundantnost predstavljena kot odvečna informacija. V članku pa dokazujemo, da niti teoretične redundantnosti ne moremo izpeljati iz naravnega jezika. Izkaže se, da je redundantnost na nivoju jezikovnega komuniciranja z napakami nadzorovano krmiljenje sistema. Jezik potrebuje tak mehanizem, ker je nujno dvoumen na vsakem nivoju komuniciranja. Na takih (teoretičnih) osnovah smo zgradili model za avtomatično obravnavanje naravnega jezika, ki je uspešnejši kot drugi sistemi.

Redundancy in Linguistics

The description of natural languages is to be found in the science of linguistics, the theoretical and the applied branch of linguistics. As far as theoretical linguistics is concerned the function of redundancy can be defined as a tool of facilitation the writing of grammar. Because:

"The redundancy rules, both phonological and syntactic, state general properties of all lexical entries, and therefore make it unnecessary to provide feature specification in lexical entries where these are not idiosyncratic." (Chomsky, 1:168)

The syntactic and phonological redundancy rules
".... both play the role of eliminating redundant specifications from the lexicon." (Chomsky, 1:168)

In other words, these redundancy rules:
".... need no specific statement in the grammar"
(Chomsky, 1:168)

Thus, in pure linguistics the word redundancy is used in the colloquial sense of this word: it means superfluous information. Therefore, it needs no treatment whatsoever in the theory. This treatment of redundancy can be declared form the fact that pure linguistics aims not to describe language as a communication system. Moreover, language is considered to consist of an infinite collection of sentences. Theoretical linguistics merely describes the structure of the sentence and language is reduced to sentence. Applied linguistics to a great extend is engaged in the study of language learning, particularly learning a second language. In this environment redundancy is treated as a tool to measure the command the learner already has of the language that should be learned (Spolsky, 1969). Spolsky worked out a noise test consisting of a number of sentences to which noise had been added. This test sorted out those students who required no further English instruction. Although a substantial use of the function of redundancy inherent in language was made, no further theoretical conclusions were drawn about the function of redundancy in the system of language itself:

"Redundancy may seem wasteful of effort, but it is in fact of great use, for it reduces the possibility of error and permits communication where there is inference in the communicating channel." (Spolsky, 1969)

This statement gives evidence of the same conception of redundancy: superfluous but convenient information. Therefore, one should come to the conclusion that linguistics does not provide the proper framework to describe the function of redundancy in language processing.

Redundancy in Language Communication

This treatment of redundancy can be declared from the fact that pure linguistics aims not to describe language as a communication system. It describes the structure of language. Moreover, language is considered to consist of an infinite collection of sentences. Describing the structure of sentences means describing states. Language communication, however, cannot be thought of as a series of states, it can only be defined as a process. So we need not only a static description of the possible states, we also need a vehiculum to define the mechanism of the process itself.

Language communication is communication through signs. Signs are composed by codes. Let us take the letters of the English alphabet for such a code. From an example we can observe, how the process of building signs with those letters works. Imagine that an English woman is asked to write down her Christian name letter by letter, so that it is possible to guess it with the help of the available letters. When she writes the letter P, the uncertainty has been reduced by a large amount, for all names that begin with another letter are excluded now. When she now adds the letter A, uncertainty is further reduced as names like Pearl, Prudence and Priscilla are ruled out. Adding the letter M makes it easy to find out the name Pamela.
This game and all similar language spelling games make in fact use of the redundancy in the encoding system. We can observe the working of redundancy in all letters. As far as the first letter P is concerned we know namely much

more than only the fact that names beginning with other letters can be excluded. We also know that after this P no Q or W etc. can follow. This is because in English only thirteen letters follow the letter P. In other words, our knowledge of the linguistic rules of word formation in English permitted us to eliminate 50% of the possibilities for the second letter of the word. The same holds mutatis mutandis for all letters in the word, till we reach the point that we are sure of the letters to come. This, however, is only possible when the linguistic rules of word formation in English are known to us, i.e. when these rules are redundant for us.

So far we payed attention to the redundancy involved in the spelling system of natural languages and came to the conclusion that redundancy is a parallel to the rules of combinabillity of letters. Those rules are described in the syntax of a language and linguistics provides a pretty complete image of those syntactic rules being almost totally devoted to syntactic studies on this level. We should now ask whether this stratum is the only one that uses redundancy in this way, i.e. for building language units of a higher level. For, the syntactic rules build evidently the transition of the unit "letter" to the unit "word". Communication can only take place, if and only if those rules are totally redundant to the communicators, i.e. if and only if they are able to recognise the words.
If we take the unit normally called "sentence" for the next higher level, we can make the following observation from daily conversation. If someone is arguing or describing his feelings and opinions to someone familiar with the topic, the listener will in his enthousiasm interrupt the communicator often before he made a full sentence because the message is understood already. Let us assume that there was no misunderstanding then this can only be declared from the fact that there was no need for a full sentence to fill in the missing information. This on the other hand can only be declared by the fact that this missing information was redundant. This redundancy must have a twofold source. Firstly from the knowledge of the syntactic properties of English sentences the correct sentence is found by the listener before it is formulated with the help of the part that is formulated already. From this correct sentence the places where the missing information fits in are derived. Consequently the type of information is derived. Then the second source of information is used namely the information about the topic itself.
In other words, in the stratum of sentence structure we again found redundancy as the important feature to allow for communication. This stratum is usually referred to with the notion syntax in linguistics. From a semiotical point of view, however, there is no need to discriminate between the formation rules of word formation and the formation rules of sentence building. It is always the same type of process: out of a so called material basis, in the first case letters, in the second case words, a more complicated sign is built. In communication these higher signs can only be used to communicate with if and only if the formation rules are totally mastered, i.e. if and only if they are redundant.

From these observations we can derive the conclusions that redundancy is vital to the process of communication and that it is by no means superfluous. The way in which redundancy works can be observed on all types of communication through signs. We could refer to the "syntax" of texts like e.g. the syllogism. Here again the structure, i.e. the syntax is "fixed". This redundant structure is used to communicate a very specific kind of "mental content". The same can be stated for texts like sonnets where we can observe such communicative possibilities like originality, humour, with one word the communication of esthetic information. The same again can be observed in the communication of "how to behave in situations". The syntax of this type of communication is given by the broad environment of cultural prejudice. This building up of signs of a higher, more abstract type is called "superiorisation" in semiotics. The vehiculum for this superiorisation is redundancy. Superiorisation, however, is not the only function of redundancy. From this function a second one can namely be derived. Redundancy is also of vital importance for solving ambiguity. Theoretically this function can immediately be derived from the superiorisation function. The mechanism of superiorisation is necessary because of the disproportion between the material repertoire of signs with which we have to operate and the repertoire of contents or mental states we wish to communicate about. The material repertoire is always restricted and finite, the contents we wish to communicate about are principally infinite. Therefore, the material repertoire must built signs that can refer to different concepts (contents). In other words the material repertoire must produce at least to some extent ambiguous signs. Let us take as an example ambiguity in the sign "word". If we take a dictionary of English and look for the word "play" we find 20 to 30 meanings to it. The different meanings are always demonstreted by a full sentence, "context", or by a reference to a specific situation. In other words by using another sign. This sign is always of a higher type than the sign that is ambiguated by it.
As we all know from our experience with the use of dictionaries this procedure of disambiguation works fairly well. In the terms we are using here we can describe this procedure as reversing the mechanism of superiorisation. This mechanism of superiorisation as was stated before is built on redundancy. We found the theoretical basis for it, as well as how it works out. With superiorisation and redundancy we found a function of language processing. If we want to simulate language processing with a processing device, say a computer, we should build this on the implementation of the superiorisation mechanism. Because computers are automata we can suppose that they are able to process that part of communication that proceeds automatically. All that is redundant is processed automatically. Syntactic rules are redundant. So the simulation of communication by signs on computers should be based on the implementation of syntactic rules. These rules are, however, not in the right format for processing, because they are not used for the process of disambiguation, but only to signal possible ambiguities. We already observed that linguistics for example only describes states and that linguistics remains within one sign, say the sentence structure or the word formation.

Simulating the Disambiguation Process

From all these observations we took the conclusion that the simulation of the normal disambiguation process should be based on a hierarchic structure of redundant information. In other words we formulated the hypothesis that disambiguation takes place by the use of the syntactic information of a higher sign. Because we are operating with nowadays computers we made the restriction that it is only allowed to use the next higher sign in the hierarchy. So homography for example should be solved only with the help of the information of the syntax of the sentence described by linguistics. For homography is a problem of word forms, the next higher sign as regards word forms is the sentence.
Homography reveals from the following sentences:

1: The professor is a mean old man.
2: I mean to go downtown tomorrow.

The two word forms "mean" refer to different linguistic classes: a noun (1) and a verb (2). This type of ambiguity could be solved on the basis and with the help of the theoretical assumptions described here. We wrote several papers reporting the results (s. in Boot, 1978). The implications of the fact that a system based on the principles outlined here could be implemented on a CDC computer, are far reaching. From systems analysis viewpoint redundancy turns out to be the error controlled regulation of the communication system in language. Because language always produces ambiguous signs on every level of production there must be an automatic regulating device on disambiguation. This device is to be found in the redundancy involved in the hierarchy of sign production. This consideration can be used on the

level of theory of perception. The algorithms can be conceived of as images of the supposed routines Miller/Johnson-Laird are speaking about:
"In order to account for people's capacity to understand sentences they hear, it is helpful to suppose that routines (or schematized information on which routines can be based) corresponding to words (or morphemes) are stored in their memories."
(Miller/Johnson-Laird: 146)

Further Applications

On the other hand the computer programs can be used as a device to measure redundancy of texts. Therefore, they can be used as a means to describe the degree of difficulty of a text. Apart from that the programs produce word classes. Those word classes are used as an important parameter in authorship studies as can be concluded from the works of Wickman and Bailey. Uptill now word classes were assigned by hand. This turned out to be a boring and very tiresome operation which produced many mistakes. (e.g. Uit den Boogaart)
The programs can also be used in automated content analysis where a decisive contribution to the problem of disambiguation and the location of statements could be given. (Boot, 2, 2)

References

1. Bailey, R.: "Authorship Attribution in a Forensic Setting." Preprints of: Computers in literary and linguistic Research. Birmingham, 1978.
2. Boot, M.: Homographie. Ein Beitrag zur automatischen Wortklassenzuweisung in der Computerlinguistik. Utrecht, 1978.
   :"Ambiguity and automated content analysis." in: MD, 1978, nr. 1, pp 117-137.
   : "An experimental Design for automated syntactic encoding of Natural Language Texts." in: ALLC Bulletin, Vol.5, 1977, pp. 237-248.
3. Chomsky, N.: Aspects of the theory of Syntax. MIT Press, 1965.
4. Eco, U.: A Theory of Semiotics. Indiana University Press, 1975.
5. Miller, G. & Johnson-Laird: Language and Perception. Harvard University Press, 1976.
6. Spolsky, B.: "Reduced redundancy as a language testing tool." in: G. Perren and J. Trim: Applications of Linguistics. Cambridge, 1971.
7. Uit den Boogaart, P.: Woordfrekwenties van het Nederlands. Utrecht, 1975.
8. Wickmann, D.: "A Discriminant Analysis of Text Homogeneity." in: Preprints of Computers in Literary and Linguistic Research. Birmingham, 1978.

metodika načrtovanja digitalnih računalniških sistemov

p.kolbezen

UDK 681.32.001.57

Institut "Jožef Stefan" Ljubljana

V članku so nanizani splošni problemi načrtovanja današnjih in bodočih generacij digitalnih računalnikov. Obravnavane so strukture avtomatiziranih postopkov načrtovanja in podane smernice za takšen razvoj digitalnih računalniških sistemov.

PROCEDURE FOR DESIGN PRESENT-DAY AND FOLLOWING GENERATION COMPUTERS. General problems of designing present-day and future generation electronic computers and structure of an automated operation line for digital systems design are considered.

1. UVOD

Generacije računalnikov se med seboj razlikujejo med drugim predvsem v tehnologiji. Elektronske cevi so zamenjali diskretni polvodniški elementi, te elemente pa integrirana vezja, ki prehajajo v vse večje integracije (LSI). Integracija vezij predstavlja tehnično bazo novih generacij digitalnih sistemov - računalnikov. Prehod k novim generacijam pa je pogojen tudi z novimi računalniškimi strukturami in organizacijo računalniških procesov. Med razvojem prvih treh generacij so se pojavile važne spremembe v strukturi in organizaciji računalniških procesov. Za prvo generacijo je značilno, da se je v vsakem trenutku izvajala ena sama instrukcija tekočega, enega samega programa. Pri izvrševanju instrukcije je lahko zaporedoma sodelovalo več naprav. Posledica takšne organizacije je bila, da je večji del računalniške opreme navadno čakala na rezultate delovanja posameznih naprav. Hitrejše odvijanje programa je bilo mogoče doseči le s hitrejšimi elementi. Prehod na drugo računalniško generacijo je bil povezan z novimi strukturami, ki so omogočale večjo izkoriščenost posameznih naprav s pomočjo instočasnega delovanja le-teh. Pojavil se je režim paketnega obravnavanja programov. Pri tem se je povečal pomen uporabe matematičnih sredstev, ki so vplivala na organizacijo računalniškega procesa. Na široko so se pričeli uporabljati sistemi avtomatiziranega programiranja. Pojavila se je specializacija računalnikov: manjši računalniki v inženirske namene, računalniki za obdelavo podatkov, procesni računalniki ter visoko zmogljivi računalniki za znanstvene raziskave.

Tudi prehod k tretji generaciji je povezan z razvojem in vpeljavo novih idej v strukturi in organizaciji računalniških sistemov. Pri teh dominira ideja o sistemu s časovnim dodeljevanjem (time-sharing), ki omogoča "istočasno" obdelovanje večjega števila programov, ter ideja o vhodno-izhodnih kontrolnih sistemih, ki uporabljajo večkratne vhodno-izhodne kanale. Pojavil se je sistem programskih prekinitev (interrupt system). Pomembna je tudi ideja o sistemu večjih računalnikov.

V zvezi s prehodom na četrto generacijo računalnikov so se pojavile nove tehnične in matematične ideje. Te slone na multiprocesorski strukturi in paralelnih organizacijah računalnika, ki omogočajo t.im. paralelne in pipe-line računalnike. Slednji so osnovani na povsem novem principu obravnavanja informacij (tipa pipe-line). Konstruiranje takšnih računalnikov

pa zahtevnejše matematično orodje. Pomembno je, da se dandanes neglede na vrsto vedno novih idej izdelujejo obsežni projekti za avtomatizacijo načrtovanja, pri čemer se upošteva nova tehnološka baza za izpopolnjevanje starih metod načrtovanja digitalnih sistemov. Pri tem je potrebno omeniti, da je za načrtovanje novih digitalnih sistemov potrebno ogromno sredstev, veliko število visoko kvalificiranih specialistov in mnogo časa, ki lahko traja tudi nekaj let. Posledica tega je, da je nastal prepad med razvojem tehnologije in razvojem takšnih računalniških struktur, ki naj bi v maksimalni možni meri izkoriščale možnosti najsodobnejših tehnologij.
Premostitev omenjenega prepada omogočajo nove metode projektiranja digitalnih sistemov. Nova metodika mora predvsem zmanjšati čas in ceno načrtovanja. Zato mora imeti naslednji dve lastnosti:

- v čim večji meri mora na vseh nivojih načrtovanja uporabljati avtomatizacijo postopkov
- mora biti neodvisna od strukture in organizacije računalniškega procesa načrtovanega digitalnega sistema.

V nadaljnem bomo obravnavali nekatera vprašanja, ki se pojavljajo v zvezi z razvojem takšne metodologije. Osnovni princip obstaja v naslednjem: Računalniški sistem, skupaj z vsemi programi in podatki, ki omogočajo matematični aparat, se smatra kot en sam kompleks, ki realizira množico algoritmov. Obdelava teh algoritmov in opisovanja le-teh s pomočjo specializiranih visokih algoritmičnih jezikov predstavlja prvi korak v načrtovanju računalniškega sistema. Tako opisani algoritmi se nato postopoma transformirajo in kot rezultat teh preoblikovanj algoritmov dobimo dvoje vrst rezultata: sheme sistema v obliki tehnične dokumentacije, ki omogočajo fizično realizacijo sistema, ali programe za avtomatično realizacijo ustreznih modulov ter programe potrebne standardne programske opreme. Vsa začetna in vmesna informacija se hrani v avtomatiziranem sistemu projektiranja s pomočjo katerega se realizira večji del transformacij omenjenih informacij.

Jasno je, da se v praksi s takšno metodiko projektiranja zlasti večjih sistemov pojavi vrsta teoretičnih in praktičnih problemov. To problematiko se ne more reševati hkrati. Logični razvoj digitalnega sistema in metode načrtovanja vedejo k taki metodiki. Med drugim bomo nadalje osvetlili tudi to dejstvo.

Najpreje si bomo ogledali predmet projektiranja - strukturo sodobnih računalnikov in njihove standardne programske oziroma programirane materialne opreme. Reševanje omenjene problematike z opisano metodologijo je tem učinkovitejše in bolj upravičljivo, čim večji je digitalni (zlasti računalniški) sistem. Obravnavali bomo osnovne probleme metodike projektiranja in njene realizacije. Ker je na tako ozko odmerjenem prostoru težko zajeti vso problematiko, bomo obravnavali le važnejšo, nekatero pa le bežno ali se je sploh ne bomo dotaknili.

2. STRUKTURA RAČUNALNIKA IN NJEGOVA STANDARDNA OPREMA

2.1. Stanje in sredstva standardne programirane materialne in programske opreme

Uporabnike računalnikov lahko razdelimo na tri grupe:

- uporabniki, ki nimajo dovolj programerske predizobrazbe in stremijo za tem, da dosežejo rešitev z najmanjšim naporom v čim krajšem času,
- specialisti, ki so kvalificirani programerji. Ti uporabniki stremijo ne le za tem, da rešijo problem, temveč tudi za tem, da ga rešijo tako, da je program učinkovit tudi pri pogostejši rabi,
- specialisti za standardno in specialno proggramsko oz. programirano materialno opremo ali sistemski programerji.

Jasno, da je navedena klasifikacija v precejšnji meri pogojna, vendar daje možnost za nekakšno klasifikacijo standardne opreme sodobnih in bodočih računalnikov: takšno klasifikacijo, ki je koristna s stališča analize struktur in projektiranja le-teh. Glede na omenjeno klasifikacijo poznamo tri različne standardne opreme:

a) Sredstva za reševanje nalog. Ta sredstva so orientirana na prvo grupo uporabnikov. Za uporabo teh sredstev ni potrebna posebna programerska kvalifikacija. Semkaj spadajo predvsem posebni programski sistemi, ki so usmerjeni v reševanje nekega razreda nalog iz določenega področja. Pomembna karakteristika takšnega sistema je sposobnost vpisovanja in izpisovanja podatkov v obliki, ki je v navadi na določenem področju in je tudi najlažje uporabljiva. Takšen sistem je lahko manjši paket programov za izračunavanje različnih tipov inženirskih konstrukcij ali sistemski programi, ki izkoriščajo najrazličnejše terminale, med drugimi tudi prikazovalne naprave z zaslonom

in pisalnike krivulj za vpisovanje in izpisovanje grafičnih informacij, ter reševanje nalog s pomočjo časovnega dodeljevanja.

V ta razred sredstev prištevamo tudi dokumentacijske sisteme, arhive, banko podatkov, problemsko usmerjene jezike in programske paktete aplikativne narave, kot so npr. programi za obračun plač ipd.

V zadnjih letih postajajo pomembnejši reševalci sistemskih problemov. Pri teh problemih se vhodni podatki izoblikujejo v pogoje, ki povezujejo (določene) vhodne podatke z rezultati. Reševalec problema mora sam določiti zaporedja operacij, ki jih zahteva rešitev problema oz. sestaviti ustrezen program. Pri nekaterih računalniških organizacijah govorimo v takšnih primerih o mikro programiranju.

Med sredstva, ki so potrebna za reševanje problemov načrtovanja, moramo prištevati tudi univerzalne algoritmične jezike, kot sta ALGOL in FORTRAN, če upoštevamo omejeno uporabo le-teh za manjše programe, ki imajo preprosto strukturo. Če pa hočemo izkoristiti vse možnosti teh jezikov, rabimo njihove izpeljanke, ki so specializirane za posamezna področja aplikacij.

Programi, kot omenjena (matematična) sredstva, se običajno vstavljajo v nek standardni (univerzalni) operacijski sistem kot specializiran dodatek.

b) Sredstva programiranja in načrtovanja računalniškega procesa. Ta so v glavnem namenjena uporabnikom, ki imajo posebno programersko izobrazbo. Med ta sredstva prištevamo predvsem programske jezike in njihove programske sisteme. Dandanes imamo že zelo širok spekter programskih jezikov, med katerimi sta posebej učinkovita univerzalna jezika tipa ALGOL-68 ali PL/1 in specializirane jezike, ki so usmerjeni predvsem v izračunavanje, obratovanje ekonomskih informacij, vzorčevanje zapletenih sistemov, obravnavanje podatkovnih struktur, transformiranje formul, obravnavanje oz. rezpoznavanje likov (pattern recognition). Avtokodi in strojno usmerjeni jeziki dopuščajo, da je možen, grobo rečeno, dostop do vseh sredstev, ki jih ima računalnik. Pospešeno se razvijajo sistemi odlaganja in generiranja programov (posebno še v režimu časovnega dodeljevanja). Raziskuje se gradnja sistemov za preverjanje pravilnosti in sinteza programov.

Programi sodobnih računalnikov se pogosto odvijajo s podporo posebnih operacijskih sistemov, ki so prirejeni za multiprogramiranje in multiprocesiranje. Zato pridobivajo na pomenu sredstva planiranja in programiranja za krmiljenje zapletenih računalniških procesov. Včasih se ta sredstva realizirajo s pomočjo visokih jezikov, drugič pa s posebno obdelavo operacijskega sistema in strojno orientiranega sistemskega programiranja. Omembe vredno pa je, da ta sredstva dandanes niso dovolj razvita in je visoko učinkovitost računalniškega sistema v večini primerov pripisovati porazdelitvi časa in nakopičenju sredstev za "sočasno" izvrševanje več nepovezanih nalog, in ne zaradi smotrne organizacije znotraj posameznih programov ali sistema med seboj povezanih programov.

c) Sredstva za razvoj standardne programske oziroma programirane materialne opreme. Do nedavnega so izdelovalci standardne opreme v najboljšem slučaju uporabljali strojno orientirana sredstva kot so strojni in zbirni jeziki. Takšna sistemska oprema, sestavljena iz sto tisoč instrukcij, se ne more uspešno razvijati brez ustrezne avtomatizacije. To dejstvo je stimuliralo intenzivne raziskave na tem področju. Največje dosežke zasledimo v avtomatizaciji načrtovanja prevajalnikov. Sodobni univerzalni programski sistemi, ki predstavljajo osnovno sredstvo za realizacijo programskih jezikov, so znatno bolj učinkoviti od prvotnih, sintaktično krmiljenih prevajalnikov in postajajo vse bolj razširjeni. Ta sredstva so namenjena predvsem reševanju sintaktičnih problemov, dočim za programiranje semantičnih podprogramov slej ko prej ne bo dovolj razvitih sredstev. Tako ostaja še dokaj nerešen problem, kako zgraditi dober specializiran jezik, ki bi omogočal efektivno realizacijo sistemskih programov.

Večjo vlogo v razvoju standardne opreme predstavljajo sredstva postopnega razširjanja sistema. Ta sredstva omogočajo poleg gradnje osnovnih sistemov (programskih, osnovnih operacijskih sistemov) tudi nadgradnjo sistemov z izkoriščanjem že razpoložljivih sredstev. Sredstva razširjanja ne dajejo samo možnost učinkovitejšega izdelovanja standardne opreme, ampak omogočajo tudi prožno menjavanje le-te v odvisnosti od pogojev delovanja standardnega programskega sistema in njegove specializacije.

Tukaj predstavlja najvažnejšo vlogo tehnika makroprocesorjev, ki se lahko uporablja tako

za razširitve programskega kot tudi operacijskega sistema.

Sredstva za načrtovanje standardne programske opreme sama sestavljajo poseben sistem standardne programske opreme. Po takšnem sistemu, ki je več ali manj še v povoju, se kažejo vse večje potrebe. V prihodnosti se bodo morala ta sredstva slej ko prej zliti z avtomatiziranimi sistemi projektiranja digitalnih sistemov (računalnikov) in bodo sestavljala tehnično bazo projektiranja novih računalnikov in njihove programske opreme.

Zaključimo lahko, da bodo programski jeziki za uporabnika računalniškega sistema vse bolj neposredno uporabljivi in ne le s pomočjo programerjev. Po drugi strani pa bodo programerji vedno bolj prevzemali vlogo sistemskih programerjev. Njihova vloga ne bo v reševanju posameznih nalog, ampak v ustvarjanju programskih sredstev za reševanje vrste nalog iz določenega področja.

2.2. Arhitektura računalnika in njegova standardna programska oprema

Sodobni računalniški sistem predstavlja skupek naprav, ki imajo od uporabnika najprej določeno organizacijo in vnaprej določeno nalogo. Organizacija naprav je podobna nekakšni hierarhiji. Na nižji stopnji se nahajajo osnovne naprave: vhodno-izhodne, povezovalne linije, pomnilniške naprave različnih tipov in naprave za procesiranje (v ožjem smislu). Te naprave več ali manj upravljajo same sebe. Združujejo se v grupe s skupnim upravljanjem. Vsaka takšna grupa se lahko obravnava kot ločena naprava na višjem nivoju, ki ima določene funkcionalne zmogljivosti in je sposobna komunicirati z drugimi napravami. Te so potrebne za specifične obdelave informacij oz. režime obratovanja. Naslednji nivoji upravljanja so vnaprej določeni za organizacijo medsebojnega delovanja naprav na nižjem nivoju in delovanja z napravami višjega nivoja v procesu izvrševanja nalog. Le-te določajo neposredno uporabniki ali izhajajo iz sistema samega. Naprave na višjem nivoju so procesorji, ki so krmiljeni s programi uporabnikov ali s programi, ki upravljajo računalniški proces. Množica algoritmov, ki omogočajo sprejemanje in izvrševanje (od zunaj) zastavljenih nalog, krmiljenje procesa izvrševanja nalog in posredovanje rezultatov naročnikom o izvršenih nalogah, predstavlja univerzalno ali standardno programsko računalniško opremo. Del teh algoritmov se realizira z materialno opremo, ki jo imenujemo materialno programsko opremo (firmware), in sestavlja notranjo standardno programsko opremo, drugi del pa predstavljajo standardni programi t.j. zunanjo standardno programsko opremo (software).

Tradicionalen pristop k projektiranju računalniškega sistema je bil razdeljen na načrtovanje materialne in programske opreme. Pri tem so se struktura opreme, notranji strojni jezik in struktura podatkov, ki so potrebni za izdelavo programske opreme, izkazali v mnogih ozirih neprilagodljivi strukturi standardne opreme. Posledica tega kaže slab časovni izkoristek oz. zmanjšano učinkovitost sistema. Zato je pri projektiranju novih računalniških sistemov potrebno obravnavati hkrati materialno in programsko standardno opremo kot en sam kompleks, ki omogoča realizacijo izbranega računalniškega procesa.

V osnovi lahko celotno standardno računalniško opremo razdelimo takole:

- operacijski sistem, ki upravlja komuniciranje podatkov, vstopanje podatkov v sistem, formiranje programov, porazdelitev nakopičenih sredstev t.j. programov in podatkov, formiranja porazdelitve oz. komuniciranja z avtonomnimi napravami na nižjih nivojih itd;
- informacijski sistem, ki omogoča upravljanje podatkov, vstopanje in izstopanje podatkov v oz. iz sistema, porazdelitev in premeščanje podatkov v pomnilnikih kot tudi nekatere oblike obravnavanja podatkov, ki dopuščajo večjo ekonomičnost informacijskega sistema;
- programski sistem, ki omogoča povezavo vhodnih programskih jezikov programskega sistema in notranjih jezikov. V ta sistem so vključeni prevajalniki, interpreterji, makroprocesorji in sistemi za pripravo in odlaganje programov v realnem času.

Izbira računalniške arhitekture in standardne opreme, kakor tudi razdelitev le-te na materialno programsko opremo in programsko opremo zavisi od režima uporabe in splošne usmerjenosti načrtovanega sistema. Osnovne karakteristike sistema, ki vplivajo na omenjeni izbor, so:

- hitrost reševanja standardnih operacij, ki se pri izkoriščanju računalnika najpogosteje uporabljajo;
- obremenitev osnovnih naprav pri multiprogramiranem in multiprocesiranem režimu obratovanja;

- odzivnost sistema pri obratovanju v režimu časovnega dodeljevanja;
- zanesljivost sistema.

Glede na perspektivo namenske rabe računalnika, lahko izluščimo naslednja osnovna področja, ki določajo usmerjenost standardne računalniške opreme:

- računalniški centri za splošno uporabo;
- banke podatkov in avtomatizirani sistemi upravljanja;
- sistemi za avtomatizacijo projektno-konstrukcijskih nalog;
- sistemi avtomatiziranih raziskav zapletenih objektov in krmiljenja tehnoloških postopkov v realnem času.

Pri sistemih, ki so namenjeni računalniškim centrom, ki niso posebej specializirani, zasledimo materialno programsko (firmwarsko) interpretacijo dokaj visokih programskih jezikov, ki povečujejo učinkovitost programiranja in realizacijo zapletenih sredstev jezika, hkrati pa bistveno olajšajo programersko delo v realnem času. Banka podatkov potrebuje visoko razvit informacijski sistem in pretežen del programske opreme v materialni programski opremi. Pri računalnikih, ki so namenjeni reševanju oz. avtomatizaciji projektivno-konstrukcijskih problemov, pa imajo veliko vlogo sredstva za obravnavanje grafičnih podatkov.

V nekaterih slučajih je smiselno, da se iz računalniške arhitekture sistema izločijo specializirani procesorji, ki realizirajo del standardne opreme in izvršujejo določene funkcije v splošnem računalniškem procesu. V zadnjih letih se pojavljajo naslednji specializirani procesorji:

- procesorji za upravljanje procesov, ki realizirajo osnovne dele operacijskega sistema;
- komunikacijski procesorji. Predstavljajo specializirane sisteme za pripravo in prenašanje podatkov med več, pogosto različnimi napravami;
- "pogovorni" procesorji. Služijo uporabnikom, ki delajo v realnem času;
- sintaktični procesorji. Obdelujejo tekste za programske in informacijske sisteme.

2.3. Paralelno obravnavanje podatkov

Nova tehnologija omogoča visoko stopnjo razčlenjevanja računalniškega procesa z izkoriščanjem velikega števila naprav, ki imajo podobne strukture. Kot primer vzemimo mrežo, ki je sestavljena iz številnih procesorjev v računalniku ILLIAC-IV. Izkoriščanje velikih modularnih mrež, grajenih iz pomnilniških elementov (množica registrov ali "logic-in-memory-arrays"), ki jih tržišče nudi v LSI tehnologiji, daje možnost hitrega izvrševanja tako zapletenih operacij kot so: naslavljanje asociativnega pomnilnika, preoblikovanja kartotečnih struktur, operacije nad grafičnimi operacijami, daje pa tudi možnosti za realizacijo novih načinov procesiranja tipa "pipe-line".

2.4. Algoritmični jeziki

Vse osnovne računalniške funkcije se konec koncem izvajajo s pomočjo interpretacije programov, ki so zapisani v nekem (notranjem) algoritmičnem jeziku. Zato učinkovitost izkoriščanja sodobnih in bodočih računalnikov bistveno zavisi od tega, kako uglašeno so razviti vhodni programski jeziki z notranjimi računalniškimi jeziki. Še posebno velik pomen ima razvoj podatkovnih struktur in sredstev za načrtovanje paralelnega obravnavanja informacij. Med temi sredstvi je najbolj razvita tehnika iskanja in opisovanja med seboj neodvisnih procesov, ki se lahko odvijajo paralelno, ter njihove sinhronizacije s pomočjo določenih karakteristik nastopajočih dogodkov oziroma mikrooperacij. Obstaja pa tudi drug način paralelnega obravnavanja informacij v računalniku. Pri tem načinu se navadno uporablja jezik strukturnih shem. Če upoštevamo, da vsaka komponenta strukturne sheme realizira nek algoritem, lahko govorimo o algoritmu, ki deluje na celotno shemo. Zato lahko takšen algoritem omogoča zelo visoko stopnjo paralelnosti. Čeprav daje tak način opisovanja algoritmov zelo zanimive možnosti, šele prehaja v programersko prakso. Rusi so razvili sistem PROJEKT in razvili jezik za opisovanje podatkov, ki vključuje sredstva opazovanja algoritmov s pomočjo strukturnih shem (jezik STRUKTURA). Tudi pri nas se je v te namene razvil osnutek jezikov ALUMIJ in SINUMIJ (glej disertacijo: P.Kolbezen, Algoritmični jezik za opis problemov analize in sinteze avtomatov), ki sta se izkazala zelo učinkovita. Izkoriščata sredstva za opisovanje modularnih struktur. ALUMIJ je prirejen matematičnemu aparatu za načrtovanje programirane materialne opreme, ki izkorišča mrežo operativnih enot (procesorjev).

3. METODIKA NAČRTOVANJA DIGITALNIH SISTEMOV

Za standardno programsko opremo je značilna hierarhija te opreme oziroma programskih sredstev. Na najnižjem nivoju se nahajajo notranji računalniški jeziki (v kompleksnejših sistemih jih je lahko več). Na naslednjem nivoju se nahajajo osnovni programi operativnega sistema. Nadalje, programi operativnega in informacijskega sistema, ki segajo po programih na nižjem nivoju itd. Programi nižjih nivojev se odvijajo v odvisnosti od programov na višjih nivojih kot makroinstrukcije. Te se generirajo z makroprocesorjem, ki se nahaja na višjem nivoju kot osnovni del operacijskega sistema. Na še višjem nivoju se nahajajo sistemski programi, ki jih tudi same lahko razdelimo na več nivojev. Nad njimi pa se nahajajo še specialni programski sistemi.

Mikroprogramsko krmiljenje, ki se izvaja s pomočjo notranjih računalniških jezikov, pa je že samo po sebi večnivojsko.

V prvih stopnjah projektiranja, ko se že odločamo za "grobo" računalniško strukturo in standardno opremo, moramo izbrati še osnovo splošne standardne opreme. Ta se mora načrtovati skupaj z materialno opremo delno ali povsem v programirani materialni opremi. Pravilen izbor te osnovne opreme oziroma sredstev ni pomemben toliko zaradi učinkovite realizacije, kolikor zaradi obstoja možnosti, da se lahko standardna oprema na višjih nivojih spreminja in prilagaja različnim konkretnim pogojem. Ta oprema daje splošno usmerjenost uporabe računalnika in je v zelo širokih mejah spremenljiva.

V osnovni sistem se morajo prištevati sredstva, ki omogočajo razširljivost in nadaljni razvoj sistema, kar lahko bistveno olajšuje kasnejša prilagajanja sredstev na višjih nivojih. Projektiranje računalnika lahko razdelimo na tri stopnje:

- sistemska
- logična
- tehnična

Cilj sistemskega projektiranja je reševanje nalog, ki so povezane z zgradbo, modularnostjo in določevanjem karakteristik sistemske strukture. Kot primer naj navedemo: izbor števila in velikost vmesnikov pomnilniških naprav (registrov), določitev mehanizmov komuniciranja med posameznimi pomnilniškimi nivoji, proučevanje karakteristik različnih delov standardne programske in materialne programske opreme projektirajočega sistema, določevanje časovnih karakteristik, pomembnih za medsebojno prilagajanje različnih računalniških blokov, itd.

Na stopnji logičnega projektiranja se določi algoritmična in logična struktura projektiranega sistema. Kot primer naj navedemo: bločno projektiranje, sinteza in analiza naprav specialnih tipov, minimizacija materialne in programirane materialne opreme, itd.

Tehnična stopnja projektiranja je namenjena določevanju zgradbe, razmeščanju in preoblikovanju konstruktivne strukture sistema. Primeri nalog, ki se rešujejo na tej stopnji so: določevanje topologije in geometrije komponent in povezav le-teh z upoštevanjem konstruktivnih posebnosti elementov in shem, realizacija sistema itd.

Delitev procesa načrtovanja na omenjene stopnje izhaja predvsem iz lastnosti in karakteristik posameznih struktur načrtovanega sistema. V naši metodiki projektiranja razlikujemo:

- sistemsko strukturo (arhitekturo)
- algoritmično strukturo (algoritem delovanja)
- logično strukturo (funkcionalna shema)

Posebej vsebuje algoritmična struktura celo vrsto karakteristik sistemske strukture, logična struktura vsebuje algoritmično, a konstruktivna struktura lahko daje informacijo o logični strukturi. V principu bi lahko reševali vse naloge projektiranja, če bi imeli eno samo strukturo. Pri tem pa bi naleteli na težave tehnične narave in neupravičljivo zapletenost reševanja nalog projektiranja.

Pri vzajemnem načrtovanju materialne in standardne programske opreme se zgoraj opisane osnovne stopnje in naloge projektiranja ohranjajo, dočim se njihova vsebina in metode reševanja bistveno spreminjajo in dopolnjujejo. Pojavljajo se nove naloge.

Jasno je, da moramo že na samem začetku načrtovanja računalniškega sistema podati začetni algoritmično-strukturni opis sistema. Določiti moramo množico algoritmov, ki omogočajo delovanje osnovnih naprav v procesu obravnavanja informacij, ki jih predpiše uporabnik sistema.

Razvija se,ali je že razvitih več jezikov, ki omogočajo omenjeno čimbolj učinkovito opisovanje sistema (STABLERJEV JEZIK, ALUMIJ, SINUMIJ, ALGORITEM, STRUKTURA, CDL in drugi).

Ti jeziki morajo vsebovati posebna sredstva za opisovanje sklopov, večregistrskih sistemov in drugih strukturnih podatkov, ki so ugodni za opisovanje informacijskih pretokov v sistemu.

Ves proces projektiranja predstavlja postopno transformacijo algoritmično-strukturnega opisa komponent projektirajočega računalnika. Kot rezultat teh transformacij dobimo programe notranje programske opreme, dokumentacijo celotne materialne opreme in programe tehnološke narave, kot so programi za avtomatizirano (strojno) realizacijo naprav, programi za kontrolo proizvodnih trakov, za testiranje in odkrivanje napak v sistemu itd.

Na sistemski stopnji načrtovanja se določi, kot je bilo že omenjeno, arhitektura računalniškega sistema in osnovna standardna programirana materealna in programska oprema v obliki algoritmično-strukturnega opisa. Ta opis dobimo s transformacijami in podrobnejšo razčlenitvijo prvotnega opisa. Stopnja detajliranja algoritmov mora biti tolikšna, da je že možno sistemsko programiranje in da je še vedno možno v dovolj širokih mejah spreminjati algoritme in strukturo projektirajočega sistema.

Ugodno je, da je prvotni opis čim preprostejši, nakar se uporabijo formalne transformacije, ki dajejo boljše karakteristike. Namen transformacij se formulira s pomočjo analize rezultatov moduliranja sistema, a sama transformacija je lahko avtomatizirana.

Oglejmo si za primer naslednjo transformacijo. Če neka naprava sprejme nalogo od drugih naprav in jo reši na ta način, da se za rešitev obrača na druge naprave, lahko v principu rečemo, da dela v tako imenovanem multipleks režimu. To pomeni, da more istočasno reševati več nalog. Zato je ugodno, da najprej zapišemo algoritem reševanja vsake posamezne naloge in šele nato preidemo k algoritmu multipleksnega delovanja s pomočjo neke formalne procedure. Ta prehod lahko izvršimo šele nato, ko dobimo rezultate sistemskega modeliranja, ki dajejo količinske karakteristike multipleksnega obratovanja.

Na stopnji algoritmičnega projektiranja se podrobneje izdelajo algoritmi osnovne standardne programske opreme in rešuje ena od osnovnih nalog združenega projektiranja materialne in standardne programske opreme računalnika, razdelitev na materialno programsko (firmware) in standardno programsko opremo. Programski del izločimo s prevodom algoritmičnega opisa na višjih nivojih v strojni jezik in z zamenjavo teh nivojev z interpretativnimi algoritmi, in če je potrebno, z upoštevanjem ustreznih programov informacijskega sistema. Istočasno z izločanjem programskega dela se izvrši izbor in korektura notranjih računalniških jezikov. Izločitev gornjega nivoja se izvrši s transformacijo takšnih algoritmičnih opisov kot je opis izločanja in ugotavljanja podprogramov, parametrov podprogramov itd. Izvrši se tudi prehod iz opisa na višjem nivoju k opisu na nižjem nivoju. N.pr. algoritem je lahko podan v obliki abstraktne strukture podatkov (spiskov, formul itd.) in prehod k nižjemu nivoju se izvaja z vstavitvijo takšne podatkovne strukture v pomnilniško strukturo, ki je sestavljena iz besed, zlogov, itd. V ta namen se uporablja običajna tehnika prevajanja, pri tem pa mora biti prevajalnik prilagojen raznim načinom realizacije operatorjev vhodnega jezika. Če se uporabijo splošne metode transformacije algoritmičnih opisov, se često prekomerno poveča število potrebnih operacij in poslabšajo karakteristike algoritmov. Algoritmi se morejo izboljšati s pomočjo posebnih optimizacijskih programov.

Navedli bomo nekaj nalog, ki jih moramo reševati na nivojih sistemskega in algoritmičnega projektiranja s pomočjo transformacij algoritmov delovanja naprav:

- porazdelitev funkcij med procesorji;
- specializacija in univerzalizacija procesov;
- povečanje hitrosti sistema na račun števila istočasno delujočih naprav;
- pomnenje in istočasno izvajanje programskih modulov;
- izbor količinskih parametrov naprav in programov s pomočjo sistemskega modeliranja;
- razdelitev materialne programske in standardne programske opreme;
- izbor in korekcije notranjih računalniških jezikov.

Nekaj omenjenih nalog je že možno avtomatizirano reševati s pomočjo pri nas izdelanih postopkov (glej raziskovalne naloge: Optimizacija struktur digitalnih sistemov, II. in III. faza in v zadnjem času: Načrtovanje organizacij digitalnih sistemov, ki jih finansira SBK).

Metode bločne sinteze so postale uporabljive in zelo učinkovite tudi z uvajanjem LSI tehnologije. Omogočajo prenos reševanja številnih nalog s stopnje tehničnega projektiranja na

stopnjo logičnega projektiranja, še posebej nalog združevanja na vseh stopnjah, nalog izbora standardnih konstrukcijskih enot in še nekaterih.

4. TEHNOLOGIJA IN ORGANIZACIJA PROJEKTIRANJA

Tehniško osnovo načrtovanja računalnikov bodočnosti predstavlja avtomatizirana tehnologija načrtovanja, konstruiranja in tehnološka priprava, ki omogoča izdelavo kompletne materialne in standardne programske opreme. Izdelali naj bi se tehnični projekti in popolna dokumentacija za izdelavo in uporabo računalniškega sistema, za standardno programsko opremo, kakor tudi za izvajanje eksperimentalnega dela pri uporabi zgoraj opisane metodike in tehnologije projektiranja.

Raziskave in obravnavanje tehničnih in matematičnih sredstev za tehnologijo projektiranja igrajo pomembno vlogo ne samo na področju avtomatizacije načrtovanja računalniških sistemov, ampak tudi za širše področje znanstveno-tehniškega napredka (npr. avtomatizacija znanstvenih raziskav, avtomatizacija sistema splošne uprave itd.).

Specializiran operacijski sistem kot nadgradnja standardnega operacijskega sistema obsežnega računalniškega kompleksa mora zagotoviti informativnost in vzajemno delo večjega števila projektantov v realnem času.

Podatkovni jezik mora vsebovati sredstva za algoritmično in strukturno opisovanje naprav, sredstva za opisovanje lastnosti algoritmov in računalniških procesov, sredstva za opisovanje tehničnih karakteristik komponent strukture. Sistem mora vsebovati tudi sredstva, ki omogoča razširitev podatkovnega jezika v smislu konkretnih pogojev projektiranja.
V sistemu imamo vrsto splošnih algoritmov, ki jih je mogoče uporabiti za izvrševanje konkretnih transformacij z upoštevanjem ustreznih informacij. Primeri takšnih algoritmov so univerzalni prevajalniki, ki prevajajo konkretne jezike, če so podane tudi vse sintaktične in semantične informacije, makroprocesorji, ki krmilijo nabore makro operacij, modelirni sistemi, ki omogočajo spreminjanje semantike operacij v modelirnih jezikih, optimizacijski procesorji, ki dopuščajo možnost spreminjanja kriterijev in pogojev, ki sicer omejujejo iskanje optimalne variante itd.

Zaslediti moremo dve osnovni periodi izkoriščanja tehnologije načrtovanja.

Prva perioda je začetno obdobje projektiranja računalniških sistemov. Za to stopnjo je značilno znanstveno-raziskovalno delo, ki daje osnovne rešitve na področju strukture in arhitekture računalnikov ter njihove standardne programske opreme (standardnega firmwara in softwara). V tem obdobju tehnologija projektiranja uporablja sredstva avtomatizacije znanstvenih raziskav in baze za izvajanje eksperimentov. Z njimi se preizkušajo osnovne znanstveno-tehnične ideje. Projektirani so že konkretni primeri ustrezne razširitve podatkovnega jezika, izdelani so posamezni programi, ki vključujejo nove programe v sistem itd. Pri projektiranju konkretnih primerov se izdela metodika aplikacij, zaporedje izvajanj operacij, koordinacija obdelave podatkov itd.

V drugi periodi se uporabi načrtovalni sistem, ki omogoča izdelavo konkretnega računalnika oziroma računalniškega sistema. Posamezni vzorci ali konfiguracije načrtovanega sistema se lahko razlikujejo po tehničnih parametrih, ki omogočajo realizacijo posameznih delov standardne programske opreme, razdelitev le-te na materialno programsko in programsko opremo itd. Pri tem se nekateri nivoji načrtovanja izkažejo univerzalni za vse konfiguracije, načrtovanje na drugih nivojih in končna dokumentacija pa se dobi z rabo vrste algoritmov projektiranja in z določitvijo parametrov, od katerih zavisi delovanje teh algoritmov. Z združitvijo algoritmov, ki se izdelajo pri projektiranju posameznih sistemskih sestavov, je mogoče z ustreznimi programi generirati standardno programsko opremo načrtovanega računalniškega sistema.

5. ZAKLJUČEK

V delu so obravnavane metode načrtovanja podobnih in bodočih digitalnih računalniških sistemov. V njem zasledimo bistven poudarek na avtomatizaciji projektiranja. Ta je namreč ena od pomembnih pripomočkov, ki omogoča načrtovalcu programske in materialne opreme kar najhitrejše prilagajanje oziroma izkoriščanje hitro se razvijajoče osnovne terminologije integriranih elektronskih vezij. Iz obstoječega stanja pa je razvidno, da "popolno" izkoriščanje najsodobnejše tehnologije še vedno zaostaja za njenim razvojem, kar je med drugim pripisovati tudi pomankljivi in nezadostni avtomatizaciji načrtovanja. Zato

se dandanes v svetu vlaga vse več naporov, da bi se obstoječe stanje izboljšalo. Tudi pri nas lahko zasledimo že vrsto uspehov na tem področju v okviru tovrstnih raziskovalnih nalog pedagoških in raziskovalnih ustanov (zlasti Fakultete za elektrotehniko in Instituta "Jožef Stefan"). To so zlasti naloge omenjenih ustanov, ki jih finansira Raziskovalna skupnost Slovenije za področje avtomatike, računalništva in informatike.

Za avtomatizacijo načrtovanja je bilo izdelanih že precej, več ali manj zaključenih programskih paketov, ki so bili že tudi uporabljeni pri reševanju nalog za potrebe našega industrijskega okolja. Naj navedemo le nekaj takšnih paketov, ki so bili izdelani na Institutu "Jožef Stefan" in deloma tudi v sodelovanju z Republiškim in Univerzitetnim računskim centrom: simulatorji računalnikov (n.pr. za računalnik ITT 1600), prevajalniki, interpreterji, zbirniki, simulatorji digitalnih vezij, programski paket za analizo in sintezo vezij, za verifikacijo in diagnostiko le-teh, ter za analizo in sintezo programov oziroma mikro programov, za diagnosticiranje prevajalnikov, za časovno in prostorsko optimizacijo programov na osnovi ugotavljanja paralelizma ter aktivnosti in pasivnosti spremenljivk v (mikro) programih, programi za časovno optimizacijo na osnovi minimalnega števila posegov v centralni pomnilnik, itd.

Omenimo naj še dela na razvoju jezikov za opis struktur in algoritmičnega opisa delovanja vezij, ter visokih programskih jezikov ali bolje rečeno izpeljank obstoječih visokih jezikov za programiranje mikro računalnikov.

V zaključku naj še pripomnimo, da so sredstva, se vlagajo v naša raziskovalno-razvojna prizadevanja in šolanje kadrov še vedno mnogo premajhna, da bi bilo mogoče zadostiti potrebam našega industrijskega okolja ter doseči in vzdržati korak z razvojem v svetu.

6. LITERATURA

1. M.A.Breuer, Design Automation of Digital Systems, N.Y. 1972

2. Raziskave, ki jih je v letih 1974-78 finansirala Raziskovalna skupnost Slovenije s področja avtomatike, računalništva in informatike

3. Raziskave, ki jih je v letih 1974-78 finansirala Raziskovalna skupnost Slovenije s področja avtomatike, računalništva in informatike

4. P.Kolbezen, Algoritmični jezik za opis problemov analize in sinteze avtomatov, Univerza v Ljubljani, Ljubljana, april 1974

5. M.A.Breuer, Editor: Digital System Design Automation: Languages, Simulation and Data Base; Pitman P.L., London 1977

7. Kibernetika, Žurnal AN USSR

8. Computer, strokovna računalniška revija, ZDA

9. IEEE Transaction on Electronic Computer, znanstveno raziskovalna revija IEEE, ZDA

mathematical models for computer-assisted solutions of non-numerical problems in chemistry

b.jerman-blažič

UDK 681.3:54

J. Stefan Institute, University of Ljubljana,
61000 Ljubljana, Yugoslavia

MATEMATIČNI MODELI ZA REŠEVANJE NENUMERIČNIH PROBLEMOV IZ KEMIJE S POMOČJO RAČUNALNIKA. V članku je dan pregled uporabe molekularne topologije pri računalniškem reševanju nekaterih nenumeričnih problemov iz kemije. Obdelani so naslednji problemi: identifikacija spojin v sistemih za iskanje in shranjevanje informacij, enumeracija in generiranje strukturnih izomerov s ciljem izločevanja molekularnih struktur na podlagi eksperimentalnih podatkov, predstavitev molekularnih struktur in procesa načrtovanja sintez v računalniškem načrtovanju organskih sintez.

The article gives a review of the applications of the molecular topology in computer-aided solving of some non-numerical chemical problems. The problems are: identification of a compound in chemical information retrieval systems, enumeration and generation of structural isomers for the purpose of special chemical studies and in computer-aided elucidation of molecular structure on the basis of experimental data, representation of molecular structures and synthesis-design process in computer-aided planning of organic syntheses.

1. Introduction

There seems to be hardly any concept in natural sciences which is closer to the notion of a graph than the molecular structure of chemical compounds. A molecular structure may be viewed as a graph composed of nodes (atoms) linked with edges (chemical bonds). In fact there is no essential difference between a graph and a structural formula. A graph is a mathematical structure which can be used to represent the topology of given molecule. The advantage of using graphs in the representation of molecular structure lies in the possibility of applying directly the mathematical apparatus of the graph theory for solving special chemical problems. The idea that metric characteristics of the molecules (that is bond lengths and bond angles) can be neglected in chemical studies is more and more popular. The molecular topology allows the non-metric relationships of the molecular structures and the totality of information contained in the molecular graphs to be investigated and applied in a very simple manner.

Concepts of topology and graph theory though not always recognized as such, are nowdays analysed and applied to various branches of chemical science: photochemistry (1), stereochemistry (2), transition metals chemistry (3), boron hydrid chemistry (4), saturated (5) and unsaturated (6) hydrocarbon chemistry, etc. Furthermore, basic concepts of chemistry such as configuration, isomerism, valency etc. are shown to have a topological basis (7).

In the present article we wish to review mainly the application of the molecular topology and the topology of a space of molecules in computer-aided solving of some non-numerical chemical problems. The problems are: identification of a compounds in chemical information retrieval systems, enumeration and generation of structural isomers for the purpose of special chemical studies and in computer-aided elucidation of molecular structure on the basis of experimental data, representation of molecular structures and synthesis design process in computer-aided planning of organic syntheses.

Let us at first to define a chemical graph and the associated notions: A chemical graph is (8) a graph consisting of nodes associated with atom names, and edges which correspond to chemical bonds. The degree of a node in the chemical graph has its usual meaning, i.e. the number of (non-hydrogen) edges connected to it. The valence of each atom determines its maximum degree in the graph. A special kind of chemical graphs are vertex-graphs. Vertex graphs are cyclic chemical graphs (8), from which nodes of degree less than three have been deleted.

2. Identification of a structure of chemical compound in an Information retrieval system

There is probably no science in greater need of mechanized information retrieval than chemistry. Millions of chemical compounds are known; new ones are produced at an even faster rate. The chemist has two main problems: first, he wants to find out whether the substance in his test tube is already known; second, given a substance, he wants to know the properties of similar substances.

Both problems can be reduced to a matching process: a description of the given compound has to be matched against descriptions of the compounds that make up the data base of the retrieval system. To assure a complete identification of the compound structure, a detailed atom-by-atom comparison is usually needed between the compound in the query and the compounds in the data base. If chemical compounds are represented as chemical graphs, the problem of matching the query item with the library item becomes identical to the problem of isomorphism of graphs, considerably simplified by the labels carried by the chemical graph nodes.

The problem of isomorphism of graphs received little attention in the literature until late 1950's (9). Research into this problem was stimulated by the development of chemical information retrieval systems, with chemical structure representation in the system's files. An approach that was implemented in several computer programs was the procedure of a node-to-node matching in search of coincidence. The nodes of two chemical graphs are matched one at time until either a valid correspondence is found or until incompatibility arises (10). In the later, it is necessary to backtrack to a point of former coincidence and start again with a different

choice of nodes. Large amount of backtracking is required in this technique due to the lack of any criteria in the decision-making step. From the computational point of view, the unavoidable backtracking is time wasting as only in rare instance is the correct choice made at each point of decision. This technique also requires information to be saved in order to restart from the last point of agreement.

The use of a standard numbering procedure for the nodes in the chemical graphs makes the problem of establishing isomorphism in graphs trivial. Many attempts have been made in order to develop standard numbering procedures. Bouman (11) has suggested ordering of the nodes in chemical graphs based on the examination of the degree of a node and the degree of the nodes to which it is connected. Randić (12) proposed a very interesting procedure for labelling the atoms in the graph by considering the rows of the adjacenty matrix of the graph as digits coded in a linear code. The search for matrices corresponding to a complete graph or to a fragment of a graph is to be carried out by ordering of the matrices according to decreasing values of the numbers representing the row vectors. A year later, the same author suggested another solution to the unique labelling of the graph nodes. The sequencing of atoms is performed according to the relative magnitudes of the coefficient of the largest eigenvector of the adjacency matric (13). Similar approach to the Bouman scheme is the Morgan algorithm (15) which exploits the concept of extended connectivity. The Morgan's algorithm was implemented in the information system of Chemical Abstract Service. The classification of the atoms is obtained by adding the initial connectivity values of nearest neighbours and assigning the sum to the node considered. As (16) recognized, this method does not always allow the maximum possible differentiation, although it generally allows the atoms to be divided into several classes depending on the number of non-hydrogen attachements to each atom.

Bart and Giordano proposed (14) a new graph matching procedure in which for the one-to-one matching the Fourier maxima of specific entities of the known chemical structure was used.

A completely different and most computer-oriented approach to graph identification was advanced by Sussenguth (17). The procedure that he suggested is based on two principles. First, if graphs G and G* are isomorphic, then the subset of nodes of G that exhibit some property must correspond to the subset of nodes of G* that exhibit this same property. Second, if the subsets of the nodes of G and G* that are characterized by some property do not have the same number of elements, then the two graphs cannot be isomorphic. The matching procedure starts with generation of subsets of nodes that represent the same type of atom.

The purpose of generation of subsets of nodes is to reduce the number of nodes of G* to which a node of G can correspond. The purpose is achieved by taking intersection of the subsets and by matching the resulting nodes. If some nodes are not matched, new subsets must be generated. The algorithm terminates when every node in G is paired off with a node in G*, or when two corresponding subsets of nodes of G and G* are found to differ in the number of nodes they contain. If the former is the case, graphs G and G* are isomorphic, if the later, isomorphism is impossible. Occasionally, the algorithm exhausts all subset generating properties before one of the two conditions is satisfied. This happens when more than one isomorphism is possible between the two graphs, or when the subset generating properties are incomplete in the sense that some property that would establish isomorphism or the lack of it has been neglected in the design of the algorithm. An improvement of the described algorithm was suggested by Ming and Tauber (18). They separated the structure search and substructure search into a distinct part of the algorithm and included first order degree (17) and second order degree in the control vector for use in structure search. A short cut of the atom-by-atom search technique and set generation procedure is the connectivity code developed by Penny (19). The connectivity code although it is not a solution in itself, can be a useful tool when used in conjunction with the two general techniques (atom-by-atom search and the set generation algorithm) as it is done in the computer program of Tauber and Ming (18).

3. Computer-aided generation and enumeration of structural isomers

Problems of structural isomerism in chemistry have received much attention for a long time, but only occasional attempts have been made toward a systematic solution of the underlying graph theoretical problems of structural isomerism. Graph theoreticians have frequently considered various aspects of this topic, but not necessarily in the context of organic molecules. Polya presented a theorem (20) which permits calculation of the number of structural isomers for a given ring system. Hill (21) and Taylor (22) pointed out that Polya's theorem permitted enumeration of geometrical and optical isomers in addition to structural isomers. More recent formulas for the enumeration of isomers of monocyclic aromatic compounds based on the graph theory, permutation groups and Polya's theorem were presented (23). Although the number of isomers may be interesting, these methods do not display the structure of each isomer. Even in simple cases, the task of specifying each structure by hand without duplication is an enormous one. Balaban published a series of papers (16) addressed in part to the problem of specification of isomeric structures. Although his method represents an important contribution to the problem of isomeric structures, it does not contain a mechanism for avoiding a duplicate structures. Most successful in solving this problem were the works based on the Dendral algorithm (24). The algorithm permits an enumeration and representation of all possible molecular structures with a given empirical formula, i.e. a given set of atoms. Chemical structures of all possible isomers are obtained by mathematical permutation of acyclic and cyclic graphs representing appropriate ring systems and attached acyclic chains of atoms. The Dendral algorithms was implemented in a computer program called Structure Generator (8). The list of the structural isomers generated by the program is in the form of a special kind of graph -AND/OR tree (25).

The ring systems in the program are constructed from vertex graphs (8), which are defined in a given problem by a series of calculations. The first level of the tree, after the specification of the initial collection of atoms, is the set of all possible partitions of the initial set of nodes. Each partition consists of the cyclic subunit and the remaining set of nodes. The cyclic subunits are a collection of atoms from which all possible ring systems can be constructed on the basis of the appropriate vertex graph. The atoms in the remaining set form acyclic parts of final structures, combined in all possible ways with the ring structures from the corresponding initial partition. The second level of the tree specifies all possible ring systems that can be constructed from the vertex graph corresponding to the cyclic subunit in the first level of the tree. The next level of the tree just beyond the node specifying a possible ring system, specifies the possible ways in which the remaining atoms can be linked to the unfilled links of the system. After the three first levels of the tree generation, the program becomes recursive. Each set of unstructured nodes is taken up as a fresh problem until there are no more unstructured nodes. The Structure Generator represents a part of a very complex and sophisticated computer program - Heuristic Dendral Program (25,8) for elucidation of molecular structure based on structural features of unknown molecules derived from chemical, physical and spectroscopic data.

Recently, a similar approach to the problem of exhaustive enumeration and generation of chemical structures was published by Sasaki and Kudo (26). Their system successfully deduce all logically valid structures, acyclic and cyclic on the basis of previously settled proposition according to the input information about the structure, of a given compound.

4. Mathematical models in the computer-aided-planning of organic syntheses

The problems of isomorphism of chemical graphs and generation and enumeration of structural isomers are closely related to the works connected with the design of chemical structure information systems. The justification of the chemical structure information systems is the assistance in the research process. It happens very often that it is not only the structure of the compound which intersets the chemist, but also the properties of the compound which the structure represents. For instance, the chemist may be interested in the synthesis of the compound, in some of its physical properties, in its behavior in a living system etc. For these reasons greater attention should be paid to the problem of collection, evaluation and correlation of the data associated with a particular compound. Closer to these desired objectives are the studies carried out in the field of the application of machine computation to the generation of chemical pathways for the synthesis of complex organic molecules. Mathematical models in the form of graphs and the associated theory in the computer-aided design of chemical synthesis are involved in two ways: first as a tool in the representation of structures stored into the programs; and second, as a model in the computer representation of synthesis pathways. Synthesis pathways can be viewed as a tree (27) in which the root is the synthetic target, the intermediates in the synthesis process are the nodes, and the chemical reactions are the edges linking the nodes of the tree.

Several alternatives to the computer-aided design have been attempted. The strongest attention from the chemists has evinced the work of the groups of Princeton and Harvard (28). Their works were based on the interactive computation with an on-line guidance by a chemist. This feature has enabled them to solve interesting chemical problems by pooling the resources of a computer with a chemist as a source of information on reaction and on strategic design decisions.

The other most significant approach is based on the Heuristic Search Paradigm of Artificial Intelligence (30) research. The program developed at Stony Brook (29) designes synthesis without the chemist's intervention using the reactions from the program reaction library and programmed design strategies.

A similar approach can be found in the works of Whitlock on the heuristic solutions of the functional group synthesis problem (32). The reaction library in his program represents the implementation of the transition graph of a finite automata, wherein the nodes are functional groups and edges are the reactions that transform one functional group into another.

The most mathematical scheme of synthesis-planning problem was suggested by a group of prominent chemists and mathematicians (31). The scheme is based on the recognition that all chemical reactions correspond to interconversions of isomeric ensembles of molecules (IEM) within a family of isomeric ensembles of molecules (FIEM) (31). Distinguishable IEM of FIEM is represented by a family of be-matrices (bond and electron matrices) $F = (M_0, M_1 \ldots, M_f)$. The be-matrix M_i of an ensemble of molecules EM_i consisting of a set A, which contain n atoms, $A = (A_1, \ldots, A_n)$ is an n x n matrix as shown below:

$$M_i = \begin{pmatrix} a_1 & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & & & \\ a_{n1} & & \cdots & a_{nn} \end{pmatrix}$$

were the entries a_{ij} $(i \neq j)$ are the formal bond orders of the bonds between pairs of atoms A_i and A_j, the diagonal entries a_{ii} are numerically equivalent to the number of free valence electrons belonging to atom A_i in EM_i.

A slightly modified version of this mathematical model of chemical systems and their relations was used as a basis for the construction of algorithm, which generates multistep syntheses of a given chemical compound. The algorithm and the former mathematical model were implemented in an organic-synthesis-planning program called HEDOS (33). The program is confined to systems consisting of the benzene ring and functional groups attached to it. Specially designed heuristic rules governing the generation of the best synthesis were incorporated into the program in view of the complexity of the synthesis pathway and the number of steps involved.

In this approach, the be-matrices of a FIEM defining metric topology, were embedded as elements of state space, that we called the state space of ensembles of molecules (33). The associated set of operators of the state space was defined as a set of reaction matrices $D(n)$; $D(n) = |-R|R$ is the reaction matrix which fits (31) the elements of FIEM|. The set of fitting matrices $F(n)$ for some state EM_i in the FIEM represented by a be-matrix B_i is obtained by the mapping γ:

$$\gamma: D(n) \times FIEM \longrightarrow F(n);\ \gamma(D(n), B_i) = |F \quad F - B_i \leq 0|$$

The target molecule Z is contained in an initial EM, denoted with EM_Z. Final states are all possible EM assigned as EM_L provided that the chemical species in EM_L are contained in the list of available chemical compounds $\mathfrak{L}$, meaning that they can be easily synthetisized or found in the commercial catalogue of the world-known suppliers of fine chemicals. The molecules in EM_L are starting materials for the synthesis of molecule Z and compose the list L, $L \subseteq \mathfrak{L}$. Thus, the problem of synthesis design for a compound Z can be reduced to the problem of finding a path K, $K = |R_1, \ldots, R_n|$ into a space of FIEM which transforms EM_Z in EM_L. The search for a path through a state space is equivalent to the travel through a directed graph, in which the nodes correspond to various EM from FIEM and edges correspond to the set of reactions D(n). The root in the directed graph which is a tree in this case is the ensemble EM_Z. The implemented synthesis-planning-algorithm in the program HEDOS generates the space of FIEM and searches for a minimal length path which leads from EM to some EM_L. This minimal path meets some prescribed criteria, which garantee the feasibility of the proposed reactions and the validity of generated structures. Information contained in the be-matrices of the EM is usually insufficient for the evaluation of the proposed reactions and intermediate structures, so additional information concerning other molecular properties was stored in the second symmetric triangle of the be-matrix. The new matrices were defined as bei matrices:

$$M_1 = \begin{pmatrix} a_{i,j} & \text{for } i \leq j & \\ i_{i,j} & \text{for } i > j & \text{additional information} \end{pmatrix}$$

The program is not interactive, i.e. the chemist cannot interrupt the program to assist in the search for synthetic intermediates or in the evaluation of the synthetic path. The program must make all decisions by itself and is strictly experimental. It was designed for the purpose of developing and testing artificial intelligence mechanism with the aid of a strongly defined topology of molecular structures and related reactions.

5. Conclusion

The recent approaches to computer-aided solving of non-numerical chemical problems have been reviewed and the merits and drawback of implemented mathematical models outlined. It seems that the use of graphs as mathematical structure in the representation of chemical compounds, as they provide a form suitable for computer manipulation, becomes more and more popular. Best results in this field was achieved in application of graph theory and permutation groups in computer programs which generate and enumerate all possible structural isomers of a given set of atoms. Thus, the problem of exhaustive isomer generation can in general be considered as solved.

The other problem, identification of chemical compounds in the information retrieval system, for the solution of which the same mathematical model was used, was not so successfully solved. The majority of information chemical systems still performe the structure and substructure searches by using logical combinations of the structure fragments, as the compounds in the system's files are presented in one of the linear notations (WLN, JUPAC-DYSON) or by different fragment codes (Mechanical Chemical Code, KWIC indexes). Both forms of representations are simple to operate. As the volume and the interdisciplinary needs of chemistry, especially in the research process have increased, and the need for fully explicit structure representation of molecules becomes essential, various chemical information systems (CAS, DARC, TOSAR) have included graphs as a form of representation of chemical notions, but only as a supplement to the files with standard records. The great deficiency of this form of representation is the time - consuming identification of compounds. The chemist and information scientists still work on the development of fast and effective graph matching techniques, as the problem is not only a chemical problem, but also a computing problem. The task is large and difficult and should require common efforts for its solution.

The computer-assisted planning of organic syntheses is just beginning. The applied mathematical models and artificial intelligence methods have exhibited many deficiences, but they can be overcome. The success of the implemented computer programs justifies the expectations that the use of the computer-assisted-synthesis analysis will become a routine in the near future. The described mathematical model of the space of ensembles of molecules, provides a useful basis for the construction of a synthesis-planning algorithm. At the same time it offers possibilities for further uses, as it is the first attempt toward a systematization of procedures for storing and handling of the vast quantity of chemical information that is currently available.

6. References

1. H.E. Zimmermann, Angew.Chem.Internat.Edit, 8 (1969) 1.
2. V. Prelog, Chem. Britain, 4 (1968) 382.
3. H.A. Schmidtke, Coord.Chem.Revs. 2 (1967); J.Chem. Phys. 48 (1968) 970.
4. S.F.A. Kettle and V. Tomlinson, J.Chem.Soc. 91 (1969) 2002.
5. H. Hosoya, Bull.Chem.Soc. Japan.44 (1971) 2332.
6. K. Ruedenberg, J.Chem.Phys. 22 (1954) 1878.
7. I. Ugi, D. Marquarding, H. Klusacek, G. Gokel and P. Gillespie, Angew.Chem. 82 (1971) 741.
8. L.M. Masinter, N.S. Sridharan, J. Lederberg, D.H. Smith, J.Am.Chem.Soc. 11 (1974) 7702; L.M. Masinter, N.S. Sridharan, D.H. Smith, J.Am.Chem.Soc. 11 (1974) 7715; R. Carhart, D. Smith, A. Brown, N.S. Sridharan, J.Chem.Infor.Comp.Sci. 15 (1975) 2.
9. A. Sachs, Publ.Math.Debrecen (1962) 270; 11 (1963) 119; L. Spialter, J.Chem.Doc. 4 (1964) 269; F. Harary, J.Math.Phys. 38 (1959) 104.
10. L.C. Ray, R.A. Kirsch, Science, 126 (1957) 814.
11. C.M. Bouman, J.Chem.Doc. 3 (1965) 92.
12. M. Randić, J.Chem.Phys. 60 (1974) 3920.
13. M. Randić, J.Chem.Infor.Comp.Sci. 15 (1975) 105.
14. J.C. Bart, N. Giordano, Proceedings of the International Conference on Computers in Chemical Research and Education, Ljubljana-Zagreb, 1973, Paper 3/1
15. H.L. Morgan, J.Chem.Doc. 5 (1965) 107.
16. W.T. Wipke and T.M. Dyott, J.Am.Chem.Soc (1974) 4825.
17. E.A. Sussenguth, Jr., J.Chem.Doc. 5 (1965) 36; E.A. Sussenguth, Jr., "Structure Matching in Information Processing, Thesis Harvard Univ., 1964.
18. T.K. Ming, S.J. Tauber, J.Chem.Doc., 11 (1971) 47.
19. R.A. Penny, J.Chem.Doc. 5 (1965) 113.
20. G. Polya, C.R. Acad.Sci. 201 (1935) 1167.
21. T.L. Hill, J.Phys.Chem. 47 (1943) 253; T.L. Hill, J.Phys.Chem. 11 (1943) 294.
22. W.J. Taylor, J.Phys.Chem. 11 (1943) 532.
23. A.T. Balaban, F. Harary, Rev.Roum.Chim. 12 (1967) 1511; ibid., 11 (1966) 1097; ibid. 12 (1967) 103.
24. J. Lederberg, "DENDRAL-64, Part I; Notational Algorithm for Tree Structures", NASA Star. No. N-65-13 158, NASA CR-57029; "Part II, Topology of Cyclic Graphs" NASA Star. No. N 66-19074, NASA Cr-68898; "Part III Complete Chemical Graphs: Embedding Rings in Trees" NASA Star. No. N 71-76061, NASA CR-123176.
25. B. Buchanan, J. Lederberg, Proceedings of IFIP Congress 1971, Ljubljana, paper TA-2-91; B. Buchanan, G. Sutherland, E.A. Feigenbaum, In Machine Inteligence 4, 5 (1969) ed. Meltzer and Michie, 209, 253; J. Lederberg, G.L. Sutherland, B.G. Buchanan, E.A. Feigenbaum, A.V. Robertson, A.M. Duffield, C. Djerassi, J.Amer.Chem.Soc. 91 (1969) 11.
26. Y. Kudo, S. Sasaki, J.Chem.Infor.Comp.Sci. 16 (1976) 1.
27. E.J. Corey, W.T. Wipke, Science 166 (1969) 178.
28. E.J. Corey, Quart.Rev. (London) 25 (1971) 455; E.J. Corey, W.T. Wipke, R.D. Cramer, W.J. Howe, J.Am.Chem.Soc. 94 (1972) 421; ibid. 94 (1972) 431.
29. N.S. Sridharan, Proceedings of IFIP Congress, 1974, Stockholm, 828-837.
30. J.R. Slagle, Artificial intelligence: The Heuristic Programming Approach, Mc Graw-Hill, New York, 1971.
31. J. Dugundji, I. Ugi, Topics in Current Chemistry, 39 (1973) 21, Springer Verlag; I. Ugi, P. Gillespie, C. Gillespie, Trans.New York Ac.Sci. 34 (1972) 416.
32. E. Blower, Jr., W. Whitlock, Jr., J.Am.Chem.Soc. 98 (1976) 1499, W. Whitlock, ibid., 98 (1976) 3225.
33. B. Džonova-Jerman-Blažič, I. Bratko, Proceedings of AFCET Congress, Gif-Sur-Yvette, 1976, 283; ibid., Proceedings of the International Conference on Information Sciecnes and Systems, Patras, 1976, Hemisphere Publishing Corporation, Washington, 461.

nekaj praktičnih izkušenj uvajanja metodologije informacijskih sistemov

j.kalan

UDK 681.3:621

Intertrade TOZD zastopstvo IBM, Ljubljana

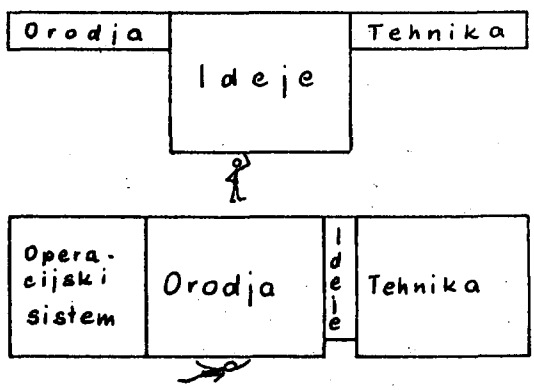
Članek podaja kratek opis metodologije razvoja informacijskih sistemov, kakršno sem zadnjih nekaj let skušal uveljaviti in verificirati v praksi. Ta metodologija je sinteza zanih principov in metod, s posebnim poudarkom na principu vodenja po ciljih. Metodologija je bila uveljavljena v nekaj primerih in je dala dobre rezultate.

EXPERIENCES WITH A METHODOLOGY IN INFORMATION SYSTEMS DEVELOPEMENT. This is the description of a practicaly proven methodology used in the developement of some information sistems. This methodology is a synthes of known principles and methods with special emphasis on managemant by objectives.

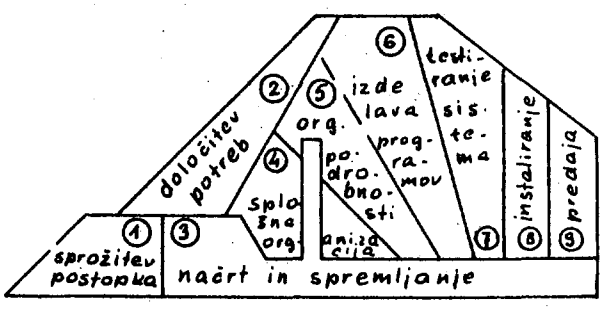
1. UVOD

V obdelavi podatkov danes že lahko govorimo o informacijskih sistemih, saj imajo skoro vsi sodobni in vsaj srednje veliki sistemi dve značilnosti, ki take sisteme omogočata. Ti značilnosti sta, da so,praktično vsi relevantni podatki shranjeni na nosilcih podatkov z direktnim pristopom in, da je pristop do shranjenih podatkov možen z velikega števila delovnih mest preko med seboj neodvisnih terminalov. Namen tega članka je predstaviti metodologijo razvoja, ki je bila preskušena v naših razmerah.

Metodologija razvoja informacijskih sistemov je danes dokaj dozorela (glej sliko 1). Da jo lahko uspešno uporabljamo, pa moramo dobro poznati osnove, na katerih je zgrajena (1,2, 3,4). Pri poskusu razdelitve razvojnega procesa na posamezne komponente so sprva skušali izdelati čim bolj splošno veljaven mrežni plan. Kasneje se je pokazalo, da je še bolj važno točno opredeliti zastavljene cilje ter določiti potrebna sredstva, ki morajo biti primerna za dosego postavljenih ciljev. Pri opisu potrebnih opravil najčešče nastopa razdelitev naloge v faze. Diagram, ki ponazarja razdelitev življenskega ciklusa razvojnega projekta v faze, izgleda da je prvi opisal Aron(2). Tak diagram v nekoliko prilagojeni obliki prikazuje slika 2.Priti do primernega besedila za posamezne točke v spisku opravil, ki pripadajo posamezni fazi, je dokaj zahtevno delo. Tako besedilo se s spremembo tehnologije lahko znatno spremeni. Posebno velik vpliv ima tu programska oprema. Najboljšo osnovo nam dajo razni priročniki (5,9), ki pa so vedno precej specifični za sredstva, s katerimi



Slika 1 Razvoj programiranja (po Aronu)



Slika 2 Faze v razvoju informacijskega sistema (Project Life Cycle)

razpolagamo. Največkrat nas izbrana programska oprema sili, da se držimo pripadajočega razporeda opravil in ostale v njej vključene standardizacije.(standards enforcing). Najnujnejša sredstva, ki jih potrebujemo pri razvoju informacijskega sistema, predstavljajo na primer sledeče komponente:

- razvojna skupina s pripadajočo organizacijsko strukturo
- čas, ki nam je na razpolago za nalogo
- spisek opravil, kot osnovno vodilo za planiranje
- spisek že obdelanih in standardi za zajemanje in označevanje tistih pojmov, na katere naletimo med delom (katalog pojmov - Data Dictionary)
- standardni postopki in oprema za izdelavo, shranjevanje in predajo dokumentacije
- računalniški sistem na katerem delamo in sistem, za katerega razvijamo

2. KOMPONENTE RAZVOJNEGA PROCESA

2.1. Razvojna skupina

Osnova za uspeh projekta je dobra razvojna skupina. Razvoj informacijskega sistema, pa tudi samo razvoj posameznih delov za tak sistem, zahteva organizirano sodelovanje velikega števila različnih profilov delavcev. Informacijski sistem predstavlja samostojno proizvodno dejavnost, ki je po svojem ustroju analogna katerikoli industrijski panogi. Ta dejavnost se navadno deli v tri dele. Prvi del predstavlja proizvodnja, drugi del so razne strokovne službe in tretji del, ki nas trenutno najbolj zanima je razvoj. Razvoj je najčešče organiziran tako, da se deli v razvojne skupine po posameznih projektih, ter v skupine specialistov, ki sodelujejo s projektnimi skupinami. Posebne skupine specialistov skrbe na primer za skladiščenje podatkov, za varnost podatkov, za daljinske obdelave, za testiranje programov itd. Skupina za vzdrževanje lahko spada v razvoj ali v proizvodnjo. Projektne skupine naj bodo sestavljene iz vodje projekta, predstavnika uporabnika odgovarjajočega dela informacijskega sistema, organizatorjev in programerjev. Slika 3 predstavlja možno organizacijo projektne skupine.

2.2. Spisek opravil pri uvajanju informacijskega sistema

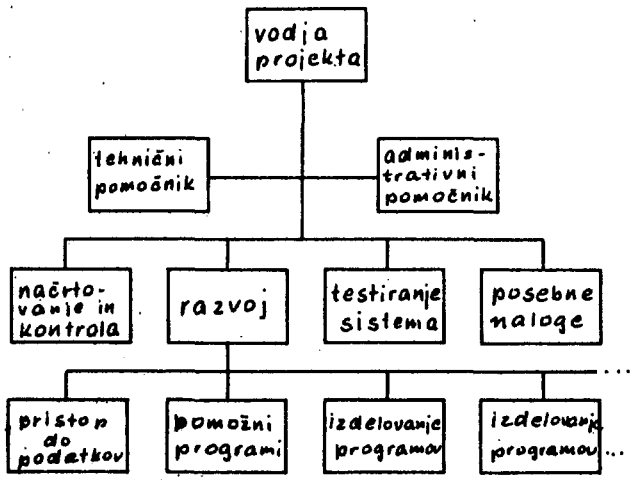
Uvajanje računalniških obdelav poteka po fazah (slika 2) in v okviru teh po opravilih. Skušajmo našteti vsaj najvažnejša opravila.

2.2.1. Sprožitev razvojnega postopka

Postopek za uvedbo neke obdelave podatkov naj načeloma sproži strokovna služba, ki obdelavo potrebuje. Pripravi naj ustrezen dokument in določi predstavnika v razvojni skupini.

2.2.2. Določitev potreb obdelave

Prva in prvenstvena naloga je postaviti in omejiti cilje, katerim naj zadovolji obdelava. Dobre in realne cilje je mogoče postaviti le na osnovi dobrega poznavanja strokovne in izvedbene problematike. Opis projekta mora cilje definirati v oprijemljivi obliki. Določiti je treba rok in pogoje za realizacijo.



Slika 3 Primer organizacije razvojne skupine

2.2.3. Načrtovanje in spremljanje projekta

Načrt izvedbe mora vsebovati idejno rešitev vsaj v najbolj grobih potezah. Oceniti je treba potrebno število in usposobljenost izvajalcev. Sledi ocena potrebnega časa in stroškov. Predvideti je treba datum pričetka. Določiti je treba postopek spremljanja in glavne kontrolne točke. Z načrtom je treba seznaniti naročnika in predvidene izvajalce. Če je načrt sprejemljiv je treba dokumentirati potrebne odobritve, ali drugačne odločitve.

2.2.4. Splošna organizacija

Zbiranje vseh obstoječih dokumentov in informacij o delovanju obstoječega sistema je prvi korak v novi sistem. Na osnovi zbranih informacij se izdela načrt bodočega obtoka podatkov. Celoto se razbije v manjše module. Pripravijo se predlogi za nove dokumente. Posebno pozornost moramo posvetiti izgledu in vsebini novih poročil. Določiti je treba nove potrebe po strojni in izdelani programski opremi. Izdelati je treba čim bolj podroben načrt za nadaljne izvedbene faze. Ponovno je treba predvideti, kako bomo kontrolirali porabo časa in sredstev, roke in kvaliteto. Še enkrat se prepričamo, če predvidena rešitev naročniku ustreza, sicer opravimo potrebne

spremembe in dopolnitve.

2.2.5. Izdelava organizacijskih podrobnosti

Napočil je čas dokončne izdelave. Celo nalogo razdrobimo na primerne dele. Za vsak del ali modul najprej določimo, kaj od njega pričakujemo, nato urotovimo kje dobimo vhodne podatke in nato še, kakšne pomožne podatke potrebuje modul interno. Podatke razporedimo v primerne logične skupine, ki jih imenujemo segmente. Za vse podatke določimo simbolična imena, ki jih takoj vnesemo v katalog. Določimo fizično organizacijo podatkov in dostop do podatkov, nakar pristopimo k izdelavi datotek za testiranje. Že v tej fazi predvidimo tudi varovanje podatkov s shranjevanjem v varovalne datoteke (backup).

2.2.6. Programiranje

Vodja programerjev skrbi predvsem za uspešno napredovanje dela v skladu s postavljenim planom. Programerji najprej skrbno prouče zahteve modulov. Programiranje bo najbolj uspešno, če najprej napišejo vse potrebne komentarje in šele nato programske ukaze. Programer skrbi za to, da se vsi izdelani in preskušeni programi čim prej znajdejo v ustreznih programskih knjižnicah.

2.2.7. Testiranje sistema

Najbolje je, da obstoji za testiranje sistema posebna skupina za to specializiranih programerjev in organizatorjev. Za testiranje sistema je potrebno pripraviti nove datoteke. Zagotoviti je treba vse procedure, ki so potrebne za redno delo.

2.2.8. Instaliranje

Za instaliranje sistema je potrebno poleg postopkov in programov zbrati tudi vse potrebne podatke. V mnogih primerih je to najtežje delo. Največkrat pa podatki že obstoje v strojno čitljivi obliki, a v neki drugačni organizaciji. V vsakem primeru morajo biti podatki zbrani in prečiščeni pred začetkom rednih obdelav, oziroma predajo.

2.2.9. Predaja

Pred dokončno uvedbo nove obdelave je potrebno še enkrat pregledati postavljene termine in oceniti, kako so dosegljivi. Uvedba nove obdelave je pot brez povratka. Po predaji bo za obdelavo skrbel uporabnik, medtem ko se bo razvojna skupina razšla. Uporabnik potrebuje pri uvajanju pomoč, ki jo lahko nudi le razvojna skupina. Važne točke so:

- kako pripraviti podatke
- kako naročiti obdelave
- kakšni so časi in stroški obdelav
- kako varovati podatke in
- katera zanimiva poročila je še mogoče dobiti iz zbranih podatkov

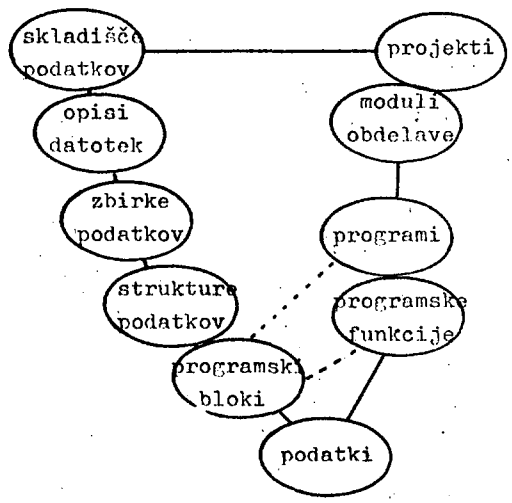
4. KATALOG POJMOV

Dokumentacija mora nastajati ves čas razvojnega procesa. Kot prva potreba po dokumentiranju se v fazi analize projekta pojavlja potreba po zajemanju pojmov. Če želimo opise pojmov uspešno uporabljati v celotnem razvojnem procesu, jih moramo že ob prvem pojavu zajeti v strogo standardizirani obliki. Če delamo tako, lahko vsak pojem tudi uspešno opišemo že ob prvem stiku z njim.

Najvažnejše formalnosti, ki jih moramo opraviti čim preje so:

- opredelitev pojma
- poimenovanje s simboličnim imenom ali šifro in vključitev v katalog
- šele po teh dveh korakih sledi potreben opis v čim bolj jedrnati in na čim več mestih uporabni obliki

V okviru delovne enote informacijskega sistema morajo obstojati standardi, ki omogočajo, da posamezne pojme čim bolj enolično razdelimo v posamezne razrede. Primer take razdelitve pojmov kaže slika 4.



Slika 4 **Primer** opredelitve pojmov informacijskega sistema

Katalog pojmov v okviru informacijskega sistema je analogen katalogu surovin, polizdelkov in izdelkov proizvodne delovne organizacije. Tako kot ta, tudi obdelava podatkov ne more poslovati brez dobre evidence materiala v delovnem postopku. Vodenje kataloga je preprosto, če je obseg majhen. Pri večjih informacijskih sistemih pa je to vodenje dokaj velike in zapletene zbirke podatkov z mnogimi medsebojnimi odvisnostimi. Na razpolago so programi (7), ki omogočajo interaktivne posege v katalog preko terminalov, kot tudi direktno vključevanje v katalogu zbranih podatkov v programe.

4. STANDARDI ZA DOKUMENTIRANJE

Standardi za dokumentiranje se neposredno navezujejo na katalog pojmov. Najbolje je, da že od začetka zasledujemo cilj čim bolj neposredne uporabe v katalogu zbranih pojmov za potrebe programiranja in da so pripadajoči opisi v skladu z zahtevami izbranega programskega jezika. Za izčrpno dokumentacijo je le malokdaj potrebno več kot to, da vsak pojem opišemo v skladu s potrebami programskega jezika ob vključitvi najnujnejših komentarjev, ki jih dovoljuje skoro vsak programski jezik. Za opise posameznih pojmov lahko predvidimo posebne obrazce, ali posebne formate na zaslonu terminala. Najvažnejše, na kar moramo paziti pri projektiranju obrazcev ali formatov je preglednost. To dosežemo z uporabo principov za načrtovanje obrazcev in poročil znanih pod imenom "Information Mapping" (8). Določen podatek naj se nahaja vedno na istem mestu, uokvirjen ali na drug način izrazito ločen od ostalega teksta. Vsak pojem lahko opišemo s hierarhično zgrajenim opisom na enem samem listu. Tak opis ima navadno tri stopnje:

- naslov
- kratek opis
- podroben opis z referencami na nižji nivo če je to potrebno

Zato naj vsak opis obsega načeloma le en list ali zaslon na terminalu. Ta opis se nato preko kataloga ali direktnih referenc navezuje na nadrejene in podrejene nivoje. Prav tako so zgrajene sestavnice v proizvodnih dejavnostih.

Zasnova sistema naj poteka vedno od zgoraj navzdol. Ob zasnovi naj se formira katalog pojmov. Tako lahko posamezne pojme razdelimo v podrobno obdelavo večim sodelavcem. Vsak dokument naj jasno izraža osnovno filozofijo: kaj, od kod, kako.

Na podoben način je zgrajena na primer IBM-ova HIPO dokumentacija (Hierarchy, Input, Processing, Output), kjer tudi vsak list podaja osnovno misel, v tem primeru hierarhijo, vhod, obdelavo, izhod. Kljub upoštevanju vseh teh načel in ob vsej tej dodelanosti pa na koncu vse te obrazce lahko smatramo le kot pomožno sredstvo, medtem ko resnične dokumente predstavljajo šele do kraja obdelani in preverjeni računalniški izpisi, ker šele ti odražajo dejansko stanje dokumentiranosti projekta. Obenem predstavlja računalnik idealno sredstvo za komuniciranje med člani razvojne skupine.

Postopek dokumentiranja je lahko v vseh fazah razvojnega procesa enak. Vse pojme lahko sproti opredeljujemo, vključujemo v katalog in dokumentiramo v čim bolj dokončni obliki. Seveda se večkrat pojavi tudi potreba po dopolnjevanju. Izogibati pa se moramo spremembam, ker imajo te lahko zelo neprijetne posledice. Sistem moramo zgraditi tako, da ostane odprt za dopolnitve, ne da bi bilo pri tem potrebno kaj spreminjati (open ended). Posebno pozornost je treba posvetiti enoličnosti in dokončnosti identifikacije in specifikacij, jasnosti in enostavnosti opisov in preglednosti oblike, kot je to opisano pri obravnavanju obrazcev.

Med delom obstojata dve stopnji dokumentiranja: dokumentacija v delu in prevzeta dokumentacija. Na računalniku se prva vodi v privatnih delovnih datotekah, druga pa v vsem dostopnih knjižnicah. Seveda mora biti jasno določeno, kdo lahko prenaša podatke iz enega področja v drugo.

5. ZAKLJUČEK

Nastanek opisane metodologije je bil povezan predvsem s težnjo po dvigu produktivnosti programerjev, vendar se jo da s pridom uporabiti tudi na ostalih razvojnih področjih. Zahteva interaktivni terminal in prostor na mediju za shranjevanje podatkov. Tako jo je mogoče uporabiti če že ne na vsakem obstoječem vsaj skoro na vsakem novo nabavljenem sistemu za obdelavo podatkov.

6. SEZNAM LITERATURE

1. H.D.Mills:Mathematical Foundations for Structured Programming,IBM Corp.Gaithersburg Md.,FSC 72-6012,February 1972
2. J.D.Aron:Characteristics of the Program System Life Cycle,Tech.Rep.,IBM Corp. Gaithersburg Md.,November 1973
3. W.P.Stevens,G.J.Myers and L.L.Constantine, Structured Design,IBM Sys.J,Vol.13,2,1974
4. M.Jackson,Data Structure as a Basis for Program Design,State of Art Conferenoe on Structured Programming,London 1975
5. J.D.Aron,The Program Developement Process, Addison-Wesley 1974
6. W.B.Cammack,H.J.Rodgers:Improving the Programming Process,Tech.Rep.,IBM Poughkeepsie Laboratory,October 1973
7. DB/DC Data Dictionary User's Guide, IBM H20-9083
8. R.E.Horn:Information Mapping,Datamation, January 1975
9. HIPO-A Design Aid and Documentation Technique, IBM C20-1851

konferenca spin 1978

Prva medvladna konferenca o strategiji in politiki na področju informatike (SPIN: Strategies and Policies for Informatics), ki jo je sklical generalni direktor UNESCO, je bila v Malagi (Španija) od 28. avgusta do 6. septembra 1978. Konferenco so organizirale UNESCO in IBI v skladu z 2133. resolucijo 19. seje generalne skupščine UNESCO in z 7. resolucijo 8. seje skupščine IBI. Cilji konference so bili:

- izmenjava izkušenj o strategiji in politiki na področju informatike, predvsem izkušenj, ki izhajajo iz razvoja domačih kapacitet in optimalnega izkoriščanja obstoječih virov,
- določiti načine in sredstva, s pomočjo katerih bo informatika lahko prispevala ekonomskemu, socialnemu in kulturnemu razvoju in napredku z ozirom na specifične potrebe dežel v razvoju,
- določiti predpogoje za razvoj strategije in politike na nacionalnih nivojih,
- zastaviti akcijski program za mednarodno sodelovanje na področju informatike.

Osrednje teme delovnega dokumenta, ki so služile kot izhodiščne točke za razpravo so bile:

- trenutna situacija v svetu na področju informatike in pregled napovedi o razvoju in uporabi informatike,
- dosedanje izkušnje v pripravi in izvajanju nacionalnih strategij,
- osnovni predpogoji za učinkovito uporabo informatike (razvoj kadrov, osvojitev računalniške tehnologije - industrijske kapacitete, raziskovalno delo in razvoj, informacije o informatiki),
- aplikacije informatike (področja aplikacij, razvoj domačih kapacitet pri uporabi računalništva, informatika in socialno-kulturni razvoj),
- regionalno in mednarodno sodelovanje (bilateralno - multilateralno),
- mednarodni standardi in patenti,
- institucije za izvajanje resolucij konference.

Konferenci so prisostvovali delegati in opazovalci iz 78 dežel članic UNESCO, ter predstavniki nekaterih organizacij Združenih narodov, mednarodnih vladnih in nevladnih organizacij ter strokovnih združenj (ITU, IFIP, WIPO, OECD).

Delo konference se je vršilo na plenarnih sejah in v komisijah. V komisiji I. so se obravnavali problemi z osnovno tematiko: osnovni predpogoj za učinkovito uporabo informatike. V komisiji II. aplikacije informatike. Delo po komisijah je bilo sklenjeno 31. avgusta. Zapisniki in priporočila pripravljena na sejah komisije I. in II. so bila predložena in sprejeta na plenarni seji dne 4. septembra 1978.

Največ pozornosti na plenarnih sejah konference je bilo posvečeno problemu nacionalnih strategij in politik na področju informatike. Če povzamemo razpravo o izvajanju in pripravi nacionalnih strategij lahko zapišemo naslednje:

Nacionalno strategijo na področju informatike opredeljujejo: sistematični koherentni in splošni principi ter standardi in cilji, ki usmerjajo delovanje in funkcioniranje računalniškega in informacijskega okolja. Splošno pravilo za formulacijo nacionalne oz. globalne strategije se sestoji v določevanju temeljnih usmeritev in ciljev, identifikaciji potreb na področju komunikacij, definiciji prioritet uvajanja računalništva, racionalizaciji in organizaciji obstoječih sistemov, krmiljenju in spremljanju akcij za uresničitev ciljev strategije ter njihovem ovrednotenju glede na zastavljene cilje. Formulacija nacionalne strategije na področju informatike je enako težaven postopek tako za razvite kot za nerazvite dežele. Cilji strategije zahtevajo usklajevanje s cilji in potrebami širših družbenih struktur ter s cilji znanstvene in ekonomske politike v deželi. Izdelava splošne politike je izredno dinamičen proces, zato ima tudi nacionalna strategija na področju informatike evolutiven značaj. Globalno nacionalno politiko na področju informatike sestavljajo:

- znanstvena in razvojna politika (teoretična in tehnična informatika, programska oprema, načrtovanje in konstrukcija, aplikacije),
- proizvodna politika (produkcija elektronskih komponent, centralnih in perifernih enot, produkcija programske in aparaturne računalniške opreme, vzdrževanje ipd.),
- politika nabave računalniške opreme (nakupovanje, najemanje, servisiranje ipd.),
- vzgoja na področju informatike (izobraževanje strokovnjakov kot so: sistemski inženirji, sistemski analitiki, programerji, operaterji, luknjačice, vzgoja uporabnikov, izobraževanje na univerzah, podiplomski študiji, uvajanje informatike v osnovne in srednje šole, tekoče izobraževanje ob delu, učni programi, nostrifikacija diplom, administrativni izpiti ipd.),
- aplikacije v administraciji, javni upravi, znanstveni dokumentaciji, avtomatizaciji, procesni tehniki ipd.,
- teleinformatika (telekomunikacije, načrtovanje in izdelava računalniških mrež za teleprocesiranje, izdelava standardov in protokolov ipd.),
- politika na področju vzpostavljanja in vzdrževanja velikih bank podatkov (postavitev splošnih identifikatorjev, ekonomičnih bank podatkov, pomoč pri odločanju ipd.),
- politika usmerjanja vpliva informatike v družbi (pospeševanje pozitivnih učinkov in nevtralizacija negativnih učinkov),
- mednarodna politika na področju informatike (uvoz/izvoz aparaturne opreme, prodaja/nakupovanje patentov, licenc, pravic, programska oprema - pomoč in usluge, mednarodno sodelovanje),
- transfer tehnologije in tehnike know-how, bilateralna in multilateralna pomoč, sprejemanje standardov in konvencij.

Realizacija nacionalne strategije in politike na področju informatike je različna, kot je različna sama strategija. V nekaterih deželah ena sama ustanova načrtuje in ureja razvoj informatike v deželi, v drugih deželah so posamezni organi

odgovorni za posamezne sektorje nacionalne politike na področju informatike in njihovo skupno delovanje usklajujejo koordinacijska telesa. Socialne, kulturne, ekonomske in tehnološke tradicije dežele določajo najprimernejši vzorec nacionalne politike. Pri tem imajo izkušnje drugih dežel izredni pomen.

Glede kategorije in nivoja ustanov in teles za izvajanje nacionalne politike lahko identificiramo več kategorij. Predvsem so tu državne ustanove kot je primer v Argentini, Iranu in Alžiriji. Državna telesa teh dežel centralizirano usmerjajo razvoj informatike tako v javni upravi kot v ostalih dejavnostih. Druga kategorija ustanov so nacionalni oz. državni računski centri, ki poleg računalniških storitev za javno upravo nudijo še konsultantske usluge, usluge na področju izobraževanja, raziskovanja in izdelovanja različnih študij. Nacionalni računski centri tega tipa obstajajo v Iraku, na Filipinih in v Tunisu. Tretja kategorija ustanov so nacionalne agencije za nabavo aparaturne in programske opreme, računalniške opreme ter opreme za vzdrževanje računalniških sistemov. Ustanove tega tipa so: Central Computer Agency v Veliki Britaniji in National Bureau of Standards v ZDA. Zelo pomembno vlogo pri izdelavi in realizaciji nacionalnih strategij v posameznih deželah imajo tudi samostojne raziskovalne organizacije kot je: Gesellschaft für Mathematik und Datenverarbeitung v ZRN, National Computer Centre v Veliki Britaniji, Institut de Recherche d'Informatique et d'Automatique v Franciji. Osnovna dejavnost teh institutov je raziskovalno delo ter vzgoja kadrov za področje informatike. Poleg tega te institucije intenzivno sodelujejo pri razvoju računalniške industrije. Podobno vlogo v okviru SR Slovenije ima tudi Institut J.Stefan.

V okviru razprav na plenarnih sejah je bila poudarjena potreba po večjem mednarodnem sodelovanju zlasti na področjih vzgoje in izobraževanja, izmenjave informacij o novih dosežkih in izkušnjah, standardizaciji proizvodov in postopkov, vzpostavljanja bolj korektnih odnosov med dobavitelji in kupci računalniške opreme in uslug, urejanju dostopa do večjih bank podatkov, organiziranju transfera tehnologije, razvoju projektov socialnega in izobraževalnega značaja (zdravje, vzgoja) ipd.

Osrednja tema razprave v komisiji I so bili osnovni predpogoji za učinkovito uporabo informatike, razvrščeni v naslednje točke: razvoj kadrov, osvojitev računalniške tehnologije oziroma razvoj proizvodnih kapacitet, raziskovalno delo in razvoj, informacije o informatiki. Vsi govorniki na sejah komisije I, so bili enotni glede temeljnega predpogoja za učinkovito uporabo informatike - vzgoje kadrov. Pomanjkanje delovne sile za različna področja informatike glede na potrebe posamezne dežele je pereče vprašanje skoraj v vseh deželah, zlasti v deželah v razvoju. Vse napovedi govorijo o vse večjih potrebah kadrov, ki se glede na profil razlikujejo od dežele do dežele z ozirom na stopnjo razvitosti. Širjenje uporabe računalnikov ima enako stopnjo rasti v razvitih in nerazvitih deželah. Širjenje in ustanavljanje izobraževalnih ustanov za vsa področja informatike je zelo neenakomerno. V okviru razprav je bilo nekajkrat poudarjeno, da je vzgoja in izobraževanje o uporabi računalniške tehnologije v procesu ekonomskega in socialnega razvoja prioritetna zahteva v vseh deželah v razvoju. Pospešeni izobraževalni programi pripravljeni s strani proizvajalcev ne zadoščajo potrebam posameznih dežel. *Hitre spremembe v računalniški tehnologiji narekujejo* nenehno dopolnilno izobraževanje, zaradi tega, ker tekoče tovrstno znanje hitro zastara. Veliko število dežel je podprlo predlog dežel članic IBI o ustanovitvi Mednarodnega centra za informatiko, t.j. posebne organizacije za izobraževanje strokovnjakov in učiteljev za področje informatike. Center bi poleg tega še koordiniral zbiranje in izmenjavo mednarodnih izkušenj s področja učnih programov, metodologije poučevanja, načrtovanja nacionalnih politik na področju izobraževanja ipd. Vzporedno s tem je bila izražena želja o ustanavljanju regionalnih centrov za informatiko s podobnimi nalogami. Veliko število dežel je podprlo predlog, tako da je bilo na plenarni seji sprejeto priporočilo, ki je uvrščeno v poročilu konference pod številko KI/11 kot sledi:

- upoštevajoč napredek na področju informatike in potrebe po izobraževanju strokovnjakov in uporabnikov,
- upoštevajoč omenjene resurse v deželah v razvoju za ustanovitev lastnih izobraževalnih struktur in raziskav na področju informatike in
- upoštevajoč potrebe po strokovnjakih in raziskovalcih v deželah v razvoju, ki potrebujejo šolanje, vsklajeno z okoljem, v katerem delajo se priporoča:

1. ustanovitev centralnega in regionalnih izobraževalnih in raziskovalnih institucij, ki bi opravljale naslednje naloge:
   a) izobraževanje strokovnjakov in raziskovalcev na področju informatike,
   b) pospeševanje raziskovalnega dela na področjih, ki so skupnega interesa za dežele v razvoju,
   c) oblikovanje sodobnih metod za poučevanje na področju informatike,
   d) ustanovitev dinamičnih skupin z nalogo priprave kratkih tečajev za določene profile računalniških strokovnjakov s ciljem obnovitve in osvežitve znanja, zlasti *za dežele brez lastnih izobraževalnih institucij*,
2. ustanavljanje fondov za pomoč raziskovalnim in izobraževalnim institucijam (obstoječim in v ustanavljanju) za širjenje lastnih dejavnosti.

Mednarodni izobraževalni center in ostali regionalni centri bi se ustanovili v sodelovanju z UNESCO in IBI. V okviru *te razprave je bila dana podpora dosedanji aktivnosti* UNESCA na področju izobraževanja v obliki tečajev, seminarjev, poletnih šol ipd. Poleg tega so bili poudarjeni vloga in prispevki, ki so jih dosegli na področju vzgoje in izobraževanja mednarodna strokovna združenja kot je IFIP, IFAC, ipd.

Druga zanimiva tema v okviru dela komisije I. je bila razprava o osvajanju računalniške tehnologije. Zelo pogosto je bilo poudarjeno, da je razvoj nacionalne računalniške industrije, ki bi zagotovila opremo za vse aspekte informatike, skoraj nemogoč. Računalniška tehnologija je zelo draga in zahteva velike investicije za razvoj in raziskave. Za ilustracijo navajamo le podatke o investicijah za razvoj v ZDA in Veliki Britaniji. Vlada ZDA letno financira razvoj računalniške industrije s 500 milj. $. Tvrdka IBM letno porabi 890 milj. $ za razvoj in to je le 7% od letnega obrata sredstev. Tvrdka ICL troši letno 36 milj. $ in to je 9% od letnega obrata sredstev. Zahodna Evropa je v obdobju 1971-1975 financirala razvoj računalniške industrije s 100 milj. $ letno.

Z druge strani obstajajo strateški, ekonomski, tehno-ekonomski in socialno-kulturni razlogi, ki govorijo v prid nacionalni računalniški industriji. Računalniška industrija (proizvodnja komponent, integriranih vezij, konstrukcija aparaturne opreme, produkcija programske opreme in ostale usluge) ima izreden tehnološki pomen, predvsem zaradi stranskih učinkov: pospešuje razvoj elektronske, mehanske in telekomunikacijske industrije. Iz razprave je bilo razvidno, da je večina dežel podprla razvoj lastne računalniške industrije na določenih področjih z ozirom na napovedi o vse večji rasti računalniške industrije v svetu (v tem trenutku je računalniška industrija *na drugem mestu po obsegu v svetovni ekonomiji*) in z ozirom na njen strateški pomen. Poudarjeno je bilo, da je računalniška industrija še eno področje, kjer je regionalno sodelovanje zelo zaželjeno in sicer posebej pri razvoju programske opreme ter produkciji in preskrbi z računalniškim potrošnim materialom (kartice, papir, magnetni trakovi ipd.).

Raziskovalno delo s področja računalništva v glavnem poteka v okviru privatnih laboratorijev multinacionalnih družb in v

okviru raziskovalnih institutov ter na univerzah. Med razpravo o raziskovalnem delu in razvoju je bila poudarjena potreba po širšem aktivnem sodelovanju med raziskovalnimi institucijami. S tem v zvezi je bila priporočena izdelava študij, ki naj jih pripravi UNESCO in ki bodo ugotovile potrebe po različnih raziskavah ter določile nadaljnje usmeritve v skladu z:
- nacionalnimi resursi (pomen informatike za posamezno deželo),
- aktivnostmi, ki obetajo večjo znanstveno-tehnološko in sociološko - ekonomsko produktivnost,
- možnostmi združevanja raziskav v okviru mednarodnih institucij z upoštevanjem bodočih tržnih potreb.

V okviru dela komisije I. so bila še posebej zanimiva za raziskovalce poročila o dostopnosti informacij s področja informatike. Dostop do obstoječega znanja ima izredni pomen za vse razvojne in raziskovalne procese, ki vse pogosteje uporabljajo najnovejše znanstvene in tehnološke dosežke. Vse dežele, posebej pa dežele v razvoju so izrazile željo za aktivnejšo udeležbo v mednarodnih komunikacijskih procesih ob uporabi lastne infrastrukture. Posebno pozornost temu problemu so posvetili tudi v Organizaciji združenih narodov, kjer so bili razviti naslednji informacijski sistemi: UNISIST za znanstveno tehnološke informacije, DEVISIS za razvoj, LATFIS za tehnološki transfer, INIS za nuklearne podatke, AGRIS za poljedelstvo, AIDS za dokumentacijo ipd. Namen vseh naštetih informacijskih sistemov je, da omogočajo učinkovit in široko zastavljen dostop do informacij uporabnikom iz vsega sveta. Glede informacij o informatiki je UNESCO zastavil dolgoročni program o realizaciji informacijskega sistema z nazivom World Information System on Informatics (WISI). Ker je ta program precej obsežen in se počasi realizira, je IBI prevzela nalogo ustanavljanja informacijsko-dokumentacijskega servisa, ki bo bolj praktičen in ki bo služil deželam, ki so bile do sedaj prikrajšane glede dostopa do informacij o informatiki. Sistem AIDS (Automatic Informatics Documentation Service) načrtuje in realizira IBI ter napoveduje začetek delovanja sistema do konca leta 1978. Sistem bo sestavljen iz mikrofilmske dokumentacije, sistema za avtomatsko shranjevanje in iskanje informacij ter iz podatkovne baze namenjene nacionalni politiki na področju informatike. V drugi fazi izdelave sistema je napovedana vsebinska razširitev, zlasti na tehničnih področjih informatike - materialna oprema, programska oprema, komunikacije ipd. Sistem bo povezan z odgovarjajočimi institucijami v posameznih deželah, medtem ko bo koordiniranje, shranjevanje, obdelava informacij in vzdrževanje, izvajala strokovna služba IBI. Delo na WISI se bo nadaljevalo. Posebej je predvidena večja učinkovitost dela na specializiranem tezaurusu, ki bo kompatibilen s splošnimi standardi UNISIST-a ter na področju komunikacij in interakcij s sistemom UNISIST.

Na sejah komisije II. je bila podana celovita slika aplikacij informatike v različnih deželah. Informatika je prodrla v skoraj vse dejavnosti posameznih dežel: znanstvene, administrativne in industrijske. Najrazličnejša področja znanosti, medicine, tehnologije, vzgoje in raziskav so bila ilustrirana z velikim številom aplikacij: poljedelstvo, meteorologija, hidrologija, knjižničarstvo, znanost ipd. Posebej je bila poudarjena vloga informatike v avtomatizaciji industrijskih procesov. Sodobne aplikacije informatike sestavljajo sliko, ki natančno ponazarja vse probleme, ki izhajajo iz prehoda k masovni uporabi informatike.
Med temi aplikacijami so bile posebej navedene naslednje:
- administracija v upravi,
- robotika, ki predstavlja razvite oblike avtomatizacije v industriji in uporabe umetne inteligence,
- načrtovanje s pomočjo računalnika,
- elektronska pošta,
- elektronski prenos bančnih transakcij,
- računalniško vodeni pouk,
- računalnik na domu.

V vseh sodobnih aplikacijah je bila vedno poudarjena vloga nove tehnologije, uporabe mikro procesorjev, ki odpirajo skoraj neomejene možnosti glede razširitve uporabe ter pocenitve proizvodnje in uslug. Poleg razvoja mikro-procesorske tehnike je bil poudarjen proces širjenja uporabe računalniških mrež, padanje cene računalniških pomnilnikov ter povečanje pomena programske opreme. Padanje cene materialni opremi je povzročilo nove trende v arhitekturi računalnika. Pri nakupu opreme ni več vitalnega pomena moč hardware-a, ampak konfiguracija opreme, ki najbolj učinkovito opravlja naloge, katerim je namenjena. V preteklosti so prevladovale razprave med strokovnjaki in uporabniki o administrativnih in poslovnih aplikacijah, danes se te razprave gibljejo v prid uporabnikom. Ta fenomen je rezultat padanja cene materialni opremi ter evolucije načrtovalnih metod. Razvoj nove in cenene mikro-elektronike omogoča proizvodnjo in uporabo materialne opreme za določeno nalogo (kot je bančni terminal). Takšna oprema je enostavna za uporabo in ne zahteva posebno znanje iz računalništva. Druga možnost razvoja na področju informatike se ponuja z razvojem teleprocesiranja. Teleprocesiranje je eno od najbolj kritičnih točk razvoja. Glede napovedi razvoja v naslednjih desetih letih z upoštevanjem pomena za širšo javnost je bilo rečeno naslednje:

Za obdobje 1977-80: pocenitev pomnilnikov, pocenitev majhnih računalniških enot ter izredno povečanje njihovih zmogljivosti, pocenitev sistemov za razpoznavanje črk ter povečanje natančnosti, velik napredek na področju razpoznavanja ljudskega govora in povečanje raznolikosti izhodnih naprav (grafika ipd.).

Za obdobje 1980-87: razvoj naprav za shranjevanje ogromnega števila podatkov (računalniki s superprevodnimi elementi), nadaljnje zmanjševanje cene in velikosti procesorjev vzporedno z izboljšavami glede zanesljivosti, "žepne računalnike za žepno ceno", kibernetične sisteme (robote) in dramatični napredek pri povezovanju računalniških sistemov z uporabo satelitov, laserjev, valovodov ipd.

Na koncu razprave o razvoju informatike je bilo opozorjeno zlasti na naslednje probleme:
- vsako uvajanje informatike v deželi zahteva velike investicije, zato je za vsako novo aplikacijo, posebej v deželah v razvoju zaželjeno, da je progresivna in natančno analizirana glede stroškov in glede uslug, katere ponuja,
- pri nabavi računalniških sistemov za nove aplikacije je izrednega pomena ocenjevanje kritične velikosti sistema z ozirom na potrebe in na optimalno koriščenje njegovih zmogljivosti,
- pri razvoju novih aplikacij je treba upoštevati telekomunikacijske usluge, ki so bistvenega pomena za nekatere uporabe, kot so knjižnice, dokumentacijski centri, rezervacija letalskih vozovnic, bančnih transakcij ipd.
- pri razvoju informatike na splošno v svetu je treba upoštevati še naslednje probleme: zaščita avtorskih pravic pri izdelavi programske opreme, mednarodno uporabo kanalov za pretok podatkov po možnosti z najnižjimi možnimi stroški in z največjo možno svobodo uporabe, standardizacija izdelkov na področju informatike ter njihova izmenjava na mednarodnem nivoju.

Problemi standardizacije izdelkov ter meddržavnega pretoka podatkov so sprožile zelo dinamično razpravo. Na zahtevo več delegacij in po posvetovanju s predsedstvom konference (Steering Committee) je bila uvrščena razprava o meddržavnem pretoku podatkov v okviru dela na plenarni seji.

Problemi, ki izvirajo iz meddržavnega pretoka podatkov so zelo kompleksni in različni in lahko imajo zakonske, socialne, ekonomske in politične implikacije:
- glede zakonov (koordinacija zakonov v posameznih deželah),
- na socialnem nivoju (zaščita osebne svobode, razdelitev funkcij v okviru meddržavnega pretoka podatkov, multinacionalne tvrdke si lahko zadržijo pomembnejše funkcije z

ozirom na možnosti, katere daje tehnologija),
- na ekonomskem nivoju (pretok podatkov lahko vpliva na plačilne bilance),
- na političnem in kulturnem nivoju (lahko pride do problema zasvojenosti dežel ipd.).

Posebej je bilo opozorjeno na dejavnosti multinacionalnih družb, ki imajo za cilj odvračanje iniciativ glede investiranja na tem področju. Z ozirom na tokove razprave je bila izrečena podpora projektom, ki prispevajo k razvoju razumevanja tako političnih kot tehničnih in organizacijskih problemov. Med takšnimi projekti je bil omenjen projekt Evropske računalniške mreže (EIN), kjer sodeluje tudi skupina iz Instituta J.Stefan ter IIASANET, ki je bil razvit pod vodstvom Mednarodnega instituta za uporabo sistemske analize IIASA. Projekt IIASANET ima za cilj povezovanje računskega centra Instituta IIASA z računskimi centri podobnih institucij ter nabiranje izkušenj v zvezi priprav za realizacijo komunikacijskega omrežja Vzhod/Zahod.

Poudarjena je bila potreba o zaščiti finančnih in tehničnih interesov majhnih uporabnikov v okviru mednarodnih podatkovnih mrež, ter potrebe po večjem sodelovanju glede uporabe lastnih in regionalnih komunikacijskih satelitov, omejevanje rasti privatnih računalniških mrež in izboljšave podrejenega položaja, v katerem se nahajajo dežele v razvoju.

Kot rezultat razprave je bilo pripravljeno naslednje priporočilo (komisija II: o končnem dokumentu):

Z ozirom na prepričanje, da je razvoj meddržavnega pretoka podatkov eno do najpomembnejših vprašanj glede razvoja informatike in meddržavnih povezav in upoštevajoč na sedanje neravnovesje v okviru teh pretokov, ki lahko vplivajo na državno suverenost in negotovost posamezne dežele, če je narava podatkov zaupljiva, ne glede ali se nanaša na državo ali kakšno podjetje, enako kot informacije glede privatnega življenja posameznikov, z ozirom na delokalizacijo podatkov in upoštevajoč ekonomske reperkusije teh pretokov, posebej glede plačilnih bilanc, konferenca priporoča:
široko zastavitev moči in naporov na mednarodnem nivoju s ciljem:
- raziskave problemov posameznih dežel glede pretoka podatkov,
- razločitev med vprašanji: pretok podatkov in prenos podatkov,
- določitev pogojev in principov o prostem pretoku podatkov in o potrebni regulativi,
- preprečevanje redukcije političnih, ekonomskih, socialnih in zakonskih dimenzij problema v enostavni in enostranski mednarodni instrument, ki bi reguliral pretok podatkov le glede enega od teh aspektov,
- podpore delu skupine v EEC, OECD ipd. z upoštevanjem posameznih okolnosti in stanj razvoja v različnih geografskih zonah.

Drugo pereče vprašanje v okviru dela komisije II, ki je sprožilo dinamično razpravo, je standardizacija izdelkov. Glede standardizacije na področju materialne opreme je Mednarodna organizacija za standarde (ISO) pripravila večje število priporočil, ki obravnavajo predvsem elektro-mehanske naprave in periferne medije (kartice, magnetni trakovi, kasete ipd.). Problemi standardizacije v komuniciranju stroj-človek, so še vedno nerešeni. Na tem področju je standardizacija najbolj problematična. Proizvajalci računalniške opreme imajo interese v tem, da svobodno spreminjajo tehnološke specifikacije zaradi vsiljevanja svojih standardov kupcem. Tako je kompatibilnost med različnimi sistemi onemogočena in s tem so uporabniki in kupci postavljeni v podrejen položaj. Poleg tega se delo na standardizaciji s področja informatike nadaljuje in ureja s pomočjo javnih mednarodnih sporazumov. Pomembni koraki v tej smeri so bili narejeni s strani CCIT v letu 1976, ki je sprejel priporočila v obliki t.k. X-25 protokola, ki zajema standardni interface za javna omrežja. Na regionalni osnovi je Evropska ekonomska skupnost v letu 1977 sprožila projekt o politiki na področju informatike. V projektu so zajeta priporočila deželam članicam o spoštovanju splošno sprejetih standardov v skupnosti.

Na področju programske opreme že obstajajo aspekti standardizacije. Eden od najbolj zanimivih so programirni jeziki. Mednarodni komite CODASYL je pripravil poročila o standardizaciji COBOLa. Priporočila je sprejel in založil ISO. Ostali najbolj univerzalni programirni jeziki se razlikujejo v odvisnosti od stroja, na katerem so implementirani. To še vedno predstavlja pomembno oviro pri prenosu programov in v osamosvojitvi uporabnikov napram proizvajalcu. Druga stran standardizacije programske opreme se nanaša na programske pakete. Zadnje aktivnosti v zvezi z računalniškimi mrežami dajejo nov značaj temu problemu, posebej standardizacija opisov programov in podatkov o njihovi dosegljivosti. Tehnologija mrež omogoča tudi neizkušenemu uporabniku dostop do najrazličnejših računalniških pripomočkov z uporabo ene in iste računalniške opreme. Prav nestandardizacija dostopov v mrežo terminalov, virtualnih terminalov in računalnikov predstavlja pomemben problem. V razpravi je bila dana podpora prizadevanjem in priporočilom o standardizaciji, katere je pripravila Evropska ekonomska skupnost.

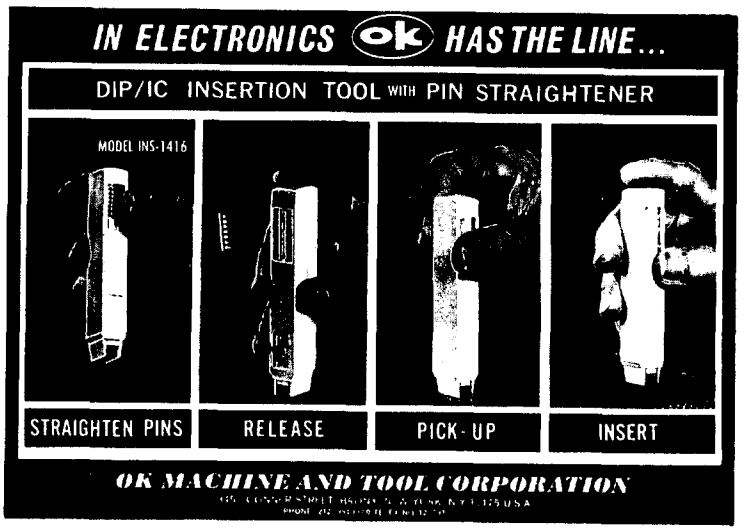
Računalniška tehnologija ni edino področje, kjer je zaželjena standardizacija. Standardi so zaželeni tudi v okviru meddržavnega pretoka podatkov. Promet s podatki postaja ena od pomembnejših svetovnih industrij, pri tem pa se podatki pretakajo nekontrolirano preko državnih mej. V okviru razprave je bila večkrat izražena tudi zaskrbljenost glede razdelitve držav na "bogate s podatki" in "revne s podatki". Postavljeno je bilo vprašanje, ali je sploh izvedljivo mednarodno soglasje, ki bo reguliralo meddržavni pretok podatkov v smislu spoštovanja mednarodnih standardov in zaščite osebnih podatkov. V okviru te razprave je bilo sprejeto naslednje priporočilo (komisija II/2):

UNESCO, IBI in ostale mednarodne organizacije naj pripravijo in preučijo možnosti o:

- ustanovitvi mednarodnega fonda projektov in programov za splošne aplikacije informatike za različne dežele in
- izvajanju spodbudnih akcij za aktivno udeležbo dežel v delu ISO v zvezi s standardizacijo na področju informatike.

B. J.-B.-Dž.

OK MACHINE AND TOOL CORPORATION / 3455 CONNER STREET, BRONX, NEW YORK 10475, U.S.A.
Phone: (212) 994-6600 • Telex: 12-5091 ; Telex: 23 2395

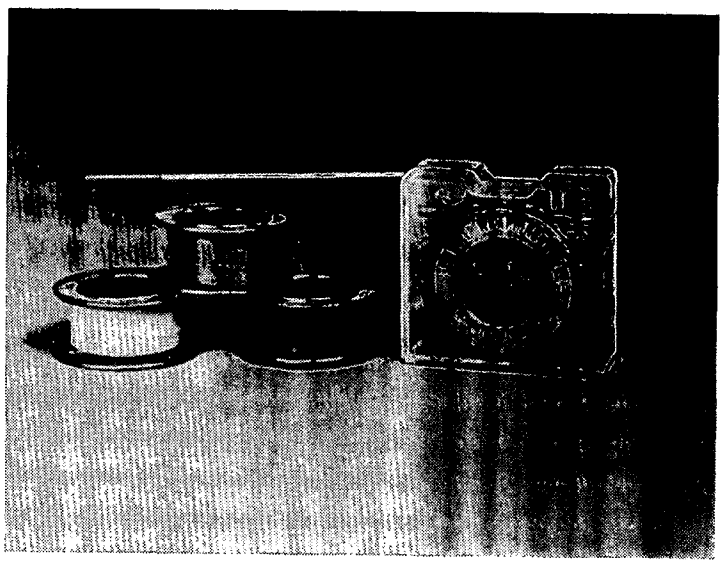


INS 1416 je orodje za vstavljanje 16- ali 14-kontaktnih integriranih vezij v podnožja ali izvrtane luknje tiskanega vezja. Posebnost je zoženi profil, ki omogoča vstavljanje vezij, ki so na plošči tesno skupaj. V držalo sta vrezani vodili za ravnanje deformiranih kontaktov integriranega vezja. Vezje potisnemo v vodilo, pri tem se poravnajo deformirani kontakti, izvlečemo pa ga s pritiskom držala navzdol.

TULEC Z ZAMENLJIVIMI KOLUTI ŽICE ZA OVIJANJE

Prednost tulca je v tem, da ne potrebujemo dodatnega orodja za rezanje žice in snemanje izolacije. Iz tulca se izvleče želena dolžina žice, katero se vstavi v vodilo na vrhu tulca. S pritiskom na vgrajeni gumb, se žico odreže. Potem se žico potegne preko posebnih vgrajenih vilic, ki snemajo izolacijo; isti postopek se ponovi tudi z drugim koncem žice.

Tulec vsebuje kolut s 15 m dolgo žico tipa 30 AWG (0,25 mm), s posebno industrijsko izolacijo in posrebreno bakreno žico. Na razpolago so koluti z belo, modro in rdečo izolacijo.



Ko pišete proizvajalcu, omenite časopis INFORMATICA

novice in zanimivosti

Dinamični 16 k x 1-bitni RAM podjetja Mostek je v prodaji; večje količine teh integriranih vezij pa bodo proizvedene že v marcu leta 1979. Ta RAM je v 16-nožičnem integriranem vezju, napaja se z eno samo napetostjo (5 V) ter doseže čas dostopa 80 ns; ima lastno osveževanje, porabi 320 mW in doseže čas cikla 100 ns. Cena enega vezja bo med 80 in 100 US $.

APŽ

Kvaliteta izdelkov podjetja Motorola.

Izdelki Motorole so bili že od nekdaj kvalitetni in zanesljivi. Te lastnosti so se pokazale tudi pri mikroračunalniški družini MC 6800, ki je postala ena izmed najbolje prodajanih družin. Sedaj pa se je pojavil na vidiku novi 8-bitni procesor MC 6809 s 16-bitnimi lastnostmi, ki je kritično ocenjen tudi v zadnji izdaji publikacije Microcomputer Analysis, ki izhaja mesečno. Pri razvoju 6809 zagovarja Motorola tri cilje: navzgornjo razširljivost , boljše specifikacije (dodatni registri, izboljšano kodiranje in adresiranje) in "tehniko programske opreme". Ta zadnji, za procesorje neobičajni terminus kaže na to, da so pri zasnovi procesorja sodelovalu tudi načrtovalci programske opreme.

APŽ

64 k RAM podjetja IBM.

Medtem ko se proizvajalci polprevodniške tehnologije napenjajo, da bi končno izdelali svoje 64 k dinamične pomnilnike tipa RAM, je podjetje IBM najavilo svoj računalnik z lastnimi, takimi integriranimi vezji. Ta vezja bodo vgrajena v sistem za porazdeljeno obdelavo 8100, ki ga bo moč dobiti v letu 1979. Najnovejša konfiguracija sistema 8100 uporablja 48 takih integriranih vezij (z diskom, tremi tiskalniki in šestimi terminali).

APŽ

Hitra integrirana vezja za krmiljenje gibkih diskov.

Ker je MOS tehnologija prepočasna je podjetje Signetics najavilo bipolarni krmilnik za gibki disk, ki je dvakrat hitrejši od zahtev za dvojno gostoto podatkov na disku, tj. s hitrostjo 8 mikro sekund na zlog. Tako je 8 x 330 prvi krmilnik tretje krmilniške generacije in ima podatkovno separacijo in predkompenzacijo že v vezju. Vsi bistveni elementi vezja so pod programsko kontrolo (tudi formati in njihove različice).To vezje je predvideno za povezavo z bipolarnim mikro procesorjem družine Signetics. Celoten krmilniški sistem bo sestavljen le iz 9 integriranih vezij in ocenjevalna konfiguracija bo kupcem na voljo v začetku leta 1979. Zanimivo je tudi, da sta že pred Signeticsovim sporočilom podjetji Western Digital in NEC (Japonska) najavili integrirani vezji krmilnikov za gibki disk z dvojno gostoto.

APŽ

Statični ROM.

Podjetje Siemens je na razstavi Electronica v Münchnu pokazalo statični 32 k MOS pomnilnik tipa ROM z oznako SAB 8332, s časom dostopa 450 ns in s 330 mW disipacije. Organizacija pomnilnika je 4096 krat 8 bitov v 24-nožičnem ohišju ter s 5 V napajanjem.

APŽ

Prodor računalnikov Amdahl/Fujitsu.

Evropa je postala novo področje za plasman računalnikov z licenco Amdahl, ki tehnološko razširjajo, dopolnjujejo in nadomeščajo dosedanje velike sisteme podjetja IBM. Tako je British Airways po petnajstih letih uporabe IBM sistemov naročil nov sistem za 14 M$ s štirimi računalniki tipa Amdahl 470, britanski Ford pa 470 V/5. Zaradi povečanega povpraševanja po Amdahlovih sistemih na evropskem tržišču je Amdahl odprl novo tovarno v Dublinu.

APŽ

Jezik Pascal na mikro računalnikih.

Pascal je bločno strukturiran programirni jezik v stilu jezika Algol. Programi v Pascalu so sestavljeni iz dveh delov: naslovni del poimenuje program in specificira spremenljivke programa, temu delu pa sledi telo, ki ga imenujemo blok. Blok je razdeljen na šest segmentov: prvi štirje določajo označitve, konstante, podatkovne tipe in spremenljivke; peti segment poimenuje in predhodi dejanski proceduri ali funkciji. Zadnji, šesti segment je stavčni ter vsebuje izvedljivi kod za imenovano funkcijo ali proceduro.

Označitve istovetijo stavke in tako lahko stavke navajamo. Konstante enačijo števila z imeni, ki se v programu pojavljajo. Podatkovnih tipov je več in definirati je mogoče strukturirane tipe, ki vsebujejo polja, zapise, množice in zbirke. Vsaki imenovani spremenljivki mora slediti njen tip. Procedure se lahko pojavljajo v procedurah (rekurzija) in njihovi stavki morajo začenjati z rezervirano besedo "begin" ter končevati z besedo "end". Definirani so operatorji za množenje, deljenje, seštevanje, odštevanje, za logiko in relacije, dopustni pa so tudi različni krmilni stavki.

Jezik Pascal je bil razvit v Švici pred 10 leti ter je dobil svoje ime po francoskem matematiku Blaise Pascalu (1623-1662). Medtem je bil preizkušen na 60 različnih računalniških družinah, v zadnjem času pa se je pojavil tudi na mikro računalnikih. Univerza v Kaliforniji (San Diego) (okrajšano UCSD) je izdala vrsto pascalskih prevajalnikov in interpreterjev, in sicer za PDP-11, LSI-11, 8080, Z-80, 6800, 9900 in 6502. Jezik Pascal se najprej prevede v vmesni kod, ki se imenuje P-code; ta kod se potem interpretira na različnih mikro računalnikih. To je tudi glavni vzrok hitrega razvoja UCSD Pascala na mikro računalnikih. UCSD operacijski sistemi vsebujejo prevajalnik za Pascal z razširitvami za nize, diskovne zbirke, interaktivno grafiko in za sistemsko programiranje ter še urejevalnik, upravljalnik zbirk, sporočevalnik napak itn. Prej ko slej bo mogoče dobiti Pascal tudi v pomnilniku tipa ROM. Podjetje Western Digital pa izdeluje že 16-bitni mikro računalnik, ki interpretira t.i. P-code z materialno opremo (v obliki štirih integriranih vezij). Očitno je prišla ponudba UCSD Pascala v pravem času in proizvajalci mikro procesorjev in uporabniki so jo sprejeli z odprtimi rokami.

Prvi je sprejel UCSD Pascal AMI, ki ga namerava ponuditi kot del mikroprocesorskega razvijalnega centra MDC-100. Pascal je namreč močnejši pri definiciji podatkovnih struktur, kot so jeziki Basic, Fortran, PL/1 in zbirni jeziki. Z uporabo pascalskih krmilnih konstruktov pa je mogoče skrajšati čas programiranja za faktor 4 do 5 ter znižati stroške vzdrževanja programov.

Uporabniška skupina za jezik Pascal pri univerzi države Minnesota (Minneapolis) združuje 2600 uporabnikov iz 41 držav. Uporaba Pascala je v zadnjih dveh letih narasla za faktor 260 (primerjava števil pascalskih poslov na

računalnikih).

Tudi DEC (Elektrotehna) ima svojo posebno interesno skupino za Pascal, ki je bila ustanovljena pred dvemi leti ter ima 600 članov. Obstaja vrsta implementacij za Pascal na DEC-ovih računalnikih. Trenutno se prodaja dvoprehodni kompilator za operacijske sisteme RSX-11/M, IAS in RSTS, ki bo razširjen za RT-11. Pascal bo uporabljan tudi na novem DEC-ovem sistemu VAX-11/780.

Pascal se izdatno uporablja tudi v podjetju Texas Instruments, kjer so ga izbrali kot ugodnejšo alternativo v primerjavi z jezikom C in Bliss. TI Pascal bo doživel nekatere spremembe, tako da bo bolj uporaben za programiranje kot za poučevanje. K TI Pascalu bo dodana še aritmetika s fiksno vejico in decimalna aritmetika ter ustreznejše procesiranje polj. V TI je že 30 do 40 % novih programov napisanih v Pascalu. V tej zbirki so programi za razvoj programske opreme, kompilatorji, subrutinske knjižnice ter pripomočki za sistemsko programiranje. V prodaji je Pascal za miniračunalnik z diskom DS 990, kjer so kompilator in moduli sistema tudi napisani v Pascalu. V Pascalski izvirni kod pa je moč vgnezditi tudi fortranske rutine.

Vendar ima Pascal tudi svoje šibke točke. Pascalska definicija ne dopušča ločene kompilacije, kar pomeni, da mora biti pascalski program vselej kompiliran v celoti; če dodamo novo rutino, se mora ta pojaviti v prvotnem izvirnem kodu in potrebna je ponovna kompilacija.

Druga ovira je zahteva, da mora biti obseg polja določen v kompilacijskem času. Pascal je zahteven tudi glede preizkušanja tipov: če je spremenljivka v nekem delu programa določenega tipa, potem mora svoj tip obdržati tudi v drugem delu. Če se spremeni obseg polja se to odrazi kot nov tip in kompilator signalizira napako. Težave se pojavijo tudi pri programiranju istočasnih procesov; zaradi tega se pojavlja nov jezik Modula, ki je osnovan na Pascalu in naj bi omogočil programiranje paralelnih procesov.

Novejši pascalski prevajalniki se naštetim težavam izognejo z vstavitvijo nemih naslovov s spremenljivkami, ki bodo uporabljene, ko dodajo dejansko proceduro kasneje. Tudi TI je uvedel izraz za spreminjanje obsega polja brez signalizacije napake. Očitno bodo uvedene nekatere "standardne" razširitve Pascala za vojaške namene (Steelmanova faza).

Kot že omenjeno se je na tržišču pojavil mikro računalnik za Pascal; to je 16-bitni računalnik s skladom, ki interpretira P-code z materialno opremo. Ta računalnik sestavljajo 4 integrirana vezja podjetja Western Digital. V tem računalniku poteka prevajanje Pascala v dveh fazah. Najprej se izvirni pascalski kod prevede v vmesni kod, ki ga imenujemo P-code, potem pa se P-code izvaja na gostiteljskem stroju z interpretiranjem. Sam interpret je idealizirani skladni stroj ter ga je moč udejaniti tudi s programsko opremo.

Z ustreznimi rutinami se tako dani procesor pretvori v psevdo stroj, katerega vodni jezik je P-code. Štiri integrirana vezja sestavljajo tako psevdo stroj P-stroj, ki ima lastnost hitre izvedbe pascalskega programa ter reducirane pomnilniške zahteve. Tak sistem je tako že prikrojen za P-code iz UCSD Pascala. Vsako od štirih integriranih vezij ima 40 nožic: imamo aritmetično vezje z mikroukaznim dekoderjem, aritmetično in logično enoto ter notranjo zbirko registrov; drugo vezje je mikro zaporednik, ki vsebuje mikroukazni dekoder, dele krmilnika, ukazni števnik ter V/I krmilno logiko; preostali vezji imata dva 512 krat 22-bitna pomnilnika tipa ROM, v katerih so shranjeni mikro ukazi in mikro diagnostika.

Štiri opisana integrirana vezja oblikujejo 16-bitni računalnik, ki deluje v povezavi s skladom. Računalnik naslovi 128 k zlogov ali 64 k besed pomnilnika, ima štiri prekinitvene nivoje ter možnost krmiljenja direktnega pomnilniškega dostopa. Vključena je tudi aritmetika s plavajočo vejico, in sicer v aritmetičnih ukazih. Sistem se napaja s tremi napetostmi (+5, +12 in -12 V), ima taktnik s frekvenco 3 MHz (s štirimi neprekrivajočimi fazami); vsi V/I signali se vodijo preko vmesnikov treh stanj in dodati je mogoče še dva dodatna pomnilnika tipa ROM za prilagoditvene (firmne) razširitve.

Western Digital proizvaja tudi ploščo, na kateri je razen 4 osnovnih integriranih vezij še 32 k zlogov RAM pomnilnika, dvoje RS-232 vrat (110 do 19 200 Baud), dvoje paralelnih vrat ter krmilnik za gibki disk z enojno ali dvojno gostoto. Priključiti je mogoče 4 gibke diske z možnostjo DMA. Sistem ima operacijski sistem s kompilatorjem za Pascal in Basic itn. Cena štirih integriranih vezij znaša $ 195. Naslov proizvajalca: Western Digital Corp., 3128 Red Hill Ave., Box 2180, Newport Beach, California 92663, U.S.A.

APŽ

Iz Elektronikschau, avstrijskega strokovnega časopisa za elektroniko, povzemamo tole novico:

Eigene Mikrocomputersysteme

bringt der jugoslawische Konzern Iskra im nächsten Jahr auf den Markt. Die in Zusammenarbeit mit dem Institut "Josef Stefan" in Ljubljana entwickelten µPs stellen einen Gesamtwert von 210 Mio. Dinar dar. Durch Inbetriebnahme weiterer Werke will Iskra ab 1982 Mikrocomputer im Wert von jährlich 1,13 Mrd. Dinar erzeugen, wobei keine ausländischen Lizenzen verwendet werden.

Elektronik Schau

Tečaji Intel.

U Velikoj Britaniji su vrlo popularni radni tečaji o mikroprocesorima koje priređuje glavni Intelov ured za Veliku Britaniju (Broadfield House, 4 Between Towns Road, Oxford). Svaki tečaj je pripremljen tako da slušaocima omogući samostalno projektiranje materialne i programske opreme za mikroprocesorsko bazirane proizvode. Tečaje vode stručnjaci sa velikim razvojnim iskustvima. Detaljne informacije pružaju prospekti koje dobavlja Intel, telef. (0865) 771431.
Na izboru su sljedeći tečaji: Oktobar: 8080/8085 mikro računari (9 do 12); Mikroračunarski razvojni sistemi (19 do 20); 8080/8085 (23 do 26). Novembar: Uvod u mikroprocesorje (13 do 15); 8086-16 bitni mikro računar (20 do 23); 8048 mikroračunalniška familija (77 do 29). Decembar: PL/M visoki programski jezik (4 do 6); 8080/8085 mikro računari (11 do 14).

MK

64 K-bitne memorije.

Kompanije cijelog svijeta koje se danas bave poluprovodničkom tehnologijom očekuju tržište za više od četvrt biliona dolara u 1980 godini.

Texas Instruments je upravo serijom uzoraka naznanio dolazak na tržište prvog 64 K x 1 bit dinamične memorije sa napajanjem sa samo jednim izvorom od 5 V. Memorija se poavljuje pod oznakom TMS 4164. U prvom kvartalu 1979 godine se očekuje serijska proizvodnja. Čip je 16-pinski u dva reda (dual-in-line) sa rasporedom nožica po Jedec Standardu dopuštajući kompatibilnost sa 16 K dinamičnim memorijama.

Druga i treća generacija dinamičnih memorija koje su trenutno u upotrebi djeluju pod tri naponska različita izvora energije: + 12, + 5 i - 5 V. Japanski proizvođači poluprovodničkih memorija zahtijevaju + 7 V i - 2 V za njihove 64 K dinamične RAM-e.

Napajanje sa samo jednim izvorom od + 5 V reducira potrošnju energije te povećava brzinu. Iz istog razloga je povaćan imunitet na šum tako da je povećana sigurnost.

Vrijeme pristupa za 4164 je u granicama od 100 do 150 ns, sa minimalnim vremenom ciklusa od 200 do 250 ns. Sipanje energije je 200 mW maksimalno ili 3 uW po bitu. U odnosu na 462 mW sipanje energije za 16 K dinamične RAM-e pri 375 ns vremenom ciklusa je sipanje energije 4164 za 60 % smanjeno uz poboljšano vrijeme ciklusa te kvadrupliranom gustinom bitova.

4116 zahtijeva 4 ms maksimalni period osvježavanja. Osnovni vremenski zahtijevi za osvježavanje se ne razlikuju mnogo od onih pri 16 K memorijama u cilju kompatibilnosti. Dodatni zahtijevi krmilnog kola za 64 K memorije su 8 bitni brojač za osvježavanje te 8 bitni multiplekser.

Fotomaske za produkciju 4164 memorija će biti urađene uz pomoć uređaja sa elektronskim ulazom čime je moguće dobiti geometrijske oblike do 0.25 u.

MK

Optično povezani računari.

Siemens će prvi u svijetu upotrijebiti vodilo (bus) iz optičnih vlakana za automatizaciju industrijskih procesa kao u slučaju za 28 visokih peći pri Bruckhausen Steel Works, Thyssen AG u Zapadnoj Njemačkoj.

Dva Siemens 310 procesna računara će biti upotrijebljeni za centralnu kontrolnu stanicu sa vizualnim kolor terminalima kao I/O napravama. Mikroprocesorske stanice su povezane međusobno te sa procesnim računarskim sistemom preko optičnog valovodnog kružnog vodila (bus) koji ima mogućnost veliko brzine prenosa podataka te potpunu neosjetljivost na elektromagnetne smetnje. Koncept prenosa podataka upotrebom vodila sa optičnim vlaknima i mikroračunarskim stanicama je razvijen u Fraunhefer Gesellschafts Institute for Information Processing (IITB).

MK

Računarski sistemi Amdahl.

Sljedeća poslovna vijest najbolje predstavlja tehničko-komercialne osobine velikih računarskih sistema Amdahl:

British Airways je naručila četiri sistema Amdahl 470 u vrijednosti 7 miliona funti, koji će zamijeniti četiri IBM računara i obezbijediti 24-satni, 7 dana-sedmično servis u srcu British Airways računarske mreže koja se proteže cijelim svijetom. Ta mreža se trenutno sastoji iz preko 200 računara sa 3500 vizualnih jedinica i 1000 teleprintera instaliranih u 650 gradova u 150 zemalja. Amdahl sistemi su bili odabrani zbog njihove kompatibilnosti sa trenutnim sistemom te potencijalnih cijena/osobine prednosti. Cijena Amdahl sistema je značajno niža od cijene alternativnih sistema. British Airways će ovim izborom uštediti 1 milion funti.

MK

Memorije sa magnetnim mjehurićima.

Memorije sa magnetnim mjehurićima kapaciteta 4 miliona bita, kasnije i 16 miliona bita će biti razvijeni u sljedećih pet godina, izjavio je John L. Archer, vođa poslova oko memorija sa magnetnim mjehurićima pri Rockwell International.

Već dostupna tri produkta na području memorija sa magnetnim mjehurićima, 256 K-bitni element (čip), 1M-bit memorijski modul i sistem za razvoj programske/materialne opreme, su rezultat desetogodišnjeg istraživanja.

Memorije sa magnetnim mjehurićima su memorijski elementi koji drže informaciju i onda kada je napajanje isključeno, a po dimenzijama se mogu uspoređivati sa poluprovodničkim memorijskim elementima. Ovaj najnoviji tehnološki novitet na području memorijskih elemenata obećava da će zamijeniti širok segment klasičnih memorijskih elemenata koji se upotrebljavaju u računarima i mikroprocesorsko baziranim produktima.

Pored osobine da zadržavaju informaciju i poslije isključenja napajanja, imaju ove memorije dodatne ugodne osobine kao što su male dimenzije te niska potrošnja energije u odnosu na mehaničke memorije (trake i diskovi). Sistemi sa memorijama sa magnetnim mjehurićima mogu u prostoru od nekoliko kubnih stopa imati kapacitet ravan onom kojega može imati "puna soba traka i diskova". Pored svega potrošnja energije je znatno manja.

Novi elementi su, također, i znatno sigurniji. Kod elemenata sa magnetnim mjehurićima firme Rockwell nastupa greška tek poslije 10^{12} operacija.

MK

literatura in srečanja

Leto 1979

4 - 5 jan. Honolulu, Hawai, ZDA

12 th HAWAII INTERNATIONAL CONFERENCE ON SYSTEM SCIENCES
Organizator: University of Hawai
Informacije: Perry G. Patteson, HICSS-12, Office of Management Programmes, University of Hawai, 2404 Mail Way, Honolulu, HI 96822, USA.

8 - 9 jan. Honolulu, Hawai, ZDA

PACIFIC TELECOMMUNICATIONS CONFERENCE
Organizator: IEEE Communication Society and University of Hawai
Informacije: PTC 79, Social Science Research Institute, University of Hawai, 2424 Maile Way No 704, Honolulu HI 96822, USA.

29 - 31 jan. San Antonio, ZDA

6th ANNUAL ACM SIGACT-SIGPLAN SYMPOSIUM ON PRINCIPLES OF PROGRAMMING LANGUAGES
Organizator: ACM
Informacije: dr. Barry K. Rosen C.S.D., IBM, Thomas J. Watson Research Center, Yorktown Heights, NY 90598 USA.

6 - 8 feb. Dunaj, Avstrija

4th INTERNATIONAL SYMPOSIUM ON COMPUTER SYSTEM MODELLING AND PERFORMANCE EVALUATION
Organizator: IIASA
Informacije: Dr. A. Butrimenko, IIASA, A-2361 Laxenburg, Austria.

12 - 14 feb. Paris, Francija

INTERNATIONAL SYMPOSIUM ON FLOW CONTROL IN COMPUTER NETWORKS
Organizator: IRIA
Informacije: IRIA, Domaine de Voluceau, Rocquencourt, BP5, 78150 Le Chesnay, France.

14 - 16 feb. Toulouse, Francija

1st EUROPEAN CONFERENCE ON PARALLEL AND DISTRIBUTED COMPUTATION
Organizator: AFCET, CNRS
Informacije: AFCET, 156 boulevard Péreire, 75017 Paris, France.

6 - 8 marec, Toulouse, Francija

IFAC/IFORS CONFERENCE ON COMPARISON OF AUTOMATICS AND OPERATION RESEARCH TECHNIQUES APPLIED TO LARGE SYSTEM ANALYSIS AND CONTROL
Organizator: AFCET
Informacije: AFCET, 156 boulevard Péreire, 75017 Paris, France.

14 - 15 marec, Tampa, Florida, ZDA

12th SIMULATION SYMPOSIUM
Organizator: ACM
Informacije: ACM Headquarters, 1133 Avenue of Americas, New York, NY 10036, USA.

19 - 22 marec, Versailles, Francija

WORLD ASSOCIATION FOR MEDICAL INFORMATICS
JOURNEES D'INFORMATIQUE MEDICALE
Organizator in informacije: Secretariat des Journées, WAMI, 74 rue de la Colonie, 75013 Paris, France.

20 - 22 marec Praga, Češkoslovaška

16 SEMINAIRE MEDA: ANALOG AND HYBRID COMPUTERS '79
Organizator: CVTS House of Technology
Informacije: ing. Jiri Kral, CVTS House of Technology, Gorkeho Nam. 23, 11282 Praha 1, CSSR

26 - 29 marec, Jahornia, Jugoslavija

III. BOSANSKO-HERCEGOVAČKI SIMPOIUM IZ INFORMATIKE
Organizator in informacije: Elektrotehnički fakultet Sarajevo, Odsjek za informatiku, 71113 Sarajevo, Toplička bb.

3 - 6 april Bowness-on-Windermere, Velika Britanija

THIRD INTERNATIONAL RESEARCH CONFERENCE ON OPERATIONAL RESEARCH AND MANAGEMENT SCIENCE
Organizator: University of Sussex
Informacije: prof. P. Rivett, Operational Research, Mantell Building, University of Sussex, Brighton BN1 9RF England.

9 - 11 april Amsterdam, Nizozemska

EURO III : 3rd EUROPEAN CONGRESS ON OPERATIONS RESEARCH
Informacije: EURO III, C/O Organisatie Bureau Amsterdam, Europeplein, 1078 GZ Amsterdam, The Netherlands.

23 - 27 Visoke Tatre, Češkoslovaška

ALGORITHMS '79
Organizator: The Czechoslovak Scientific and Technical Society, The Union of Slovak Mathematicians and Physicists, The Institute of Technical Cybernetics of the Slovak Academy of Sciences, Dept. of Cybernetics - EF of the Slovak Technical University.
Informacije: Czechoslovak Scientific and Technical Society, Central Council, Prague 1, PO Box 20, Czechoslovakia.

8 - 10 maj London, Velika Britanija

WORKSPACE
Organizator: Online Conference Limited
Informacije: Online Conference Ltd., Cleveland Road, Uxbridge UB8 2DD, Middlesex, UK.

14 - 18 maj Paris, Francija

SEE-GIEL: COLLOQUE INTERNATIONAL DE COMMUTATION
Organizator in informacije: 11 rue Hamelin, 75783 Paris Cédex 16, France.

16 - 18 maj Stuttgart, ZRN

IFAC WORKSHOP: SAFECOMP - SAFETY OF COMPUTER CONTROL SYSTEMS
Organizator in informacije: VDI/VDE - GMR, C/O M.A. Kaaz, Graf Recke Str. 84, P.O. Box 1139, D- 4000 Dusseldorf 1, BDR.

21 - 22 maj Bari, Italija

IFAC SYMPOSIUM ON CRITERIA FOR SELECTING APPROPRIATE TECHNIQUES UNDER DIFFERENT CULTURAL, TECHNICAL AND SOCIAL CONDITIONS
Organizator: FAST, CSATA
Informacije: IFAC Symposium 1979, c/o FAST, Piazzale

Morandi 2, 20121 Milano, Italia.

21 - 23 maj Ann Arbor, Michigan, ZDA

FOURTH IFIP/IFAC CONFERENCE ON PROGRAMMING RESEARCH AND OPERATIONS LOGISTICS IN ADVANCED MANUFACTURING TECHNOLOGY (PROLAMAT-79)
Organizator: IFIP, IFAC, Computer Automated Systems Association
Informacije: Society of Manufacturing Engineers, 2051 Ford Road, POB 930, Dearborn, MI 48128, USA (Attn. Peter L. Blake, Technical Activities Dept.).

21 - 27 maj Moskva, SSSR

8th IMEKO CONGRESS: MEASUREMENT FOR PROGRESS IN SCIENCE AND TECHNOLOGY
Organizator: IMEKO
Informacije: IMEKO Secretaiat, 1371 Budapest, BP 457, Hungary.

11 - 13 junij Paris, Francija

IFIP INTERNATIONAL CONFERENCE - TELEINFORMATICS 1979
Organizator: AFCET
Informacije: Conference Secretariat, AFCET, 156 boulevard Péreire, 75017 Paris, France.

junij Helsinki, Finska

IFAC/IFIP WORKSHOP ON REAL-TIME PROGRAMMING
Organizator: IFAC, IFIP
Informacije: IFAC Secretariat, c/o EKONO OY, Box 27, SF-00131 Helsinki, Finland.

4 - 7 jun. New York, ZDA

NATIONAL COMPUTER CONFERENCE
Organizator: American Federation of Information Processing Societies, 210 Summit Avenue, Montvale NJ07645, USA.

5 - 7 jun. Barcelona, Španija

CIL 79: CONVENCION INFORMATICA LATINA
Organizator: Asociación de Técnics de Informática (ATI), Cámara Oficial de Comercio, Industria y Navigación de Barcelona, Centro de Estudios y Asesoramiento Metalúrgico (CEAM), Facultad de Informática de la Universidad Politécnica de Barcelona, Feria International de Barcelona (FIB)
Informacije: CIL 79, Pza Comercial 5, Barcelona-3, Spain.

10 - 13 jun. Boston, Massachusetts, ZDA

INTERNATIONAL CONFERENCE ON COMMUNICATIONS
Organizator: IEEE
Informacije: IEEE Computer Society, 5855 Naples Plaza, Long Beach CA 90803, USA.

10 - 16 jun. Praga, Češkoslovaška

2nd IFAC/IFIP SYMPOSIUM ON SOFTWARE FOR COMPUTER CONTROL (SOCOCO 79)
Organizator: IFAC, IFIP
Informacije: IFAC, IFIP Symposium on Software for Computer Control, General Computing Centre of the Czechoslovak Academy of Sciences, 18207 Prague 82, POB 5, Czechoslovakia.

17 - 20 Jun. Denver, Coloradó, ZDA

1979 JOINT AUTOMATIC CONTROL CONFERENCE
Informacije: prof. T.F. Edgar (Prog. Chm), Dept. of Chemical Engineering, University of Texas, Austin, TX 78712, USA.

2 - 6 julij Oxford, Velika Britanija

8th IFAC SYMPOSIUM ON AUTOMATIC CONTROL IN SPACE
Organizator: IFAC
Informacije: Institute of Measurement and Control, 20 Peel Street, London W8 7PD, UK.

16 - 20 julij Graz, Avstrija

6th INTERNATIONAL COLLOQUIUM ON AUTOMATA, LANGUAGES AND PROGRAMMING
Informacije: Prof. H. Maurer, Institut für Informationsverabeitung, Techn. Universität Graz, Steyrergasse 17, A-8010 Graz, Austria.

16 - 18 avg. New, Delhi, India

IFAC: SYMPOSIUM ON COMPUTER APPLICATIONS IN LARGE SCALE POWER SYSTEMS
Informacije: prof. M.A. Pai, Dept. of Electrical Engineering Indian Institute of Technology, Kanpur 208 016, India.

29 - 31 avg. Zurich, Švica

IFAC SYMPOSIUM ON COMPUTER-AIDED DESIGN OF CONTROL SYSTEMS
Organizator: IFAC
Informacije: IFAC Secretariat, c/o EKONO OY, PO Box 27, SF-00131 Helsinki, Finland.

avgust/september Dunaj, Avstrija

CONFERENCE DES NATIONS UNIES SUR LA SCIENCE ET LA TECHNIQUE AU SERVICE DU DEVELOPMENT
Informacije: Secrétariat de la Conférence, Room DC 1148, United Nations, New York, N.Y. 10017, USA

27 - 31 avg. Montréal, Kanada

DIXIEME SYMPOSIUM INTERNATIONAL DE PROGRAMMATION MATHEMATIQUE
Informacije: Secrétariat du Dixième Symposium de Programmation Mathématique, 772 ouest, rue Sherbrooke, Montréal, Québec, Canada

12 sept. Namur, Belgija

INTERNATIONAL SYMPOSIUM ON CYBERNETICS AND SOFTWARE
Informacije: Prof. Tuncer I. Oren, Computer Science Dept., University of Ottawa, Ontarion KIN 6N5, Canada

17 - 20 sept. Berlin, ZRN

MEDICAL INFORMATICS BERLIN 79
Organizator: Online Conference Limited, AMK-Berlin
Informacije: Online Conference Ltd, Cleveland Road, Uxbridge UB8 2DD, Middlesex, UK

24 - 28 sept. Darmstadt, ZRN

5th IFAC SYMPOSIUM ON IDENTIFICATION AND SYSTEM PARAMETER ESTIMATION
Organizator: VDI/VDE-Gesellschaft Mess - und Regelungstechnik
Informacije: IFAC-IDENTIFICATION 1979, c/o VDI/VDE Gesellschaft Mess -und Regelungstechnik, Postfach 1139, D-4000 Düsseldorf 1, BDR.

raziskovalne naloge prijavljene na rss v letu 1978

V tej rubriki objavljamo kratke povzetke raziskovalnih nalog, ki jih financira Področna raziskovalna skupnost za avtomatiko, računalništvo in informatiko, ki so s področja računalništva in informatike.

Naslov naloge: Materialna/programska oprema za računalniško industrijo
Projekt: Računalniška tehnika in proizvodnja
Nosilec naloge: A.P. Železnikar, Institut "J. Stefan", Ljubljana

Program raziskave:

1. Metode za načrtovanje digitalnih (mikroračunalniških) sistemov.
2. Metode za logično in tehnično načrtovanje in za ustrezno dokumentacijo.
3. Metode za sistematičen in organiziran industrijski razvoj programske opreme.
4. Raziskave podsistemov s centralnimi procesorskimi rnotami, statističnimi in dinamičnimi pomnilniki, direktnim prenosom podatkov, prekinitvami, perifernimi vmesniki, časovniki, periferijo itn.
5. Raziskave mikroračunalniških prečnih in rezidenčnih ter inverznih zbirnikov, prevajalnikov in simulatorjev.
6. Raziskave mikroračunalniških monitorjev, operacijskih sistemov in urejevalnikov.
7. Raziskave programskih in direktnih jezikov.
8. Raziskave konfiguracij multimikroprocesorskih sistemov in problematike koordinacije.
9. Družbena in tehnološka izhodišča pri oblikovanju in organizaciji domače računalniške in spremljajoče proizvodnje.

Naslov naloge: Modularna gradnja mikroračunalnikov in programov
Projekt: Individualna naloga
Nosilec naloge: Radovan Tavzes, Institut "J. Stefan" Ljubljana

Program raziskave:

Izdeleli bomo najpomembnejše periferne enote, ki zagotavljajo uspešno vključitev modularnega mikroračunalnika v upravljanje procesov s posebnim poudarkom na njihovi uporabi v industrijskih pogojih.

Izdelali bomo programsko opremo, to je modularni mikroračunalniški operacijski sistem, ki bo zaokrožila modularni mikroračunalnik v instrument procesne tehnike, uspešen v enostavnih primerih industrijskega upravljanja procesov.

Naslov naloge: Prenos gospodarnih in vernih informacij III
Projekt: Računalniška tehnika in proizvodnja
Nosilec naloge: Bogomir Horvat, VTŠ Maribor

Program raziskave:

Naloga predstavlja III. fazo raziskave. V tej fazi bi bil zasnovan in fizikalno realiziran predprocesorski komunikacijski podsestav. Ta podsestav bi bil na eni strani povezan s kanalom na drugi strani s terminalnimi procesorji ali ustreznimi moduli. Zmogljivost, funkcionalnost in zanesljivost bi preverili ob realizaciji.

Naslov naloge: Podsestavi avtomatskih industrijskih merilnih sistemov
Projekt: Avtomatski industrijski merilni modularni sistemi
Nosilec naloge: Peter Šuhel, Fakulteta za elektrotehniko, Ljubljana

Program raziskave:

Kompleksna študija industrijskih merilnih sistemov bo zajela:

- avtomatski merilni procesni sistem za kontrolo kvalitete, sortiranje in nadzor serijskega proizvodnega procesa elementov

Študija vhodnih adapterjev:
- transducerji
- problemi vhodnikov in prenosnih poti

Programiranje merilnega sistema:

- mikroprocesor

Hitro, precizno merjenje parametrov:

- merjenje izmeničnih napetosti in toka
- merjenje faznega kota

Registriranje in prikaz izmerjenih parametrov:

- CRT terminal
- magnetni spomin-floppy disk

Obdelava izmerjenih parametrov:

- mikroprocesor

Sistemi merilnih robotov:

- študija inteligence sistema ter osnovni principi

Naslov naloge: Upravljanje materialnih tokov v DO
Projekt: Informacijski sistemi
Nosilec naloge: Franc Šetina, Krka, Novo mesto

Program raziskave:

Naloga je enoletna.
Program raziskave obsega analizo materialnih tokov, informacijsko analizo in sintezo - modeliranje formalnega informacijskega sistema glede na planiranje, izvrševanje in kontrolo.

Naslov naloge: Računalniško upravljanje pilotnega obrata oddelka za kemijo
Projekt: Računalniška avtomatizacija industrijskih procesov
Nosilec naloge: Čedo Jakovljević, Krka, Novo mesto

Program raziskave:

Definiranje parametrov, merilnih in regulacijskih zank na osnovi zahtevane tehnologije. Določiti je treba glavne značilnosti obravnavanih procesov, meje sistema in podsistemov, ugotoviti značilnosti in cilje osnovnih operacij tehnološkega postopka.

Razvoj in implementacija programskega sistema. Izdelava programskih modulov in povezovanje v sistem nadzora na računalnik PDP 1140. Definiranje strukture modulov, razvoj regulacijske strategije, nadzornih funkcij.

Definiranje, izbor in instalacija opreme. Določitev funkcij vmesnika, naročilo dodatne opreme, instalacija in testiranje vmesniškega sistema.

Naslov naloge: Računalniška analiza signalov v nevrofiziologiji
Projekt: Individualne naloge
Nosilec raziskave: Bogdan Oblak, Medicinska fakulteta in Klinični center, Ljubljana

Program raziskave:

A) Okvirni program je delno dopolnjen program prejšnjih let:
a) Dopolnitev programske zbirke za vodenje meritev, zbiranje in za analizo bioloških signalov KOMB 7.
b) Določitev standardov statokinzigrama.

B) Podrobni program:
a) Dopolnitev programske zbirke KOMB 7.
   1. izdelati gonilne programe za programirano draženje v zanki
   2. izdelati programe za avtomatsko klicanje podprogramov po časovnem zaporedju
   3. dopolniti programe za programirano draženje
   4. razširitev programov za matematično-statistične analize.
   5. dokončna izdelava dokumentacije programske izbirke.
b) Analiza statokinezigrama:

V tem letu nameravamo izdelati programe za analizo statokinezigrama pri električnem draženju.

Naslov naloge: Razvoj metode, ki omogoča razpoznavanje nepopolnih prstnih odtisov in sledi
Projekt: Individualne naloge
Nosilec naloge: Ludvik Gyergyek, Fakulteta za elektrotehniko, Ljubljana

Program raziskave:

- določitev in računalniška implementacija postopkov, ki naj omogočijo razpoznavanje takih vzorcev prstnih odtisov, ki so s stališča razpoznavanja slabe kvalitete.
- določitev in računalniška implementacija postopkov, ki odpravijo vpliv translacije in rotacije pri delno odtisnjenih prstnih odtisih.
- določitev in računalniška implementacija razpoznavalnega postopka delnih prstnih odtisov ter sledi.
- proučitev možnosti grobe razvrstitve vzorcev prstnih odtisov.
- izdelava ustreznih računalniških postopkov.
- preizkus razpoznavalnih postopkov.

Naslov naloge: Premična telemehanska postaja
Projekt: Sistem daljinskega nadzora in upravljanja
Nosilec naloge: Nikola Pavešić, Fakulteta za elektrotehniko Ljubljana

Program raziskave:

1. Idejna rešitev premične telemehanske končne postaje, ki omogoča neposredno povezavo med službenim avtomobilom in računalniškim sistemom IBM 370/...
2. Obisk tovarne Motorola v Wiesbadnu, ZRN.
3. Realizacija testne aparaturne opreme premične telemehanske postaje in vmesnika za priključitev na UKV premično postajo.
4. Realizacija testne aparaturne opreme nepremične telemehanske postaje in vmesnika za neposredno priključitev na računalniški sistem IBM 370/...
5. Izdelava programske opreme, ki omogoča neposredno komunikacijo med službenim vozilom in računalniškim sistemom IBM 370/...
6. Preizkus zgrajene aparaturne in programske opreme.

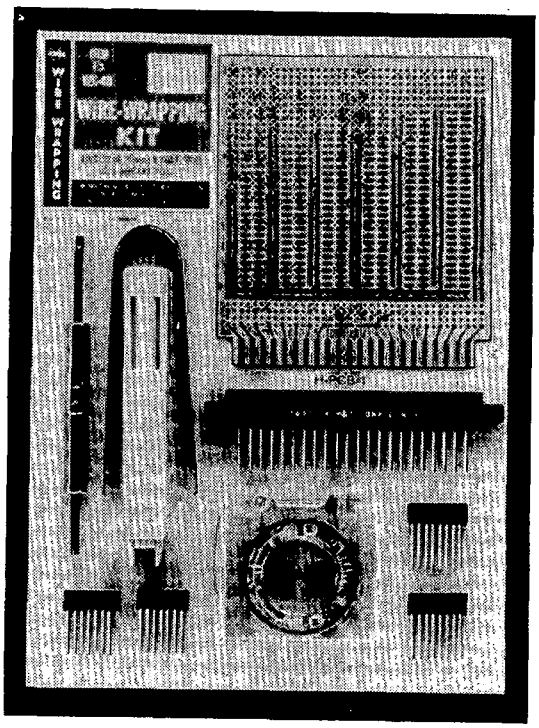
Naslov naloge: Mikroprocesor v diagnostiki digitalnih sistemov na osnovi MSI in LSI
Projekt: Računalniška tehnika in proizvodnja
Nosilec naloge: Ljubo Pipan, Fakul. za elektrotehniko, Lj.

Program raziskave:

- kritična analiza nekaterih deterministčnih in naključnih metod odkrivanja prisotnosti in izvora napak s stališča uporabnosti pri digitalnih sistemih, ki vsebujejo strukture MSI in LSI
- opredelitev funkcij, ki jih pri izvajanju zgornjih metod v smislu lastne diagnostike sistema prevzema mikro procesor kot gradnik v sistem vgrajene diagnostične instrumentacije
- določitev parametrov, ki vplivajo na izbiro ustreznega mikroprocesorja
- nekateri praktični zgledi realizacije diagnostične instrumentacije na osnovi mikroprocesorja
- kriteriji za diagnostiko sestavljenih digitalnih sistemov in njihov odraz na praktične rešitve.

OK MACHINE AND TOOL CORPORATION / 3455 CONNER STREET, BRONX, NEW YORK 10475, U.S.A.
Phone: (212) 994-6600 • Telex: 12 5091 • Telex: 23 2395

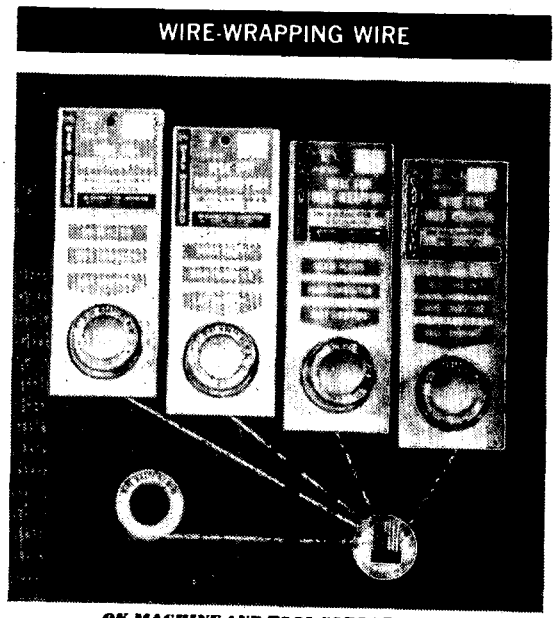
WK-4B WIRE-WRAPPING KIT



OK MACHINE & TOOL CORPORATION
3455 CONNER ST., BRONX, N.Y. 10475 U.S.A.
TELEX 125091

WK-4B OŽIČEVALNA SESTAVLJENKA

Sestavljenka vsebuje orodje in dele, ki so potrebni za izdelavo prototipne ali amaterske ploščice. Deli, ki jih ima sestavljenka, so: univerzalna plošča s tiskanim vezjem, standardni 2 x 22-polni konektor z izvodi za ožičevanje, dve 14-in dve 16- kontaktni podnožji za integrirana vezja z izvodi za ožičevanje, orodje za vstavljanje in izvlačenje integriranih vezij, tulec s 15 m žice in posebno orodje WSU-30, ki je kombinacija orodja za ožičevanje in odvijanje žice na trnih s premerom 0,63 mm; v ročaju je vgrajeno rezilo za snemanje izolacije.



ŽICA ZA OVIJANJE

Žica ima najvišjo industrijsko kvaliteto z oznako AWG 30 (0,25 mm), ki je navita v 15 m zvitkih. Žica je primerna za manjše proizvodne serije, razvojna dela, izdelavo prototipov ali za amaterske projekte. Žica je prevlečena s plastjo srebra in je izolirana s posebno plastjo, ki prenese velike mehanske in električne obremenitve.

Na razpolago so štiri barve izolacije: bela, modra, rumena in rdeča. Žica je navita na 40 mm kolutih, ki omogočajo boljše rokovanje in skladiščenje.

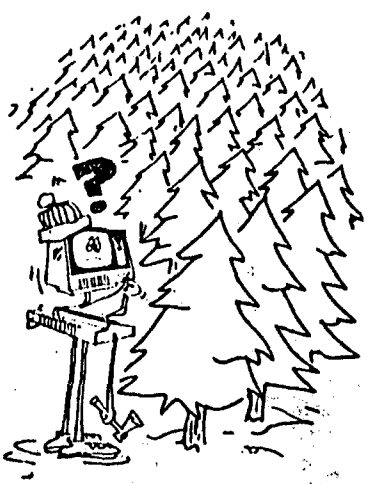
Avtorji in sodelavci

Bojan HLADNIK (1949), diplomiral leta 1973 na Fakulteti za strojništvo v Ljubljani. Magistriral leta 1976 s področja procesne tehnike. Zaposlen v Iskri Elektromehaniki v Kranju - organizacijsko področje. Poleg organizacijsko-operativnega dela se ukvarja z uvajanjem projektnega vodenja, planiranja in operacijskimi raziskavami.

Bojan JARC (1949), diplomiral leta 1973 na Fakulteti za naravoslovje in tehnologijo, smer tehnična matematika. Zaposlen je v organizacijskem področju v Iskri Elektromehaniki v Kranju. Do sedaj se je ukvarjal z reševanjem aplikativnih nalog s področja telefonije, sedaj pa z uvajanjem projektnega vodenja, planiranjem in operacijskimi raziskavami

Andrej JERMAN-BLAŽIČ (1942), diplomiral leta 1965 na Elektrotehnični fakulteti v Beogradu. Več let je bil zaposlen na Institutu J.Stefan v Ljubljani, kjer je delal na raziskovalnih in aplikativnih nalogah na področju prevajalnikov, poslovne informatike in programske opreme za vodenje Telekomunikacijskih procesov. Sedaj opravlja dela in naloge pomočnika predsednika Rep. komiteja za družbeno planiranje in informacijski sistem. Je sekretar Slovenskega društva Informatika in član uredniškega odbora časopisa Informatica.

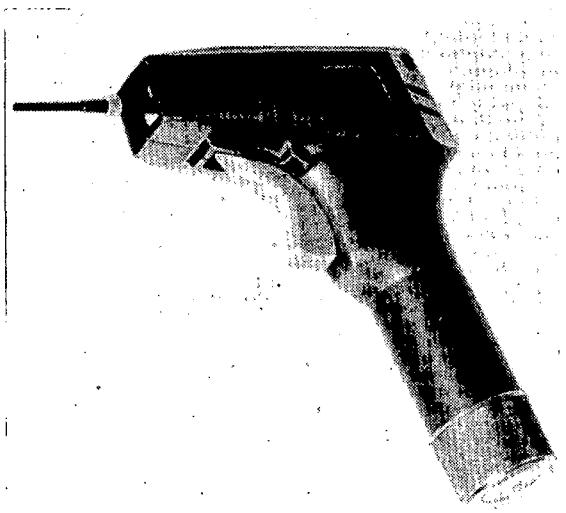
Saš HADŽI (1953), diplomiral leta 1976 na FNT Ljubljana, smer tehnična fizika. Zaposlen na Institutu J.Stefan, na oddelku za računalništvo in informatiko, kjer se ukvarja z mikroračunalništvom.



SREČNO 1979

OK MACHINE AND TOOL CORPORATION / 3455 CONNER STREET, BRONX, NEW YORK 10475, U.S.A.
Phone: (212) 994-6600 • Telex: 12-5091 • Telex: 23-2395

HOBBY-WRAP TOOL



OK MACHINE AND TOOL CORPORATION
3455 CONNER STREET, BRONX, N.Y. 10475 U.S.A. PHONE: (212) 994-6600
TELEX NO: 125091 TELEX NO: 232395

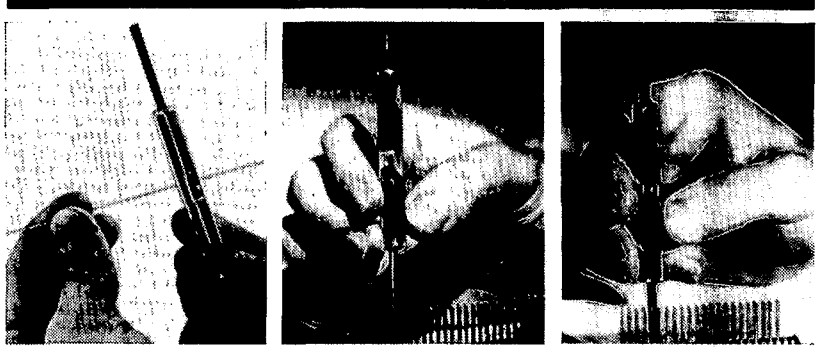
AMATERSKO ORODJE ZA OŽIČEVANJE

Model BW-630 je orodje na baterijski pogon za ožičevanje žice tipa 30 AWG na standardne trne, ki so med seboj oddaljeni 1,65 mm. Orodje je opremljeno s kompletom, ki omogoča izdelavo "modificiranega" načina ožičevanja. Vgrajena je tudi naprava, ki preprečuje nategovanje žice. Konstrukcija je prilagojena delu resnih amaterjev; teža orodja je 40 dkg in se napaja preko standardnih ali akumulatorskih baterij velikosti "C". Ohišje pištole je izdelano iz hrapave površine in zavarovano pred udarci. Baterije niso vključene v komplet.

ORODJE ZA OŽIČEVANJE-ODVIJANJE IN SNEMANJE IZOLACIJE

Ceneno orodje, ki opravlja funkcijo treh orodij, s podobno ceno. Z orodjem je mogoče ožičevati, odvijati in snemati izolacijo, s posebnim rezilom, vgrajenim v ročaj. Orodje primerno za delo z žico tipa 30 AWG (0,25 mm), katero se ovije na standardne (0,6 mm) trne podnožij za integrirana vezja. Uporabe se naučimo v nekaj minutah, žico pa ovijemo v nekaj sekundah ne da bi uporabili spojko. K orodju je priloženo tudi navodilo za uporabo.

HOBBY-WRAP-30



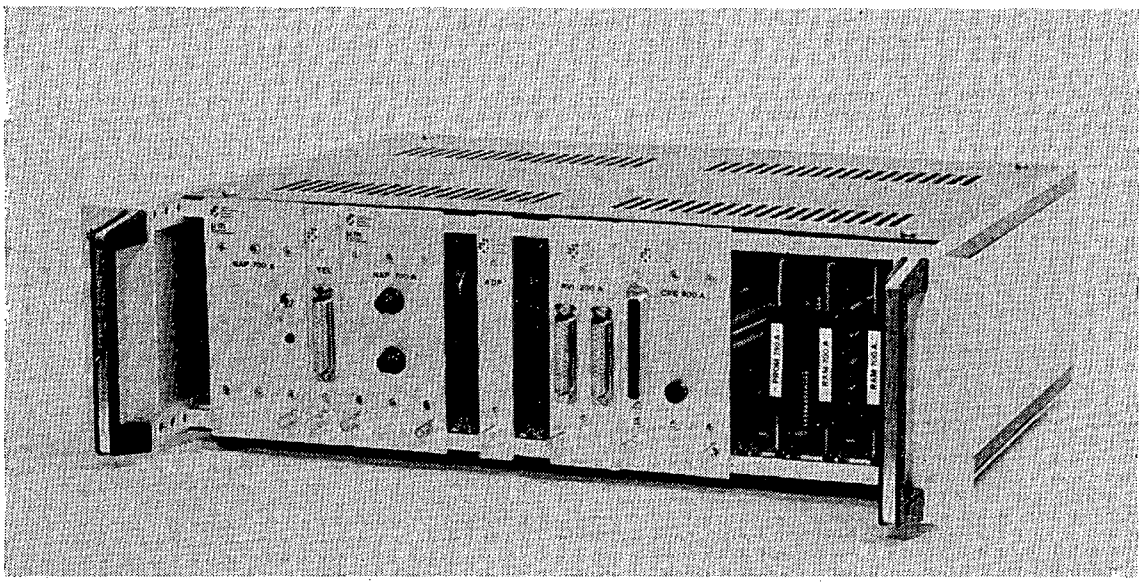
STRIP WRAP UNWRAP

OK MACHINE & TOOL CORPORATION
3455 CONNER STREET, BRONX, NEW YORK, N.Y. 10475 U.S.A. • PHONE (212) 994-6600
TELEX: 125091 TELEX: 232395

Ko pišete proizvajalcu, omenite časopis INFORMATICA.

modularni sistem instrumentacije mikro - m na osnovi mikroračunalnika

Vaš problem avtomatizacije upravljanja procesov in zbiranja podatkov lahko hitro in ekonomično rešite z modularnim sistemom instrumentacije "mikro-m". Iz modulov, ki smo jih razvili in preizkusili, lahko sami ali z našo pomočjo sestavite instrument, ki ustreza vašim zahtevam.



VHODNO/IZHODNE ENOTE:

- PROM-programator za EPROM 1702A (PRP-702)
- programirana ura (PKU-600)
- paralelni vhod/izhod s prekinitvenimi linijami, ojačanimi izhodi ali optično izolirani vhodi/izhodi (PVI-200, PVI-201, PVI-202)
- serijski vhod/izhod, 20 mA zanka, optično izoliran (SVI-220, SOE-250, TEL-001, TEL-010)
- A/D pretvornik, točnost 8 bitov, preklop na sistemsko analogno vodilo (ADP-100)
- izbiralnik analognih kanalov z ojačevalnikom (IKA-400, IKA-410)
- mostični ojačevalnik (MOS-300)

PROCESOR IN SPOMINSKE ENOTE:

- centralna procesna enota s procesorjem tipa 8080 A, 1 K RAM-a, 256 besed PROM-a, prekinitveno strukturo in 14 pozivnimi vodi (CPE-800, CPE 800/1, CPE 800/2)
- aktivni spomin (RAM), statični, 1 K ali 8 K (RAM-700, RAM-701)
- modul za mrtvi spomin (ROM), prostor za 2 K ali 16 K EPROM-a (ROM 750, ROM 751)

OHIŠJE IN NAPAJALNIKI:

- 19" ohišje po DIN 41494, kompletno z vtičnicami in sistemskimi povezavami (OHI-504)
- napajalnik, katerega moč omogoča polno zasedbo ohišja (NAP-700).

Na razpolago so tudi naslednji SISTEMI:

- modularni mikroračunalnik MKR-80: vsebuje enote in programsko opremo za vpisovanje programov s pisalnika v aktivni spomin in izvajanje programov. Možna je razširitev z vsemi vhodno/izhodnimi enotami.
- univerzalni digitalni programator UDP-80: omogoča razvoj in izvajanje funkcij digitalnega avtomata. Osnovne funkcije digitalnega avtomata so razširjene z vgrajenimi programskimi časovniki in števci, ki so nastavljivi s kodirnimi stikali.
- sistem za izravnavanje konic električne energije SIK-80: izravnava porabo električne energije tako, da v obdobjih velike porabe programirano izklaplja razpoložljiva bremena.

Primeri realiziranih sistemov: avtomatizacija stiskalnice, avtomatizacija reaktorja, mobilni sistem za toplotne meritve in drugi.

INSTITUT "JOŽEF STEFAN", 61000 Ljubljana, Jamova 39
ODSEK ZA REAKTORSKO IN PROCESNO TEHNIKO
Telefon: (061) 313-022 int. 35

Iskradata

Računalnik Iskradata C 18

- STROJNA OPREMA ZA POSLOVNO IN PROCESNO UPORABO
- FLEKSIBILNA MODULARNA ZGRADBA IN MOŽNOST RAZŠIRITVE
- PROCESOR Z MOŽNOSTJO MIKROPROGRAMIRANJA
- CENTRALNI POMNILNIK DO 512 K BYTE, DISKOVNE ENOTE DO 400 M BYTE, DO 4 TRAČNE ENOTE, DO 64 ZASLONOV S TASTATURO
- SISTEMSKA IN APLIKACIJSKA PROGRAMSKA OPREMA ZA PROCESNO IN POSLOVNO UPORABO
- POVEZAVA NA VEČJE RAČUNALNIKE

Mikroračunalnik Iskradata 1680

-MODULARNO GRAJEN
-OMOGOČA POLJUBNO KONFIGURIRANJE
-NASLAVLJANJE DO 64 K BYTE
-SERIJSKI IN PARALELNI VHODNO/IZHODNI MODULI
-PERIFERNE NAPRAVE: DISKETNE ENOTE, ZASLONI S TASTATURO, TISKALNIK, TELEPRINTER
-ŠOLSKI ENOMODULNI RAČUNALNIK
-LABORATORIJSKI RAČUNALNIK
-PROCESNI RAČUNALNIK
-POSLOVNI RAČUNALNIK

spremljajoče dejavnosti

-LASTEN RAZVOJ STROJNE, SISTEMSKE IN APLIKACIJSKE PROGRAMSKE OPREME
-ŠOLANJE
-TEHNIČNO VZDRŽEVANJE
-STROKOVNA POMOČ

ISKRA, Industrija za telekomunikacije, elektroniko
in elektromehaniko, Kranj, TOZD Računalniki,
64000 Kranj, PE Ljubljana, Titova 81,
tel (061) 326-367, 322-241