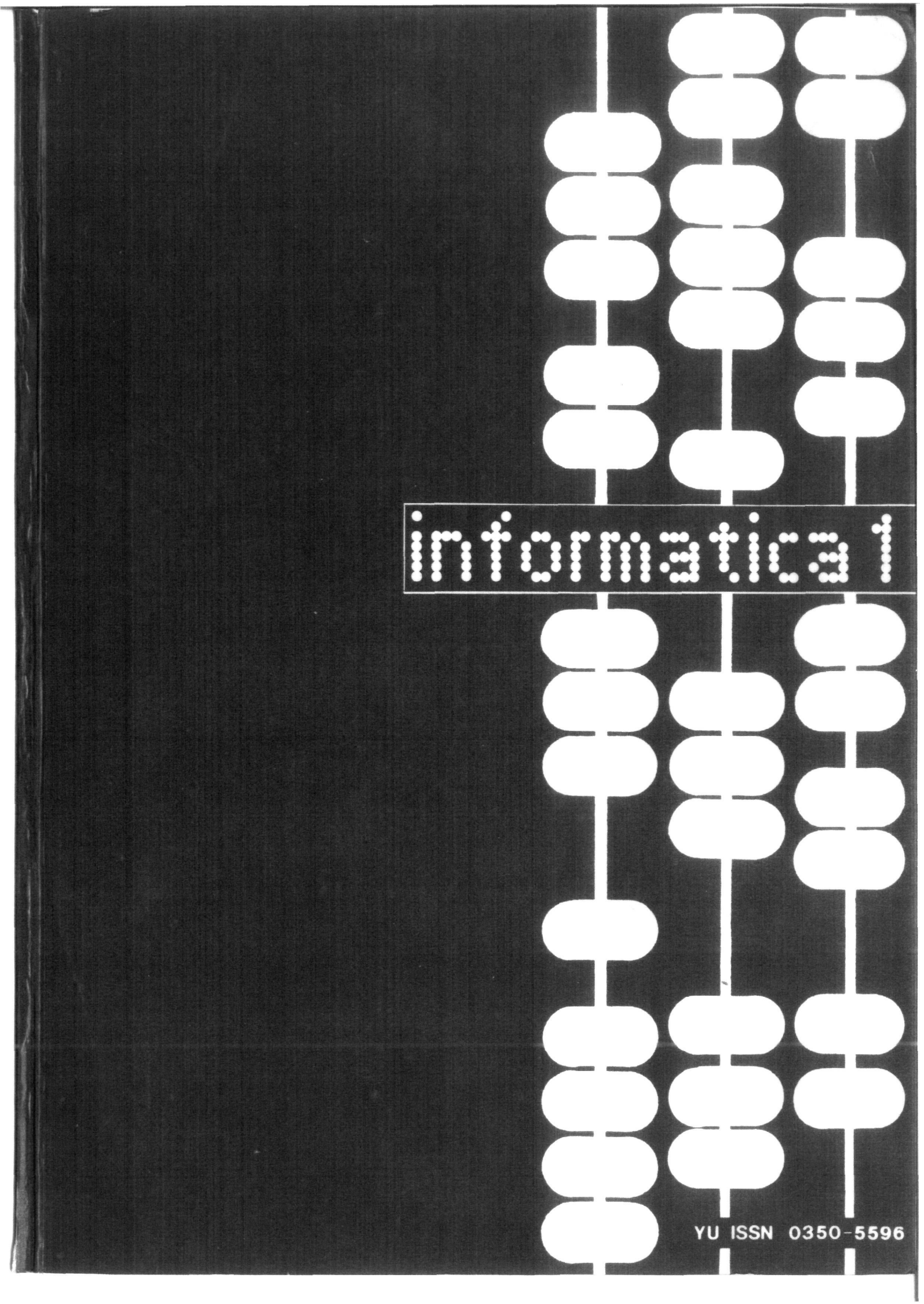
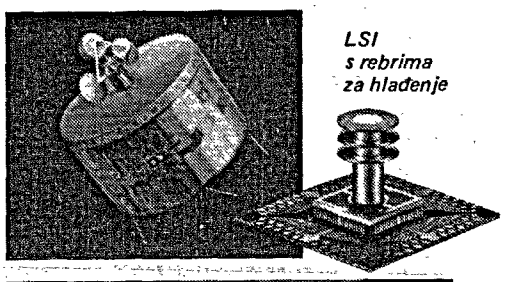


YU ISSN 0350-5596

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



LSI s rebrima za hlađenje

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

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

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

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

Za dalje informacije obratite se na:

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

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

FUJITSU

Fujitsu Limited·Tokyo, Japan

informatica

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

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

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

Printed by: Tiskarna KRESIJA, Ljubljana

DESIGN: Rasto Kirn

JOURNAL OF COMPUTING AND INFORMATICS

YU ISSN 0350 – 5596

VOLUME 3, 1979 – No. 1

CONTENTS

informatica

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

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

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

Pri financiranju revije sodeluje tudi Raziskovalna skupnost Slovenije.

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

Tisk: Tiskarna KRESIJA, Ljubljana

Grafična oprema: Rasto Kirn

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

YU ISSN 0350 – 5596

LETNIK 3, 1979 – št. 1

V S E B I N A

SPEKTRALNE ČRTE INFORMACIJSKIH SISTEMOV IN KRITIČNI POGLEDI NANJE

S. LESKOVAR

UDK: 007 : 681.3

FAKULTETA ZA ELEKTROTEHNIKO, LJUBLJANA

Članek opisuje koncept informacijskega sistema sedemdesetih let, in ga kritično vrednoti. Opis se naslanja na spektralne črte s katerimi označuje nekatere karakteristične poteze, ki jih osvetljuje literatura in praktične izkušnje. Po prikazu rasti in oblikovanje koncepta informacijskih sistemov se članek posveti relacijam, med podatki in informacijami, od katerih preide na informacijsko analizo, iz te na baze podatkov in končno na okolje informacijskih sistemov.

THE SPECTRAL LINES OF INFORMATION SYSTEMS AND THEIR CRITICAL APPRAISAL. The paper deals with the concept and its critical appraisal of Information System of seventieth. The concern is based on spectral lines which are used for denoting some characteristic lines which are ilemunated by literature and by practical experience. After the growth of the concept of information systems is presented, the paper deals with relations between data information and further on with information analysis, data base concepts and with information systems environment at the end.

1. Uvod

Ko danes ocenjujemo kakšna je bera učinkovitosti informacijskih sistemov v primerjavi z obeti, lahko vedno znova ugotavljamo, da informacijski sistemi niso izpolnili pričakovanj uporabnikov, še zlasti pa ne uporabnikov v gospodarstvu. Ni še 10 let, ko so informacijski sistemi doživljali pravo evforijo kot MIS (upravljavski informacijski sistemi). Toda velike ambicije upravljavskih informacijskih sistemov, ki jih je rodil prehod iz šestdesetih v sedemdeseta leta so kmalu zbledele kot neuresničljive /17/.

Informacijski sistemi sedemdesetih let so bili manj ambiciozni. Nič več niso poskušali biti totalni. Svoje poslanstvo in naloge so iskali v transformiranju klasičnih neformalnih informacijskih sistemov gospodarskih organizacij v formalizirane sisteme z računalniško obdelavo podatkov. V vsaki konvencionalni organizaciji eksistira namreč v implicitni obliki njen (neformalen) informacijski sistem, ki je vgrajen v njenih poslovodnih in upravljavskih funkcijah.

S tehnične plati ni bilo videti nobenih preprek, ekonomski učinki so izgledali ugodni in tudi z organizacijske plati ni bilo pričakovati nepredvideno velikih problemov. V resnici pa so nastopili ne samo tehnični in organizacijski problemi ampak so se tem problemom pridružili še drugi, ker so bile upravljavske in vodstvene dejavnosti napačno opredeljene, napačno ocenjene informacijske potrebe, nastopili pa so tudi problemi, ki izvirajo iz narave upravljavskih funkcij, in ki se jim pridružujejo problemi, ki izvirajo iz človeške narave /35, 42, 31/.

Razumevanje tega dogajanja je nekoliko lažje, če ga navežemo na 20 letni razvoj informacijskih sistemov. Poskusimo ga na kratko orisati v skopih obrisih in nanj naslonimo obravnavanje spektralnih črt informacijskih sistemov. Pri tem naj spektralne črte nakazujejo dinamične sestavine konceptualne in tehnične narave, ki v medsebojnem učinkovanju ustvarjajo koncept informacijskih sistemov sedemdesetih let.

2. Rast in oblikovanje koncepta informacijskih sistemov

Ko so v sredi pedesetih let računalniki začeli vstopati v poslovno področje, je kmalu postalo jasno, da je potrebno razvijati čimbolj generalizirane postopke za uporabo računalnikov. Prvi korak v to smer je bilo rojstvo COBOL-a v začetku šestdesetih let, kar je bila generalizacija, ki je omogočala učinkovitejše programiranje za poslovne aplikacije. Ta je sicer večala strojne stroške, pri tem pa je omogočala veliko manj časa za izdelavo programov, kar je zmanjševalo stroške za programiranje. Zmanjševanje človeških naporov na račun računalnikov je postal splošen razlog za iskanje vedno bolj generaliziranih postopkov /39/ čeprav je razumljivo, da generalizirane rešitve v splošnem niso tako učinkovite kot k problemu ukrojene.

Generalizacijo metod za rabo računalnikov je pogojevala v veliki meri tudi izgradnja amerikanskega računalniškega zračnega obrambnega sistema SAGA, ki so ga v ZDA začeli razvijati sredi pedesetih let /48/. To je bilo pravo računalniško omrežje za zbiranje in hranjenje podatkov iz radarskih postaj in drugih virov za podatke zračne obrambe. V teh sistemih nahajamo že tudi zametke generaliziranih podatkovnih baz, ki so jih nazivali komandni in kontrolni sistemi.

Poleg obrambnih sistemov se je kazalo kot neizmerno področje za generalizirano uporabo računalnikov upravljanje in poslovodenje v industriji, trgovini, bančništvu in v gospodarstvu nasploh, pa tudi v upravnih dejavnostih in v državni administraciji. Najprej so računalniki našli pot v računovodstva za knjigovodske obdelave podatkov. Kmalu pa je vzbrstela ideja, da bi začeli z računalniki pripravljati ne samo knjigovodske podatke ampak tudi upravljavske informacije. Idejo, da bi se računalniška raba povezala z upravljanjem,so pomembno podpirali trije koncepti iz začetka šestdesetih let, ki obravnavajo informacijske sestavine upravljanja in vodenja.

Prvi koncept se nanaša na obravnavanje upravljanja kot na transformacijo informacije v akcijo. To transformiranje teče v posebnem informacijskem omrežju, ki ga sestavljajo nivoji, frekvenca dogodkov in decizijske funkcije /20/. Za ta koncept je pomembno, da začenja obravnavati dogajanja v organizaciji povezano z vzroki in posledicami in da uvaja v upravljavske funkcije tudi povratne zanke.

Drugi koncept /50/ se nanaša na obravnavanje problemov in na odločanje. S tem, da razdeli odločitve na programirane in neprogramirane kaže na možnost, da se programirane odločitve avtomatizirajo z računalniki. Oba koncepta dopolnjuje koncept, ki obravnava planiranje in upravljanje s sistemi /4/. Ta koncept opredeljuje hierarhične nivoje upravljanja z njihovimi opravili in lastnostmi. Vsi trije koncepti so komplementarni in jih druga polovica šestdesetih let obravnava že v sintetični obliki /6/.

Vzporedno s koncepti, ki so povezovali informacije in upravljanje, in ki so s tem rojevali koncept tako imenovanega upravljavsko informacijskega sistema, se je ob koncu šestdesetih let močno povečala računalniška moč, hitrost delovanja računalnikov in njihove spominske kapacitete. Te računalniške lastnosti so močno pospeševale razvoj tehnik za gradnjo in vzdrževanje podatkovnih zbirk, tehnik za sortiranje podatkov in tehnik za pripravljanje poročil. Nastajali so vedno bolj generalizirani sistemi podatkovnih zbirk /45/, ki so proti sedemdesetim letom prerasli v koncepte podatkovnih baz /7, 8, 9/.

Računalniški sistemi so bili na prehodu iz šestdesetih v sedemdeseta leta dorasli domala vsaki obdelavi podatkov, ne glede na njeno kompleksnost in obsežnost podatkovnih struktur. Ker pa so bili stroški za uporabo velikih računalniških sistemov relativno veliki, so o njihovi rabi in o velikih podatkovnih strukturah razmišljale predvsem velike gospodarske organizacije.

Razvoj računalniške tehnologije je spremljalo padanje stroškov za računalniško opremo, kar je postalo posebej očitno, ko so začeli prihajati na trg miniračunalniki. Ti so imeli čedalje več lastnosti velikih sistemov in so kmalu tudi prevzemali njihove posle. Z miniračunalniki se je začelo tudi distribuirano procesiranje /17, 37/, ki je pomenilo decentralizacijo v gradnji podatkovnih baz in tudi v obdelavi podatkov. Krog uporabnikov računalniške obdelave se je večal, toda informacijski sistemi še vedno niso mogli zaživeti na predviden način.

Že zgodaj se je začelo obravnavati tudi razmerje med stroški in koristi informacijskih sistemov /46, 19/. Vrednotenje informacijskih sistemov je lahko teklo le preko vrednotenja informacij. Te so vredne samo toliko, kolikor so vredni učinki, ki slonijo na njihovi uporabi v dejavnostih upravljanja in poslovodenja.

Na tej osnovi je mogoče vrednotiti informacijske sisteme predvsem teoretično, ker praktično ni mogoče priti do soodvisnosti med upravljavskimi učinki in informacijami v kvatitativni obliki. Poleg tega pa velikokrat tudi ocene učinkov niso dognane in ne dovoljujejo realne ekonomske analize.

Ker pri nas dolgo ni bilo omembe vredne računalniške opreme, so problematiko informacijskih sistemov odprla šele sedemdeseta leta /28/. Ni pa se razmahnila, ker jo je utesnjevala predvsem domača računalniška politika, ki med drugim ni omogočila Univerzi, da bi prišla do univerzitetnega računalniškega sistema. Zato je na tem področju za več kot pol desetletja izpadel pomemben univerzitetni delež ne samo pri oblikovanju računalniških kadrov ampak tudi v raziskovalno - razvojnem delu.

Med najpomembnejše probleme informacijskih sistemov v sedemdesetih letih lahko uvrstimo skoraj nepremostljive težave v odnosih med uporabniki in nastajajočimi informacijskimi sistemi. Vsa prizadevanja, ki jih nakazuje literatura sedemdesetih let /51, 27, 21, 35, 15, 30, 40, 41, 42/, niso v tem pogledu kaj prida zalegla, kar velja tudi za naše razmere /29, 31/. Zaradi tega se lahko, podobno kot v začetku sedemdesetih let, ponovno vprašamo, ali so informacijski sistemi utvara /16/ ali stvarnost, in kje je meja med obojim.

3. Od podatkov k informacijam

Z uporabo računalnikov v gospodarskih dejavnostih in v administraciji se je začela tudi diskusija, kaj naj pomeni podatek in kaj informacija. Velikokrat sta se oba pojma enačila, računalniki pa so se obravnavali kot stroji za procesiranje informacij in kot stroji za avtomatsko oziroma elektronsko obdelovanje podatkov. Zato se je že sredi šestdesetih let v okviru IFIP-a /22/ izoblikovalo naslednje gledanje na podatke in informacije: podatki so "formalizirana predstava dejstev (faktov) ali idej, ki omogoča, da se z nekim postopkom prenašajo, izmenjujejo ali se z njimi vršijo kakšne manipulacije"; informacijo pa je razumeti "kot pomen, ki ga prireja človek podatkom v skladu z znanimi konvencijami, na katerih sloni predstavljanje dejstev in idej s podatki". IFIP-ovo opredelitev je mogoče še danes smatrati kot temeljno, ki pa jo je za operativno rabo potrebno ustrezno dograjevati. Največkrat je pri tem potrebno sintaktično in semantično nedvoumno opredeliti "znane konvencije".

Problematika informacijskega strukturiranja in informacijske selekcije, ki je prišla v ospredje v drugi polovici šestdesetih let, je odprla vprašanje identifikacije informacije /22/. Informacijsko strukturiranje naj bi opredelilo vrednosti in razrede, imena, razredne soodvisnosti, identifikatorje, vse do slovarja, ki naj podaja povezave med identifikatorji in opredeljenimi razredi.

Na tej osnovi se razredi vrednosti lahko delijo v dve vrsti: v razrede vrednosti, ki se dajejo v sistem in v razrede vrednosti, ki se s sistemom derivirajo. V modificirani obliki zasledimo podobne prijeme tudi pri Langeforsu /26/ in Lundebergu /32/.

Ker je bil eden začetnih problemov računalniške rabe operativno uporabno povezovanje dveh konceptov, koncepta podatkov in koncepta informacij, je CODASYL že v letu 1962 oblikoval informacijsko algebro za opisovanje informacijskih elementov, ki jih sestavljajo podatkovni elementi. Za CODASYL-om so se zvrstili še koncepti Chapina, Langeforsa, McDonougha in Taggarta /52/. Vsem je skupno to, da obravnavajo informacijski element kot kompozicijo, ki najprej opredeljuje, katere podatkovne vrste in v kaki sestavi bodo s svojimi vrednostmi opredeljevale informacijsko vsebino. Sistematično grupiranje podatkovnih elementov z natanko opredeljenimi pomeni naj predstavlja določen operativni informacijski element z identifikacijsko lastnostjo.

Koncept informacijskih elementov se je pokazal zelo koristen pri informacijski analizi in še posebej pri oblikovanju poročil. Da bi se lažje seznanili s tem, kako podatkovni elementi gradijo informacijski element, si oglejmo CODASYL-ov koncept. Zanj je značilno, da gradi informacijski element z lastnostmi. Kot primer vzemimo informacijski element, ki ga podajajo tri lastnosti: študent, študijska smer in letnik. Ta element lahko opišemo z naslednjo shemo:

Lastnost	Lastnost	Lastnost
Priimek in ime študenta	Študijska smer	Letnik
Vrednost	Vrednost	Vrednost
FIAS JANKO	AVTOMATIKA	4

Vsaka lastnost se izraža s svojim podatkovnim elementom, ki mu pripada določen razred vrednosti. Vsaka vrednost se prikazuje z nekim nizom simbolov. Drugi koncepti nastavljajo podatkovne elemente po drugih kriterijih in ne uporabljajo samo lastnosti. Langefors ima naprimer najprej identiteto objekta, nato časovno opredelitev in za tem lastnosti; pri McDonoughu zamenjujejo lastnosti najprej imena nato časovne postavke zatem pa podatkovni elementi, ki opredeljujejo kvantiteto in kvaliteto.

Taggartov koncept informacijskega elementa je precizneje opredelil lastnosti, čas in atribute. Prinjem pomeni atribut podatkovni element, ki označuje kak dogodek ali stanje ali razred objektov, ki so zanimivi za uporabnika. Čas je podatkovni element, ki identificira časovno periodo za kako dogajanje ali čas ko se je spremenilo stanje v določeni množici objektov. Bitnost pa je eden ali več podatkovnih elementov, ki identificirajo razred objektov, na katere se nanaša dogajanje ali stanje, ki ga podaja atribut.

Nakazani koncepti kažejo prizadevanja, kako priti po formalni poti od podatkov k informacijam. Toda za uporabnika je bolj pomembno, do katerih podatkov lahko pridejo, s kakšnimi napori in koliko lahko tem podatkom zaupajo.

Kljub vsemu se podatek in informacija velikokrat rabita kot sinonima. Včasih pa se zgodi, da se rabi pojem informacija kot sinonim za obdelane podatke, čeprav to ni skladno z osnovno IFIP-ovo opredelitvijo. Računalniki očitno ne morejo niti rojevati informacij, niti ničesar početi z njimi. Za računalnike eksistirajo samo podatki in njihove strukture. Ko pa podatki prihajajo k uporabniku, jih ta dojema kot informacije, ki lahko imajo različne lastnosti npr. natančnost, nezamujenost, nadrobnost, kompletnost itn./15, 19/.

4. Informacijska analiza in razvoj informacijskih sistemov

Uporaba računalnikov v gospodarstvu pa tudi v administraciji se je začela z elektronsko obdelavo podatkov - EOP (EDP) jo pri nas pogosto označujemo tudi kot avtomatsko obdelavo podatkov - AOP. Za to obdelavo so značilne takoimenovane aplikacije, ki slonijo vsaka na svojih zbirkah podatkov. Aplikacije so npr.: vodenje saldakontov, obračunavanje osebnih dohodkov, itn.

Pomankljivosti AOP, ki so se kazale v nepovezljivosti med podatki in izredno težavni poti do kompleksnejših obdelav, naj bi bilo mogoče odpraviti z gradnjo informacijskih sistemov z bazami podatkov. Ti naj bi povsem nadomestili AOP in omogočali obdelavo podatkov, ki bi bila vključena v funkcije upravljanja.

Pri AOP je bilo mogoče dokaj preprosto organizirati kako aplikacijo, potrebno je bilo le dovolj natanko posneti ročne manipulacije in na tej osnovi pripraviti računalniško obdelavo. Razvijanje informacijskih sistemov, pri katerih naj bi se obdelave podatkov vključevale v funkcije upravljanja, je narekovalo mnogo zahtevnejše pristope od onih v AOP. Potrebno je ugotavljati, katere informacije oziroma podatki so potrebni funkcijam upravljanja, v katerih oblikah in v kaki dinamiki.

Potreba po opredeljevanju informacij za upravljavske funkcije je vodila ob koncu šestdesetih let in še posebej v sedemdesetih letih k sistemski informacijski analizi. Informacijska analiza je kmalu postala temeljna dejavnost v snovanju informacijskih sistemov. Z njo naj bi se opredelile informacije, ki jih rabijo vodstvene dejavnosti kolikor je mogoče soodvisno. Opredeljene naj bi bile tudi za uporabnika pomembne informacijske lastnosti in informacijske povezave. Taka informacijska specifikacija naj bi bila osnova za razvoj podatkovnih baz, ki bi bile prilagojene potrebam uporabnikov.

Informacijska analiza se začenja s strukturiranjem vodstvenih dejavnosti /6, 29, 33, 34/. Za samo informacijsko analizo pa je potrebno poznavanje in razumevanje procesov odločanja v vodstvenih dejavnostih in za te procese potrebno informacijsko ozadje /27, 15, 33, 35/.

Proizvajalci računalniške opreme so si prizadevali, da bi bila analiza čim bolj pragmatična in da bi vodila po čimkrajši poti k rezultatom. Tej tedenci se je zoperstavljala teoretska usmeritev /26, 32, 33/, ki se je prizadevala, da bi najprej formalno predstavila realni svet in ga šele s pomočjo take predstavitve analizirala.

Pragmatična analiza pravzaprav sploh ni analizirala funkcij upravljanja, ampak je izhajala iz dejstev, na kakršne je naletela ter na tej osnovi specificirala potrebe po podatkih. Teoretična poglobljena analiza je terjala več napora, a je omogočala tudi odkrivanje slabosti v dejavnostih upravljavskih funkcij, kot so npr. več-

tirnost, neusklajenost, uporaba različnih podatkov za iste objekte itn.

Pomembno orodje analize so od njenih začetkov naprej diagrami potekov. V sedemdesetih letih pa postajajo uspešnejše orodje analize procesov decizijske tabele /25, 38/. Z njimi je mogoče predstaviti kombinacije pogojev in akcij in na ta način modelirati odločitvene situacije. Uporabljamo jih lahko tudi za opisovanje kompleksnih odločitvenih procesov /54/ in pri tem uporabljamo danes že standardno programsko opremo.

V visoko razvitih okoljih poskušajo informacijsko analizo razviti v formaliziranih oblikah in jo izvajati z računalniki /47, 34/. Koliko to lahko prispeva k kvaliteti informacijskih specifikacij je še vedno odprto vprašanje, čeprav je pričakovati, da bi določen del analize, ki se nanaša na dobro strukturirane dejavnosti bilo mogoče avtomatizirati.

Ne glede na to, da je bilo vloženega mnogo truda v prizadevanja, ki naj bi vodila do poti za učinkovito informacijsko analizo, na kateri naj bi slonel konkretni razvoj informacijskih sistemov, se je pokazalo, da je informacijska analiza še vedno močno nedorečeno poglavje informacijskih sistemov. Še vedno je preveč poudarkov na formalni plati, premalo pa na vsebinski.

Čeprav se je v zadnjih letih močno razvila informacijska kultura samih uporabnikov, da lažje sodelujejo pri informacijski analizi, pa informacijski analitiki še zmeraj premalo poznajo in upoštevajo lastnosti in naravo uporabnikov /31, 40, 35/. Enega izmed razlogov, da je skoraj polovica informacijskih sistemov zgrešenih /35/ je prav gotovo iskati tudi v tej smeri. Že nekaj let pa je očitno tudi, da ni mogoče razvijati mehanike za analizo in razvoj informacijskih sistemov neodvisno od okolja, kjer naj bi delovali. Zato postaja okolje pomemben dejavnik informacijskih sistemov.

5. Podatkovne baze informacijskih sistemov

Podatkovne baze lahko obravnavamo povezano z informacijskimi sistemi, lahko pa jih obravnavamo neodvisno od njih kot tehnične stvaritve. Če vemo, da se največji del računalniških opravil nanaša na delo s podatki, na njihovo zajemanje, hranjenje, urejanje, povezovanje in na računske in logične operacije z njimi, lahko razumemo, da je razvoj računalniške opreme vse skozi spremljal tudi ustrezen razvoj "podatkovnih tehnik".

Računalniška obdelava podatkov je kmalu oblikovala osnovno podatkovno strukturo - podatkovno zbirko (file), ki jo sestavljajo podatkovni elementi. Na to strukturo se je naslonil ves razvoj računalniške obdelave podatkov šestedestih let. Za vse obdelave so bile potrebne predvsem naslednje osnovne operacije z zbirkami:
- kreiranje zbirk,
- vzdrževanje zbirk,
- seganje k podatkom v zbirkah,
- sortiranje podatkov v zbirkah in
- pripravljanje poročil s podatki, ki so v zbirkah.

Naštete operacije so tako pogoste, da se je kmalu kazala potreba po izdelavi tipizirane programske opreme zanje /39/. Ta programska oprema naj bi razbremenila programerje pri mukotrpnem pisanju programov za navedene operacije. Pri preprostih obdelavah se je sicer dalo veliko opraviti že s COBOL-om, toda aplikacije so postajale v šestdesetih letih vedno bolj zahtevne, kar je večalo potrebo po posebni programski opremi za delo s podatkovnimi zbirkami.

V prvi polovici šestdesetih let je bila že na voljo programska oprema, v katero so bile vgrajene funkcije za vse osnovne operacije z zbirkami podatkov, in ki so jo nazivali zbirkovno upravljanje (File Management) ali pa sistem za zbirkovno upravljanje (File Management System). Iz teh sistemov je v drugi polovici šestdesetih let začel rasti nov koncept /39/ generalizirane programske opreme za delo z zbirkami, ki je kmalu dozorel v CODASYL-ov koncept generaliziranih podatkovnih baz in sistemov za upravljanje z njimi /8, 9, 36/.

Po tem konceptu je baza podatkov zbirka podatkovnih n-terk v kateri eksistirajo relacije med tako imenovanimi rekordi, podatkovnimi agregati in podatkovnimi elementi. Pod rekordom razumemo kak imenovan razred, ki vsebuje bodisi podatkovne elemente bodisi podatkovne agregate. Pod podatkovnimi agregati razumemo vsak imenovan podrazred podatkovnih elementov, ki sestavljajo rekord.

Velikokrat imamo opravka z večimi bazami podatkov v enem sistemu, kar lahko imenujemo sistem podatkovnih baz. Kot sinonim za sistem podatkov-

nih baz srečujemo tudi izraz podatkovna banka. Izraz podatkovna banka pa je v literaturi pogosto uporabljen v precej ohlapno opredeljenih oblikah in ga zato zmeraj bolj poredko srečujemo v strokovni literaturi.

Za oblikovanje, vzdrževanje in uporabo podatkovne baze je potrebna posebna programska oprema, ki jo pogosto nazivamo upravljalni sistem podatkovne baze (Data Base Management System). Ta programska oprema naj po CODASYL-u zagotavlja:

- možnost za struktuiranje podatkov, ki bi se najbolj prileglo posameznim aplikacijam ne glede na to, da utegne več aplikacij rabiti iste podatke, kar zahteva fleksibilnost, ki ne bi smela večati podatkovne redundančnosti,
- učinkovit dostop do podatkov in učinkovito ažuriranje podatkov v podatkovni bazi,
- uporaba različnih strategij iskanja v podatkovni bazi ali v njenih delih,
- zaščito podatkov v bazi pred nedovoljenimi posegi
- spremljanje in nadzor nad fizično razporeditvijo podatkov v spominskih medijih,
- možnost za predpisovanje različnih podatkovnih struktur, začenši s takimi, pri katerih ni nobenih povezav med podatkovnimi elementi pa vse do drevesnih in zančnih struktur,
- možnosti za uporabnikovo interakcijo s podatki, ne da bi moral ppznati, kako je fizično zgrajena podatkovna baza in ne da bi se mu bilo potrebno ukvarjati z mehaniko prenašanja podatkov iz spominskih medijev do ekrana, ali druge izhodne enote,
- da so uporabniški programi praktično neodvisni od fizične razporeditve podatkov v spominskih medijih,
- možnosti za tak opis podatkovne baze, ki ne omejuje njene uporabe samo na določen programski jezik.

Pomembna lastnost podatkovnih baz je strukturiranje podatkov na osnovi podatkovnih modelov. Podatkovni model je konstrukcija iz podatkovnih elementov in njihovih medsebojnih povezav. Te modele je mogoče računalniško implementirati s tem, da se jih prevede v sheme s pomočjo jezikov za opisovanje podatkov (DDL). Podatkovni elementi in njihovi agregati se v shemi opisujejo z DDL na način, ki je podoben njihovemu opisovanju v Data Division v COBOL-u.

V sredini sedemdesetih let se je CODASYL-ovemu konceptu pridružil še ANSI X3 SPARC koncept, ki sloni na treh shemah: na zunanji shemi, na konceptualni shemi in na notranji shemi. S konceptualno shemo se podaja logični opis celotne baze podatkov, medtem ko se zunanja shema nanaša na logični opis podatkov za določeno aplikacijo in jo uporablja programer. Notranja shema pa podaja opis podatkovne baze na strojnem oziroma spominskem nivoju.

Pomemben napredek pri oblikovanju podatkovnih struktur v bazah podatkov omogočajo Coddove relacije in normalizacijske tehnike /10,11,12,14/. Z njihovo pomočjo se dajo oblikovati kanonske strukture, ki omogočajo minimiziranje konceptualne sheme. Implementacija take sheme pa je tudi najbolj prilagojena na nepričakovane zahteve po razširitvah baze, ki jih je mogoče izvesti, ne da bi jo bilo potrebno drastično prestrukturirati /36/.

S tehnične plati gledano je lahko kaka baza podatkov prava mojstrovina. Kakšna pa je njena vrednost za uporabnike zavisi od tega, katere podatke jim lahko nudi.

Kaj bo v bazi, naj bi opredeljevale informacijske specifikacije, ki služijo kot osnova za razvoj konceptualne sheme za bazo podatkov. Ker zavisijo informacijske specifikacije od informacijske analize, je razumljivo, da je z njo pogojena vsa vsebinska plat podatkovnih baz. Rečemo lahko, da predstavljajo baze podatkov tehnično plat informacijskih sistemov, ki so po vsebinski plati opredeljeni z informacijsko analizo.

6. Okolje kot dejavnik v razvoju informacijskih sistemov

Informacijski sistemi, ki so nastajali v prvi polovici sedemdesetih let, so kljub manjšim ambicijam, kot so jih gojili predhodni MIS koncepti, doživljali mnogo neuspehov. Pokazalo se je, da del teh neuspehov izvira iz preprostega dejstva, da se pri razvoju informacijskih sistemov ni dovolj upoštevalo lastnosti okolja, za katere se je razvijal kak informacijski sistem.

Ko so se začele razvijati metode in tehnike za informacijsko analizo, skoraj ni bilo govora o posebnostih različnih okolij in o prilagojevanju nanje. Zato se pri informacijski analizi lastnosti okolja niso kaj prida upoštevale, čeprav je od njih zaviselo med drugim tudi to, kako bo reagiralo na informacijski sistem in kako ga bo sprejemalo. Poznavanje okolja pa bi omogočalo tudi tovrstna predvidevanja in njihovo upoštevanje.

Pri obravnavanju klasične organizacije ni mogoče govoriti o informacijskem sistemu in njegovi okolici. Pri njej imamo opravka samo z upravljavskimi in vodstvenimi funkcijami, v katerih so vtkane oziroma z njimi integrirane tudi informacijske dejavnosti, ki sicer ustvarjajo informacijski sistem organizacije, vendar le v implicitni obliki.

Ko začnemo razvijati informacijski sistem za kako organizacijo, začnemo ločevati informacijske dejavnosti od ostalih. Z uvajanjem informacijskega sistema, naj bi se izločale informacijske dejavnosti iz upravljavskih funkcij in prehajale v informacijski sistem. Na ta način se namesto klasičnega sistema funkcij za upravljanje in vodenje ustvarjata dva komplementna sistema, eden za "čiste" dejavnosti upravljanja in vodenja in drugi za informacijske dejavnosti. Te naj zagotavljajo komplementnemu razredu za njegovo delo potrebne informacije.

Okolje informacijskega sistema postaja torej nek sistem funkcij upravljanja in vodenja, ki ga ne poznamo v nobeni klasični organizaciji. Pri informacijski analizi je bilo sicer mogoče ugotoviti katere informacije so potrebne za delo upravljavskih in vodstvenih funkcij. Veliko težja, če ne domala nemogoča naloga pa je ugotavljanje, kaj pomeni za upravljavske in vodstvene funkcije prenos informacijskih dejavnosti na informacijski sistem, ki je od njih odvojen. Če gledamo na informacijsko okolje s tega vidika je razumljivo, da je upravičeno razlikovati npr. okolje v banki od okolja, ki ga ustvarja metalurški obrat, slednje okolje od okolja trgovske organizacije itn.

Vpliv okolja se je v drugi polovici sedemdesetih let začel kazati kot tako vpliven dejavnik v razvoju informacijskih sistemov, da se je v okviru IFIP-ovega tehničnega komiteja TC 9 oblikovala delovna grupa WG2, ki obravnava okolje informacijskih sistemov z namenom, da bi na ta način prispevala k razvoju informacijskih sistemov, ki bi bili bolj prilagojeni na svoje okolje.

Čeprav se je okolje informacijskih sistemov šele začelo raziskovati, se že kaže, da so informacijski sistemi neglede na vsa prizadevanja preveč odtujeni svojemu okolju, da bi lahko z njimi skladno zaživeli. Zato je pričakovati, da bo tehnološki napredek na področju računalništva spet omogočil reintegracijo informacijskih dejavnosti v funkcijo upravljanja in vodenja vendar v novi kvaliteti /32/ in z mnogimi izkušnjami uspešnih in neuspešnih informacijskih sistemov.

7. Zaključek

Informacijski sistemi so področje, ki je že od svojega začetka dalje polno problemov. Nekaj teh problemov je članek poskušal orisati z osvetljevanjem nekaterih učinkovin informacijskih sistemov. Tiste, ki jih je osvetlil je imenoval spektralne črte informacijskih sistemov. Kaj so odkrili pogledi na te črte, je bilo odvisno od njihove osvetlitve. Ta je izvirala delno iz literature, ki je navedena na kraju, pomemben delež v tem osvetlevanju pa je izviral iz praktičnih izkušenj, nabranih v desetletnem delu na problematiki informacijskih sistemov.

Članek poskuša razkrivati predvsem tiste probleme informacijskih sistemov, ki kažejo na današnje zagate pri njihovem uvajanju v operativo, kar postavlja mnoge samohvalne snovalce informacijskih sistemov v dokaj dvomljivo luč. Po svoje pa želi prispevati tudi k tisti informacijski kulturi, v kateri zaenkrat nemočno zaostajamo za razvitim svetom.

LITERATURA:

1. M.J. Alexander, "Information Systems Analaysis", SRA 1974

2. ANSI X3 SPARC/DBMS, Study Group report, American National Standard Institute, Washington, D.C., 1975

3. H.I. Ansoff, "Corporate Strategy", McGraw-Hill, N. York, 1965

4. R. Anthony, "Planing and Control Systems, A Fromework for Analysis", Harward Business School, 1965

5. C. Argyris,"Organizational Learning and Management Information Systems", Accouting Organizations and Society, Vol. 2, No.2, 1977

6. S.C. Blumenthal, "Management Information Systems", Prentice-Hall, Inc., 1969

7. G.M. Booth, "Distributed Information Systems", AFIPS konferenčni materiali, 1976

8. CODASYL Systems Comittee, "Feature Analysis of Generalized Data Base Management Systems" Technical Report, ACM, N. York, London, and Amsterdam, 1971

9. Data Base Task Group (DBTG) of CODASYL Programming Languages Committee, Report, ACM, N. York, London, and Amsterdam, 1971

10. E.F. Codd, "A Reational Model of Data for Large Shared Data Banks", Comm. ACM 13, No.6, 1970

11. E.F. Codd, "Further Normalisation of the Data Base Relational Models",Courant Computer Science, Symposia Series, Vol.6, Prentice Hall, 1972

12. E.F. Codd, "Recent Investigations in Relational Data Base Systems", Information Processing 74, North-Holland Publishing Company, Amsterdam, 1974

13. R.M. Cyert, J.G. March,"A Behavioral Theory of the Firm", Prentice Hall, Inc., Englewood Cliffs, 1963

14. C.J. Date, "An Introduction to Database System", Addison-Wesley Publishing Company, 1976

15. G.B. Davis, "Management Information Systems" McGraw-Hill, 1974

16. J. Dearden, F.W. McFarlan, "Management Information Systems", Richard D.Irwin, Inc., Honewood, 1966

17. J. Dearden, "MIS is a Mirage", Harward Business Review, Jan., Feb., 1972

18. The Diebold Group, "Automatic Data Processing Handbook", McGraw-Hill, 1977

19. J.C. Emery, "Cost/Benefit Analysis of Information Systems", SMIS konferenčni materiali No.1, The Society for Management Information Systems, 1971

20. J.W. Forester, "Industrial Dynamics", M.I.T. Press, 1961

21. W. Goldberg, "Why Should Management use Computers", Gradivo za seminar Informacijska tehnologija in upravljanje, JUIAG, Opatija, 1972

22. G.B.B. Grindley, W.G.R. Stevens, "Principles of the Identification of Information", Konferenčni materiali, File 68 (IAG), Studentlitteratur, Lund, 1968

23. IFIP-ICC Vocabulary of Information Processing, North-Holland Publishing Company, Amsterdam, 1966

24. D.A. Jerdin (ed), "The ANSI/SPARC DBMS Model", North-Holland Publishing Company, Amsterdam, 1977

25. F.J.J. Johnston, J.C. Davis, "Decision Tables in Data Processing", NCC Manchester, 1970

26. J.W. Klimbie, K.L. Koffeman (ed), "Data Base Management", North-Holland Publishing Company, Amsterdam, 1974

27. B. Langefors, "Theoretical Analaysis of Information Systems", Studentlitteratur/ Auerbach, Third Edition, 1973

28. S. Leskovar, V. Rupnik, "Makroprojekt - Sistemi informacija u upravljanju", Fakulteta za elektrotehniko, Ljubljana, 1970

29. S. Leskovar, "Upravljavski informacijski sistem", v A.P. Železnikar, S. Leskovar (ured.), Računalništvo v gospodarskih organizacijah, DZS, 1973

30. S. Leskovar, "Some Social and Managerial Implications of Informatics", Human Choice and Computers, Vienna, 1974

31. S. Leskovar, "Upravljalsko poslovna informatika", GZ Slovenije in FE Ljubljana, 1975

32. S. Leskovar, "Technological Impact on Information System Development and User's Appreciation Evolution", Konferenčni materiali, IFIP TC 8 - WG 8.2, The Information Systems Environment, Bonn, 1979

33. M. Lundeberg, "Information Analysis (IA)", Management Informatics, Vol.3, No.1, 1974

34. M. Lundeberg, J. Bubenko Jr., "Systemeering 75", Studentlitteratur, Lund, 1975

35. N.B. Macintosh, R.L. Daft, "User Department Technology and Information Design", North-Holland Publishing Company, Information and Management 1, 1978

36. J. Martin, "Computer Data-Base Organization", Prentice-Hall, Inc., 1977

37. F.J. Maryanski, "A Survey of Developments in Distributed Data Base Management Systems" Computer, Feb., 1978

38. H. McDaniel, "An Introduction to Decision Logic Tables", John Wiley and Sons, 1968

39. W.C. McGee, "Generalization: Key to Successful Electronic Data Processing", Journal of the ACM, January 1959

40. E. Mumford, "Job Satisfaction", A Study of Computer Specialists, London: Longmans, 1972

41. E. Mumford, B. Hedberg, "The Design of Computer Systems: Problems of Philosophy and Vision", Human Choice and Computers, Vienna, 1974

42. E. Mumford, F. Land, J. Hawgood, "A Participative Approach to the Design of Computer Systems", Impact of Science on Society, 1978

43. H. Nielsen, "The Management Information System - A Basis for File Considerations, Konferenčni materiali, File 68 (IAG), Studentlitteratur, Lund, 1968

44. G.M. Nijssen (ed), "Architecture and Models in Data Base Management Systems", North-Holland Publishing Company, Amsterdam, 1977

45. T.W. Olle, "Data Structures and Storage Structures for Generalized File Processing", File 68, Studentlitteratur, Lund, 1968

46. J. Orlicky, "The Successful Computer System" McGraw-Hill, N. York, 1969

47. D. Teichroew, H. Sayani, "Automation of System Building", Datamation, August, 1971

48. H. Sackman, "Computers, System Science, and Evolving Society", John Wiley and Sons, Inc., 1967

49. H. Sackman, "Computers and Social Options", Konferenčni materiali, Human Choice and Computers, Vienna, 1974

50. H.A. Simon, "The New Science of Management Decision", Harper and Row, Inc., 1960

51. S. Sjöberg, "Methods of Analysis and Construction; The Interplay Between Decision Model and Data Base", Konferenčni materiali, MIS 70, Studentlitteratur, Lund, 1970

52. I. Škraba, T. Mohorič, "Generalizirani sistemi za upravljanje podatkovnih baz", Konferenčni materiali za ADP seminar 71 v Zagrebu

53. W.M. Taggart Jr., "Developing an Organization's Information Inventory", Management Informatics, Vol.3, No.6, 1974

54. M. Verhelst, "Decision Tables Revisited", Gradivo za IAG seminar, Brussels, 1977

INTEGRIRANI MIKRORAČUNALNIK ISKRA – EMZ 1001

DUŠAN RAIČ

FAKULTETA ZA ELEKTROTEHNIKO, LJUBLJANA

UDK: 681.325

Prikazane so prednosti in težave integriranih mikroračunalnikov. Opisana je pot, po kateri smo pri mikroračunalniku EMZ 1001 rešili probleme testiranja in eksperimentiranja na prototipih. Navedene so glavne lastnosti vezja in orisane možnosti za njegovo uporabo.

ONE-CHIP MICROCOMPUTER ISKRA - EMZ 1001

Some drawbacks and advances of one-chip microcomputer systems are indicated and a solution to the problems is described such as it was used at the EMZ 1001 design. The main characteristics and application suggestions are given for the potential EMZ 1001 user.

UVOD

Odkar so se v letu 1971 na trgu pojavili prvi mikroprocesorji, nobeno leto še ni minilo brez občutnega napredka na tem področju moderne elektronike. Novi proizvodi prinašajo vedno večje hitrosti delovanja, večje nabore ukazov, večje dolžine besed, večjo izbiro priključenih vezij za prenašanje in pomnenje podatkov in podobno.

Pri vsej živahnosti razvoja pa opažamo, da poteka delo vodilnih svetovnih proizvajalcev predvsem v dveh smereh: po eni strani nastajajo vedno sposobnejši izdelki splošne narave, na drugi strani pa je znaten interes pri izdelkih z minimalno možno ceno, kateri se podrejajo tehnične lastnosti. Obe veji razvoja sta prisotni že nekaj let in obe gradita svojo uspešnost na izpopolnjevanju tehnologije ter vrhunskih načrtovalskih naporih.

Sistemi z najnižjo možno ceno so težili k stalnemu zmanjševanju števila potrebnih vezij v dani napravi. Končno je prišlo do združitve vseh potrebnih funkcij računalnika v enem samem vezju, v integriranem mikroračunalniku. Beveda so taka vezja poleg obvezne specializacije prinesla tudi obilico težav proizvajalcem, predvsem zaradi težavnega testiranja. Kljub temu pa so prednosti na posameznih področjih ogromne, saj so si zaradi izredno nizke cene utrli pot v vrsto novih izdelkov.

Pred uporabnika se postavlja vprašanje, kdaj uporabiti integrirani mikroračunalnik in kdaj univerzalni mikroprocesor s standardnimi dodatnimi vezji. Odgovor je trenutno razmeroma enostaven in lažji, kot pa je npr. izbira med posameznimi mikroprocesorji. Upoštevati moramo dva pomembna činitelja : velikost sistema in število kosov. Monolitni mikroračunalniki danes vsebujejo notranje pomnilnike velikosti 0,5 do 4 k za programe in 32 do kakih 256 besed za podatke. Sistemi z večjimi informacijskimi zahtevami trenutno ne morejo biti integrirani v enem samem ohišju. Prav tako moramo upoštevati stroške programiranja, ki so proti mikroprocesorjem večji. Kot najcenejši način vpisovanja programov v monolitnih mikroračunalnikih se namreč uporablja zapis v sami difuzijski maski pomnilnika ROM, to pa bistveno vpliva na ceno programiranja in na način uporabe takih vezij. Mikroračunalnik z vpisanim programom lahko zaradi tega obravnavamo tudi kot univerzalno "vezje po naročilu", katero šele po vpisu programa dobi končne lastnosti. Kadar je število naročenih kosov večje od kakih 5000 do 10000, se stroški izdelave programov, oziroma maske porazdelijo in niso pomembni, pri manjšem številu kosov pa postanejo nedopustno visoki.

Integrirani mikroračunalniki se zaradi tega uporabljajo predvsem v velikoserijskih izdelkih in tam, kjer njihovo uporabo terjajo posebne sistemske zahteve.

POSEBNOSTI INTEGRIRANIH MIKRORAČUNALNIKOV

Prednosti, ki jih prinese integracija vseh mikroračunalniških komponent v enem monolitnem vezju, so vsekakor znatne ne samo iz komercialnega, temveč tudi iz tehničnega vidika. Povezave med procesno enoto, pomnilnikom ROM in RAM ter z vhodno-izhodnimi posredniki so narejene znotraj ohišja in ne obremenjujejo več izvodov. Poleg tega, da na ta način ostanejo skoraj vsi izvodi prosti za prenašanje koristnih I/O podatkov, se za cel velikostni razred zmanjšajo parazitne kapacitivnosti v informacijskem kanalu med procesno enoto in pomnilnikom. Pri neki izbrani hitrosti delovanja to pomeni manjšo porabo energije in manjše dimenzije krmilnih elementov. Druga, tudi pomembna prednost integracije pa je povečana zanesljivost sistema.

Monolitna izvedba sistema ima seveda tudi svoje slabe strani. Predvsem se moramo zavedati, da bo vezje do neke mere specializirano za uporabo v določenih aplikacijah, katerim je prilagojena velikost pomnilnikov in zmogljivost procesne enote z I/O kanali. Razširitve takih sistemov so problematične in navadno ekonomsko neupravičene. Nadalje nastopajo problemi tudi v sami proizvodnji elementov zaradi težavnega testiranja. Na izvodih vezja namreč lahko zasledujemo samo odzive funkcionalne enote, ki jo tvori procesor skupaj z vpisanim programom; te funkcije so praviloma zelo kompleksne in poleg tega popolnoma drugačne za vsak vpisan program. Sorodna vrsta težav nastopa pri izdelavi prototipov, ko je program za neko aplikacijo šele v razvoju. Izdelava programske opreme je, kot vemo, vedno vezana na dolgotrajna preizkušanja in na vnašanje sprememb, ki jih želimo sproti preverjati.

Za izdelavo prototipov nudijo proizvajalci večinoma posebne izvedbe mikroračunalnikov v posebnih ohišjih z velikim številom izvodov. Preko dodatnih izvodov je možen dostop do notranjih podatkovnih vodil, na katera priključimo zunanje pomnilnike. Firma INTEL se poslužuje drugega načina, pri katerem so pomnilnik ROM zamenjali z električno programiranim pomnilnikom EPROM na sami silicijevi ploščici.

Oba načina reševanja problema prototipov sta se nam pokazala kot neprimerna. Povezana sta z dodatnimi stroški (razviti moramo dve različni vezji), potrebujeta dodatno tehnologijo in zahtevata od razvijalcev prototipa nakup posebnega razvojnega vezja, ki navadno tudi ni poceni. Zavedali smo se, da je za popularizacijo novega izdelka pomembno omogočiti eksperimentiranje na čim lažji način. Problem smo rešili tako, da smo že v zasnovi sistema omogočili različne načine delovanja vezja. Kot bomo videli v naslednjih odstavkih, smo s tem v eni potezi rešili probleme testiranja, eksperimentiranja in razširjanja sistema.

ZGRADBA MIKRORAČUNALNIKA EMZ 1001

Mikroračunalnik ISKRA EMZ 1001 je zaključeni mikroračunalnik na eni ploščici silicija. Z veliko gostoto integriranih elementov prinaša vse prednosti krmiljenja z mikroračunalnikom pri minimalnih stroških opreme. Njegova notranja zgradba je usmerjena zlasti v povezavo s tastaturami in izpisi na segmentiranih svetlečih diodah (LED). Ima zelo bogato organizacijo vhodov in izhodov ter nabor ukazov, ki je optimiran glede na predvideno uporabo v sistemih z nadzornimi ali aritmetičnimi funkcijami, krmilnih avtomatih in sistemih človek/stroj. V teh primerih najpogosteje srečamo naslednje zahteve :

- vgrajena ura, koledar in s tem povezane časovne odločitve,
- sinhronizacija z omrežno frekvenco,
- izpis številčnih prikazov,
- vhodi iz tastature,
- pomnenje dogodkov in logične odločitve,
- aritmetične operacije, itd.

Obdelava podatkov v mikroračunalniku EMZ1001 poteka preko 4-bitnih besed. Tako deluje tudi aritmetična enota, ki jo zato običajno programiramo za obdelave v serijskem BCD načinu. V aplikacijah, kakršnim naj bi največ služil EMZ 1001, aritmetika predvidoma ne bo pomembni parameter sistema. Važnejše bo pomnenje in odločanje na osnovi posameznih dogodkov, zaradi česar smo omogočili obdelave tudi nad posameznimi biti v pomnilniku RAM.

Nadzorni del in vhodno-izhodna konstrukcija mikroračunalnika EMZ 1001 sta 8-bitno zasnovana. Ukazi se dekodirajo v besedah po 8 bitov; vsi ukazi so enobesedni in skoraj vsi potrebujejo za izvršitev en sam cikel. Dolžina ukaznega cikla je 4,5 μs, kar zaradi enobesednih ukazov daje razmeroma veliko hitrost delovanja.

Povezava mikroračunalnika EMZ 1001 z zunanjim svetom poteka preko več specializiranih vodil. Vodilo A ima 13 bitov, ki služijo samo kot izhodi. Vodilo D ima 8 bitov in prenaša podatke v dveh smereh, lahko pa je tudi v t.im. nevtralnem "floating" stanju. Vodili I in K, od katerih ima vsako 4 bite, služita samo kot vhoda.

Sistem programskih prekinitev ni bil realiziran. Isto velja tudi za prenose DMA, ki zaradi specifičnosti in omejene velikosti pomnilnika RAM nimajo nobenega pravega pomena.

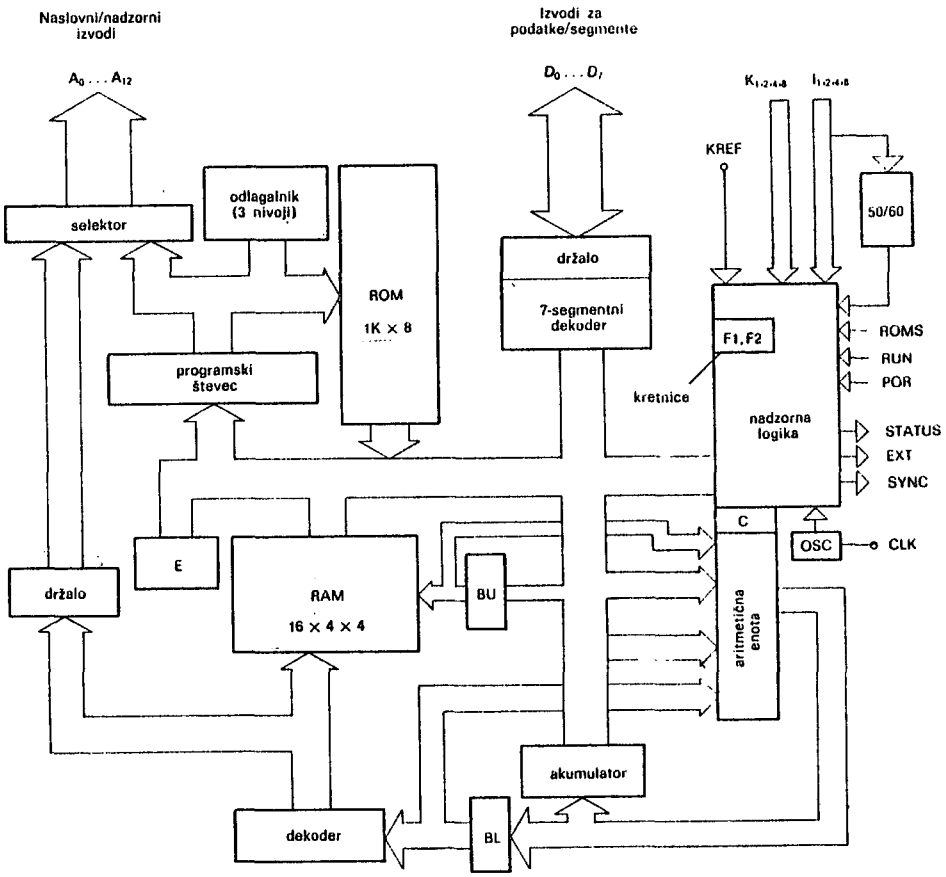
Na sliki 1 vidimo poenostavljeni blokovni diagram, časovni diagram in pregled izvodov mikroračunalnika EMZ 1001.

Nihanje oscilatorja pogojuje zunanja konstanta RC z vozliščem na izvodu CLK. Kot vidimo na priloženem diagramu, je osnovni ukazni cikel (T1, T3, T5, T7) sestavljen iz štirih nihajev oscilatorja. Na izvodu SYNC je stalno prisoten signal pravokotnih impulzov, ki delijo posamezne cikle na dva dela (T1+T3 in T5+T7). S tem signalom krmilimo predvsem zunanja vezja, kadar delamo v multipleksnem načinu delovanja.

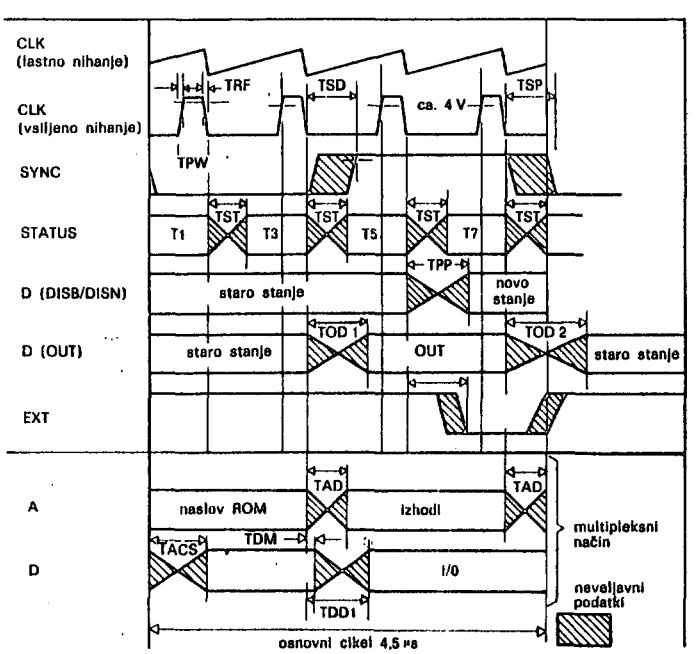
Organizacija procesne enote je klasična, z enim 4-bitnim akumulatorjem, ki je povezan s paralelnim 4-bitnim seštevalnikom in z registrom prepolnitve C. Možne so logične in aritmetične operacije, kot so seštevanje, komplementiranje, štetje navzgor in navzdol, primerjanje ter konjunkcija. Kot vidimo na blokovnem diagramu, prihajajo podatki za operacije iz različnih delov vezja glede na to, kateri ukazi se izvajajo. Akumulator uporabljamo za izvor in za ponor podatkov pri notranjih operacijah in pri I/O posegih. Kadar potrebujemo dva argumenta ali 8-bitno informacijo, akumulator dopolnimo z adresirano besedo iz pomnilnika RAM.

Poenostavljeni blokovni diagram

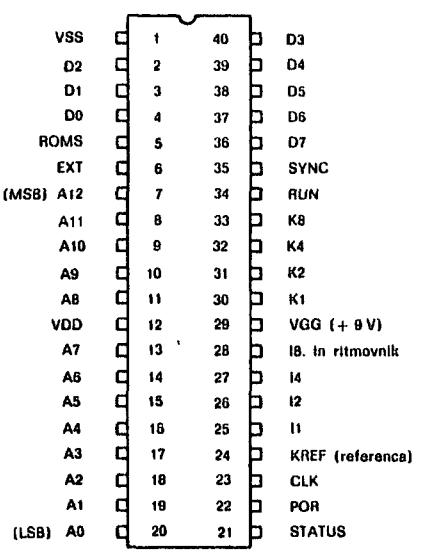
EMZ 1001



Časovni diagram



Izvodi

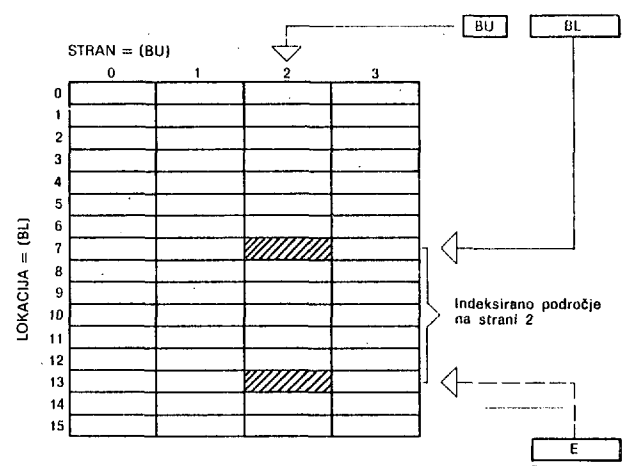


Slika 1

POMNILNIK RAM

256 bitov pomnilnika RAM je namenjenih za hranjenje vmesnih rezultatov programma. Pomnilnik je razdeljen na 4 strani, od katerih ima vsaka 16 lokacij. Besede v posameznih lokacijah so dolge 4 bite. Pomnilnik RAM naslavljamo z registroma BL (4 biti) in BU (2 bita). Register BL naslavlja 16 besed v okviru ene strani. Ta register lahko uporabljamo tudi kot splošni register in ga spreminjamo z aritmetičnimi ukazi. Na voljo so ukazi za polnenje registra BL, izmenjavo z akumulatorjem, povečanje za 1 in zmanjšanje za 1. Prek dekoderja se register BL uporablja tudi za formiranje izhodov na linijah vodila A.

Preostala dva bita naslova v pomnilniku RAM sta določena v registru BU. Operacije v registru BU se izvršijo hkrati z nekaterimi operacijami v drugih registrih. Tako imamo npr. nekaj ukazov, ki poleg osnovne operacije spremenijo tudi register BU.



Slika 2. Organizacija pomnilnika RAM

V pomnilniku RAM poleg besednih operacij s 4 biti izvajamo tudi operacije nad vsakim posameznim bitom. Pri naslavljanju pomnilnika RAM ima določeno vlogo tudi register E. Služi lahko kot splošni register (tako kot register BL), poleg tega pa je z njim možno nastaviti in nadzorovati vsebino registra BL in s tem indeksiranje v pomnilniku RAM.

Kadar v pomnilniku RAM ne želimo uporabljati posameznih bitov, lahko uporabimo direktno naslovljivi kretnici F1 in F2, ki sta neodvisni od vseh ostalih registrov.

POMNILNIK ROM

Naslovni prostor programskega pomnilnika je razdeljen na 8 bank, od katerih vsaka vsebuje 1K besed. Banka 0 je tisti del pomnilnika, ki se nahaja že na sami ploščici, ostale banke pa po želji dodajamo od zunaj.

Vsaka banka v pomnilniku ROM je razdeljena na 16 strani in vsaka stran vsebuje 64 besed. Programski števec je dolg 13 bitov: spodnjih 6 bitov naslavlja lokacije v okviru ene strani, naslednji 4 biti določajo stran, zadnji trije biti z največjo težo pa določajo banko. Ob priključitvi napajanja programski števec kaže banko 0, stran 0 in lokacijo prav tako 0.

Po izvršitvi vsakega ukaza se programski števec poveča za 1 in tako adresira naslednji ukaz v programu. Izjemo predstavljajo edino skočni ukazi (JMP x, JMS x, RT in RTS). Ukaz JMP skoči na naslov x na tisti strani, kjer je ta ukaz zapisan; ukaz JMS x skoči na naslov x na zadnji strani (stran 15) banke, v kateri je zapisan. Naslove na drugih straneh dosežemo tako, da neposredno pred ukazom JMP oz. JMS zapišemo ukaz za pripravo strani (ukaz PP x).

Ukaz za klicanje subrutin (JMS x) shrani v odlagalniku naslov ukaza, ki sledi ukazu JMS x. Ko se izvrši ukaz za vračanje (RT ali RTS), se ta naslov prenese iz odlagalnika nazaj v programski števec. Odlagalnik ima globino za 3 nivoje subrutin, tj. hrani lahko tri različne povratne naslove hkrati.

VODILO A

Izhodi na vodilu A so namenjeni splošnemu nadzoru nad zunanjimi priključenimi elementi, strobiranju izpisa in strobiranju tipk. Izvor podatkov za vodilo A je dekodirani register BL. Tega registra ne uporabljamo le za naslavljanje v pomnilniku RAM, temveč tudi za generiranje stanj na vodilu A. Ukaza PSH in PSL nastavita posamezne bite v izhodnem držalu na vrednosti "0" ali "1". Dekodirana vrednost registra BL pove, kateri bit se bo nastavil. Na vodilo A ne moremo neposredno pošiljati binarne vsebine delovnih registrov, pač pa moramo posebej nastaviti vsak bit v izhodnem držalu. Prenos iz registra tipa "master-slave" na vodilo A izvršimo z ukazom MVS za vseh 13 bitov istočasno. Spremembe, ki smo jih individualno nastavljali za vsak bit posebej, se torej prenesejo na izhod vzporedno z enim samim ukazom. Ukaz MVS povzroči tudi prehod vodila D v nevtralno stanje.

Če ne potrebujemo večjega števila mest v izpisu številk, lahko krmilimo segmente in katode kazalnika LED neposredno iz vodil A in D. V tem primeru vodilo D predstavlja tokovni izvor za tiste segmente, ki so osvetljeni, vodilo A pa uporabimo kot ponor toka na katodah vsake posamezne številke. Za to funkcijo so zlasti primerni izvodi A0...A3, ki lahko požirajo tudi do 25 mA. Če uporabljamo izpis s skupno anodo ali obračalne ojačevalnike, lahko polariteto linij A in D programsko spremenimo.

VODILO D

Vodilo D ima tri možna stanja: logični stanji "1" in "0", ter nevtralno stanje velike upornosti ("floating"). Običajno za prenos vhodnih podatkov postavimo vodilo D v nevtralno stanje, tako da prek njega zunanje enote lahko posredujejo svoja logična stanja. Zunanjo enoto izberemo s posebnim ukazom (MVS), ki avtomatično postavi vodilo D v nevtralni položaj.

Vodilo D je v nevtralnem položaju tudi vedno po vklopu napajanja, kadar je izvod RUN vezan na maso in v času T13 multipleksnega delovanja. Nevtralni položaj vodila D je moč regulirati tudi pod vplivom programa.

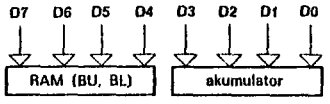
Vodilo D je s svojimi osmimi biti namenjeno predvsem za prenose I/O med standardnimi vezji z 8 biti ter za krmiljenje segmentov pri izpisu številk. Izhodi na vodilu D so vedno vzporedni za vseh 8 bitov.

Vhod podatkov programiramo z ukazom INP. Ta ukaz prenese bite D0...D3 v akumulator, bite D4...D7 pa v pomnilnik RAM na tisto lokacijo,

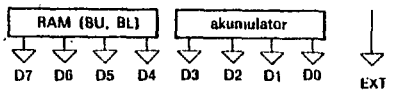
ki je trenutno določena z registroma BU, BL. Iste bite z ukazom OUT prenesemo iz akumulatorja in pomnilnika RAM na vodilo D. V času izvajanja ukaza OUT se na izvodu EXT pojavi strobirni impulz, s katerim zagotovimo sprejem podatkov v pravem trenutku. Po izvršenem ukazu OUT se vodilo D vrne v prejšnje stanje.

Za krmiljenje segmentov izpisa se poslužujemo ukazov DISN (display number) ali DISB (display binary). V obeh primerih se izhodni podatki shranijo v držalu D in so od takrat naprej stalno prisotni na vodilu D, dokler s kakim ukazom ne povzročimo prehoda vodila v nevtralno stanje. Ukaz DISN pretvori vsebino akumulatorja v 7-bitno segmentno kodo. Osmi bit se prenese iz registra C in ga običajno uporabimo za krmiljenje decimalne pike. Ukaz DISB prenese vsebino akumulatorja in vsebino pomnilnika RAM skupaj, brez segmentnega kodiranja. Primeren je za izpis poljubnih binarnih vzorcev.

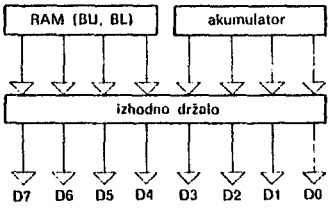
Vhod podatkov pri ukazu INP



Izhod podatkov pri ukazu OUT



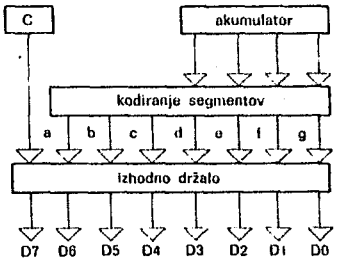
Izhod podatkov pri ukazu DISB

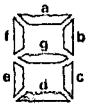


Kodiranje pri ukazu DISN (normalna polarizacija)

Vsebina akumulatorja	a D6	b D5	c D4	d D3	e D2	f D1	g D0	Izpis
0	1	1	1	1	1	1	0	0
1	0	1	1	0	0	0	0	1
2	1	1	0	1	1	0	1	2
3	1	1	1	1	0	0	1	3
4	0	1	1	0	0	1	1	4
5	1	0	1	1	0	1	1	5
6	1	0	1	1	1	1	1	6
7	1	1	1	0	0	0	0	7
8	1	1	1	1	1	1	1	8
9	1	1	1	1	0	1	1	9
10	1	1	1	0	1	1	1	A
11	0	0	1	1	1	1	1	b
12	1	0	0	1	1	1	0	C
13	0	1	1	1	1	0	1	d
14	1	0	0	1	1	1	1	E
15	1	0	0	0	1	1	1	F

Izhod podatkov pri ukazu DISN





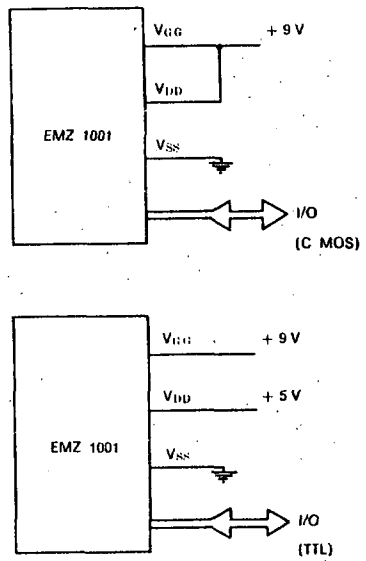
Slika 3. Operacije na vodilu D

VHODI I IN K

Štirje biti vhoda K in štirje biti vhoda I prenašajo v notranjost mikroračunalnika logična stanja posameznih diskretnih spremenljivk. Preko njiju ne moremo vstopati s paralelnimi podatki v notranje registre, pač pa preko skočnih ukazov SZI in SZK krmilimo programski tek. Posamezen vhod izberemo programsko.

Vhodi I in K se med seboj hardversko razlikujejo. Vhodi K imajo veliko notranjo upornost, poleg tega pa se ojačujejo še v posebni diferencialni stopnji z zunanjim izvodom KREF. Tako je mogoče nastaviti prag med logično 0 in 1, kar je zlasti ugodno, če priključujemo kapacitivne tipke ("Touchmos"). Diferencialna vhodna stopnja omogoča tudi gradnjo nezahtevnih pretvornikov A/D z minimalnim številom dodatnih elementov.

Vhod I8 je vezan še na poseben element, na generator sekundnih impulzov. Če vhod I8 krmilimo s signalom 50 ali 60 Hz, to vezje generira sekundne impulze, katere lahko programsko testiramo z ukazom SOS. S tem smo omogočili enostavno programiranje časovnih funkcij na osnovi omrežne frekvence.



Slika 4. Dva načina napajanja EMZ 1001

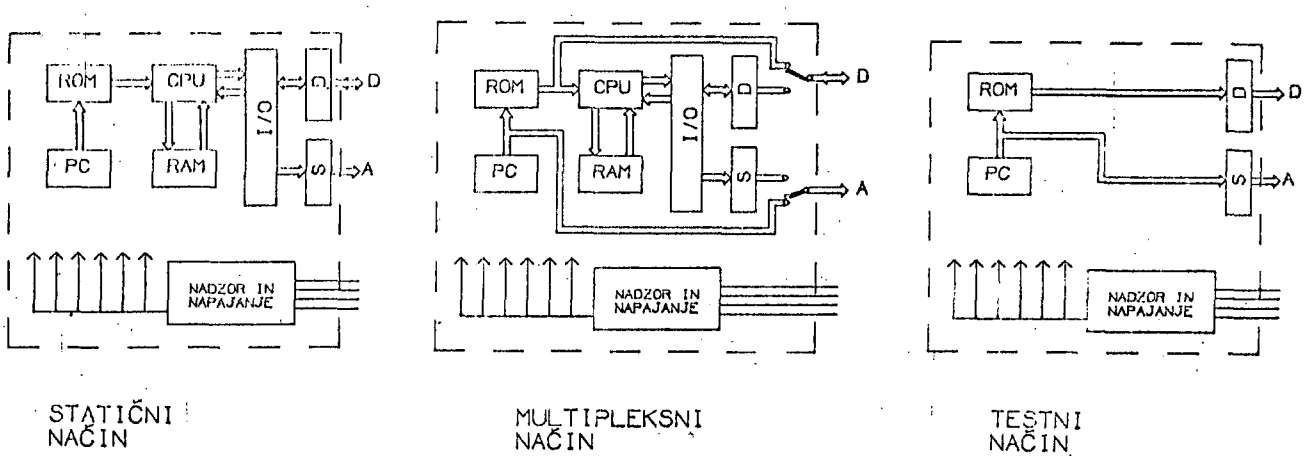
NADZORNI IZVODI

Mikroračunalnik EMZ 1001 je opremljen z dodatnimi izvodi, preko katerih določamo, na kak način naj vezje deluje.

- Izvod ROMS določa izbiro pomnilnika ROM in prehode med statičnim, multipleksnim ter testnim delovenjem. Več o tem bomo pojasnili v naslednjem rezdelku.
- Izvod RUN v logični "1" omogoča normalno obratovanje, v logični "0" pa zaustavi programski tek. Ta izvod je zlasti koristen pri testiranju, ker preko njega dosežemo koračno delovanje (t.im. "single - step").
- Izvod POR služi za resetiranje sistema, največkrat v trenutku vklopa napajalne napetosti.
- Izvod VDD določa visoke napetostne nivoje za vse izhodne signale. Povežemo ga lahko z glavno napajalno napetostjo VGG (+9 V), ali pa s kako nižjo napetostjo, npr. +5 V. Na tak način mikroračunalnik EMZ 1001 povežemo z vezji, ki so TTL kompatibilna.

RAZVOJNA OPREMA, TESTIRANJE IN RAZŠIRITVE

Mikroračunalnik EMZ 1001 lahko deluje v treh različnih načinih delovanja, prikazanih na sliki 5. Razlike nastopajo predvsem v povezavi pomnilnika ROM s procesno enoto in v načinu prenašanja podatkov po vodilih A in D.



Slika 5. Poenostavljen prikaz mikroračunalnika EMZ 1001 v treh osnovnih načinih delovanja.

Statični način predstavlja normalni način uporabe vezja. Procesor izvršuje program iz notranjega pomnilnika ROM, izvodi pa prenašajo podatke I/O. Ker sta vodili A in D opremljeni z držali, lahko držimo podatke dalj časa v stabilni obliki.

Multipleksni način se odlikuje po tem, da osnovni cikel vsakega ukaza razpade na dve polovici: na čas T13 in na čas T57. V času T13 imamo na vodilu A programsko adreso, na vodilu D pa ukaz iz priključenega zunanjega pomnilnika. V času T57 izvodi preklopijo nazaj na podatke I/O in vezje deluje popolnoma enako kot v statičnem načinu.

Posredovanje programskih podatkov preko vodil A in D omogoča, da EMZ 1001 lahko dela z notranjim, z zunanjim (notranji izključen) ali z obema pomnilnikoma ROM (notranji je v tem primeru razširjen z zunanjim). Posamezne možnosti izberemo z ustreznimi prevezavami izvoda ROMS.

Multipleksni način je namenjen predvsem testiranju in eksperimentiranju, uporabiti pa ga moramo tudi pri razširitvah pomnilnika ROM .

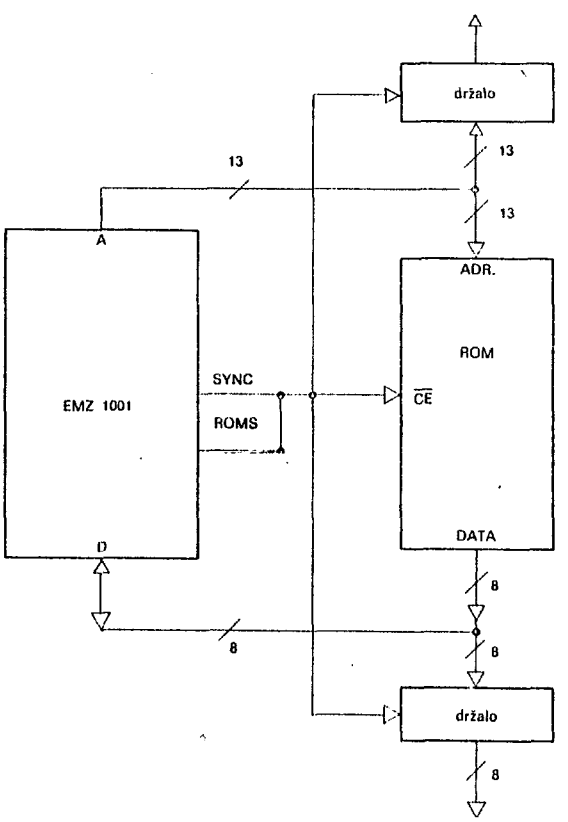
Testni način. V tem načinu delovanja mikroračunalnik ne izvršuje ukazov. Programski števec avtomatično šteje in adresira notranji pomnilnik ROM. Adrese zasledujemo na vodilu A, vsebino pomnilnika pa na vodilu D. Vezje se kaže kot samonaslovljivi pomnilnik ROM in v 1024 ukaznih ciklih posreduje celo vsebino notranjega pomnilnika. Tudi testni način delovanja določimo s posebno prevezavo izvoda ROMS.

Z uporabo navedenih načinov delovanja lahko problem testiranja zelo učinkovito rešimo. Pomembno je predvsem to, da je možno uporabiti isto strategijo za vsa vezja, neodvisno od vpisanega programa. Poslužimo se naslednjih postopkov:

1. Z uporabo multipleksnega načina izključimo notranji pomnilnik ROM in vezje povežemo z zunanjim pomnilnikom, ki vsebuje standardni testni program. Ta testira posamezne logične module, kot so pomnilnik RAM, registri I/O, aritmetična enota in procesna enota. Na isti način preverimo tudi povezave med omenjenimi moduli.

2. V testnem načinu preverimo vsebino notranjega pomnilnika ROM. Tu gre za enostavno primerjavo dveh tabel, ki zahteva borih 4,6 ms preizkusnega časa.

Če se poslužimo multipleksnega načina delovanja, nas razmeroma lahko delo čaka tudi pri razvoju in preizkušanju prototipov. Notranji pomnilnik nas v tem primeru sploh ne zanima, zato ga bomo izključili in nadomestili z zunanjimi pomnilniki (verjetno tipa EPROM). Multipleksiranje podatkov I/O na izvodih odpravimo z dodatnimi zunanjimi držali, s katerimi signali postanejo zopet enakovredni signalom, kakršni nastopajo v statičnem načinu delovanja. Tako dobimo popolnoma enako sliko, kot če bi uporabili vezje z notranjim pomnilnikom. Poleg tega, da prototipni program sedaj lahko menjavamo, imamo tudi možnosti za uporabo standardnih pomagal, kot so lovci adres (breakpoints), logični analizatorji, koračno delovanje itd.



Slika 6. Priključitev zunanjega pomnilnika ROM

Na sliki 7 je prikazan koncept hierarhično zasnovanega razvojnega sistema, ki izhaja iz multipleksnega načina delovanja mikroračunalnika EMZ 1001. Na najnižjem nivoju, ki smo ga imenovali " statični emulator ", poskrbimo za razmejitev med multipleksnim načinom in med signali statičnega načina delovanja, katere vodimo v prototip. Nadgradnjo predstavljata adresna in ukazna enota, ki posredujeta stanja na vodilih A in D in hkrati dovoljujeta aktivne posege v izvajanje programa.

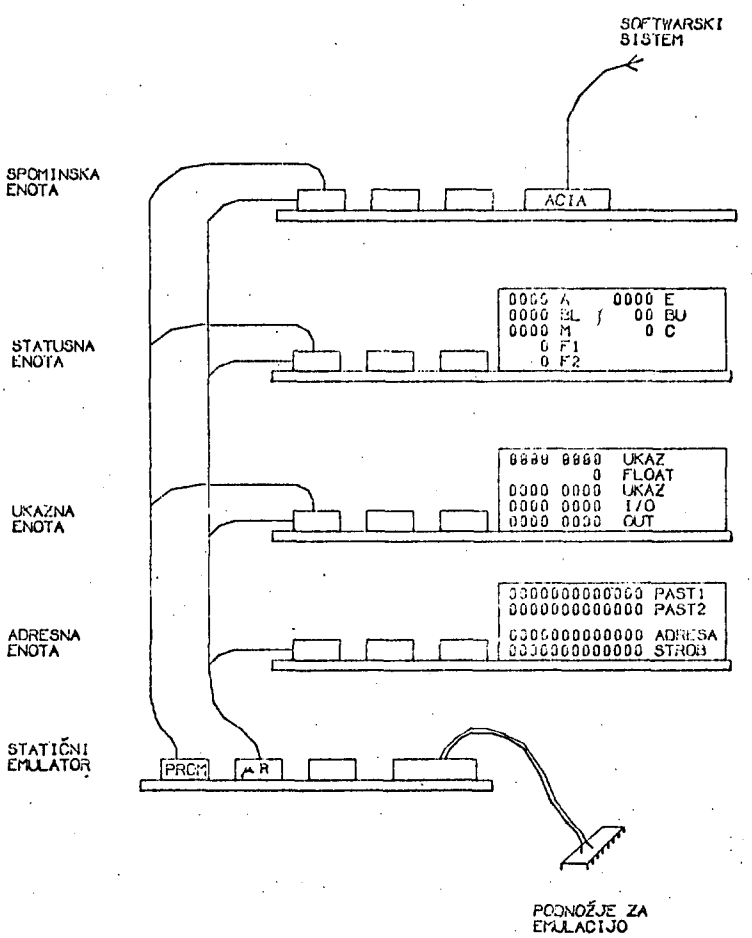
Nadalje smo predvideli še statusno enoto, ki omogoča izpis in menjavo stanja notranjih registrov procesne enote v točkah, kjer smo program ustavili. Spominska enota je namenjena hitremu nalaganju programov, ki prihajo iz podpornega softverskega sistema pri sprotnem popravljanju in spreminjanju delov aplikativnega programa.

MOŽNOSTI UPORABE

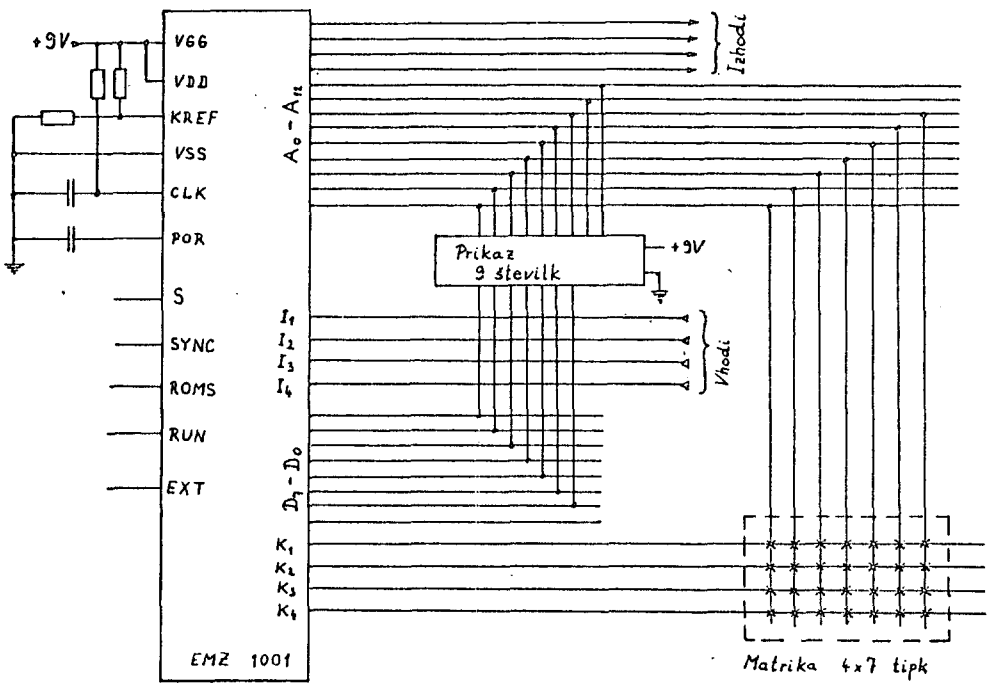
Uporaba mikroračunalnika EMZ 1001 bo prinesla ekonomske in tehnične prednosti zlasti na področju izdelkov, ki se proizvajajo v večjem številu. To so npr.

- gospodinjski avtomati,
- elektronske tehtnice,
- merilni instrumenti,
- elektronske igre in igralni avtomati,
- blagajne,
- registratorji dogodkov,
- enote za zbiranje vzorčnih podatkov,
- programatorji, itd.

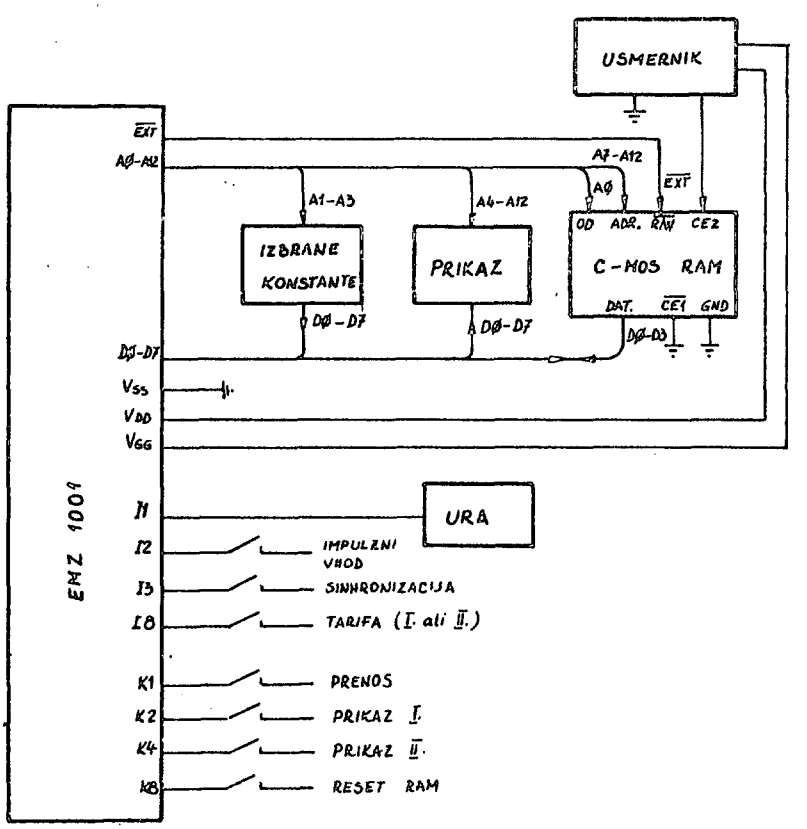
Elektronska oprema takih sistemov naj bi bila v največji možni meri združena v enem samem vezju, v integriranem mikroračunalniku. V ilustracijo navajamo dva primera. Na sliki 8 vidimo enostaven sistem, podoben žepnemu kalkulatorju. Za izbiranje operacij je predvidenih 28 tipk, ki so povezane v matrični organizaciji 4 x 7. Podatki se izpisujejo s pomočjo devetih kazalnikov LED, katere naslavljamo z istimi biti vodila A kot naslavljamo tudi posamezne stolpce tipk. Mikroračunalnik uporablja samo notranje pomnilnike in deluje zato v statičnem načinu. Oblike signalov na vodilu A in D so pod polnim nadzorom notranjega programa. Linije A0...A8 programiramo tako, da se na njih pojavijo zaporedni strobirni impulzi, ki serijsko vključujejo posamezne kazalnike za prikaz številk. V nekem danem času je vedno vključen en sam kazalnik. Segmentno kodo za izpis posredujemo preko vodila D. Hkrati s tem na vodilih K testiramo pripadajoče vhode iz stolpca tipk, ki se vzbuja z istim strobirnim signalom.



Slika 7. Hierarhično zasnovani razvojni sistem.



Slika 8. Primer enostavnega sistema z mikroračunalnikom.



Slika 9. Primer uporabe mikroračunalnika EMZ 1001 v merilniku elektroenergetskih konic.

Čas trajanja posameznih strobirnih impulzov v programu omejimo na 1...2 ms, tako da zaradi hitrega zaporedja dobimo vizualni občutek istočasne prisotnosti vseh izpisanih številk. Poleg tipk in številčnega izpisa imamo še 4 vhode in 4 izhode za diskretne spremenljivke.

Slika 9 prikazuje blokovno shemo merilnika elektroenergetskih konic, kakršen se uporablja za nadzor pri večjih porabnikih. Priključena ura nam daje časovne impulze na vhod I1, merilnik moči pa na vhod I2. Računalnik zasleduje porabo moči v času s štetjem impulzov v notranjem pomnilniku RAM. Pri tem uporablja določene konstante, ki so mu dostopne preko vodila D in selektorskih impulzov na linijah A1...A3. Prikaz trenutnih in maksimalnih vrednosti poteka na segmentiranih kazalnikih LED, ki jih napajamo preko vodila D in A. Posebno kritični podatki se hranijo še v zunanjem pomnilniku CMOS RAM, ki ima rezervno baterijsko napajanje.

Na osnovi vseh prikazanih lastnosti mikroračunalnika EMZ 1001 ne bo težko narediti osnutka električne sheme za kakšno podobno novo aplikacijo. Najbolj smiselne so rešitve, v katerih zadošča število razpoložljivih vodil, s katerimi neposredno krmilimo naš sistem. Nekaj več dela zahteva od nas analiza algoritmov in podatkov, na osnovi katere ugotovimo, če sta razpoložljiva pomnilnika ROM in RAM dovolj velika za dano nalogo.

Dosedanje izkušnje so pokazale, da imamo precej primerov, ko pomnilnika zadoščata in ko sta njuni velikosti medsebojno dovolj dobro uravnoteženi. To potrjujejo tudi uspešne realizacije sistemov v različnih področjih, kot so komunikacije, regulacija in merilna tehnika.

DODATEK - NABOR UKAZOV EMZ 1001

Medregisterski ukazi

Mnemonika	Operand	Opis	Predstavitev
LAB		Napolni ACC z vsebino BL	BL → ACC
LAE		Napolni ACC z vsebino E	E → ACC
LAI	X	Napolni ACC s konstanto* Izberi vhode K in vhode I	X → ACC 0 ≦ X ≦ 15 X → vhodna maska
LBE	Y	Napolni BL z vsebino E, BU pa s konstanto Y	E → BL Y → BU 0 ≦ Y ≦ 3
LBEP	Y	Napolni BL z vsebino E, povečano za 1, BU pa s konstanto Y*	E + 1 → BL Y → BU 0 ≦ Y ≦ 3
LBF	Y	Napolni BL z 1111, BU pa s konstanto Y*	15 → BL Y → BU 0 ≦ Y ≦ 3
LBZ	Y	Napolni BL z 0000, BU pa s konstanto Y*	0 → BL Y → BU 0 ≦ Y ≦ 3
XAB		Zamenjamo vsebini ACC in BL	BL ←→ ACC
XABU		Zamenjamo vsebini ACC (biti 0, 1) in BU. ACC (2, 3) *se ne spremeni*	BU ←→ ACC (0, 1)
XAE		Zamenjaj vsebini ACC in E	E ←→ ACC
STC		Napolni C z »1«	1 → C
RSC		Napolni C z »0«	0 → C
SF1		Napolni kretnico F1 z »1«	1 → F1
RF1		Napolni kretnico F1 z »0«	0 → F1
SF2		Napolni kretnico F2 z »1«	1 → F2
RF2		Napolni kretnico F2 z »0«	0 → F2

* Prvi ukaz po vklopu napajanja ne sme biti tipa LB ali LAI. Kadar program vsebuje zaporedje ukazov LAI ali (LB-), se v tem zaporedju vrši samo prvi ukaz.

Ukazi za delo s pomnilnikom RAM

V vseh navedenih ukazih uporabljamo kratico M za označitev tiste lokacije v pomnilniku RAM, ki jo določata registra BU in BL.

Mnemonika	Operand	Opis	Predstavitev
LAM	Y	Napolni ACC z M, nato spremeni BU	M → ACC BU ⊕ Y → BU
XC	Y	Zamenjaj vsebini ACC in M in nato spremeni BU	M ←→ ACC BU ⊕ Y → BU
XCI	Y	Zamenjaj vsebini ACC in M, nato povečaj BL za 1 in spremeni BU. Preskoči, če po povečanju velja BL = 0*	M ←→ ACC BL + 1 → BL BU ⊕ Y → BU IF BL = 0 SKIP*
XCD	Y	Zamenjaj vsebini ACC in M, nato zmanjšaj BL za 1 in spremeni BU. Preskoči, če po zmanjšanju velja BL = 15*	M ←→ ACC BL — 1 → BL BU ∓ Y → BU IF BL = 15 SKIP*
STM	B	Napolni izbrani bit v celici M z »1«	1 → M (bit B) 0 ≦ B ≦ 3
RSM	B	Napolni izbrani bit v celici M z »0«	0 → M (bit B) 0 ≦ B ≦ 3

Posamezni biti v celici M so izbrani takole:

MSB			LSB
B = 3	B = 2	B = 1	B = 0

Spremembe registra BU potekajo z operacijo XOR:

		stari BU				
		0	1	2	3	
novi BU	0	0	1	2	3	Y
	1	1	0	3	2	
	2	2	3	0	1	
	3	3	2	1	0	

Aritmetični in logični ukazi

Mnemonika	Operand	Opis	Predstavitev
ADD		Seštej ACC in M. C se ne spremeni.	M + ACC → ACC
ADCS	X	Seštej ACC, M in C. Preskoči, če je vsota manjša od 16.*	M + ACC + C → ACC, C IF SUM ≦ 15 SKIP
ADIS	X	Seštej ACC in konstanto X. C se ne spremeni. Preskoči, če je vsota manjša od 16.*	X + ACC → ACC IF SUM ≦ 15 SKIP
AND		Logični IN med ACC in M.	M & ACC → ACC
XOR		Logični ekskluzivni ALI med ACC in M.	M ⊕ ACC → ACC
CMA		Logični eniški komplement ACC.	15 — ACC → ACC

Ukazi za preskoke*

Mnemonika	Operand	Opis	Predstavitev
SAM		Preskoči, če ACC = M	IF ACC = M SKIP
SZM	B	Preskoči, če M (bit B) = 0	IF M (bit B) = 0 SKIP 0 ≦ B ≦ 3
SBE		Preskoči, če BL = E	IF BL = E SKIP
SZC		Preskoči če C = 0	IF C = 0 SKIP
SOS		Preskoči, če je sekundni izhod časovnika enak 1. Po preskoku vrni izhod časovnika na 0.	IF SF = 1 SKIP 0 → SF
SZK		Preskoči, če je vhod K enak 0. Vhodi K so izbrani z argumentom zadnjega izvršenega ukaza LAI. Neizbrani vhodi so zvezani proti masi. Za preskok je potrebno, da so izbrani vhodi K ≦ KREF.	IF $K_{8,4,2,1}$ ≦ KREF SKIP
SZI		Preskoči, če je vhod I enak 0. Vhodi I so izbrani z argumenta zadnjega izvršenega ukaza LAI.	IF $I_{8,4,2,1}$ = 0 SKIP
TF1		Preskoči, če je kretnica F1 = 1	IF F1 = 1 SKIP
TF2		Preskoči, če je kretnica F2 = 1	IF F2 = 1 SKIP

* Kadar pride do preskoka, se preskoči prvi naslednji ukaz. Če preskočimo ukaz PP, se preskoči še naslednji ukaz.

Ukazi za usmerjanje programa

Mnemonika	Operand	Opis	Predstavitev
PP	Y	Pripravi naslednjo stran če predhodni ukaz ni bil PP. Pripravi naslednjo banko, če je predhodni ukaz bil PP.	Y → (PPR/PBR)
JMP	X	Skoči na lokacijo X na isti strani, kjer je napisan ukaz. Če je predhodni ukaz bil PP, spremeni registra za stran in banko z vsebino pripravljenih registrov. Uporabljajo se naslednje 3 sekvence: 1. Skok na isti strani JMP LL 2. Skok na neko stran v isti banki PP LP/64 JMP LP 3. Skok v drugo banko PP LB/64 PP LB/1024 JMP LB	X → LR 0 ≦ X ≦ 63 X → LR PPR → PR PBR → BR
JMS	X	Skoči na lokacijo X na strani 15. Shrani PR + 1 in LR + 1 v odlagalnik. Če je predhodni ukaz bil PP, spremeni registra za stran in banko z vsebino pripravljenih registrov. Uporabljata se dve sekvenci: 1. Klic subrutine na strani 15 JMS LS 2. Klic subrutine na poljubni strani v isti banki PP LP/64 JMS LP	LR + 1 → L STACK PR → P STACK X → LR 15 → PR LR + 1 → L STACK PR → P STACK X → LR PPR → PR PBR → BR
RT		Vračanje iz subrutine	L STACK → LR P STACK → PR
RTS		Vračanje iz subrutine in preskok prvega ukaza v glavnem programu.*	L STACK → LR P STACK → PR SKIP
NOP		Neoperativna koda.	

Ukazi za vhode in izhode

Mnemonika	Opis
INP	Prenos podatkov iz vodila D v ACC in pomnilnik RAM. D_{3-0} → ACC, D_{7-4} → M
OUT	Izhod podatkov iz ACC in pomnilnika na vodilo D. V času prisotnosti podatkov se generira impulz $\overline{EXT}$. Podatkovni register D ostane nespremenjen. ACC → D_{3-0}, M → D_{7-4}
DISB	Binarni prenos podatkov iz ACC in M v register D. Poveži register D z vodilom D in ukini nevtralno stanje. ACC → reg. D (3—0) → vodilo D_{3-0} M → reg. D (7—4) → vodilo D_{7-4} Izhod registra lahko invertiramo z ukazom EUR.
DISN	Prenos vsebine ACC, kodirane v segmente za prikaz numerične vrednosti. Kodo zapiši v register D. Poveži register D z vodilom D in ukini nevtralno stanje. ACC → koda → reg. D (6—0) → vodilo D_{6-0} C → reg. D (7) → vodilo D_7
MVS	Prenos iz registra A-master v register A-slave. Prehod vodila D v nevtralno stanje. A_{0-12} (master) → A_{0-12} (slave)
PSH	Zapiši »1« v izbrani bit registra A (master). Izbiro bita določa dekodirana vrednost registra BL. IF 0 ≦ BL ≦ 12 : 1 → bit (BL) registrira A (master) IF BL = 13 : nastavi multipleksni način delovanja IF BL = 14 : ukini nevtralno stanje v vodilu D IF BL = 15 : 1 → vse bite registrira A (master)
PSL	Zapiši »0« v izbrani bit registra A (master). Izbiro bita določa dekodirana vrednost registra BL. IF 0 ≦ BL ≦ 12 : 0 → bit (BL) registrira A (master) IF BL = 13 : nastavi statični način delovanja IF BL = 14 : nastavi nevtralno stanje v vodilu D IF BL = 15 : 0 → vse bite registrira A (master)
EUR	Pomožni ukaz za programski nadzor nad polarizacijo registra D in nad modulom štetja v časovniku. Operacije so določene s pomočjo vsebine akumulatorja. ACC bit 0 = 1 normalna polarizacija za register D (isto kot po vklopu napajanja) = 0 invertirani register D ACC bit 3 = 1 50 Hz = 0 60 Hz (isto kot po vklopu napajanja)

MIKRORAČUNALNIŠKA KRIPTOGRAFIJA I

A. P. ŽELEZNIKAR

UDK: 681.3.06 : 003.6

INŠTITUT JOŽEF STEFAN, LJUBLJANA

ČLANEK OPISUJE OSNOVNE POJME KRIPTOGRAFIJE, TERMINOLOGIJO TER KRIPCIJSKE POSTOPKE. OPISANA JE OSNOVNA SHEMA INKRIPCIJE IN DEKRIPCIJE IN V PREGLEDNI OBLIKI SO PRIKAZANE T.I. ROČNA, PREMESTITVENA, SUBSTITUCIJSKA, PLAYFAIRJEVA (WHEATSTONOVA), VERNAMOVA KRIPCIJA TER KRIPCIJA Z UPORABO PASTI. NADALJE JE OPISAN T.I. STANDARDNI KRIPCIJSKI ALGORITEM, KI JE OSVETLJEN Z VEČ VIDIKOV, ŠE POSEBEJ Z VIDIKA NJEGOVEGA PROGRAMIRANJA NA MIKRO RAČUNALNIKIH. V SKLEPNEM DELU SO NAVEDENE BISTVENE UGOTOVITVE GLEDE NA MOŽNOSTI MIKRORAČUNALNIŠKE IMPLEMENTACIJE STANDARDNEGA KRIPCIJSKEGA ALGORITMA. ČLANEK PREDSTAVLJA LE PRVI DEL ALI UVOD V MIKRORAČUNALNIŠKO KRIPCIJO. KONKRETNO PROGRAMIRANJE IN UPORABA POSEBNIH INTEGRIRANIH VEZIJ BO OPISANA V NADALJEVANJIH TEGA ČLANKA.

CRYPTOGRAPHY USING MICROCOMPUTERS I. THIS ARTICLE DEALS WITH FUNDAMENTALS OF CRYPTOGRPHY, TERMINOLOGY AND WITH ENCRYPTION PROCESSES. BASIC SCHEME OF ENCRYPTION AND DECRYPTION IS DESCRIBED AND SOME METHODS LIKE FIELD, TRANSPOSITION, SUBSTITUTION, PLAYFAIR, VERNAM, AND TRAP DOOR ARE PRESENTED. THE DATA ENCRYPTION STANDARD (DES) ALGORITHM IS DESCRIBED FROM SEVERAL POINTS OF VIEW AND HIS PROGRAMMING POSSIBILITIES ARE DISCUSSED. IN THE CONCLUSION OF THE ARTICLE FEASIBILITIES OF DES ALGORITHM IMPLEMENTATION IN VARIOUS WAYS (HARDWARE AND SOFTWARE) ARE POINTED OUT. THIS ARTICLE REPRESENTS INTRODUCTION TO THE MICROCOMPUTER CRYPTION. PROGRAMMING AND ENCRYPTION INTEGRATED CIRCUITS APPLICATION WILL BE PRESENTED IN THE ARTICLES FOLLOWING THIS ONE.

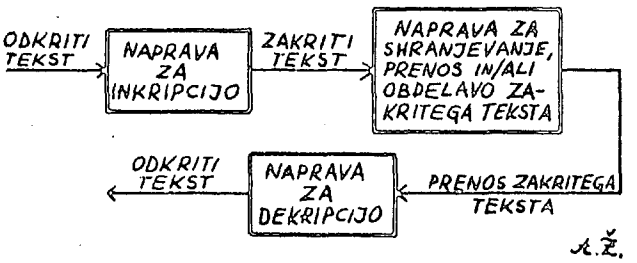
1. UVOD

V ČASOPISU INFORMATICA SMO ŽE PISALI O SPLOŠNI PROBLEMATIKI, MOŽNOSTIH IN UPORABI KRIPTOGRAFIJE (1). DANES ŽE IMAMO NA TRŽIŠČU VRSTO NAPRAV ZA T.I. KOMERCIALNO INKRIPCIJO IN DEKRIPCIJO PODATKOV, KI TEMELJIJO NA UPORABI T.I. STANDARDNEGA PODATKOVNEGA KRIPCIJSKEGA ALGORITMA (DES JE KRATICA ZA DATA ENCRIPTION STANDARD). NA VOLJO SO TUDI ŽE POSEBNA INTEGRIRANA VEZJA (NPR. DRUŽINA VEZIJ 9114 IN VEZJE 8294), S KATERIMI JE MOGOČE OB DOLOČENI PROGRAMSKI PODPORI REALIZIRATI ZAPLETENE TER VARNE IN ZANESLJIVE KRIPCIJSKE ALGORITME. STANDARDNI (KOMERCIALNI) ALGORITEM JE BIL SPREJET IN ZAŠČITEN V ZDA TER OBJAVLJEN V FIPS PUBLIKACIJI 46 (US DEPARTMENT OF COMMERCE). TA ALGORITEM JE POSTAL V ZDA OSNOVA OZIROMA STANDARD PRI RAZVOJU KRIPCIJSKE TEHNOLOGIJE, TO JE INTEGRIRANIH VEZJIH IN NAPRAV ZA SHRANJEVANJE, PRENOS IN OBDELAVO TEKSTA Z NAMENOM INKRIPCIJE IN DEKRIPCIJE.

BESEDA KRIPTOGRAFIJA JE PRIVZETA IZ GRŠČINE IN POMENI TAJNO PISAVO PA TUDI UMETNOST PRESLIKAVE ODKRITE INFORMACIJE V ZAKRITO OBLIKO. KRIPTOANALIZA PA JE TEHNIKA RAZKRIVANJA PRVOTNE, ODKRITE INFORMACIJE IZ KRIPTOGRAMA Z UPORABO RAZLIČNIH PRIPOMOČKOV; TO RAZKRIVANJE IMENUJEMO VČASIH TUDI LOMLJENJE KODA.

KRIPTOGRAFIJA VELJA ZA UMETNOST TAJNE PISAVE, KI BI JO NAJ OBVLADALI LE POSVEČENI, IZBRANI. KRIPTOGRAFSKI ELEMENT ALI ZNAK IMENUJEMO ŠIFRA; TA BESEDA JE IZPELJANA IZ FRANCOSKE BESEDE ZA ŠTEVILKO (CIFRA). ŠIFRA JE TAJNI ZNAK (ŠTEVILKA) ALI TUDI ZAPOREDJE TAJNIH ZNAKOV.

UMETNOST KODIRANJA (ŠIFRIRANJA) JE BILA ZNANA ŽE V STARI GRČIJI. ŠPARTANSKI ŠIFRANT JE NAVIL SVOJ PAS SPIRALNO NA OKROGLO PALICO, SPOROČILO PA JE NAPISAL NA PAS PONAVLJAJOČE OD VRHA NAVZDOL NAVITEGA PASU. RAZVITI PAS JE VSEBOVAL SPOROČILO BREZ POMENA, ČE SO GA ČITALI OD LEVE PROTI DESNI. TU JE BIL ŠIFRIRNI KLJUČ



SLIKA 1. OSNOVNA SHEMA KRIPTIVNE NAPRAVE

ODVISEN OD PREMERA PALICE. SPREJEMNIK JE PAS ZOPET NAVIL NA PALICO IN SPOROČILO PREČITAL. TUDI JULIJ CEZAR JE UPORABLJAL ŠIFRIRANJE IN JE VSAKO ČRKO SPOROČILA ZAMENJAL S ČRKO, KI JE BILA V ABECEDI ZA TRI MESTA POMAKNJENA V DESNO; NPR. ZA

JULIJ JE DOBIL BESEDO MYOLM

METODA S PASOM JE ČRKE PREMESTILA (PERMUTIRALA), CEZARJEVA METODA PA JIH JE ZAMENJALA (SUBSTITUIRALA).

IZ BESEDE KRIPCIJA JE MOČ IZPELJATI BESEDI INKRIPCIJA IN DEKRIPCIJA. INKRIPCIJA POMENI TRANSFORMACIJO RAZUMLJIVEGA, ODPRTEGA TEKSTA ALI ZAPOREDJA ZNAKOV V ZAPOREDJE ŠIFER, KI JE NERAZUMLJIVO OZIROMA PREDSTAVLJA ZAKRITJE PRVOTNO ODKRITEGA TEKSTA. DEKRIPCIJA JE INVERZNI POSTOPEK K DANI INKRIPCIJI IN OMOGOČA, DA DOBIMO IZ ZAKRITEGA TEKSTA PRVOTNI, POMENSKO ODKRITI TEKST. UPORABLJAMO LAHKO TUDI GLAGOLSKO OBLIKO KRIPTIRATI, KAR LAHKO POMENI INKRIPTIRATI (ŠIFRIRATI) IN DEKRIPTIRATI (DEŠIFRIRATI). NADALJE JE MOČ IZPELJATI ŠE PRIDEVNIŠKE OBLIKE KRIPTEN, KRIPTIVEN, KRIPTIČEN IN KRIPCIJSKI.

V ČLANKIH O UPORABI KRIPTOGRAFIJE BOMO OPISALI TUDI OSNOVNE MIKRORAČUNALNIŠKE KONFIGURACIJE IN PROGRAMSKO OPREMO, KI ZAGOTAVLJA OBDELAVO, PRENOS IN SHRANJEVANJE ŠIFRIRANIH PODATKOV. ŠIFRIRANE PODATKE LAHKO POŠILJAMO NPR. MED DVEMA TELEPRINTERJEMA, ČE IMA ODDAJNI TELEPRINTER INKRIPCIJSKO, SPREJEMNI TELEPRINTER PA DEKRIPCIJSKO NAPRAVO. ŠIFRIRANE PODATKE LAHKO SHRANJUJMO V POMNILNIKE, NA DISKE IN TRAKOVE IN TAKO VARUJEMO NJIHOVO TAJNOST OZIROMA JIH ZAŠČITIMO PRED UPORABO V NEPRAVEM ČASU IN NA NEPRAVEM MESTU.

KOT SMO ŽE NAPISALI, SE KRIPTOGRAFIJA UKVARJA S TRANSFORMACIJO ODKRITEGA V ZAKRITI TEKST S CILJEM, DA KASNEJE Z INVERZNO TRANSFORMACIJO TEKST ZOPET ODKRIJE. TAKO DOBIMO OSNOVNO SHEMO KRIPTIRANJA NA SLIKI 1.

PRI REALIZACIJI INKRIPCIJSKEGA IN DEKRIPCIJSKEGA POSTOPKA SE UPORABLJA T.I. KLJUČ, KI OMOGOČA ZAKRIVANJE IN ODKRIVANJE TEKSTA V DANEM TRENUTKU. ŠTEVILO MOGOČIH KLJUČEV MORA BITI ČIM VEČJE, TAKO DA VSEH NI MOGOČE UPORABITI TUDI V DOVOLJ DOLGEM RAZDOBJU. STANDARDNI KRIPTIVNI ALGORITEM UPORABLJA KODNO KNJIGO, KO SE 64-BITNI BLOK ZAKRIJE ALI ODKRIJE V 64-BITNI BLOK. KRIPCIJA SE PRI TEM KRMILI S 56-BITNIMI KLJUČI, TAKO DA JE ŠTEVILO MOŽNIH KOMBINACIJ KLJUČEV ENAKO 2 NA POTENCO 56 OZIROMA

$$7{,}2.10^{16}$$

ZA ODKRITJE TAKO KRIPTIRANEGA TEKSTA BI NPR. Z RAČUNALNIKOM CDC 7600 POTREBOVALI VSAJ 2500 LET.

2. NEKATERE KRIPCIJSKE METODE

VEČ KRIPCIJSKIH METOD JE BILO OPISANIH V DELU (1). NAS ZANIMAJO PREDVSEM POSTOPKI KRIPTIRANJA, KI SO DOVOLJ ENOSTAVNI ZA PROGRAMIRANJE. OGLEJMO SI BEŽNO T.I. ROČNO, TRANSPOZICIJSKO, WHEATSTONOVO IN VERNAMOVO METODO.

ROČNA KRIPTIVNA METODA UPORABLJA ENOSTAVEN POSTOPEK IN ZADOSTUJETA ŽE SVINČNIK IN PAPIR, PRIMEREN JE PA TUDI ŽEPNI KALKULATOR. OSNOVA TE METODE JE UPORABA KLJUČA V KRIPCIJSKEM POSTOPKU. IMEJMO LAHKO POMNLJIV KLJUČ, NPR. V ANGLEŠČINI

THE QUICK BROWN FOX JUMPS OVER THE LAZY DOG'S BACK

TER PRIPADAJOČO KRIPTIVNO TABELO

ODKRITA ABECEDA
ABCDEFGHIJKLMNOPQRSTUVWXYZ
THEQUICKBROWNFXJMPSVLAZYDG
ZAKRITA ABECEDA

IMEJMO SPOROČILO (BREZ ŠUMNIKOV)

PRIDI JUTRI OB DVEH V MARIBOR ST

KI SE PRETVORI V TAJNO SPOROČILO Z UPORABO GORNJE TABELE, IN SICER

JPBQB RLVPB XHQAU KANTP HXPSV

KO IMAMO SKUPINE PO PET ZNAKOV. ČE BI POSLALI SAMO KRATKO SPOROČILO "TAKOJ", KI IMA KRIPTOGRAM "VTOXR", BI TE SAME BESEDE NE MOGLI ANALIZIRATI, DALJŠE TEKSTE PA BI SČASOMA LAHKO ODKRILI.

VEČJA ZANESLJIVOST GLEDE NA ODKRIVANJE SE DOSEŽE S T.I. POLIKRIPTIČNIMI METODAMI, KI TEMELJIJO NPR. NA TABELAH VIGENERJA. TAKŠNO TABELO IMAMO NA SLIKI 2, PRIMER ODKRITEGA TEKSTA, KLJUČA IN ZAKRITEGA TEKSTA PA NAJDEMO NA SLIKI 3. DEKRIPCIJA ZAKRITEGA TEKSTA POTEKA TAKO, DA VZAMEMO VRSTICO ZNAKA KLJUČA, KI PRIPADA ZNAKU ZAKRITEGA TEKSTA IN TA ZNAK DOLOČA STOLPEC V TABELI NA SLIKI 2, KI JE OZNAČEN S PRIPADAJOČIM ZNAKOM ODKRITEGA TEKSTA. TOREJ NAPIŠEMO NAD ZAKRITIM TEKSTOM KLJUČ TOLIKOKRAT, DA CELOTNI ZAKRITI TEKST POKRIJEMO, NATO PA UPORABIMO TABELO NA SLIKI 2.

		ODKRITA ABECEDA ABCDEFGHIJKLMNOPQRSTUVWXYZ
D	A	ABCDEFGHIJKLMNOPQRSTUVWXYZ
E	B	BCDEFGHIJKLMNOPQRSTUVWXYZA
S	C	CDEFGHIJKLMNOPQRSTUVWXYZAB
I	D	DEFGHIJKLMNOPQRSTUVWXYZABC
G	E	EFGHIJKLMNOPQRSTUVWXYZABCD
N	F	FGHIJKLMNOPQRSTUVWXYZABCDE
A	G	GHIJKLMNOPQRSTUVWXYZABCDEF
T	H	HIJKLMNOPQRSTUVWXYZABCDEFG
O	I	IJKLMNOPQRSTUVWXYZABCDEFGH
R	J	JKLMNOPQRSTUVWXYZABCDEFGHI
	K	KLMNOPQRSTUVWXYZABCDEFGHIJ
Z	L	LMNOPQRSTUVWXYZABCDEFGHIJK
A	M	MNOPQRSTUVWXYZABCDEFGHIJKL
K	N	NOPQRSTUVWXYZABCDEFGHIJKLM
R	O	OPQRSTUVWXYZABCDEFGHIJKLMN
I	P	PQRSTUVWXYZABCDEFGHIJKLMNO
T	Q	QRSTUVWXYZABCDEFGHIJKLMNOP
E	R	RSTUVWXYZABCDEFGHIJKLMNOPQ
	S	STUVWXYZABCDEFGHIJKLMNOPQR
A	T	TUVWXYZABCDEFGHIJKLMNOPQRS
B	U	UVWXYZABCDEFGHIJKLMNOPQRST
E	V	VWXYZABCDEFGHIJKLMNOPQRSTU
C	W	WXYZABCDEFGHIJKLMNOPQRSTUV
E	X	XYZABCDEFGHIJKLMNOPQRSTUVW
D	Y	YZABCDEFGHIJKLMNOPQRSTUVWX
E	Z	ZABCDEFGHIJKLMNOPQRSTUVWXY

SLIKA 2. VIGENERJEVA TABELA RABI ZA GENERIRANJE VEČABECEDNIH ŠIFER. PRI INKRIPCIJI SE NAJPREJ NAPIŠE ODKRITI TEKST IN POTEM KLJUČ NAD NJIM. ŠIFRIRANI TEKST SE DOBI S PRESEKOM STOLPCA ODKRITE ABECEDE IN VRSTICE ZNAKOV KLJUČA, KOT KAŽE SLIKA 3.

KLJUČ:	THEQU ICKBR OW NFOX J UMPSOVE RT	VRST
ODKRITO:	PRIDI JUTRI OB DVEH V MARIBOR ST	STOL
ZAKRITO:	IYMTC RWDSZ CX QASE E GMGAPJV JM	SEČI

SLIKA 3. ZAKRITI TEKST JE DOBLJEN IZ ODKRITEGA TEKSTA, KLJUČA IN TABELE NA SLIKI 2.

V TEM, PRAVKAR OBRAVNAVANEM PRIMERU JE LAHKO KLJUČ TUDI TEKST IZ KAKE KNJIGE IN LAHKO SE UPORABLJA TUDI T.I. LASTNI TEKSTOVNI KLJUČ

(AVTOKLJUČ). V SISTEMU Z AVTOKLJUČEM SE ODKRITI TEKST SAM UPORABI KOT KLJUČ.

PREMESTITVENA KRIPCIJA UPORABLJA NAČIN PREMEŠČANJA ZNAKOV (ČRK) V TEKSTU. ZNAKI KLJUČA SE NAJPREJ OŠTEVILČIJO, IN SICER OD 1 NAPREJ V SMERI VRSTNEGA REDA V ABECEDI, S TEM DA SE DVA ENAKA ZNAKA V KLJUČU ŠTEJETA OD LEVE PROTI DESNI. NPR. ZA KLJUČ LJUBLJANA BI IMELI OSTEVILČENJE IN PRIMER NA SLIKI 4.

```
KLJUČ:     L J U B L J A N A
STOLPCI:   6 4 9 3 7 5 1 8 2
ODKRITO:   P R I D I J U T R
           I O B D V E H V M
           A R I B O R S T .
ZAKRITO:   U H S R M D D B R
           O R J E R P I A I
           V O T V T I B I .
```

SLIKA 4. ZAKRITI TEKST JE SESTAVLJEN IZ PREMEŠČENIH ZNAKOV ODKRITEGA TEKSTA; ZAPOREDJE ZAKRITEGA TEKSTA USTREZA ZAPOREDNIM ŠTEVILKAM STOLPCEV KLJUČA.

SEVEDA JE MOČ UPORABITI TUDI METODE VEČKRATNEGA PREMEŠČANJA. DEKRIPTOR POZNA DOLŽINO TEKSTA IN DOLŽINO KLJUČA IN ODTOD DOLOČI DOLŽINO VSAKEGA STOLPCA. NPR. DOLŽINA TEKSTA JE 26, DOLŽINA KLJUČA 9, IMAMO 26 : 9 JE ENAKO 2 IN OSTANE 8; TOREJ IMAMO 8 STOLPCEV S TREMI ZNAKI IN EN STOLPEC Z DVEMA ZNAKOMA, OSTEVILČENJE KLJUČA PA POZNAMO.

S SUBSTITUCIJSKO KRIPCIJO (1) NADOMEŠČAMO ZNAKE TEKSTA, TODA JIH NE PREMEŠČAMO. ZARADI TEGA JE LAHKO DEKRIPTIVNA ANALIZA LAŽJA IN BOLJ USPEŠNA. PREMEŠČANJE IZNIČI OZIROMA PREDRUGAČI VRSTNI RED ZNAKOV, TODA NE ZAKRIJE NJIHOVE PRAVE VREDNOSTI. KOMBINACIJA SUBSTITUCIJSKE IN PREMESTITVENE KRIPCIJE LAHKO POVZROČI ŠIFRE, KI JIH JE TEŽJE ODKRITI.

PLAYFAIRJEVA KRIPCIJA UPORABLJA KLJUČ, KI JE ZAPISAN V OBLIKI KVADRATA S 5 KRAT 5 MESTI IN KJER ČRKI "I" IN "J" ZASEDATA ISTO MESTO TER SE UPORABLJATA IZMENOMA. ZA NAŠ PRIMER (THE QUICK BROWN FOX ...) IMAMO KLJUČ

```
T H E Q U
I/J C K B R
O W N F X
M P S V L
A Z Y D G
```

ZA INKRIPCIJO IMAMO TALE PRAVILA:

(1) ODPRTI IN ZAPRTI TEKST PREVAJAMO TAKO, DA VZAMEMO VSELEJ ZAPOREDNI PAR ČRK V TEKSTU (DVA SOSEDNJA ZNAKA);

(2) ČE SE PAR ZNAKOV NAHAJA V DIAGONALNIH OGLJIŠČIH PRAVOKOTNIKA V KLJUČU (GLEJ GORNJI KVADRAT), VZEMI ZNAKA V OSTALIH DVEH OGLJIŠČIH V VRSTNEM REDU PARA (V KVADRATU OD LEVE PROTI DESNI);

(3) ČE JE PAR ZNAKOV V ISTI VRSTICI, VZEMI DESNA ZNAKA OD ZNAKOV PARA; PO POTREBI ZAKLJUČI VRSTICO CIKLIČNO;

(4) ČE JE PAR ZNAKOV V ISTEM STOLPCU, VZEMI SPODNJA ZNAKA OD ZNAKOV PARA; PO POTREBI ZAKLJUČI STOLPEC CIKLIČNO;

(5) DVA ENAKA ZNAKA TEKSTA LOČI Z ZNAKOM "X" (TUDI I IN J STA ENAKA ZNAKA).

NAŠE SPOROČILO (PRIDI JUTRI ...) DOBI TEDAJ OBLIKO

CLBAI XJTHI CJFQD QELPI GCRJX ME

DEKRIPCIJA POTEKA ENOLIČNO Z UPORABO GORNJIH PRAVIL. TAKŠNA SUBSTITUCIJA JE LAHKO ZELO UČINKOVITA, SAJ JE DOKAJ ENOSTAVNA, ČEPRAV IMAMO ZA 26 ČRK V KLJUČU LE 676 RAZLIČNIH PAROV. PRI UPORABI TE METODE POSTANE POJAVNA DISTRIBUCIJA ZNAKOV BOLJ ENAKOMERNA OD NARAVNE IN VELJA NPR. ZA ANGLEŠČINO POJAVNOST ZA "E" 12 PROCENTOV, ZA "T" 9 PROCENTOV, ZA PARA "TH" IN "HE" PA LE 3,25 IN 2,5 PROCENTOV.

VERNAMOVA KRIPCIJA (1918) VELJA ZA ABSOLUTNO ZANESLJIVO, SAJ UPORABLJA SISTEM T.I. ENKRATNEGA KLJUČA. VERNAMOVA METODA SE UPORABLJA TUDI NA VROČI LINIJI MED WASHINGTONOM IN MOSKVO. VERNAM JE VPELJAL V SVOJO METODO OPERACIJO SEŠTEVANJA PO MODULU 2 Z NAKLJUČNIMI BINARNIMI KLJUČI. UPORABIMO KOT PRIMER OPERACIJO PO DRUGEM MODULU, KI GA PRIKAZUJE SLIKA 5 (MODUL 26).

KLJUČ		IN	ODPRTO		JE	ZAPRTO		MANJ	KLJUČ		JE	ODPRTO	
		(+)			(=)			(−)			(=)		
T	19		P	15	I	8		T	19		P	15	
H	7		R	17	Y	24		H	7		R	17	
E	4		I	8	M	12		E	4		I	8	
Q	16		D	3	T	19		Q	16		D	3	
U	20		I	8	C	2		U	20		I	8	
I	8		J	9	R	17		I	8		J	9	
C	2		U	20	W	22		C	2		U	20	
K	10		T	19	D	3		K	10		T	19	
B	1		R	17	S	18		B	1		R	17	
R	17		I	8	Z	25		R	17		I	8	
O	14		O	14	C	2		O	14		O	14	
W	22		B	1	X	23		W	22		B	1	
N	13		D	3	Q	16		N	13		D	3	
F	5		V	21	A	0		F	5		V	21	
X	23		E	4	B	1		X	23		E	4	
J	9		H	7	O	16		J	9		H	7	
M	12		V	21	H	7		M	12		V	21	
P	15		M	12	B	1		P	15		M	12	
S	18		A	0	S	18		S	18		A	0	
V	21		R	17	M	12		V	21		R	17	
L	11		I	8	T	19		L	11		I	8	
A	0		B	1	B	1		A	0		B	1	
Z	25		O	14	N	13		Z	25		O	14	
Y	24		R	17	P	15		Y	24		R	17	
D	3		S	18	V	21		D	3		S	18	
G	6		T	19	Z	25		G	6		T	19	

---INKRIPCIJA---------DEKRIPCIJA----------

SLIKA 5. ČRKE ABECEDE "A" DO "Z" SE OŠTEVILČIJO S ŠTEVILKAMI "0" DO "25". SLIKA KAŽE PRIMER INKRIPCIJE IN DEKRIPCIJE, KO IMAMO SEŠTEVANJE (OPERATOR "IN") IN ODŠTEVANJE (OPERATOR "MANJ") PO MODULU 26. TO JE PRIMER MODIFICIRANE VERNAMOVE METODE.

VERNAM JE UPORABIL ZA SVOJO METODO KRIPCIJE GENERIRANJE ZAPOREDJA NAKLJUČNIH ŠTEVIL OZIROMA KLJUČEV, KJER SE UPORABLJENI KLJUČ UNIČI TAKOJ PO UPORABI NA ODDAJNI IN SPREJEMNI STRANI. KLJUČ JE LAHKO NA PERFORIRANEM TRAKU TER SE S POSEBNIM NOŽEM NA ČITALNIKU TAKOJ PO VČITANJU UNIČUJE AVTOMATIČNO. KLJUČI SO LAHKO NAPISANI TUDI NA PAPIRNIH TRAKOVIH V OBLIKI DESETIŠKIH ŠTEVILK IN TRAK SE PO UPORABI TAKOJ UNIČI.

KRIPCIJA, KI UPORABLJA PASTI. REZIDUALNA ARITMETIKA, KOT JO JE UVEDEL VERNAM IN SMO JO POKAZALI NA SLIKI 5, PREDSTAVLJA OSNOVO SODOBNIH KRIPCIJSKIH METOD. NAJ BO INKRIPCIJSKI KLJUČ SESTAVLJEN IZ PARA CELIH ŠTEVIL (I,N) IN SPOROČILO NAJ BO NEKO CELO ŠTEVILO V INTERVALU (0, N-1). INKRIPCIJSKI POSTOPEK BODI

$$C \equiv M^{I} \pmod N$$

KJER JE C ŠIFRA. PRI I = 9007 JE N 129-MESTNO DESETIŠKO ŠTEVILO. DEKRIPCIJSKI KLJUČ, KI JE SEVEDA TAJEN, JE SESTAVLJEN IZ PARA CELIH ŠTEVIL (D,N) IN DEKRIPCIJSKI POSTOPEK JE DOLOČEN S FORMULO

$$M = C^{D} \ (MOD\ N)$$

TU JE M ODKRITI TEKST, DOBLJEN IZ ZAKRITEGA TEKSTA. ŠTEVILI I IN D IMENUJEMO INKRIPCIJSKA IN DEKRIPCIJSKA PAST.

PRI GENERIRANJU N, D IN I SE UPORABLJA POSTOPEK NASLEDNJIH TREH KORAKOV:

1.. GENERIRAJ DVE NAKLJUČNI PRAŠTEVILI P IN Q (KI IMATA PRIBLIŽNO 100 DECIMALNIH MEST.) IN NAJ BO
$$N = P.Q$$
TAKO DA IMA N 200 DECIMALNIH MEST.

2.. IZBERI ŠTEVILO D KOT PRAŠTEVILO, KI JE VEČJE OD PRAŠTEVIL P IN Q.

3.. IZBERI I TAKO, DA VELJA
$$I.D = 1 \ (MOD(P-1).(Q-1))$$

VZEMIMO DVA PARTNERJA, NPR. BANKO B IN UPORABNIKA BANČNIH USLUG U. NAJ BOSTA
IU IN IB
JAVNA INKRIPCIJSKA POSTOPKA ZA UPORABNIKA U IN BANKO B. NADALJE NAJ BOSTA
DU IN DB
TAJNA DEKRIPCIJSKA POSTOPKA K PREJŠNJIMA INKRIPCIJSKIMA POSTOPKOMA.

NAJ BO T ODKRITO SPOROČILO ZA DENARNI TRANSFER, KI IMA PO TAJNEM DEKRIPCIJSKEM POSTOPKU UPORABNIKA U OBLIKO
DU(T)
TA TRANSFORMACIJA JE OBRNLJIVA Z UPORABO JAVNE OPERACIJE IU. KO ŽELI UPORABNIK U POSLATI BANKI B SVOJE SPOROČILO T, MORA OPRAVITI SESTAVLJENO OPERACIJO
IB(DU(T))
BANKA DOBI IZ TE INFORMACIJE SPOROČILO DU(T) Z UPORABO POSTOPKA DB IN NADALJE KONČNO ODPRTI TEKST S
T = IU(DU(T))
ČE POSTOPEK V CELOTI PONOVIMO, IMAMO TELE KORAKE:

$$T \xrightarrow{U} DU(T) \xrightarrow{U} IB(DU(T)) \xrightarrow{\text{PRENOS V BANKO}} \xrightarrow{B} IB(DU(T)) \xrightarrow{B} DB(IB(DU(T)) \xrightarrow{B} \xrightarrow{B} IU(DB(IB(DU(T)))) = T$$

PUŠČICI $\xrightarrow{U}$ IN $\xrightarrow{B}$ OZNAČUJETA TRANSFORMACIJI UPORABNIKA U IN BANKE B.

NA PODOBEN NAČIN LAHKO POŠLJE SPOROČILO T1 TUDI BANKA B UPORABNIKU U, IN SICER:

$$T1 \xrightarrow{B} DB(T1) \xrightarrow{B} IU(DB(T1)) \xrightarrow{\text{PRENOS K UPOR.}} \xrightarrow{U} IU(DB(T1)) \xrightarrow{U} DU(IU(DB(T1))) \xrightarrow{U} \xrightarrow{U} IB(DU(IU(DB(T1)))) = T1$$

ZA OBA OPISANA POSTOPKA JE TEDAJ ZNAČILNO, DA IMAMO JAVNA INKRIPCIJSKA POSTOPKA IU IN IB, DOČIM STA TAJNA DEKRIPCIJSKA POSTOPKA DU IN DB ZNANA LE KONKRETNEMU UPORABNIKU (DU) IN BANKI (DB, LAHKO PA TUDI DU).

3. STANDARDNI PODATKOVNI KRIPCIJSKI ALGORITEM

VZEMIMO TOLE OSNOVNO SHEMO: ODKRITI TEKST RAZBIJEMO V SKUPINE PO OSEM ZNAKOV (ČRK, ŠTEVILK, INTERPUNKCIJ, PRESLEDKOV, KRMILNIH ZNAKOV V ASCII). NAŠE VHODNO ZAPOREDJE BO TEDAJ VSELEJ SESTAVLJENO IZ 8 KARAKTERJEV, OD KATERIH BO VSAK PREDSTAVLJEN Z OSMIMI BITI. TOREJ BOMO IMELI VHODNO ZAPOREDJE
64 BITOV
ZA INKRIPCIJSKI ALGORITEM, KATEREGA IZHOD BO ZOPET 64 BITOV ZAKRITEGA TEKSTA (ŠIFRA). IMEJMO TALE OSNOVNI PSEVDO KOD:

```
RUTINA INKRIPCIJA
  VZEMI 64 BITOV ODKRITEGA TEKSTA;
  PERMUTIRAJ BITE TEGA TEKSTA;
  OPRAVI PODATKOVNO TRANSFORMACIJO TEKSTA;
  OPRAVI BLOČNO TRANSFORMACIJO TEKSTA;
  OBRATNO PERMUTIRAJ BITE TEKSTA;
  IZDAJ 64 BITOV ZAKRITEGA TEKSTA;
KONEC RUTINE
```

TA RUTINA OPRAVI NAJPREJ ZAČETNO PERMUTACIJO OZIROMA PREMESTITEV BITOV. NAJVEČJI DEL INKRIPCIJSKEGA POSTOPKA SE OPRAVI S PRODUKTNO TRANSFORMACIJO, KI BO OPISANA KASNEJE. BLOČNA TRANSFORMACIJA POMENI IZMENJAVO LEVIH IN DESNIH 32 BITOV. ZADNJI KORAK JE OBRATNA PERMUTACIJA K ZAČETNI PREMESTITVENI OPERACIJI. OB IZSTOPU IZ ALGORITMA "INKRIPCIJA" DOBIMO ZAKRITI TEKST.

INKRIPCIJSKI KLJUČ JE SESTAVNI DEL KRMILNIH FAKTORJEV V PRODUKTNI TRANSFORMACIJI. BISTVENA JE TUDI RELACIJA MED ZAČETNO IN OBRATNO PERMUTACIJO, KI IMA POMEN DOLOČENE FUNKCIJE IN K NJEJ INVERZNE FUNKCIJE.

OGLEJMO SI PRIMER ZAČETNE IN K NJEJ INVERZNE PERMUTACIJE NA SLIKI 6. OPREDELIMO OPERACIJO PERMUTACIJE TAKOLE:

-1-. VHODNI NIZ 64 BITOV OPAZUJEMO OZIROMA ŠTEJEMO VEDNO OD LEVE PROTI DESNI;

-2-. SKLADNO Z RAZPREDELNICO (1) NA SLIKI 6 BO PRVI BIT PERMUTIRANE BESEDE (64 BITOV) BIT ŠTEVILKA 58 VHODNE BESEDE, DRUGI BIT IZHODNE BESEDE BO PETDESETI BIT VHODNE BESEDE ITN.

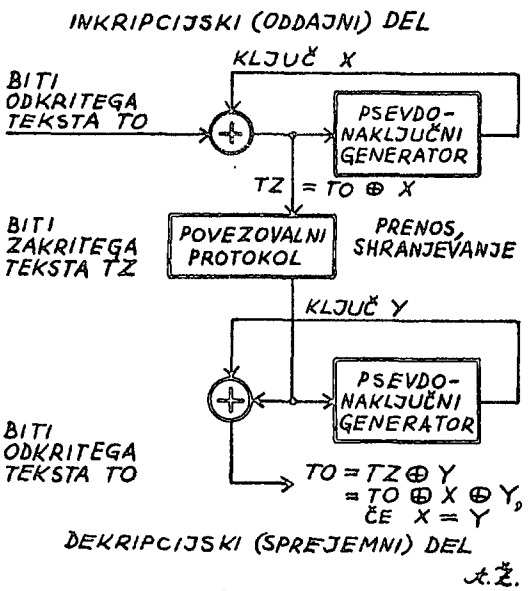
-3-. KORESPONDENCA MED 64-BITNO VHODNO OZIROMA IZHODNO BESEDO IN RAČUNALNIŠKIMI ZLOGI (8 ZLOGOV Z 8 BITI) PA JE TALE:

INKRIPCIJ-	00000000	01111111	...	55566666
SKI BITI	12345678	90123456	...	78901234
BITI ZLOGOV	76543210	76543210	...	76543210
ZLOGI	0	1	...	7

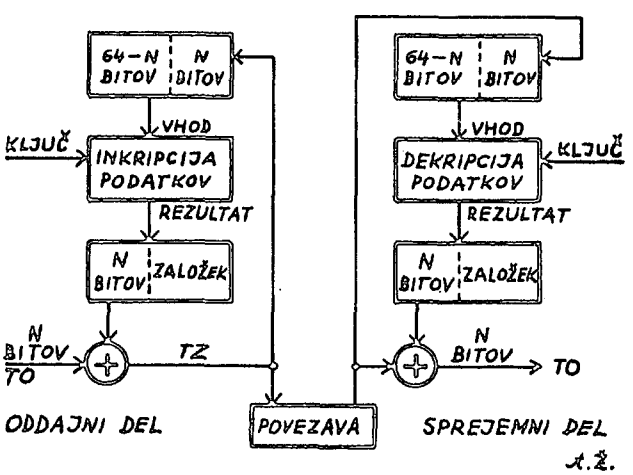
58	50	42	34	26	18	10	02	
60	52	44	36	28	20	12	04	
62	54	46	38	30	22	14	06	
64	56	48	40	32	24	16	08	
57	49	41	33	25	17	09	01	(1)
59	51	43	35	27	19	11	03	
61	53	45	37	29	21	13	05	
63	55	47	39	31	23	15	07	
40	08	48	16	56	24	64	32	
39	07	47	15	55	23	63	31	
38	06	46	14	54	22	62	30	
37	05	45	13	53	21	61	29	
36	04	44	12	52	20	60	28	(2)
35	03	43	11	51	19	59	27	
34	02	42	10	50	18	58	26	
33	01	41	09	49	17	57	25	

SLIKA 6. PERMUTACIJSKI RAZPREDELNICI ZA ZAČETNO (1) IN INVERZNO ZAČETNO PERMUTACIJO (2), KI PREDPISUJETA PREMEŠČANJE BITOV NA ZAČETKU IN NA KONCU KRIPCIJSKEGA POSTOPKA. PRI INVERZNI PERMUTACIJI JE POSTOPEK TALE: BIT 1 JE 40. BIT, BIT 2 JE 8. BIT, BIT 3 JE 48. BIT ITN. VHODNEGA NIZA. TO JE TEDAJ NAVADNO PREMEŠČANJE BITOV.

KOT SMO ŽE OMENILI, JE NBS (NATIONAL BUREAU OF STANDARDS) LETA 1977 IZDAL T.I. PODATKOVNI KRIPCIJSKI STANDARD (DES). TA STANDARD TEMELJI NA IBM-OVEM ALGORITMU TRANSFORMACIJE 64-BITNE BESEDE Z UPORABO 56-BITNEGA KLJUČA. DES ALGORITEM ZAGOTAVLJA ZADOSTNO ZAŠČITO PRED LOMLJENJEN KODA. PRI TEM SE LAHKO UPORABLJA METODA S POVRATNO ZANKO ZA TRANSFORMACIJO ASCII ZNAKOV, KOT JE PRIKAZANO NA SLIKI 7. V TEM SISTEMU SE SIGNAL IZ PSEVDONAKLJUČNEGA GENERATORJA, KI JE FUNKCIJA PREJŠNJEGA BLOKA ODKRITEGA TEKSTA, PRIŠTEJE BITNO PO MODULU 2 K TRENUTNEMU BLOKU ODPRTEGA TEKSTA TO. REZULTAT TEGA PRIŠTETJA JE INKRIPCIJA, KI JO POŠLJEMO SPREJEMNIKU ALI SHRANIMO NA TRAK ALI DISK. TA INKRIPCIJA JE ZAKRITI TEKST TZ. ZAKRITI TEKST SE DEKRIPTIRA Z UPORABO PSEVDONAKLJUČNEGA GENERATORJA, KI JE ENAK IZVIRNEMU PSEVDONAKLJUČNEMU GENERATORJU. PO DOLOČENI INICIALIZACIJSKI PERIODI EMULIRA SPREJEMNI GENERATOR ODDAJNEGA IN KO S TEM POSTANE X = Y, SE DEKRIPCIJA LAHKO ZAČNE.

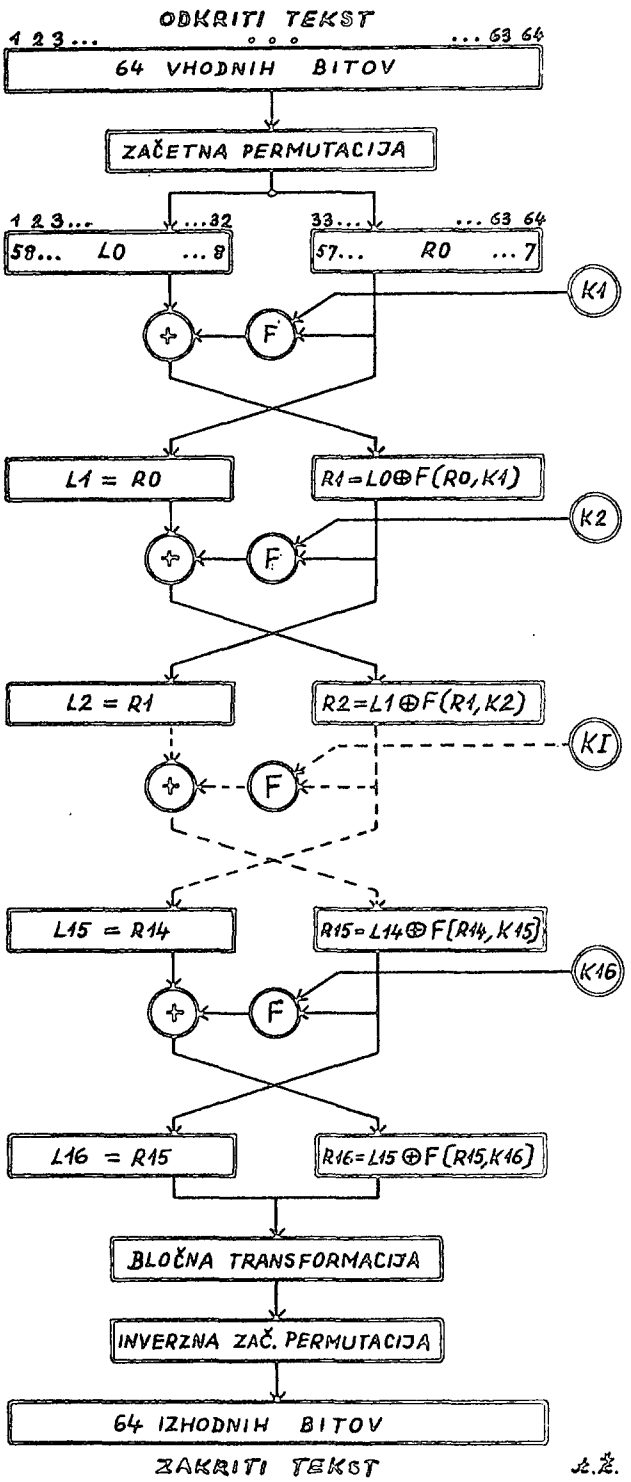


SLIKA 7. ENOSTAVEN KRIPCIJSKI SISTEM, KI UPORABLJA POVRATNO ZVEZO NA ODDAJNI IN SPREJEMNI STRANI.

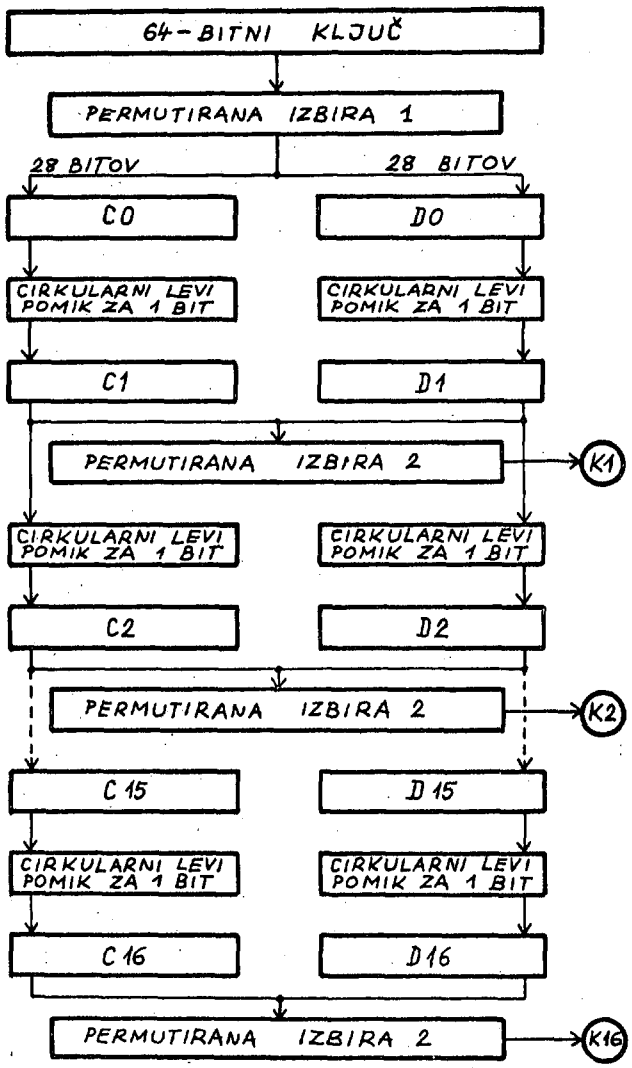


SLIKA 8. KRIPCIJSKA POVRATNA VEZAVA, KI UPORABLJA T.I. DES ALGORITEM.

SEVEDA PA JE MOČ UPORABITI TUDI SAM DES ALGORITEM KOT PSEVDONAKLJUČNI GENERATOR, KO IMAMO ZOPET SISTEM S KRIPTIVNO POVRATNO VEZAVO. TAK SISTEM JE PRIKAZAN NA SLIKI 8. PO ODDAJI 64 BITOV IMATA SPREJEMNA IN ODDAJNA STRAN ENAKO BESEDO (IZHOD JE ENAK VHODU). ČE SE UPORABLJATA ENAKA TRANSFORMA IN KLJUČA NA



SLIKA 9. PODROBNI DIAGRAM OPERACIJ STANDARDNEGA KRIPCIJSKEGA ALGORITMA. TA ALGORITEM JE OBRNLJIV TER RABI TAKO ZA INKRIPCIJO IN DEKRIPCIJO PODATKOV, ČE SE OBRNE VRSTNI RED UPORABE KLJUČEV K1, K2, ..., K16 NA SLIKI. FUNKCIJA F(R,K) JE PODROBNO OPISANA V TEKSTU; TO VELJA TUDI ZA GENERIRANJE KLJUČEV.



SLIKA 10. DIAGRAM NA TEJ SLIKI KAŽE GENERIRANJE 48-BITNIH PODKLJUČEV K1, K2, ... , K16 IZ 64-BITNEGA (OZIROMA 56-BITNEGA) KLJUČA.

ODDAJNI IN SPREJEMNI STRANI, SE VHOD Z DVEMA SEŠTEVANJIMA PO MODULU 2 IZENAČI Z IZHODOM. SEVEDA PA ENA SAMA NAPAKA PRI PRENOSU (POVEZAVA) POVZROČI VEČ NAPAK V ODKRITEM TEKSTU NA SPREJEMNI STRANI V TRENUTNEM BLOKU (8 ZLOGOV) IN V NASLEDNJIH BLOKIH, DOKLER SE ODDANE IN SPREJETE BESEDE ZOPET NE UJAMEJO.

KER SO VHODNI IN IZHODNI BLOKI (8 ZLOGOV) V DES ALGORITMU ENAKIH DOLŽIN, JE REALIZACIJA OBSTOJEČIH PRENOSNIH PROTOKOLOV ENOSTAVNA, SAJ NIMAMO MODIFIKACIJE DOLŽINE BESEDE. DOLOČENE NAPAKE, KI SE LAHKO POJAVIJO V TEKSTU, JE MOČ OBVLADATI Z NAVADNIMI PRIPOMOČKI ZAZNAVANJA NAPAK TAKO V ODKRITEM KOT ZAKRITEM TEKSTU. TO POMENI, DA JE NBS-OV KRIPCIJSKI STANDARD MOČ UPORABITI TUDI ZA KOMUNICIRANJE V REALNEM ČASU TER PRI SHRANJEVANJU ŠIFRIRANIH PODATKOV NA TRAK ALI DISK.

OGLEJMO SI NADROBNO DIAGRAM KRIPCIJSKEGA ALGORITMA NA SLIKI 9, KJER JE PREDVSEM POJASNJENA PRODUKTNA TRANSFORMACIJA (GLEJ PSEVDO KOD NA ZAČETKU TEGA POGLAVJA).

SLIKA 9 KAŽE NAJPREJ ZAČETNO PERMUTACIJO IN RAZDELITEV 64 BITOV V LEVI BLOK (L) IN DESNI BLOK (R). INDEKSI PRI L IN R KAŽEJO ITERACIJSKI KORAK V ALGORITMU. NAJPREJ SE GENERIRA PODKLJUČ K1 IZ KRIPCIJSKEGA KLJUČA K. TUDI VSI NADALJNI PODKLJUČI K2, ... , K16 SE GENERIRAJO IZ KLJUČA K. POTEM SE GENERIRA FUNKCIJA F(R0,K1), KI UPORABLJA VHODA R0 IN K1. NASLEDNJA OPERACIJA JE SEŠTEVANJE PO MODULU 2 BITNIH NIZOV L0 IN F(R0,K1). TAKO PRVOTNI L0 NADOMESTIMO Z L0 ⊕ F(R0,K1). POTEM IZMENJAMO ŠE 32 LEVIH IN 32 DESNIH BITOV. PRODUKTNA TRANSFORMACIJA PONOVI PRAVKAR OPISANO SESTAVLJENO OPERACIJO ŠE 15-KRAT, KOT KAŽE SLIKA 9. ALGORITEM SE KONČA Z BLOČNO IN INVERZNO ZAČETNO PERMUTACIJO. VSOTA PO MODULU 2 JE OBRNLJIVA OPERACIJA. ALGORITEM NA SLIKI 9 JE UPORABLJIV TAKO ZA INKRIPCIJO KOT ZA DEKRIPCIJO (VZAJEMNA OBRNLJIVOST).

PRI INKRIPCIJI SMO IMELI ZAPOREDJE LEVIH, DESNIH DELOV IN KLJUČEV

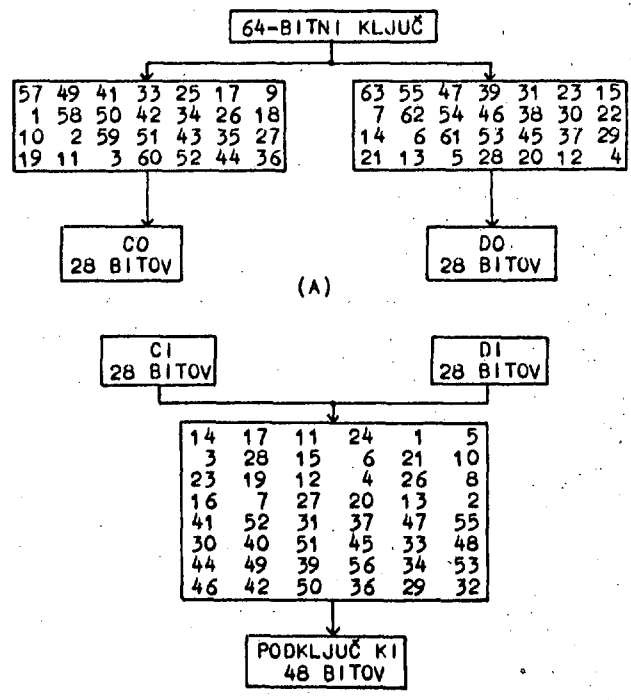
$$L0, L1, \ldots, L15, L16$$
$$R0, R1, \ldots, R15, R16$$
$$K1, \ldots, K15, K16$$

PRI DEKRIPCIJI PA SO TA ZAPOREDJA OBRATNA. TAKO IMAMO PRI DEKRIPCIJI NAJPREJ

$$R16 \oplus F(L16,K16) = (L15 \oplus F(R15,K16)) \oplus F(R15,K16) = L15$$

PRI DEKRIPCIJI SE TOREJ KORAKI OBRNEJO IN REZULTAT TEGA JE PRVOTNI ODKRITI TEKST.

OGLEJMO SI NA KRATKO GENERIRANJE PODKLJUČEV K1, K2, ... , K16 NA SLIKI 10. VZEMIMO 64-BITNI KLJUČ, ČEPRAV BOMO POTREBOVALI SAMO 56 BITOV. PREOSTALIH OSEM BITOV LAHKO UPORABIMO KOT BITE PARNOSTI. PRVO TRANSFORMACIJO KLJUČA IMENUJEMO PERMUTIRANA



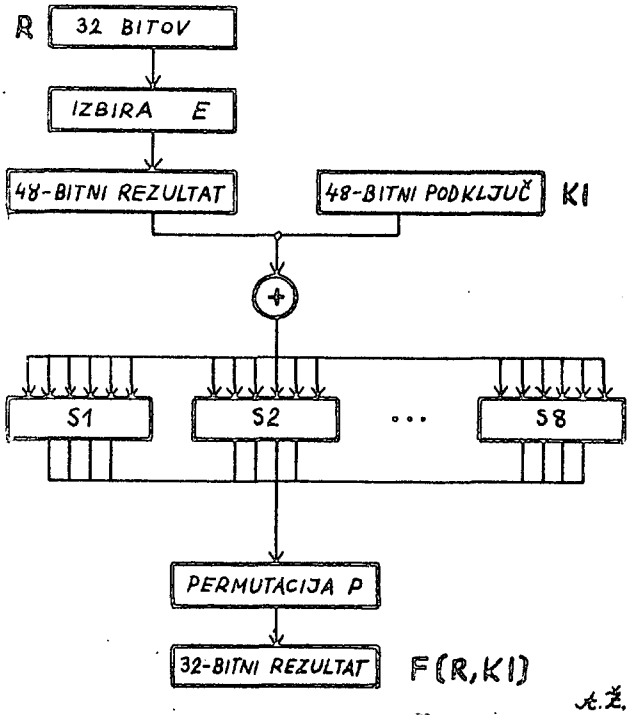
SLIKA 11. PERMUTIRANA IZBIRA 1 (SLIKA A) IN PERMUTIRANA IZBIRA 2 (SLIKA B). PERMUTIRANA IZBIRA 1 (A) SE UPORABLJA ZA PRIDOBIVANJE BESED C0 IN D0 IZ KLJUČA. BITI S ŠTEVILKAMI 8, 16, 24, 32, 40, 48, 56 IN 64 V KLJUČU SE NE UPORABIJO. PERMUTIRANA IZBIRA 2 (B) SE UPORABLJA ZA PRIDOBIVANJE PODKLJUČEV "KI" IZ BESED CI IN DI (I = 1, 2, ... , 16).

IZBIRA 1 (KRATKO PI 1). PI 1 PERMUTIRA 56-BITNI KLJUČ TER GA PREUREDI V DVE 28-BITNI BESEDI, KI JU OZNAČIMO S CO IN DO. GENERIRANJE PODKLJUČA SE OPRAVI S CIRKULARNIM LEVIM POMIKOM BESED CO IN DO, KO DOBIMO BESEDI C1 IN D1; IZ NJIJU IZVEDEMO S PERMUTIRANO IZBIRO 2 PODKLJUČ Z 48 BITI. TRANSFORMACIJI, KI JU IMENUJEMO PERMUTIRANA IZBIRA 1 IN PERMUTIRANA IZBIRA 2 STA PRIKAZANI NA SLIKI 11. VSAK NADALJNI PODKLJUČ SE GENERIRA NA PODOBEN NAČIN: BESEDI CI IN DI SE CIRKULARNO POMAKNETA V LEVO ZA EN ALI DVA BITA, POTEM PA SE UPORABI PERMUTIRANA IZBIRA 2. SLIKA 12 KAŽE ŠTEVILO LEVIH POMIKOV ZA BESEDE CI IN DI PRI VSAKI ITERACIJI (KORAKU) GENERIRANJA PODKLJUČEV.

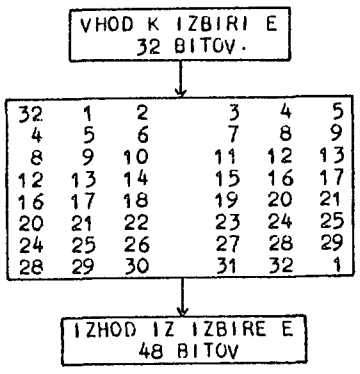
ITERACIJSKI KORAK I	ŠTEVILO LEVIH POMIKOV
1	1
2	1
3	2
4	2
5	2
6	2
7	2
8	2
9	1
10	2
11	2
12	2
13	2
14	2
15	2
16	1

SLIKA 12. PREGLED LEVIH CIRKULARNIH POMIKOV ZA BESEDE CI IN DI (I = 1, 2, ..., 16)

KO SO BILI PODKLJUČI K1, K2, ... , K16 GENERIRANI, JIH LAHKO SHRANIMO IN UPORABLJAMO DO TRENUTKA, KO SE POJAVI NOV KLJUČ. ISTA MNOŽICA PODKLJUČEV SE UPORABLJA ZA INKRIPCIJO IN DEKRIPCIJO, LE NJIHOV VRSTNI RED JE OBRATEN SKLADNO S SLIKO 9.



SLIKA 13. SHEMA REALIZACIJE FUNKCIJE F(R,KI). PRESLIKAVE S1, S2, ... , S8 FUNKCIJE F PRESLIKAJO 6-BITNI VHOD V 4-BITNI IZHOD. PRESLIKAVE S1, S2, ... , S8 SO DEFINIRANE NA SLIKI 15, PERMUTACIJI IZBIRA E IN P PA NA SLIKAH 14 IN 17.



SLIKA 14. RAZPREDELNICA PRESLIKAVE IZBIRA E. IZHOD IMA VEČ BITOV KOT VHOD (32 PROTI 48), DA BI BIL PRILAGOJEN DOLŽINI PODKLJUČA.

S1

14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13

S2

15	1	8	14	6	11	3	4	9	7	2	13	12	0	5	10
3	13	4	7	15	2	8	14	12	0	1	10	6	9	11	5
0	14	7	11	10	4	13	1	5	8	12	6	9	3	2	15
13	8	10	1	3	15	4	2	11	6	7	12	0	5	14	9

S3

10	0	9	14	6	3	15	5	1	13	12	7	11	4	2	8
13	7	0	9	3	4	6	10	2	8	5	14	12	11	15	1
13	6	4	9	8	15	3	0	11	1	2	12	5	10	14	7
1	10	13	0	6	9	8	7	4	15	14	3	11	5	2	12

S4

7	13	14	3	0	6	9	10	1	2	8	5	11	12	4	15
13	8	11	5	6	15	0	3	4	7	2	12	1	10	14	9
10	6	9	0	12	11	7	13	15	1	3	14	5	2	8	4
3	15	0	6	10	1	13	8	9	4	5	11	12	7	2	14

S5

2	12	4	1	7	10	11	6	8	5	3	15	13	0	14	9
14	11	2	12	4	7	13	1	5	0	15	10	3	9	8	6
4	2	1	11	10	13	7	8	15	9	12	5	6	3	0	14
11	8	12	7	1	14	2	13	6	15	0	9	10	4	5	3

S6

12	1	10	15	9	2	6	8	0	13	3	4	14	7	5	11
10	15	4	2	7	12	9	5	6	1	13	14	0	11	3	8
9	14	15	5	2	8	12	3	7	0	4	10	1	13	11	6
4	3	2	12	9	5	15	10	11	14	1	7	6	0	8	13

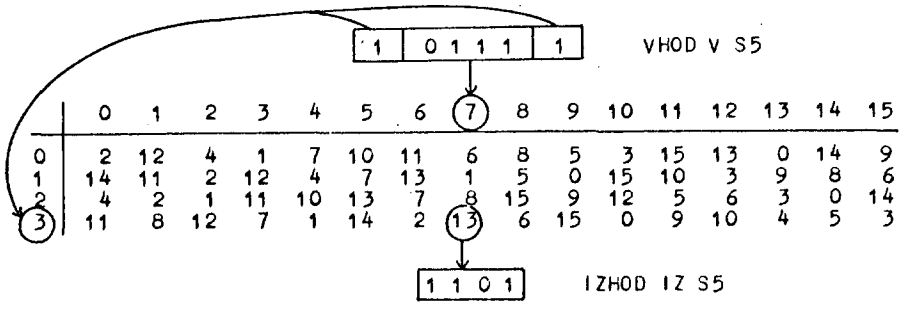
S7

4	11	2	14	15	0	8	13	3	12	9	7	5	10	6	1
13	0	11	7	4	9	1	10	14	3	5	12	2	15	8	6
1	4	11	13	12	3	7	14	10	15	6	8	0	5	9	2
6	11	13	8	1	4	10	7	9	5	0	15	14	2	3	12

S8

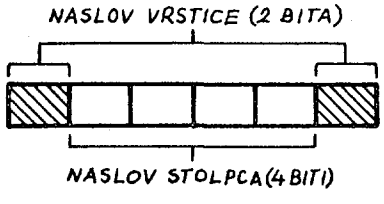
13	2	8	4	6	15	11	1	10	9	3	14	5	0	12	7
1	15	13	8	10	3	7	4	12	5	6	11	0	14	9	2
7	11	4	1	9	12	14	2	0	6	10	13	15	3	5	8
2	1	14	7	4	10	8	13	15	12	9	0	3	5	6	11

SLIKA 15. MATRIKE IZBIRALNIH FUNKCIJ S1, S2, ... , S8. VSAKA FUNKCIJA "S1" PRESLIKA 6-BITNI VHOD V 4-BITNI IZHOD.



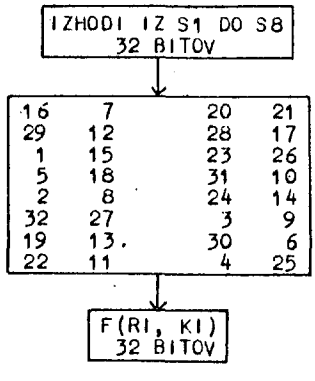
SLIKA 16. PRIKAZ PRESLIKAVE S5 IN NJENE UPORABE PRI DANEM VHODU, KO MORAMO DOLOČITI IZHOD. SREDNJI ŠTIRJE BITI 6-BITNEGA VHODA PREDSTAVLJAJO STOLPNI INDEKS; PRVI IN ZADNJI BIT VHODA SESTAVLJATA VRSTIČNI INDEKS. BINARNA VREDNOST IZBRANEGA VSTOPA V TABELI S5 JE IZHODNA VREDNOST.

SHEMA FUNKCIJE F(R,K) JE PRIKAZANA NA SLIKI 13. NJENA PRVA OPERACIJA JE IZBIRA E, KI JE PODOBNA PREJ OPISANI ZAČETNI PERMUTACIJI, LE DA IMA IZHOD VEČ BITOV KOT VHOD. SLIKA 14 PRIKAZUJE PRESLIKAVO IZBIRA E. 48-BITNI REZULTAT TE OPERACIJE SE SEŠTEJE PO MODULU 2 S PODKLJUČEM. DOBLJENI REZULTAT SE REDUCIRA NA 32 BITOV Z UPORABO POSEBNIH FUNKCIJ S1, S2, ... , S8. NJIHOVA UPORABA JE NAKAZANA NA SLIKI 13. SKUPINE PO 6 BITOV SE PRESLIKAJO S TEMI FUNKCIJAMI V 4-BITNE BESEDE. PRESLIKAVE S1, S2, ... , S8 SO DEFINIRANE NA SLIKI 15, PRIMER NJIHOVE UPORABE PA JE PRIKAZAN NA SLIKI 16. ŠESTBITNI VHOD SE UPORABI PRI VSAKI PRESLIKAVI SI (I = 1, 2, ... , 8), IN SICER TAKO, DA SE DOBI NASLOVNI PAR (TO STA INDEKSA) MATRIKE SI. VHODNI FORMAT JE TALE:



A.Ž.

S TEM FORMATOM JE DOLOČEN KAZALEC, KI KAŽE NA VREDNOST V MATRIKO SI. VREDNOST V MATRIKI SI SE IZ DECIMALNE VREDNOSTI PRETVORI V BINARNO IN TAKO SE DOBIJO IZHODNI ŠTIRJE BITI. SKLADNO S SLIKO 13 SE NAD TEMI IZHODNIMI BITI OPRAVI ŠE PERMUTACIJA P (PRESLIKAVA JE DEFINIRANA NA SLIKI 17) IN SE TAKO DOBI 32-BITNI REZULTAT ZA FUNKCIJO F(R,KI).



SLIKA 17. RAZPREDELNICA PERMUTACIJE P, KI JE SESTAVNI DEL REALIZACIJE FUNKCIJE F(R,KI) NA SLIKI 13.

OPISANI POSTOPKI V TEM POGLAVJU SESTAVLJAJO STANDARDNI PODATKOVNI KRIPCIJSKI ALGORITEM IN OMOGOČAJO, DA LAHKO PRISTOPIMO K PROGRAMSKI ALI MATERIALNI REALIZACIJI ALGORITMA.

4. MOŽNOSTI MIKRORAČUNALNIŠKE INKRIPCIJE IN DEKRIPCIJE

ALGORITEM, KI JE BIL OPISAN V PREJŠNJEM POGLAVJU, OMOGOČA, DA PRISTOPIMO K REALIZACIJI DES ALGORITMA. TA ALGORITEM LAHKO PROGRAMIRAMO, UPORABIMO PA LAHKO TUDI POSEBNA INTEGRIRANA VEZJA ZA KRIPCIJO, KI SO SESTAVNI DELI MAJHNEGA MIKRO RAČUNALNIKA.

V DVEH NADALJEVANJIH TEGA ČLANKA BOMO OPISALI PROGRAM ZA KRIPCIJO S PROCESORJEM Z-80. TA PROCESOR IMA ZELO BOGATO ZALOGO UKAZOV ZA MANIPULACIJO Z BITI (TESTIRANJE, NASTAVLJANJE IN BRISANJE BITOV) IN JE UPORABNIŠKO DOBRO PRILAGOJEN NALOGAM KRIPCIJE. SEVEDA LAHKO ŽE SEDAJ PRIČAKUJEMO, DA BO HITROST KRIPTIRANJA (PREDELAVA ZNAKOV V ŠIFRE IN OBRATNO) DOVOLJ VELIKA ZA OBIČAJNO RABO, KOT STA PRENOS IN SHRANJEVANJE PODATKOV. V MIKRORAČUNALNIŠKIH KONFIGURACIJAH S KRIPTIVNIMI PERIFERNIMI VEZJI BO HITROST PROIZVODNJE ŠIFRIRANEGA ALI DEŠIFRIRANEGA TEKSTA DOSEGLA ALI CELO PRESEGLA VREDNOST

13 MEGABITOV NA SEKUNDO

S TAKŠNO HITROSTJO PA JE MOGOČE ZADOSTITI ZAHTEVE VELIKEGA KROGA UPORABNIKOV.

POZOREN BRALEC JE LAHKO OPAZIL, DA SMO IMELI V OKVIRU PODATKOVNEGA KRIPCIJSKEGA ALGORITMA VRSTO RAZPREDELNIC, VNAPREJ DOLOČENIH PRESLIKAV OZIROMA FUNKCIJ. TO SEVEDA NE POMENI, DA NE SMEMO Z UPOŠTEVANJEM DODATNIH POGOJEV, KI JIH NISMO NAVEDLI, KONSTRUIRATI SVOJE PERMUTACIJSKE IN DRUGE PRESLIKAVE. MIKRORAČUNALNIŠKA UPORABA V KRIPCIJI OMOGOČA UPORABO TAKŠNIH IN TAKO SVOJSKIH KRIPTIČNIH POSTOPKOV, DA SE VERJETNOST ZLOMITVE KODA V DANEM PRIMERU LAHKO POMAKNE IZVEN REALNIH PLANETARNIH KOMBINATORIČNIH MOŽNOSTI. V RAZDOBJU, KI PRIHAJA IN V KATEREM BO POTREBNO INFORMACIJO ZAŠČITITI ZARADI VRSTE ŽIVLJENSKIH IN EKSISTENCIALNIH OKOLIŠČIN, BO POSTALO KRIPTIRANJE NUJEN IN POMEMBEN PRIPOMOČEK V JAVNEM IN ZASEBNEM ŽIVLJENJU.

NA KONCU TEGA KOMPILACIJSKEGA ČLANKA BI SE BRALCU RAD OPRAVIČIL ZARADI NENAVADNEGA GRAFIČNEGA OBLIKOVANJA ČLANKA. SESTAVLJAL SEM GA S SVOJIM PROCESORJEM TEKSTA (Z-80) IN IZPISAL S STARIM POŠTNIM TELEPRINTERJEM. NADALJE, LITERATURA ZA INKRIPCIJO JE BILA ZBRANA ŽE V ČLANKU (1), IZČRPNEJŠI SPISEK PA BO NAVEDEN NA KONCU ZAPOREDJA ČLANKOV O MIKRORAČUNALNIŠKI KRIPTOGRAFIJI.

LITERATURA

(1) J. ZUPAN, KRIPTOGRAFIJA NA PRELOMU, INFORMATICA 1 (1977), ŠTEV. 4, STR. 22 - 28.

KRMILNO VEZJE ZA GIBKI DISK

B. KASTELIC
M. KOVAČEVIĆ
A. HADŽI

UDK: 681.327.63

INŠTITUT JOŽEF STEFAN, LJUBLJANA

Članek opisuje krmilno vezje za gibki disk, ki povezuje mikroračunalnik s procesorjem M 6800 (Iskradata 1680) in gibki disk MFE 701. Podan je tudi pregled zanimivejših krmilnikov za gibki disk, podrobnejši opis krmilnika INS 1771 ter osnovna programska oprema za programirano prenašanje podatkov med mikroračunalnikom in krmilnikom za gibki disk.

FLOPPY DISK INTERFACE. In the paper the interface between processor M 6800 (Iskradata 1680) and floppy disk drive MFE 701 is deskribed. The paper includes detailed description of the INS 1771 controller, the brief deskription of some other controllers and the basic software for programed data transfer.

1. UVOD

Univerzalni računalniški sistemi nujno potrebujejo periferne enote za shranjevanje podatkov, predvsem programov. Med danes obstoječimi spominskimi sredstvi se največ uporabljajo digitalne kasete, luknjani trak in gibki diski. Najuporabnejši je vsekakor gibki disk, saj ob dobri programski opremi predstavlja najudobnejšo in najzanesljivejšo pomnilno periferno enoto.

Gibki disk priključimo na mikroračunalniški sistem preko posebnega vmesnega vezja. To vmesno vezje imenujemo krmilno vezje, mikroračunalniško komponento, ki je osrednji del tega vezja, pa krmilnik. Prenos podatkov med gibkim diskom in računalnikom poteka s pomočjo procesorja. To je programirani prenos podatkov. Obstaja pa še druga možnost, to je tako imenovani DMA prenos podatkov, kjer se prenašajo podatki med gibkim diskom in računalniškim spominom s pomočjo DMA krmilnika. Mikroprocesor lahko med tem časom opravlja druge operacije.

Članek obsega materialno opremo krmilnega vezja za gibki disk prilagojeno za mikroračunalnik Iskradata 1680, pregled zanimivejših krmilnikov za gibki disk in najnujnejšo programsko opremo za programirano prenašanje podatkov med gibkim diskom in mikroračunalniškim spominom.

2. GIBKI DISK (FLOPPY DISK)

Običajna pogonska enota vsebuje: čitalno/pisalno glavo, pogonski mehanizem za nastavljanje glave, čitalno, pisalno in krmilno elektroniko ter sistem za vstavitev diskete v enoto. Vsi ti elementi omogočajo: brisanje in zapisovanje podatkov na disketo, čitanje podatkov, nastavitev čitalno/pisalne glave na željeno stezo ter generiranje in interpretiranje krmilnih signalov.

Gibki diski imajo sedem standardnih vhodnih in štiri standardne izhodne linije. Poleg teh imajo običajno še dodatne vhodne in izhodne linije.

Za premikanje čitalno/pisalne glave so uporabljene linije korak, smer in vključitev glave. Vsak impulz na liniji korak (Step) povzroči premik glave na naslednjo stezo v smeri, ki jo določa linija smer (Direction). Aktivni nivo na liniji vključitev glave (Head Load) primakne čitalno/pisalno glavo v neposredni stik z disketo.

Podatki, ki se bodo zapisovali na disketo, prihajajo po liniji zapis podatkov (Write Data). Pisalna logika se aktivira s pomočjo linije zapis možen (Write Gate). Linija napaka zapisa (Write Fault) se aktivira, kadar niso izpolnjeni vsi pogoji, ki omogočajo zanesljiv zapis na disketo. Linijo nižji tok (Low Current) aktivira krmilnik, kadar se nahaja čitalno/pisalna glava med stezama 43 in 77. Po liniji zaščita zapisanega (Write Protect) se prenaša informacija, če je v pogon vstavljena zaščitena disketa.

Prečitani podatkovni in urini biti prihajajo po liniji čitani podatki (Read Data), ločeni podatkovni biti po liniji ločeni podatki (Separated Data) in urini biti po liniji ločena ura (Separated Clock).

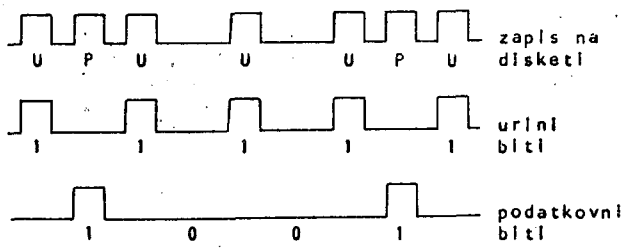
Med važnejše linije spadajo še steza nič (Track 00), ki se aktivira, kadar je čitalno/pisalna glava nad stezo nič, linija indeks (Index), po kateri prihajajo indeksni impulzi, linija pripravljen (Ready), ki je aktivna, kadar je gibki disk sposoben opravljati svoje funkcije.

S pomočjo štirih linij izbira pogona (Drive Select) izbiramo enega izmed štirih diskov.

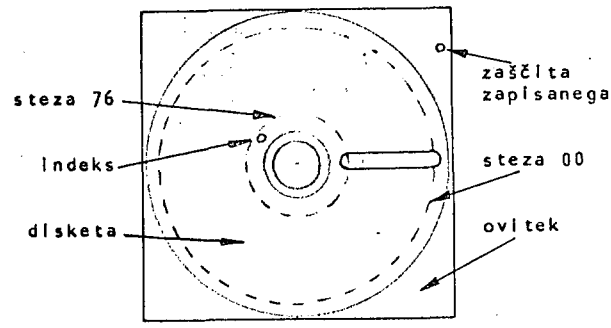
Za napajanje gibkega diska (MFE 701) so potrebne naslednje napetosti:

220VAC, 50Hz, 0,3A
+5VDC, 1,3A
-5VDC, 0,2A
+24VDC, 2,0A

Podatki se shranjujejo na diskete, ki so izdelane iz lahkega gibkega meteriala. Shranjeni so na površini, ki je prevlečena z magnetnim materialom. Poleg enostranskih obstajajo še dvostranske diskete, kjer so



SLIKA 1: Podatki so zapisani na disketo s pomočjo frekvenčne modulacije, tako da so podatkovni biti vrinjeni med urine bite.



SLIKA 2: Disketa je spravljena v ovitku, ki ima običajno štiri odprtine: osno odprtino za pogon diskete, odprtino za čitalno/pisalno glavo ter dve manjši luknjici za indeks in zaščito zapisanih podatkov.

podatki shranjeni na obeh straneh. Premer diskete je 19,8cm, v sredini pa ima osno odprtino premera 3,81cm. Vstavljena je v kvadratni ovitek velikosti 20,3 X 20,3cm iz papirja ali plastike. Notranja stran ovitka je obdana s snovjo, ki omogoča pri vrtenju majhno trenje in dobro ščiti površino diskete.

Podatki so shranjeni na 77 koncentričnih stezah. Vsaka je razdeljena na enake sektorje. Poznamo dva načina sektoriranja: mehko ali programsko in trdo ali stalno sektoriranje. Pri stalnem sektoriranju (Hard Sectoring) ima disketa poleg luknjice za indeks še največkrat 32 enakomerno porazdeljenih luknjic za identifikacijo posameznih sektorjev. Pri programskem sektoriranju (Soft Sectoring) si na stezi izmenično sledita polje za identifikacijo sektorja in podatkovno polje.

Najbolj razširjeni način programskega sektoriranja predstavlja standardni IBM 3740 format. S pomočjo formata zapisanega na disketi, dosežemo zanesljiv zapis in sinhronizacijo prenosa podatkov. Format sestavljajo identifikacijska in podatkovna polja med katerimi so vrzeli. Vrzeli razmejujejo posamezna polja zaradi varnosti in tako preprečujejo, da ne bi variacije v hitrosti pogonskega motorja in v sinhronizaciji vplivale na zanesljivost delovanja. Zlogi vrzeli so običajno kodirani tako, da vsebujejo le urine bite, medtem ko so vsi podatkovni biti ničle. Adresne oznake se razlikujejo od ostalih zlogov po manjkajočih urinih bitih.

3. KRMILNIKI ZA GIBKI DISK

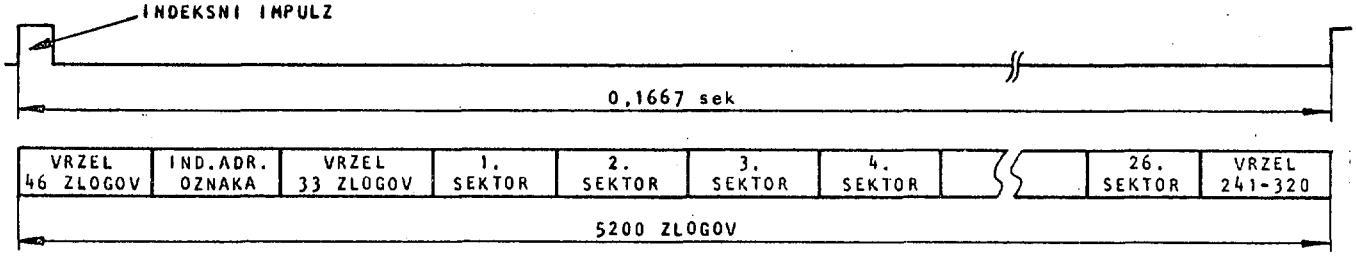
Razlika med krmilniki za gibki disk, ki jih je mogoče kupiti na svetovnem trgu, je predvsem v manjših, manj pomembnih funkcijah in v načinu komuniciranja z mikroračunalnikom. Pri izbiri ustreznega krmilnika je vsekakor odločilnega pomena tip mikroprocesorja, na katerega vodilo bomo priključili krmilnik.

V tem pregledu so zajeti naslednji krmilniki za gibki disk:

MC6843	Motorola
uPD372D	NEC
8271	INTEL
FD1771	Western Digital
FD1791	Western Digital

Krmilniku FD1771 je popolnoma ekvivalenten INS 1771, ki ga proizvaja National Semiconductor Corp..

Vsi omenjeni krmilniki za gibki disk uporabljajo za identifikacijo sektorjev programsko sektoriranje. Uporabljajo lahko standardni IBM 3740 ali pa kakšen drugi format. Formati se razlikujejo med seboj po dolžini podatkovnih polj v sektorju. Dolžina sektorja se lahko programira, vendar pa vsi krmilniki nimajo enakih možnosti. Največji



SLIKA 3: Standardni IBM 3740 format steze

ID.ADR. OZNAKA	ŠTEVILKA STEZE	NIČLE	ŠTEVILKA SEKTORJA	NIČLE	CRC	IDENT. VRZEL	PODAT. ADR.OZN.	P O D A T K I	CRC	PODAT. VRZEL
5 ZLOGOV					2	17	1	128	2	33

SLIKA 4: Standardni IBM 3740 format sektorja

razpon dolžine sektorja ima krmilnik uPD372D in sicer od enega zloga do enega sektorja na stezo. Vsi krmilniki so tudi sposobni zapisati ustrezni format na disketo, jo inicializirati. Nadalje lahko programiramo razdaljo med impulzi na liniji korak, kot to zahteva priključeni gibki disk. Vsi krmilniki razen uPD372D sami generirajo CRC zloge in imajo možnost programiranega ali DMA prenosa podatkov. DMA prenos podatkov poteka brez sodelovanja procesorja. Podatki se direktno prenašajo med diskom in računalniškim spominom.

Za krmilnik MC6843 je značilno, da se lahko direktno priključi na vodila mikroprocesorja M6800. Programira se lahko dolžina stabilizacijskega časa čitalno/pisalne glave, kar omogoča operacije z različnimi gibkimi diski. S pomočjo enostavnega dodatnega vezja lahko krmili več diskov. Za napajanje potrebuje samo napetost +5V. Ima dvanajst direktno dostopnih registrov, preko katerih je vzpostavljena komunikacija s procesorjem. Krmilnik izvaja naslednje ukaze: iskanje steze nič, iskanje poljubne steze, zapis sektorja, čitanje sektorja, zapis več sektorjev, čitanje več sektorjev, zapis enega sektorja z brisalno podatkovno oznako, čitanje CRC zlogov, pisanje v prostem formatu in čitanje v prostem formatu.

Krmilnik uPD372D lahko krmili štiri gibke diske, saj ima štiri izhodne linije za izbiro ustreznega pogona. Sposoben je čitati ali pisati na en gibki disk in istočasno premikati čitalno/pisalno glavo na drugem. Poleg koračnega časa se lahko programira tudi dolžina koračnega impulza. Za napajanje potrebuje napetosti +12V, +5V in -5V. Ima devet registrov, tri za čitanje in šest za zapisovanje. V te registre nalagamo krmilne besede za vsako najmanjšo operacijo. Krmilnik nudi res največ možnosti, toda sam izvede bisveno manj operacij kot ostali. Tak način delovanja zahteva mnogo obsežnejšo programsko opremo. Krmilnik ne moremo uporabiti za DMA prenos podatkov.

Krmilnik 8271 je popolnoma prilagojen na vodila procesorja 8080 in za DMA prenos podatkov s pomočjo DMA krmilnika 8257. Krmili lahko dva gibka diska, z razvojem minimalne programske opreme pa štiri. Programira se lahko stabilizacijski čas, čas vključitve in čas izključitve čitalno/pisalne glave. Avtomatično poišče stezo in izvede verifikacijo, če se je čitalno/pisalna glava stabilizirala nad pravo stezo. Krmilnik vsebuje štiri registre in sprejema petnajst ukazov, ki se nanašajo na premikanje čitalno/pisalne glave, čitanje in pisanje podatkov, formatiranje in na še nekaj manj pomembnih operacij.

Krmilnik INS 1771 avtomatično premakne čitalno/pisalno glavo na poljubno stezo in preveri, če se je operacija pravilno izvedla. Sposoben je čitati enega ali več sektorjev tako, da sam najde ustrezni sektor. Nastavimo lahko stabilizacijski čas glave in izbiramo med korak-smer in tri fazno kontrolno logiko za krmiljenje koračnega motorja. Ta krmilnik je podrobneje opisan v naslednjem poglavju.

Zelo podoben pravkar opisanemu je krmilnik FD1791, ki ima še možnost zapisa podatkov na gibki disk s tako imenovano dvojno gostoto (MFM). Za tak zapis obstaja standardni format IBM sistem 34, ki ima na eni stezi prav tako 26 sektorjev, toda z dolžino 256 zlogov. Omenjeni krmilnik nima notranjega ločevalca podatkov, za napajanje pa potrebuje napetosti +12V in +5V.

3.1. KRMILNIK INS 1771

Krmilnik INS 1771 vsebuje pet registrov, ki so direktno dostopni preko adresnega in podatkovnega vodila. Za izbiro posameznih registrov uporablja računalnik vhoda A0 in A1 (Register Select Linies), ki v konjunkciji z aktivnim vhodom RE ali WE omogočita preko podatkovnega vodila dostop do ustreznega registra. Delni naslovi posameznih registrov:

A1	A0	RE	WE	IZBRANI REGISTER
0	0	0	1	statusni register
0	0	1	0	ukazni register
0	1	0	1	stezni register
0	1	1	0	stezni register
1	0	0	1	sektorski register
1	0	1	0	sektorski register
1	1	0	1	podatkovni register
1	1	1	0	podatkovni register

Pri izbiranju posameznih registrov sta pomembni tudi liniji RE in WE. Vhod RE (Read Enable) dovoljuje računalniku čitanje podatkov ali statusne informacije iz naslovljenega registra. Vhod WE (Write Enable) dovoljuje vpisovanje podatkov ali kontrolnih besed v naslovljeni register. Oba vhoda ne smeta biti istočasno aktivna (logična ničla), medtem ko mora biti vedno, kadar je aktiven eden izmed RE ali WE vhodov, aktiven tudi CS (Chip Select) vhod, ki omogoča komunikacijo med računalnikom in krmilnikom.

Logična ničla na vhodu MR (Master Reset) izbriše ukazni register, resetira sedmi bit (ni pripravljen) v statusnem registru in naredi PH1/STEP izhod aktiven. Ko se MR vrne na logično enico, se nastavi čitalno/pisalna glava nad stezo nič.

Vhod CLK (Clock) dovaja krmilniku pravokotne urine impulze frekvence 2MHz.

Krmilnik INS 1771 ima tudi ločevalec podatkov, ki ga aktiviramo s pomočjo vhoda XTDS. Kadar je ta vhod na logični enici, prihajajo po liniji FDDATA v krmilnik še neločeni podatki. Ti se ločujejo na urine in podatkovne bite v samem krmiliku. Vendar ima ta ločevalec podatkov premajhno zanesljivost. Če pa je vhod XTDS nizek, prihajajo po liniji FDDATA zunaj ločeni podatkovni biti in po liniji FDCLK zunaj ločeni urini biti.

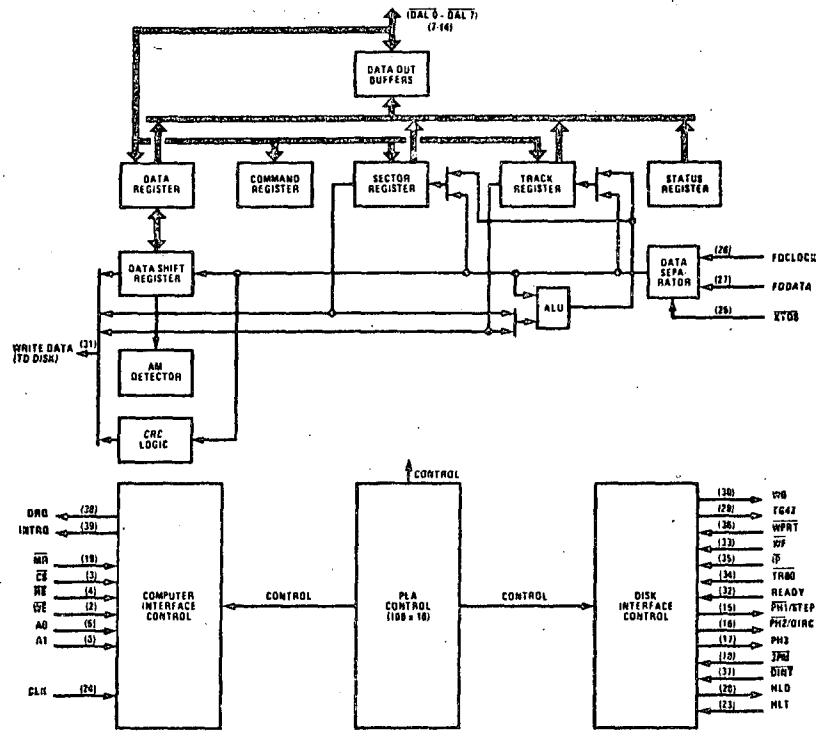
Za pravilno zapisovanje podatkov na disketo sta pomembna dva vhoda: zaščita zapisanega WPRT (Write Protect) in napaka zapisa WF (Write Fault). Če je vhod WPRT nizek se takoj prekine ukaz za pisanje in se nastavi šesti bit (zaščita zapisanega) v statusnem registru. Prav tako se prekine ukaz za pisanje, kadar je nizek vhod WF in se nastavi peti bit (napaka zapisa) v statusnem registru.

Preko vhoda IP (Index Pulse) prihajajo v krmilnik vsaj 10ms dolgi impulzi, kadar je zaznana indeksna oznaka na disketi.

Vhod TR00 (Track 00) je nizek, kadar je čitalno/pisalna glava nad stezo nič.

Vhod pripravljen (Ready) sporoči krmilniku, da je gibki disk pripravljen za čitalno ali pisalno operacijo (logična enica). Kadar je nizek, se čitalna ali pisalna operacija ne izvrši.

Za inicializacijo diskete je pomemben vhod DINT (Disk Initialization). Inicializacija je dovoljena, kadar je ta vhod na logični enici. Če je nizek se ukaz zapis steze (Write Track) prekine in nastavi se šesti bit (zaščita zapisanega) v statusnem registru.



SLIKA 5: Blokovni diagram krmilnika INS 1771

Vhod TEST se uporablja za določitev časovnega presledka med koračnimi impulzi.

S pomočjo HLT (Head Load Timing) vhoda lahko nastavimo zakasnitev pri vključevanju čitalno/pisalne glave, če je to potrebno. Če ni potrebno, ga vežemo na +5V.

Izhodna signala, ki povezujeta krmilnik in računalnik sta le dva. Zahteva podatka (Data Request - DRQ) je izhod, ki se postavi na logično enico, kadar je krmilnik pripravljen na prenos zloga podatkov med čitalno ali pisalno operacijo. Prekinitvena zahteva (Interrupt Request - INTRQ) je izhod, ki se postavi na visok nivo, ko je končana ali prekinjena katerakoli operacija. INTRQ se postavi na nizek nivo, kadar je naložen v ukazni register nov ukaz.

Zapis podatkov na disketo poteka preko linije zapis podatkov (Write Data - WD) in zapis možen (Write Gate - WG). Izhod WD pošlja gibkemu disku podatke (združene podatkovne in urine bite), ki se zapisujejo na disketo. Izhod WG je visok, kadar se zapisujejo podatki na disketo. Pomembna pri zapisovanju je še linija TG43 (Track Greater Than 43), ki sporoči gibkemu disku, da se čitalno/pisalna glava nahaja med stezama 44 in 76.

Krmiljenje koračnega motorja je možno na dva načina, odvisno od vhoda 3PM (Three - Phase Motor Select). Kadar je 3PM nizek, prihajajo preko vhodov PH1/STEP, PH2/DIRC in PH3 zaporedni tri fazni impulzi za krmiljenje tri faznega koračnega motorja. Če je 3PM visok, prihaja preko izhoda PH2/DIRC nivo, ki določa smer premikanja čitalno/pisalne glave. Preko izhoda PH1/STEP pa 4ms dolgi impulzi, ki premikajo glavo s steze na stezo.

Izhod HLD (Head Load) pritisne z logično ničlo čitalno/pisalno glavo k disketi.

Krmilnik INS 1771 ima še osem vhodno/izhodnih signalov (Data Access Linies - DAL), ki so vezani na podatkovno vodilo. Te linije omogočajo direktno komunikacijo med računalnikom in krmilnikom, saj se podatki, kontrolne besede in statusne informacije prenašajo preko podatkovnega vodila.

3.2. DELOVANJE KRMILNIKA INS 1771

Krmilnik INS 1771 pozna enajst ukazov, ki jih računalnik nalaga v ukazni register:

- Nastavitev čitalno/pisalne glave na stezo 00 (Restore)
- Iskanje steze s poljubno številko (Seek)
- Premik glave na sosednjo stezo (Step)
- Premik glave na sosednjo stezo proti centru diska (Step In)
- Premik glave na sosednjo stezo proti robu diska (Step Out)
- Čitanje identifikacijskega polja (Read Adress)
- Čitanje enega ali več sektorjev (Read Command)
- Zapis enega ali več sektorjev (Write Command)
- Čitanje celotne steze (Read Track)
- Zapis celotne steze (Write Track)
- Prekinitev operacije (Force Interrupt)

Posamezni ukazi imajo lahko več variant. Tako lahko določimo pri prvih petih ukazih, to je pri ukazih, kjer gre za premikanje čitalno/pisalne glave, časovno razliko med posameznimi impulzi na liniji korak, položaj glave med izvajanjem operacije (ali je v stiku z disketo ali ni) in izvršitev ukaza brez ali z verifikacijo steze, nad katero se je čitalno/pisalna glava stabilizirala. Pri čitanju ali zapisu sektorja lahko z enim ukazom prečitamo ali zapišemo en sam sektor ali pa več sektorjev naenkrat, dokler ni zahtevana prekinitev ali dokler ni prečitan zadnji sektor na stezi. Pri teh dveh operacijah moramo še določiti dolžino sektorja in pa zakasnitev, ki

je potrebna, da se čitalno/pisalna glava stabilizira.

ZAPIS SEKTORJA: Glava je že nastavljena na ustrezno stezo in v sektorskem registru je zapisana številka sektorja, v katerega hočemo shraniti podatke. Ko sprejme krmilnik ukaz za pisanje, primakne čitalno/pisalno glavo k disketi in čita identifikacijska polja, dokler ne najde ID polje s pravilno številko steze, pravilno številko sektorja in pravilnima CRC zlogoma. Preko linije zahteva podatka (DRQ) sporoči računalniku, da naj naloži prvi zlog podatkov v podatkovni register. Krmilnik sproži operacijo pisanja enajst zlogov za ID poljem. Najprej zapiše šest zlogov ničel, nato podatkovno adresno oznako in podatke. Na koncu podatkovnega polja zapiše še dva ustrezna CRC zloga. S tem je sektor zapisan.

ČITANJE SEKTORJA: Iskanje sektorja poteka enako kot pri pisanju. Ko sprejme krmilnik prvi podatkovni bit čitanega sektorja, sproži linijo zahteva podatka (DRQ) in računalnik lahko prečita zlog iz podatkovnega registra ter ga shrani v svoj spomin. Operacija se ponavlja, dokler ni izpisan zadnji zlog. Na koncu ugotovi krmilnik s pomočjo CRC zlogov, če je bilo čitanje opravljeno brez napake.

S čitanjem statusnega registra dobi računalnik informacijo o poteku in zaključku določene operacije. Statusni register vsebuje naslednje informacije:

- Bit 0: Kadar se izvršuje kakšen ukaz je postavljen na enico, sicer je na ničli.
- Bit 1: Ta bit je dejansko kopija stanja na liniji zahteva podatka in je enica, kadar je podatkovni register poln med operacijo čitanja ali podatkovni register prazen med operacijo pisanja.
- Bit 2: Pri operacijah, s katerimi premikamo čitalno/pisalno glavo, je ta bit komplement vhoda steza 00. Pri ostalih operacijah pa nam sporoči, da računalnik ni pravočasno zapisal ali prečital zloga iz podatkovnega registra.
- Bit 3: Ta bit se postavi na enico, ko pride do CRC napake pri verifikaciji identifikacijskega polja ali pri čitanju podatkov.
- Bit 4: Če se ne ujema številka steze pri verifikaciji, ali če ne najde želenega sektorja, se postavi na enico.
- Bit 5: Bit pokaže, odvisno od operacije: tip zapisa, napako zapisa ali če je glava vključena.
- Bit 6: Ta bit je komplement vhoda zaščita zapisanega. Kadar je enica pokaže, da je v gibki disk vstavljena zaščitena disketa.
- Bit 7: Enica pokaže, da gibki disk ni pripravljen.

4. KRMILNO VEZJE ZA GIBKI DISK

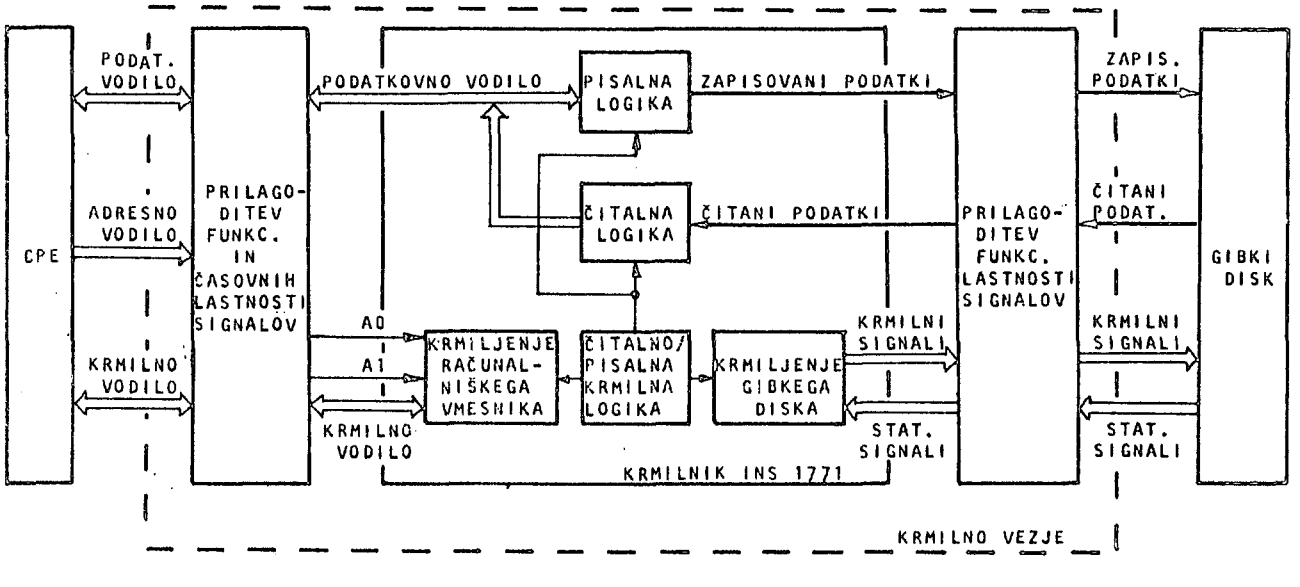
Krmilno vezje (stikalni načrt predstavljata sliki 7 in 8) je realizirano s 26 integriranimi vezji na ploščici dvojnega evropskega formata. Mikroprocesor M 6800 in krmilnik INS 1771 nimata časovno prilagojenih posameznih signalov, zato je krmilno vezje dokaj obsežno. Vse linije: podatkovno, adresno in kontrolno vodilo ter vhodne in izhodne linije gibkega diska so povezane s krmilnim vezjem preko ojačevalnikov s tremi stanji (TRI STATE).

Preko adresnega vodila naslavlja računalnik vseh pet registrov v krmilniku. Adresni liniji A1 in A0 sta direktno spojeni s krmilnikom, medtem ko z ostalimi adresnimi linijami tvorimo s pomočjo VMA signala CS signal, ki omogoča ob ustrezni adresi komunikacijo med mikroračunalnikom in krmilnikom. V realiziranem vezju imajo posamezni registri naslednje naslove:

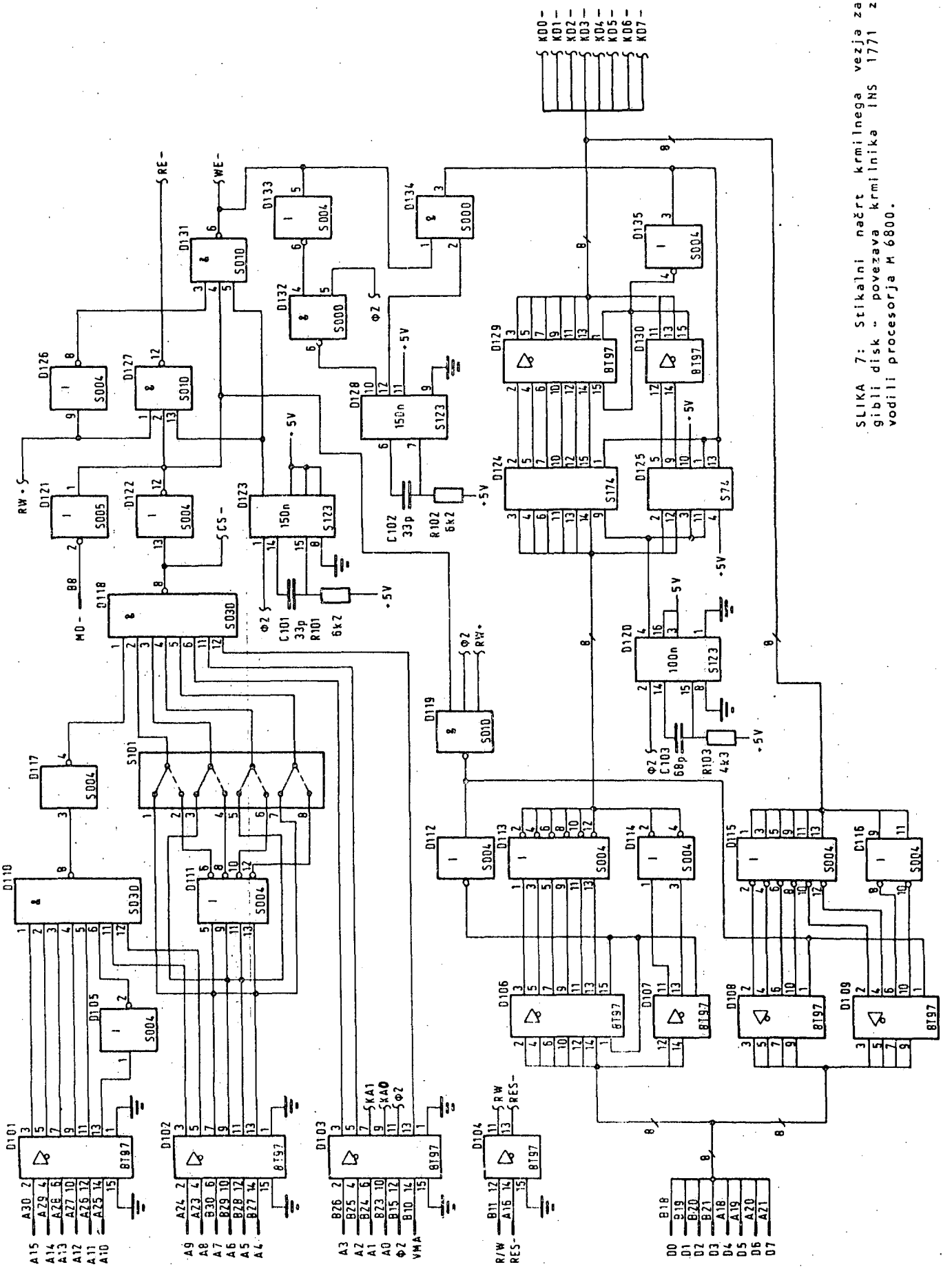
ukazni in statusni register	FBCC
stezni register	FBCD
sektorski register	FBCE
podatkovni register	FBCF

Za časovno prilagoditev posameznih signalov so uporabljeni monostabilni multivibratorji in pa pomnilne celice. Krmilno vezje tvori ustrezno časovno izoblikovana signala RE in WE ter povzroča dodatno zakasnitev podatkov na podatkovnem vodilu. Na mikroprocesorjeva prekinitvena vhoda NMI in IRQ sta vezani liniji prekinitvena zahteva (INTRQ) in podatkovna zahteva (DRQ).

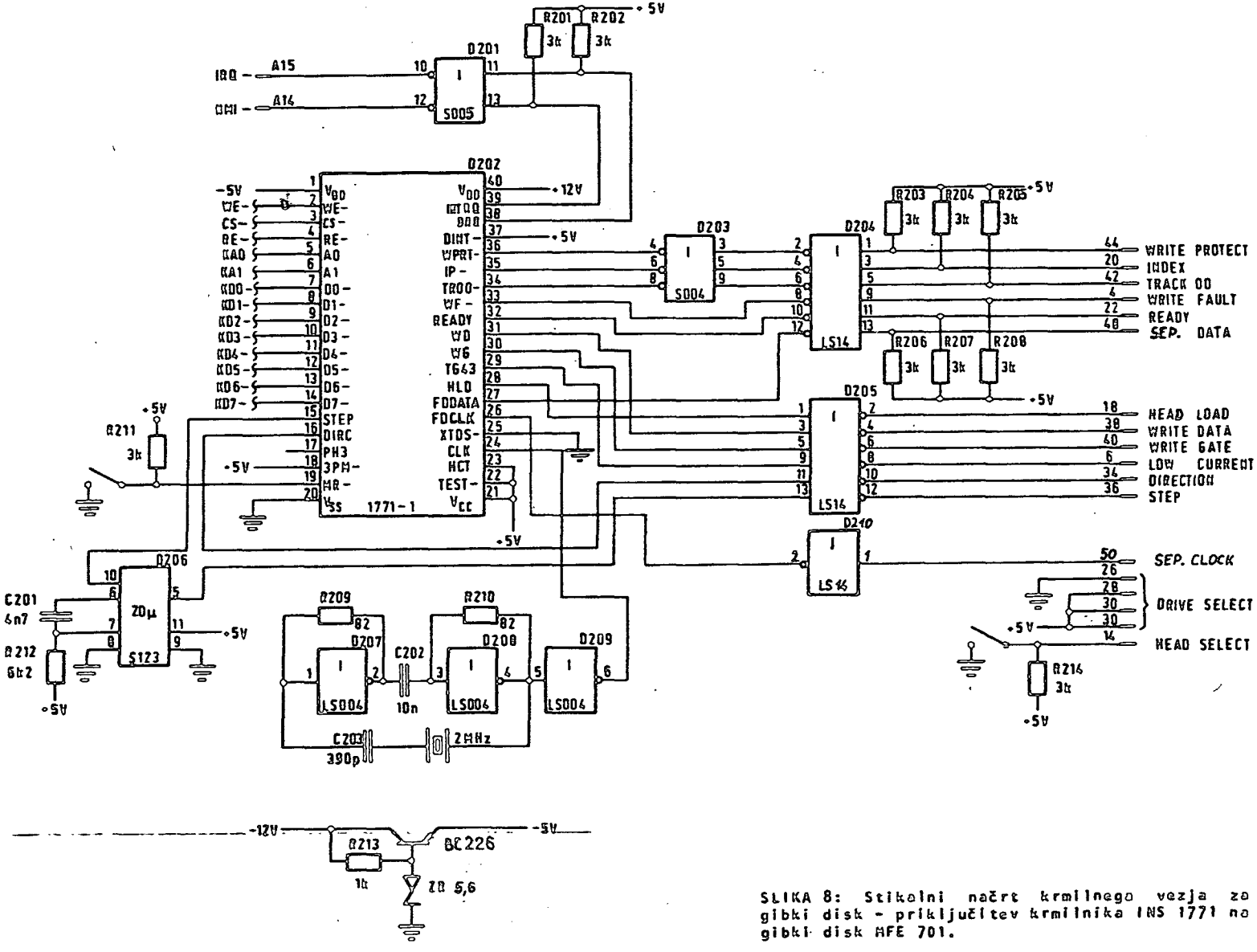
Povezava med krmilnikom in gibkim diskom je enostavna, saj je potrebnih le nekaj



SLIKA 6:Blokovni diagram krmilnega vezja za gibki disk



SLIKA 7: Stikalni načrt krmilnega vezja za gibki disk - povezava krmilnika INS 1771 z vodili procesorja M 6800.



SLIKA 8: Stikalni načrt krmilnega vezja za gibki disk - priključitev krmilnika INS 1771 na gibki disk MFE 701.

negatorjev. Časovno je potrebno prilagoditi linijo korak.

Krmilnik je vključen v krmilno vezje tako, da se podatki ločujejo na podatkovne in urine bite v ločevalcu gibkega diska. Izkušnje so pokazale, da je ločevalec v samem krmilniku premalo zanesljiv!

5. PROGRAMSKA OPREMA

Vse programiranje sloni na vpisovanju podatkov in krmilnih besed v registre ter čitanju podatkov in statusnih informacij iz registrov krmilnika. Krmilnik ima pet registrov: stezni, sektorski, podatkovni, statusni in ukazni.

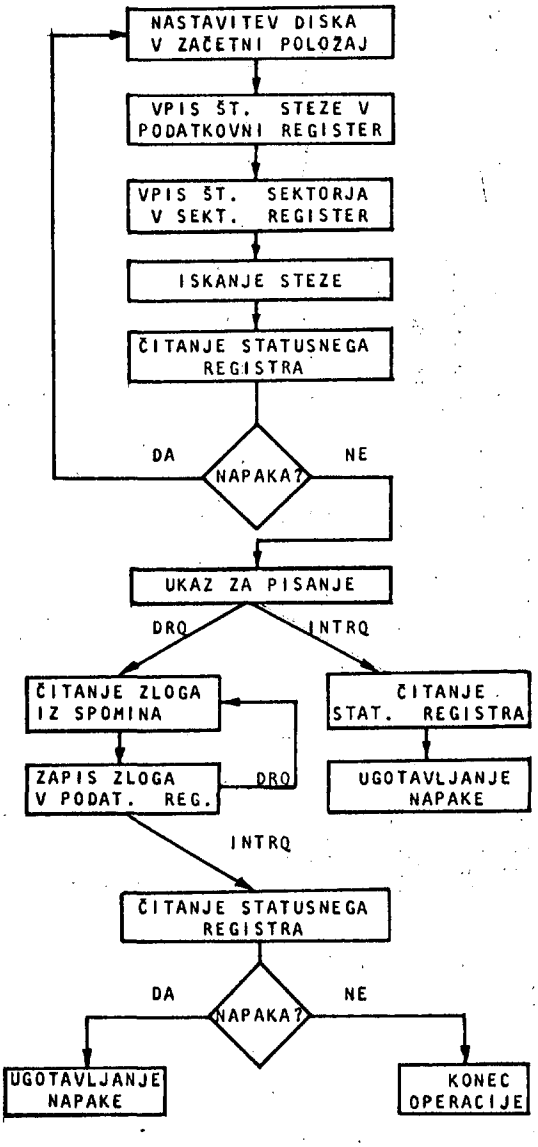
Stezni register vsebuje številko steze, na kateri je čitalno/pisalna glava. V sektorski register vpišemo številko sektorja, katerega hočemo prečitati ali zapisati. V podatkovni register se zapisujejo oziroma čitajo podatki. Vanj naložimo tudi številko steze, na katero hočemo premakniti čitalno/pisalno glavo z ukazom iskanje steze. Statusni register vsebuje v vsakem trenutku informacijo o stanju krmilnika in gibkega diska med izvajanjem ali po izvršitvi operacije. V ukazni register se vpisujejo krmilne besede, ki definirajo enajst ukazov z več variantami.

Računalnik naloži ukaz v ukazni register in krmilnik začne izvrševati operacijo. Krmilnik sporoči računalniku preko linije prekinitvena zahteva, kdaj je operacija zaključena ali prekinjena. Program se nadaljuje na drugi lokaciji, ki je definirana z monitorskim programom. V času, ko izvaja krmilnik ukaz, mora računalnik ustaviti izvajanje programa in počakati na prekinitveno zahtevo. Ker se nato program navadno nadaljuje na naslednji lokaciji, ne potrebujemo posebnega prekinitvenega programa. Zato zapišemo v lokacijo pomnilnika, kamor skoči program ob prekinitvi, le instrukcijo za vrnitev iz prekinitvenega programa (RTI) ali pa instrukcijo za skok na lokacijo spomina, kjer se program nadaljuje.

Linija podatkovna zahteva, ki je priključena na prekinitveni vhod procesorja (IRQ), sproži prekinitev, kadar je med operacijo pisanja podatkovni register prazen, ali če je v podatkovnem registru podatek med poeracijo čitanja. Ker mikroprocesor M 6800 ni dovolj hiter, moramo prekinitveni program, ki obsega čitanje podatka iz spomina in vpisovanje v podatkovni register ali obratno, zapisati na mesto, kjer se začne izvajanje programa ob prekinitvi na liniji IRQ. Ta prekinitveni program se ponavlja, dokler se ne sproži prekinitvena zahteva, vezna na procesorjev vhod NMI.

Vsako disketo je potrebno, preden začnemo zapisovati nanjo podatke, inicializirati oziroma formatirati. To pomeni, da je na vsaki stezi zapisan format, ki omogoča programsko sektoriranje. Formatiranje izvedemo tako, da najprej zapišemo celotni format steze v mikroračunalniški spomin in ga nato prepišemo s pomočjo ukaza zapis steze na ustrezno stezo diskete. Z ukazom zapis steze (Write Track) se zapišejo vsi zlogi na stezo, istočasno pa krmilnk spusti ob ustreznih podatkovnih zlogih posamezne urine bite, ki so značilni za adresne oznake.

Na koncu članka je priložen še program za zapis sektorjev na disketo. Računalnik najprej nastavi gibki disk v začetni položaj in zahteva naslednje informacije: s katerega naslova naj začne prepisovati podatke, na katero stezo in



SLIKA 9: Blokovni diagram poteka programa za zapis podatkov v poljubni sektor diskete.

kateri sektor jih naj zapiše in koliko sektorjev bo obsegal zapis. Nato se začne zapisovanje s pomočjo podprograma za zapis enega sektorja.

Program za čitanje podatkov z diskete je zelo podoben programu za pisanje. Razlikujeta se v tem, da ima ta program namesto podprograma za zapis podprogram za čitanje sektorja. Bistvena razlika med tema dvema podprogramoma pa je le v prekinitvenemu programu. Pri zapisovanju se podatki čitajo iz mikroračunalniškega spomina in se nalagajo v podatkovni register krmilnika, pri čitanju pa poteka prenos podatkov v obratni smeri.

6. ZAKLJUČEK

Pri razvijanju materialne opreme krmilnega vezja pridejo do izraza časovne karakteristike posameznih signalov krmilnika in mikroprocesorja. Ker krmilnik INS 1771 nima časovno prilagojenih signalov s signali mikroprocesorja M6800, mora izdelano vmesno vezje generirati med krmilnikom in vodili procesorja potrebne zakasnitve posameznih

```
                   ***********************************************
                   *                                             *
                   *          ZAPIS PODATKOV NA DISKETO          *
                   *                                             *
                   ***********************************************
                   *
BB00                     ORG   $BB00
                   * CELICE ZA SHRANITEV VMESNIH VREDNOSTI
BB00               IR1   RMB   2        INDEKSNI REGISTRI
BB02               IR2   RMB   2
BB04               STP   RMB   2        SKLAD
BB06               NAP   RMB   1        NAPAKA PRI CITANJU/PISANJU
BB07               NAD   RMB   2        ZAC.NASLOV PRI CITANJU/PISANJU
BB09               STSEK RMB   1        STEVILO SEKTORJEV
                   * MONITORSKI PODPROGRAMI
FC00               MON   EQU   $FC00    MONITOR
FEE6               OUTST EQU   $FEE6    IZPIS BESEDILA
FD41               BADDR EQU   $FD41    VPIS STIRIH HEKS. STEVIL
FD50               BYTE  EQU   $FD50    VPIS DVEH HEKS.STEVIL
                   * KOMUNIKACIJA Z DISKOM
FBCC               UKST  EQU   $FBCC    UKAZNI/STATUSNI REGISTER
FBCD               STEZA EQU   $FBCD    STEZNI REGISTER
FBCE               SEKTR EQU   $FBCE    SEKTORSKI REGISTER
FBCF               PODAT EQU   $FBCF    PODATKOVNI REGISTER
                   *
0003               PREK  EQU   $0003    NASLOV PREKINITVENEGA PROGRAMA
                   *
                   ***********************************************
                   *
BC00                     ORG   $BC00
BC00 BD BC BC            JSR   NAST     NASTAVITEV DISKA
BC03 49                  ROLA
BC04 49                  ROLA           DISKETA ZASCITENA ?
BC05 24 20               BCC   VPPOD    NE
BC07 CE BC 10            LDX   #TEKS5   DA
BC0A BD FE E6            JSR   OUTST    IZPIS BESEDILA
BC0D 7E FC 00            JMP   MON
BC10 0A            TEKS5 FCB   $A,$D
BC12 44                  FCC   "DISKETA JE ZASCITENA"
BC26 04                  FCB   4
BC27 CE BC 9C      VPPOD LDX   #TEKS3
BC2A BD FE E6            JSR   OUTST    IZPIS BESEDILA
BC2D BD FD 41            JSR   BADDR    VPIS NASLOVA
BC30 FF BB 07            STX   NAD
BC33 CE BC 80            LDX   #TEKS1
BC36 BD FE E6            JSR   OUTST    IZPIS BESEDILA
BC39 BD FD 50            JSR   BYTE     VPIS STEVILKE STEZE
BC3C B7 FB CF            STAA  PODAT
BC3F 86 1C               LDAA  #$1C     NASTAVITV STEZE
BC41 B7 FB CC            STAA  UKST
BC44 3E                  WAI
BC45 CE BC 8E            LDX   #TEKS2
BC48 BD FE E6            JSR   OUTST    IZPIS BESEDILA
BC4B BD FD 50            JSR   BYTE     VPIS STEVILKE SEKTORJA
BC4E B7 FB CE            STAA  SEKTR
BC51 CE BC AC            LDX   #TEKS4
```



```
BC54 BD FE E6            JSR   OUTST    IZPIS BESEDILA
BC57 BD FD 50            JSR   BYTE     VPIS STEVILA SEKTORJEV
BC5A B7 BB 09            STAA  STSEK
BC5D BD BD 12            JSR   ZAPIS    ZAPIS SEKTORJA
BC60 7A BB 09      PONOV DEC   STSEK
BC63 27 18               BEQ   KONEC    ZAPIS KONCAN ?
BC65 B6 FB CE            LDAA  SEKTR
BC68 81 1A               CMPA  #26      ZADNJI SEKTOR STEZE ?
BC6A 26 09               BNE   PIS      NE
BC6C 7F FB CE            CLR   SEKTR    DA
BC6F 86 5C               LDAA  #$5C     PREMIK GLAVE NA NASLEDNJO STEZO
BC71 B7 FB CC            STAA  UKST
BC74 3E                  WAI
BC75 7C FB CE      PIS   INC   SEKTR    NASTAVITEV STEVILKE SEKTORJA
BC78 BD BD 1E            JSR   ZAPI1    ZAPIS NASLEDNJEGA SEKTORJA
BC7B 20 E3               BRA   PONOV
BC7D 7E FC 00      KONEC JMP   MON
BC80 0A            TEKS1 FCB   $A,$D
BC82 4E                  FCC   "NA STEZO : "
BC8D 04                  FCB   4
BC8E 0D            TEKS2 FCB   $D,$A
BC90 53                  FCC   "SEKTOR :   "
BC9B 04                  FCB   4
BC9C 0D            TEKS3 FCB   $D,$A,$A
BC9F 5A                  FCC   "Z NASLOVA : "
BCAB 04                  FCB   4
BCAC 0D            TEKS4 FCB   $D,$A
BCAE 53                  FCC   "ST.SEKTORJEV "
BCBB 04                  FCB   4
                   ***********************************************
                   *
                   * NASTAVITEV DISKA V ZACETNI POLOZAJ
                   *
BCBC 86 3B         NAST  LDAA  #$3B     VRNITEV V PROGRAM PO PREKINITVI
BCBE 97 03               STAA  PREK
BCC0 86 0C               LDAA  #$0C     NASTAVITEV NA STEZO 00
BCC2 B7 FB CC            STAA  UKST
BCC5 CE FF FF            LDX   #$FFFF   ZAKASNITEV
BCC8 09            ZA1   DEX
BCC9 26 FD               BNE   ZA1
BCCB B6 FB CC            LDAA  UKST     TESTIRANJE STATUSNEGA REGISTRA
BCCE 81 04               CMPA  #4       STEZA 00 ?
BCD0 27 1E               BEQ   PRAV     DA
BCD2 81 44               CMPA  #$44     STEZA 00 + ZASC. DISKETA ?
BCD4 27 1A               BEQ   PRAV     DA
BCD6 CE BC DF            LDX   #TEKS7   NE
BCD9 BD FE E6            JSR   OUTST    IZPIS BESEDILA
BCDC 7E FC 00            JMP   MON
BCDF 0D            TEKS7 FCB   $D,$A
BCE1 4E                  FCC   "NI PRIPRAVLJEN"
BCEF 04                  FCB   4
BCF0 39            PRAV  RTS
                   ***********************************************
                   *
                   * PREPIS PROGRAMA IZ NASLOVA V IR1 NA 0006
                   *
BCF1 CE 00 06      PREP  LDX   #0006
```

```
BCF4 FF BB 02        STX   IR2
BCF7 FE BB 00  PR1   LDX   IR1
BCFA A6 00           LDAA  0,X
BCFC 08              INX
BCFD FF BB 00        STX   IR1
BD00 FE BB 02        LDX   IR2
BD03 A7 00           STAA  0,X
BD05 08              INX
BD06 8C 00 11        CPX   #$11
BD09 27 06           BEQ   PR
BD0B FF BB 02        STX   IR2
BD0E 7E BC F7        JMP   PR1
BD11 39        PR    RTS
               ********************************************************
               *
               *  ZAPIS SEKTORJA
               *
BD12 CE BD 3F  ZAPIS LDX   #Z1     NASTAVITEV ZACETKA PREPISOVANJA
BD15 FF BB 00        STX   IR1
BD18 BD BC F1        JSR   PREP    PREPIS PROGRAMA IZ Z1 NA 0006
BD1B FE BB 07        LDX   NAD     ZACETNI NASLOV
BD1E BF BB 04  ZAPI1 STS   STP     SHRANITEV SKLADA
BD21 7F BB 06        CLR   NAP
BD24 86 7E           LDAA  #$7E
BD26 97 03           STAA  PREK    NADALJEVANJE PO PREKINITVI
BD28 8E BD 4A        LDS   #ZKON
BD2B 9F 04           STS   PREK+1
BD2D FF BB 00        STX   IR1     SHRANITEV IND. REGISTRA
BD30 FE BB 00  Z2    LDX   IR1
BD33 8E 7F FF        LDS   #$7FFF  NASTAVITEV SKLADA
BD36 A6 00           LDAA  0,X     PODATEK IZ SPOMINA
BD38 C6 AC           LDAB  #$AC    ZAPIS SEKTORJA
BD3A F7 FB CC        STAB  UKST
BD3D 0E              CLI
BD3E 3E              WAI
               *
               *     PREKINITVENI PROGRAM
               *
BD3F B7 FB CF  Z1    STAA  PODAT   ZAPIS NA DISK
BD42 08              INX
BD43 A6 00           LDAA  0,X     PODATEK IZ SPOMINA
BD45 0E              CLI
BD46 8E 7F FF        LDS   #$7FFF  NASTAVITEV SKLADA
BD49 3E              WAI
               *
               *
BD4A FF BB 02  ZKON  STX   IR2
BD4D CE 10 00        LDX   #$1000  ZAKASNITEV
BD50 09        Z3    DEX
BD51 26 FD           BNE   Z3
BD53 FE BB 02        LDX   IR2
BD56 B6 FB CC        LDAA  UKST    TESTIRANJE STAT.REGISTRA
BD59 26 08           BNE   Z4      NAPACEN ZAPIS
BD5B 86 3B           LDAA  #$3B    PRAVILEN ZAPIS
BD5D 97 03           STAA  PREK    VRNITEV PO PREKINITVI (NMI)
BD5F BE BB 04        LDS   STP     NASTAVITEV SKLADA
BD62 39              RTS
```



```
BD63 7C BB 06  Z4    INC   NAP
BD66 F6 BB 06        LDAB  NAP     NAPAKA
BD69 C1 0A           CMPB  #10     DESET PONOVITEV
BD6B 26 C3           BNE   Z2      PONOVITEV ZAPISA
BD6D CE BD 76        LDX   #TEKS6
BD70 BD FE E6        JSR   OUTST   IZPIS BESEDILA
BD73 7E FC 00        JMP   MON
BD76 0A        TEKS6 FCB   $A,$D
BD78 4E              FCC   "NAPACEN ZAPIS"
BD85 04              FCB   4
                     END
      NO ERRORS DETECTED
```

SYMBOL TABLE:

BADDR	FD41	BYTE	FD50	IR1	BB00	IR2	BB02
KONEC	BC7D	MON	FC00	NAD	BB07	NAP	BB06
NAST	BCBC	OUTST	FEE6	PIS	BC75	PODAT	FBCF
PONOV	BC60	PR	BD11	PR1	BCF7	PRAV	BCF0
PREK	0003	PREP	BCF1	SEKTR	FBCE	STEZA	FBCD
STP	BB04	STSEK	BB09	TEKS1	BC80	TEKS2	BC8E
TEKS3	BC9C	TEKS4	BCAC	TEKS5	BC10	TEKS6	BD76
TEKS7	BCDF	UKST	FBCC	VPPOD	BC27	Z1	BD3F
Z2	BD30	Z3	BD50	Z4	BD63	ZA1	BCC8
ZAPI1	BD1E	ZAPIS	BD12	ZKON	BD4A		

signalov. Ta problem seveda ne bi prišel do izraza, če bi bil v krmilnem vezju uporabljen Motorolin krmilnik MC6843, ki ga lahko priključimo direktno na vodila procesorja istega proizvajalca.

Priložena programska oprema je uporabna dejansko le za testiranje delovanja krmilnega vezja in kot osnova za obširnejši programski paket, ki omogoča udobno uporabo gibkega diska kot perifrne pomnilne enote.

LITERATURA

1. M6800 Microprocessor Applications Manual, Motorola Inc., 1975
2. INS1771-1 Floppy Disk Formatter/Comtroller, National Semiconductor Corp., 1977
3. Users Manual,LSI Floppy Disk Controller Chip uPD372D NEC MICROCOMPUTERS, INC., 1977
4. MC6843 Floppy Disk Controller, Motorola Inc., 1978
5. FD 1791 Floppy Disk Formatter / Controller, Western Digital Corporation, October 1978
6. I. Rampil, A Floppy Disk Tutorial, BYTE, december 1977

PROGRAMSKA REŠITEV IZVAJANJA PROGRAMA PO KORAKIH ZA M6800

I. ROZMAN

UDK: 681.3.06

VISOKA TEHNIŠKA ŠOLA, MARIBOR

Izvajanje programa po korakih, oziroma ukaz za ukazom je pri mikroračunalnikih ponavadi rešeno z dodatnim sorazmerno obsežnim logičnim vezjem, ki v precejšni meri podraži ceno sistema, kar je občutno takrat, kadar gre za manj zmogljive mikroračunalniške sisteme, kot je n.pr. UMRS-1 (učni mikroračunalniški sestav). V delu je prikazan način oziroma opisan program za procesor M6800, ki omogoča omenjen način izvrševanja programa in ne potrebuje nobenega dodatnega logičnega vezja.

THE SOFTWARE SOLUTION OF A SINGLE INSTRUCTION EXECUTION ON THE M6800 - The problem of a single instruction execution or the step by step execution is usually solved by the additional hardware, wich increases the price of microcomputer system. This is especially expressed with the small system such as UMRS-1 (the training microcomputer system). This article shows the way and describes the program wich anables the step by step execution of user's program and needs no additional hardware.

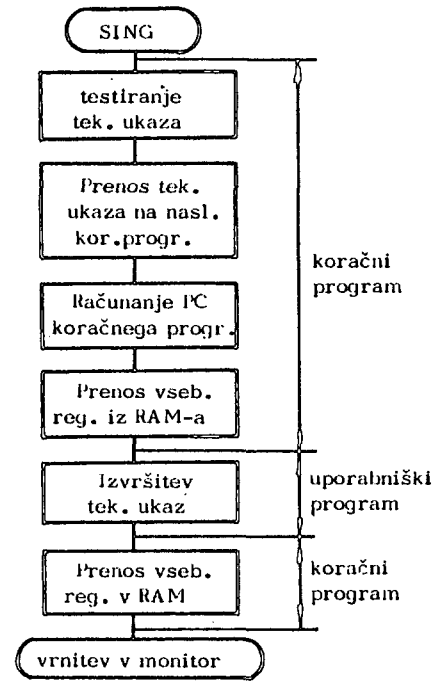
Uvod

Ponavadi mikroračunalniki za izvajanje programa po korakih koristijo linijo $\overline{HALT}$ ali linijo $\overline{NMI}$. Linija $\overline{HALT}$ po izvršenem ukazu ustavi aktivnost CPU-a zato nastane težava, kako prikazati vsebine registrov po izvršenem ukazu. Za prikaz vsebine je potreben aktiven CPU. Možna rešitev je prikazana v literaturi [1]. Pri rešitvi, ki koristi linijo $\overline{IRQ}$ je poleg dodatnega logičnega vezja potreben še krajši program literatura [2], medtem ko je pri programski rešitvi potreben koračni program, ki omogoča izvajanje programa po korakih.

Zasnova programa

Rešitev problema bazira na zahtevi, da računalnik v določenih trenutkih izvaja koračni program, v določenih trenutkih pa uporabniškega. Blokovno je diagram poteka prikazan na sliki 1. Ker se v koračnem programu računajo vrednosti programskega števca, lahko rečemo, da koračni program deloma simulira izvajanje uporabniškega programa. Dejanske vrednosti programskega števca se nanašajo na koračni program.

Algoritem na sliki 1 najprej testira tekoči ukaz. Ugotoviti mora, če je ukaz 1-byten, 2- byten ali 3-byten. To je pomembno zato, da se lahko ustrezno poveča vrednost programskega števca. Testiranje ukaza je sorazmerno enostavno, ker prvo šestnajstiško število kode mnemonikov vseh 1-bytnih ukazov zajema znake 1, 3, 4, 5, vseh 3-bytnih ukazov znake 7, B, F in še kode 8C, 8E in CF, kar so kode takojšnjega naslavljanja (immediate) 2-bytnih registrov. Vse ostale kode z drugačnimi začetnimi znaki mnemonikov so namenjene za 2-bytne ukaze. Računalnik v koračnem programu nato prenese tekoči ukaz iz naslovov nahajanja (naslovi, ki jih zavzame tekoči ukaz v uporabniškem programu) na fiksne naslove, ki so namenjeni za izvršitev tekočega ukaza. Pred izvršitvijo tekočega ukaza je potrebno naložiti registre z vsebinami, ki so se po izvršitvi predhodnega ukaza shranile na fiksne naslove. Za ta prenos vsebin registrov lahko uporabimo ukaz RTI (return from interrupt), vendar bi morali v tem primeru vsebine registrov ležati v skladu (stack). Slabost bi bila, da pokazatelja sklada (stack pointer) v uporabniškem programu ne bi mogli uporabiti kot splošno namenski 2-bytni register. Če bi vanj vpisali vsebino, ki bi bila izven naslovnega področja RAM-ov, bi po izvršitvi takšnega ukaza izgubili vsebine registrov, zato bi bilo nadaljnje izvajanje programa onemogočeno. Zato je za uporabnika boljša rešitev, če za prenos vsebin registrov namesto ukaza RTI uporabimo del programa, ki naredi isto kot ukaz RTI, samo da ne uporabi sklada. Ta del programa je sorazmerno dolg, ker v njem ne smemo uporabljati podprogramov, ker smo tako že vezani na sklad. Po prenosu vsebin iz fiksnih naslovov v registre računalnik izvr-



Slika 1: Blokovni prikaz izvajanja programa po korakih

ši tekoči ukaz, nakar zopet shrani vsebine registrov na fiksne naslove. Za prenos vsebin registrov v spomin lahko uporabimo ukaz SWI. Vendar je to smiselno samo takrat, kadar za obraten prenos uporabimo ukaz RTI, ker je uporaba ukaza SWI tudi vezana na sklad. V primeru, ko prenos vsebin registrov v registre realiziramo programsko potem tudi obraten prenos moramo programsko realizirati.

Diagram na sliki 1 velja za večino ukazov iz nabora ukazov za M6800. Obstaja pa nekaj izjem kot so ukazi, ki se nanašajo predvsem na programski števec. To so: RTS, JSR, JMP in vsi ukazi BRA. Te ukaze koračni program posebej razpozna in jih samo simulira, kar je razumljivo, ker se predvsem nanašajo na spremembe programskega števca. Pri skokih v podprograme, ukazi JSR in pri ukazih povratka RTS koračni program spremeni ustrezno vsebino pokazatelja sklada SP. Izjema so pogojne razvejitve (vsi ukazi BRA), kjer se ukaz izvaja do takšne mere, da ne razpozna pogoj, medtem ko skok koračni program ustrezno izračuna.

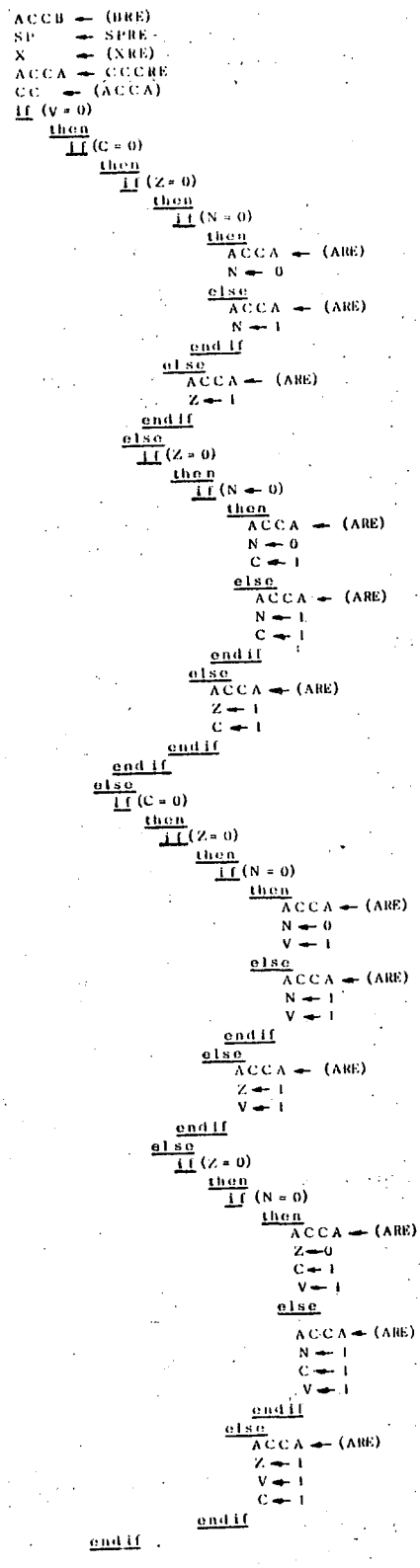
Naložitev registrov z vsebinami

Ta del koračnega programa naloži vsebine registrov iz fiksnih registerskih naslovov v registre. Ker je komuniciranje s pogojnim registrom edino možno preko akumulatorja A, zato vsebino pogojnega registra naložimo najprej v akumulator A, nato pa jo z ukazom TAP prenesemo v pogojni register. Nato akumulator A naložimo z vsebino. Ustrezno vsebini, ki jo naložimo v akumulator A se spremenita bita Z in N v pogojnem registru. Prav tako ukaz LDAA, ki ga uporabimo za naložitev akumulatorja A postavi na nič bit V. Zato moramo te bite predhodno zaznati in jih po naložitvi akumulatorja A postaviti na prvotno vrednost. Ker pa pri korigiranju bita N moramo uporabiti ukaz TST, ta ukaz pa še resetira bit C moramo pred ukazom LDAA testirati bite Z, N, V in C in jim po naložitvi akumulatorja A vrniti prvotno vrednost. Psevdo program, ki opisuje naložitev registrov prikazuje slika 2.
ARE, BRE, SPRE, XRE, CCR so fiksne adrese odkoder blok prenese vsebino v registre ACCA, ACCB, SP, X in CC.

Podoben program bi lahko uporabili tudi za obraten prenos, le da bi namesto ukazov nalaganja uporabili ukaze shranjevanja. Vendar je za obraten prenos smiselno vsak ukaz izvajati dvakrat. Enkrat, da prenesemo vsebino pogojnega registra, drugič pa da prenesemo vsebino ostalih registrov na fiksne naslove, ker takšna rešitev precej skrajša koračni program.

Zaključek

Opisani koračni program zavzame 471 bytov in še 9 bytov v RAM-u, kar predstavlja cenejšo varianto kot realizacija z uporabo linije $\overline{\text{HALT}}$. V primeru realizacije z uporabo linije $\overline{\text{NMI}}$ (cenejša varianta) vendar smo omejeni, ker smo vezani na sklad, zato pokazatelja sklada v uporabniškem programu ne moremo uporabiti kot splošno namenski register. Opisana rešitev je uporabljena v šolskem mikroračunalniškem sestavu UMRS-1, ki ga ISKRA TOZD računalniki vključuje v proizvodnji program.



Slika 2: Psevdo program

Literatura

1. D. Kodek, J. Kožuh, Komandna plošča za mikroprocesor 6800, Elektro tehniški vestnik 1978, št. 1 str. 41 - 44.

2. I. Rozman, R. Babič, V. Žumer, Problemi posamičnega izvajanja inštrukcij z mikroprocesorjem M6800. XXII. Jugoslovenska konferencija za ETAN 12-16. juna 1978, Zadar; str. III. 443 - III 449.

```
*****************************************
*          KORACNI PROGRAM - SING       *
*        PROGRAMIRAL : IVO ROZMAN       *
*****************************************

*
* EQUATE TABELA
*

ADI     EQU     $7E0        ;NASLOVI ZA
ADI1    EQU     $7E1        ;TEKOCI UKAZ
ADI2    EQU     $7E2        ;V RAM-U
***************
ADI3    EQU     $7E3        ;NASLOVI
ADI4    EQU     $7E4        ;ZA SKOK
ADI5    EQU     $7E5        ;IZ RAM-A
***************
ADI6    EQU     $7E6        ;NASLOVI ZA SKOK
ADI7    EQU     $7E7        ;IZ RAM-A PRI
ADI8    EQU     $7E8        ;UKAZIH BRA
***************
CRE     EQU     $7E9
BRE     EQU     $7EA
ARE     EQU     $7EB        ;FIKSNI
XH      EQU     $7EC        ;REGISTERSKI
XL      EQU     $7ED        ;NASLOVI
PH      EQU     $7EE        ;V
PL      EQU     $7EF        ;RAM-U
SH      EQU     $7F0
SL      EQU     $7F1
***************
CRE1    EQU     $7DF        ;POMOZNI CC NASLOV
PPRY    EQU     $F8CB       ;MONITORSKI NASLOV
STP     EQU     $704        ;VREDNOST SKLADA
***************

        ORG     $FC5A

*****************************************
        LDX     PH          ;DOLOCITEV TEK.UKAZA
        LDAA    0,X         ;PRENOS 1 BYTA NA
        STAA    ADI         ;FIKSNI NASLOV
*
* TESTIRANJE UKAZA
*
        CMPA    #$60        ;SO 2 ALI 3 BYTNI
        BPL     DTBY        ;UKAZI ?
        ANDA    #$F0
        CMPA    #$20        ;SO UKAZI BRA ?
        BEQ     BRAN
        LDAA    0,X
        CMPA    #$39        ;JE UKAZ RTS ?
        BNE     INC2

* UKAZ RTS

        LDX     SH
        LDX     1,X         ;RACUNANJE PC
        STX     PH
        LDX     SH          ;POMIK SP
        INX
        INX
        STX     SH
INC     JMP     WRIT        ;SKOK V MONITOR
*
* 1-BYTNI UKAZI
*
INC2    LDAA    #1
        STAA    ADI1        ;NOP NA PROSTE
        STAA    ADI2        ;FIKSNE NASLOVE
        JMP     INS         ;IZVEDBA UKAZA
DTBY    ANDA    #$F0
        CMPA    #$70
        BEQ     TBYT
        CMPA    #$E0        ;LOCITEV 2-BYTNIH
        BEQ     TBYT        ;UKAZOV OD 3-BYTNIH
        CMPA    #$F0
        BEQ     TBYT
        LDAA    0,X
        CMPA    #$8C
        BEQ     TBYT
        CMPA    #$8E
        BEQ     TBYT
        CMPA    #$CE
        BEQ     TBYT
        BRA     DBYT

*
* 2-BYTNI UKAZI
*
BRAN    LDAA    #4
        STAA    ADI1        ;UKAZI BRA
        BRA     WRI1
DBYT    LDAA    0,X
        CMPA    #$8D
        BNE     NOBR

* UKAZ BSR

        JSR     STO1        ;SHRAN. NASL. POVR.
        INX
        INX
        STX     PH          ;RACUNANJE PC SKOKA
        JSR     BRA
        JSR     SP          ;POMANJSANJE SP
        JMP     WRIT        ;SKOK V MONITOR
NOBR    CMPA    #$6E
        BEQ     STK1
        CMPA    #$AD
        BNE     DBY1

* JSR-INDEKSNO

        JSR     STO1        ;SHRAN. NASL. POVR.
        JSR     SP          ;POMANJSANJE SP

* JMP-INDEKSNO

STK1    LDAB    XL
        LDAA    XH          ;RACUNANJE PC
        CLC
        ADCB    1,X
        BCC     ENA1
        INCA
ENA1    STAB    PL
        STAA    PH
        JMP     WRIT        ;SKOK V MONITOR
DBY1    LDAA    1,X         ;PRENOS 2-BYTA UKAZA
        STAA    ADI1        ;FIKSNEGA NASLOVA
WRI1    LDAA    #1          ;NOP NA PROSTE
        STAA    ADI2        ;FIKSNE NASLOVE

        INX
        BRA     INS         ;IZVEDBA UKAZA
*
* 3-BYTNI UKAZI
*
TBYT    LDAA    0,X
        CMPA    #$7E
        BEQ     STAK
        CMPA    #$BD
        BNE     TBY1

* UKAZ JSR-EXTENDED

        LDX     PH
        INX
        STX     PH
        JSR     STO1        ;SHRANITEV NASL.,POVR.
        JSR     SP          ;POMANJSANJE SP
        DEX

* UKAZ JMP-EXTENDED

STAK    LDX     1,X
        STX     PH          ;DOLOCITEV PC SKOKA
        JMP     WRIT        ;SKOK V MONITOR
TBY1    LDAA    1,X
        STAA    ADI1        ;PRENOS 2 IN 3 BYTA
        LDAA    2,X         ;NA FIKSNE NASLOVE
        STAA    ADI2
        INX
        INX
INS     INX
*
* IZVAJANJE UKAZA
*
        STX     PH
        LDX     #RAM2       ;NASL. SKOKA V SING
        STX     ADI7        ;PRI UKAZIH BRA
        LDX     #RAM        ;NASL. SKOKA V SING
BACK    STX     ADI4        ;PRI OSTALIH UKAZIH
        LDAA    #$7E        ;MNEMONIK SKOKA
```

```
        STAA    ADI3
        STAA    ADI6
        LDAB    BRE             ;NALOZITE B,X,SH
        LDX     XH              ;IN CC V REGISTRE
        LDS     SH
        LDAA    CRE
        TAP
*
* KOREKTURA N,Z,V,C BITOV V CC REGISTRU
* PRI NALOZITVI ACCA Z VSEBINO
*
        BVS     VS
        BCS     VCSS
        BEQ     ICC
POZ     BMI     IC1

* V=0 ; C=0 ; Z=0 ; N=0

NEG1    LDAA    ARE
        TST     POZ
        BRA     RT7             ;SKOK NA IZVRS. UK.

* V=0 ; C=0 ; Z=0 ; N=1

IC1     LDAA    ARE
        TST     NEG1
        BRA     RT7             ;SKOK NA IZVRS. UK.

* V=0 ; C=0 ; Z=1 ; N=0

ICC     LDAA    ARE
        BITA    #0
RT7     BRA     KON             ;SKOK NA IZVRS. UK.
VCSS    BEQ     ICC1
POZ1    BMI     IC2

* V=0 ; C=1 ; Z=0 ; N=0

NEG2    LDAA    ARE
        TST     POZ1
        BRA     RT3             ;SKOK NA SEC

* V=0 ; C=1 ; Z=0 ; N=1

IC2     LDAA    ARE
        TST     NEG2
        BRA     RT8             ;SKOK NA SEC

* V=0 ; C=1 ; Z=1 ; N=0

ICC1    LDAA    ARE
        BITA    #0
RT8     SEC                     ;SETIRANJE BITA C
        BRA     KON             ;SKOK NA IZVRSIT.UK.
VS      BCS     VSCS
        BEQ     ICC2
POZ2    BMI     IC3

* V=1 ; C=0 ; Z=0 ; N=0

NEG3    LDAA    ARE
        TST     POZ2
        BRA     RT9             ;SKOK NA SEV

* V=1 ; C=0 ; Z=0 ; N=1

IC3     LDAA    ARE
        TST     NEG3
        BRA     RT9             ;SKOK NA SEV

* V=1 ; C=0 ; Z=1 ; N=0

ICC2    LDAA    ARE
        BITA    #0
RT9     SEV                     ;SETIRANJE BITA V
        BRA     KON             ;SKOK NA IZVRSIT.UK.
VSES    BEQ     ICC3
POZ3    BMI     IC4

* V=1 ; C=1 ; Z=0 ; N=0

NEG4    LDAA    ARE
        TST     POZ3
        BRA     RTA             ;SKOK NA SEC IN SEV

* V=1 ; C=1 ; Z=0 ; N=0

IC4     LDAA    ARE
        TST     NEG4
        BRA     RTA             ;SKOK NA SEC IN SEV

* V=1 ; C=1 ; Z=1 ; N=0

ICC3    LDAA    ARE
        BITA    #0
RTA     SEV
        SEC                     ;SET BITOV V IN C
KON     JMP     ADI             ;SKOK NA IZVRSILT.UK.
*
* VRNITEV V RUTINO SING
*
RAM     TPA
        STAA    CRE1
        LDX     #RAM1           ;SHRANITEV CC V RAM
        JMP     BACK            ;PONOVITEV IZVRS.UK.
RAM1    STAA    ARE
        STAB    BRE
        STX     XH              ;SHRANITEV A,B,IX,SP
        STS     SH              ;V RAM
        LDAB    CRE1            ;ZAMENJAVA STARE
        STAB    CRE             ;VREDN. Z NOVO V CC
WRIT    JMP     PPRY            ;SKOK V MONITOR
*
* VRNITEV V SING PRI UKAZU BRA
*
RAM2    LDS     #STP            ;DEFIN.MON.SKLADA
        BSR     BRA
        BRA     WRIT            ;SKOK V MONITOR
**************

*
* RACUNANJE PC PRI UKAZIH BRA
*

BRA     LDX     PH
        DEX
        LDAA    0,X
        CLC
        TSTA
        BMI     NEG

* SKOK NAPREJ

        ADCA    PL
        BCC     PRES            ;PRISTETJE OPERANDA
        INC     PH
PRES    BRA     STOR

* SKOK NAZAJ

NEG     ADCA    PL
        BCS     STOR            ;PRISTETJE KOMPLEM.
        DEC     PH              ;OPERANDA
STOR    STAA    PL
        RTS
*
* POMANJSANJE SP ZA 2
*
SP      LDAB    #2
        LDAA    SL
        CLC                     ;ODSTETJE 2 OD SL
        SBA
        BCC     DEC
        DEC     SH              ;DEKREMENTIRANJE SL
DEC     STAA    SL
        RTS
*
* SHRANITEV ADRESE POVRATKA PRI BSR IN JSR
*
STD1    LDX     SH
        LDAA    PH
        LDAB    PL
        CLC
        ADCB    #2              ;PRISTETJE 2 ZA
        BCC     IPP2            ;2-BYTNE UKAZE
        INCA
IPP2    STAB    0,X
        DEX
        STAA    0,X
```

```
        LDX     PH
        RTS
**************************************************
        END
```

```
S11EFC5AFE07EEA600B707E0B1602A2AB4F0B1202742A600B1392613FE07F013
S11EFC75EE01FF07EEFE07F00808FF07F07EFDE0B601B707E1B707E27EFD28CE
S11EFC90B4F0B1702766B1B02762B1F0275EA600B1BC2758B1BE2754B1CE2781
S11EFCAB502007B604B707E1203FA600B16D2611BDFE1A0808FF07EEBDFDEAD3
S11EFCC6BDFE0A7EFDE0B16E270AB1AD261BBDFE1ABDFE0AF607EDB607EC0C37
S11EFCE1E90124014CF707EFB707EE7EFDE0A601B707E1B601B707E208202CF4
S11EFCFCA600B17E2712B1BD2616FE07EE08FF07EEBDFE1ABDFE0A09EE01FF11
S11EFD1707EE7EFDE0A601B707E1A602B707E2080808FF07EECEFDE3FF07E748
S11EFD32CEFDC4FF07E4B67EB707E3B707E6F607EAFE07ECBE07F0B607E906B7
S11EFD4D293925182712280B8B607EB7DFD53200DB607EB7DFD552005B607EBA3
S11EFD68B500205527122B08B607EB7DFD6E200DB607EB7DFD702005B607EBFA
S11EFD83B5000D2039251C2712280B8B607EB7DFDBC200DB607EB7DFDBE200519
S11EFD9EB607EB05000B201B2712280B8B607EB7DFDA8200DE607EB7DFDAA2084
S11EFDB905B607EB05000B0D7E07E007B707DFCEFDCE7EFD35B707EBF707EAFE
S11EFDD4FF07EC8F07F0F607DFF707E97EF80B8E0704BD0220F6FE07EE09A644
S11EFDEF000C9D2B0AB907EF24037C07EE2008B907EF25037A07EEB707EF39CC
S11EFE0AC602B607F10C1024037A07F0B707F139FE07F0B607EEF607EF0CC966
S110FE250224014CE70009A700FE07EE3996
```

MIKROPROCESORSKI TERMINALSKI KONCENTRATOR

D. VRSALOVIĆ
N. FILIPOVIĆ

UDK: 681.3.07

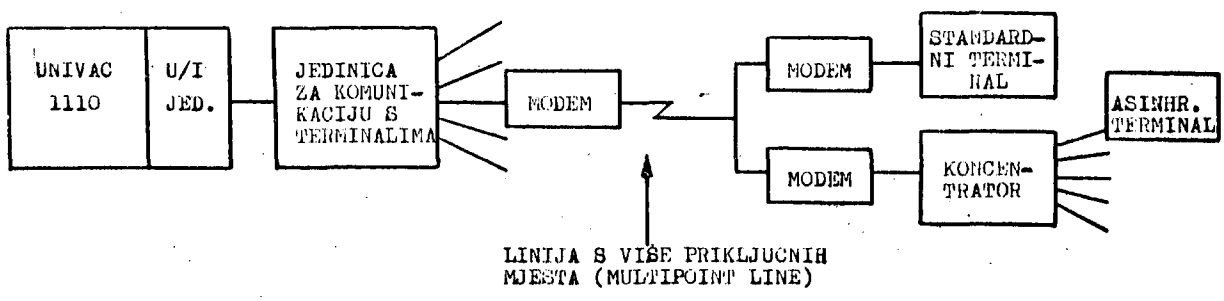
ZAVOD ZA ELEKTRONIKU
ELEKTROTEHNIČKI FAKULTET, ZAGREB

U članku je razmotrena mogućnost priključenja većeg broja terminala na jednu ulazno - izlaznu liniju računala. Opisana je struktura uredjaja koji obavlja koncentratorsku funkciju, kao i karakteristike priključenih terminala. Analizirana su svojstva nadzornog programa u ovisnosti o povezivanju koncentratora s računalom.

TERMINAL MULTIPLEXER WITH MICROPROCESSOR The article deals with the possibilities of connecting a certain number of terminals to one input - output line of a large computer. The structure of the multiplexer is described, as well as the characteristics of the connected terminals. The article also provides an analysis of the structure of the supervisor programmes which are dependent upon the complexity of the performed tasks.

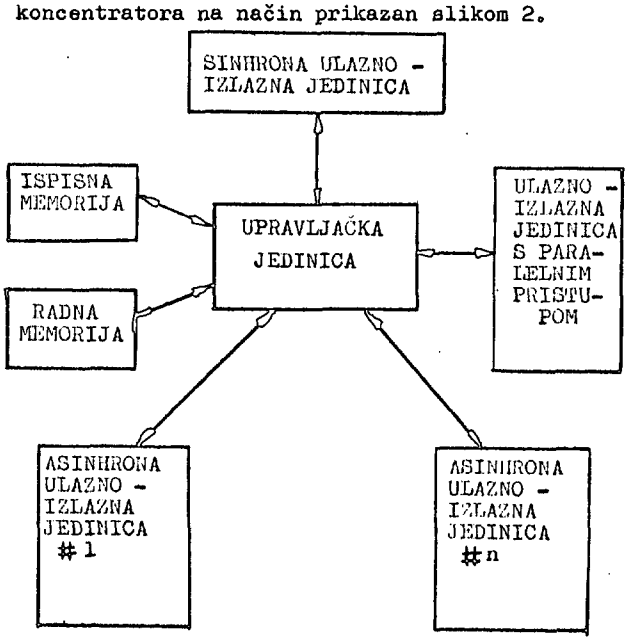
1. UVOD

Jedan od problema koji se javljaju pri organiziranju velikih računskih centara je uspostavljanje mreže terminala namijenjene povezivanju širokog kruga korisnika s centralnim računalom. Ovo je povezivanje ograničeno mnogim faktorima, medju kojima se ističu cijena upotrebljene opreme i problemi ostvarenja i održavanja veze (najčešće telefonskom linijom) izmedju terminala i računala. U nastavnim aktivnostima Zavoda za elektroniku Elektrotehničkog fakulteta u Zagrebu uključen je intenzivan rad s računalom UNIVAC 1110 Sveučilišnog računskog centra, a problemi povezivanja riješeni su korištenjem terminalskog koncentratora vlastite izrade.(Sl. 1.) Računalo UNIVAC 1110 šalje i prima poruke prema pravilima sinhrone komunikacije. Struktura poruka uvjetovana je znakovnim komuni-



Sl. 1. Povezivanje terminala s računalom UNIVAC 1110

kacijskim protokolom. Protokol definira točan oblik kontrolnih i tekstovnih poruka koje se razmjenjuju izmedju računala UNIVAC 1110 i standardnih terminala (npr. UNISCOPE 100, UNISCOPE 200, DCT 1000), kao i njihov pravilan slijed. Ukoliko se kao terminal koristi neki od tvorničkih uredjaja, u njemu su ugradjeni sklopovi koji omogućuju sinhronu komunikaciju prema zadanom protokolu. Za povezivanje nekoliko ovakvih terminala na jednu liniju koristi se multipleksor u kojemu su sklopovski riješeni problemi opisani u trećem poglavlju. Visoka cijena ovih uredjaja njihov je osnovni nedostatak. Ukoliko se oni zamijene znatno jednostavnijim asinhronim terminalima koji nemaju sklopove za podržavanje protokola, potrebno je izmedju terminala i korisničkog modulatora/demodulatora (MODEMa) uključiti odgovarajući medjusklop, u ovom slučaju terminalski koncentrator. Terminalski koncentrator zamišljen je kao uredjaj koji rješava opisane probleme i omogućuje priključenje većeg broja jednostavnijih terminala na jednu ulazno - izlaznu liniju računala. U tu svrhu koncentrator mora razmjenjivati sinhrone poruke s matičnim računalom i brinuti se o poštivanju komunikacijskog protokola. Pored toga, koncentrator proslijedjuje tekstovne poruke terminalima (asinhrono), te prima tekst koji terminali šalju računalu. Navedene dužnosti uvjetuju gradju terminalskog koncentratora na način prikazan slikom 2.



Sl. 2. Struktura terminalskog koncentratora

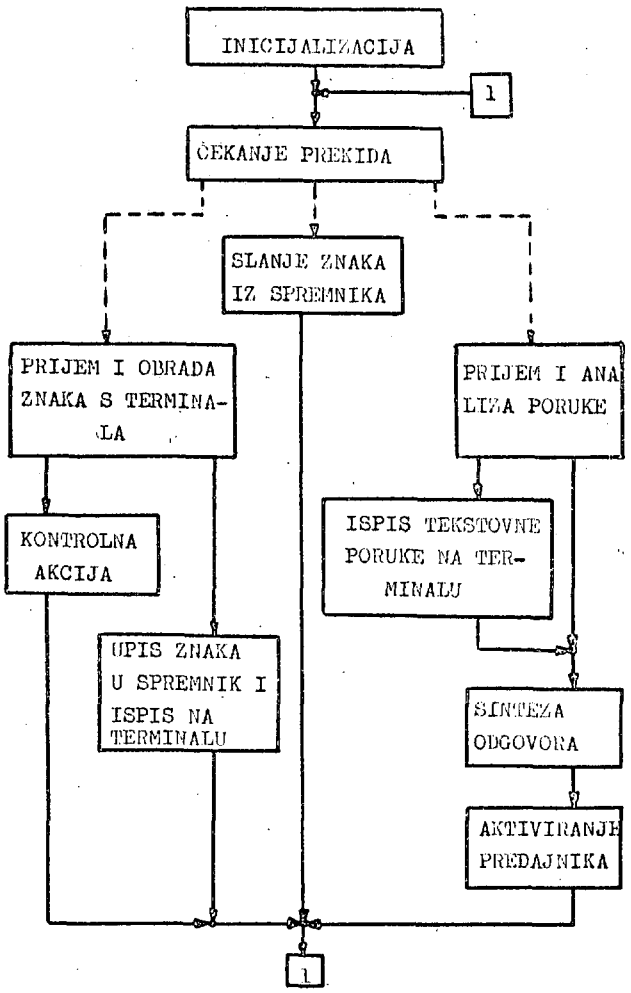
U ispisnoj memoriji smješten je nadzorni program čije izvodjenje omogućuje normalan rad koncentratora. Radna memorija služi za privremeno smještanje poruka koje se šalju računalu ili terminalima putem sinhrone ili jedne od asinhronih U/I jedinica. U/I modul s paralelnim pristupom služi za priključenje brze periferne jedinice (npr. brzi pisač, čitač trake i sl.). Upravljačka jedinica realizirana je korištenjem mikroprocesora Fairchild F8 i memorijskog medjusklopa, dok su U/I jedinice izvedene korištenjem odgovarajućih sklopova visokog stupnja integracije.

2. NADZORNI PROGRAM

Nadzorni program terminalskog koncentratora napisan je u mnemoničkom jeziku za F8 mikroprocesor, te u radnoj verziji zaprema manje od 4 K riječi ispisne memorije. Korištenje mnemoničkog jezika uvjetovano je optimiranjem vremena izvodjenja pojedinih odsječaka programa, kao i minimiziranjem njegovih dimenzija. Pored toga, ustrojstvo F8 mikroprocesora sugerira niz programskih rješenja koja su najlakše ostvariva u strojnom jeziku.

Struktura nadzornog programa prikazana je na slici 3. Nakon inicijalizacije pristupnih sklopova i pripreme registara oznaka i brojača, program ulazi u petlju čekanja prekida. Petlja čekanja je pasivna jer se zbog specijalizirane namjene sistema odustalo od uvodjenja bilo kakve pozadinske aktivnosti. Prekid može izazvati neka od U/I jedinica u trenutku kada se želi aktivirati. Istovremeno posluživanje dvije U/I jedinice nije moguće, te je zbog toga potrebno zabraniti njihovo simultano aktiviranje. To je izvedeno korištenjem prioritetnog lanca dozvole prekida. Jedinica spojena u lanac može se aktivirati samo u slučaju kad niti jedna od važnijih jedinica ne traži pažnju procesora. Najveću važnost dobile su jedinice čije se posluživanje mora izvesti u najkraćem vremenu po pojavi signala zahtjeva za prekidom.

Nakon odobravanja prekida jedinica koja ga je izazvala postavlja na sabirnicu adresu programskog odsječka koji taj prekid poslužuje. Time je izbjegnut dugotrajan postupak ispitivanja izvora prekida, a pojednostavljena su i neka sklopovska rješenja. Programski odsječak za posluživanje prekida ispituje stanje pristupnog sklopa i provodi odgovarajuću akciju



Sl. 3. Struktura nadzornog programa

(najčešće prijenos podataka iz/u pristupni sklop uz promjene oznaka i brojača). Zahvaljujući modularnosti programa, svaki je odsječak za posluživanje prekida dovoljno kratak da sam bude neprekidiv. Time su izbjegnuti mnogi problemi koji bi se pojavili pri izvodjenju programa u realnom vremenu. Obrada prekida traje u prosjeku 50 us. Najmanji razmak izmedju dvaju prekida uvjetovan je brzinom komuniciranja i u promatranom sistemu iznosi 1/2400 sek. Na osnovu statističkih procjena, vrijeme koje nadzorni program provede u čekanju prekida iznosi do 90% ukupnog vremena. Modularnost programa donijela je dvije značajne prednosti: dodavanje novih U/I jedinica ili promjena načina rada postojećih, provodi se jednostavnim dodavanjem odgovarajućih modula i ne unosi bitnije promjene u strukturu programa, a većina postojećih odsječaka ostaju nedirnuti. Pored toga, znatno je olakšano ispitivanje programa, a omogućena je i potpuna verifikacija većine funkcija koje se obavljaju.

3. POVEZIVANJE KONCENTRATORA S RAČUNALOM

Dio nadzornog programa namijenjen podržavanju komunikacije izmedju računala i koncentratora nastao je emulacijom DCT 1000 protokola te primjenom pravila za rad multipleksora namijenjenog povezivanju više terminala. Svaka poruka koju šalje računalo analizira se znak po znak u toku prijema. Ukoliko se ustanovi da je format poruke neispravan, ili da je poruka upućena drugoj grupi terminala priključenoj na istu telefonsku liniju (slika 1.), poruka se ignorira. U slučaju primitka formalno ispravne poruke, analiza je završena neposredno po prijemu posljednjeg znaka, pri čemu su postavljene odgovarajuće oznake u registru stanja terminala kojima je poruka bila upućena. Ukoliko se radilo o tekstu, započinje se slanje znakova adresiranom terminalu, a ako se radilo o kontrolnoj poruci, provodi se sinteza odgovora u ovisnosti o trenutnom stanju terminala. Svako odstupanje od slijeda poruka definiranog protokolom smatra se pogreškom nastalom uslijed gubitka (odnosno deformiranja) poruke u prenosu. U tom se slučaju traži ponavljanje poruka koje traje sve do uspostavljanja ispravnog stanja.

U slučaju većeg broja aktivnih terminala povećava se vjerojatnost da dva terminala istovremeno zatraže slanje poruke računalu. Do konflikta ne dolazi zahvaljujući cikličkom ispitivanju stanja terminala, u toku kojeg se definira koji terminal ima pripremljenu poruku najvišeg prioriteta (tekst). Taj terminal postaje nosilac poruke u slijedećoj predaji, a poruci se mogu dodati kontrolne poruke nižeg prioriteta u skladu s definiranim protokolom. U slučaju da niti jedan terminal nema pripremljenu tekstovnu poruku, provodi se ciklično ispitivanje na slijedećoj razini prioriteta (kraj zauzeća terminala), sve do najniže razine. Ukoliko niti jedan terminal nema poruke, šalje se kontrolna poruka "nema prometa". Terminal koji je u jednoj predaji bio nosilac poruke, u slijedećoj biva ispitan posljednji u nizu. Ovime je ostvareno malo vrijeme posluživanja

terminala, a broj poruka koje se prenose između računala i koncentratora sveden je na minimum. Ispitivanje s četiri terminala u paralelnom radu pokazalo je da je vrijeme odziva terminalskog koncentratora znatno ispod vremena odziva operacionog sistema računala UNIVAC 1110, te su sa stanovišta korisnika terminali transparentno spojeni na računalo. S druge strane, vrijeme analize i sinteze poruke dovoljno je kratko da ne unosi nikakvo kašnjenje u odziv terminala na prozivku računala (500 µs prema približno 30 ms), čime je osiguran normalan rad sistema.

4. FUNKCIONALNE KARAKTERISTIKE TERMINALSKOG KONCENTRATORA

Izvedeni koncentrator omogućuje priključenje do osam asinhronih terminala (korištenjem 20 mA petlje ili V24 standarda) na jednu U/I liniju računala UNIVAC 1110. Pored alfanumeričkih znakova koji sačinjavaju poruku (do 256 znakova po poruci), uvedeni su i posebni kontrolni znakovi sa slijedećim značenjima:

- "RETURN" zaključenje jedne linije teksta (poruke)
- "CTR L" brisanje prethodno upisanog teksta
- "CTR X" prekid ispisivanja teksta na terminalu (pogodno za terminale s video izlaznom jedinicom); djelovanje se poništava tipkom "RETURN"
- "CTR Z" tekst namijenjen terminalu ispisuje se na brzom pisaču (služi za veće ispise); pridjeljivanje pisača ne provodi se ukoliko je on već zauzet, kraj pridjeljivanja ostvaruje se ponovnim pritiskom na tipku "CTR Z"
- "CTR E" znakovi primljeni od računala šalju se terminalu bez dodavanja kontrolnih znakova za povrat glave pisača i novi red (koristi se pri crtanju slike na grafičkim terminalima).

Definiranje novih kontrolnih znakova provodi se dodavanjem odgovarajućeg modula u rutinu za posluživanje prekida tastature, čime je moguće znatno poboljšati funkcije lokalnog editiranja teksta, odnosno povećati izbor opcionalnih ulazno - izlaznih jedinica.

5. ZAKLJUČAK

Upotreba mikroprocesora F8 i pridruženih sklopova visokog stupnja integracije omogućila je realizaciju terminalskog koncentratora koji programski simulira ponašanje standardnih terminala sistema UNIVAC 1110. Budući da je cijena ovakvog uredjeja te upotrijebljenih asinhronih terminala znatno niža od cijene odgovarajućeg broja standardnih terminala, kao i da su u postojećoj konfiguraciji ostvarena neka funkcionalna poboljšanja u odnosu na te terminale, potvrdjena je ispravnost razvoja opisanog sistema. Modularna gradja koncentratora omogućuje njegovu primjenu u raznim situacijama. Uz odgovarajuće nadopune moguće je ostvariti univerzalnost koncentratora u odnosu na komunikacijske procedure, a time i omogućiti povezivanje s velikim brojem postojećih sistema računala. Pored toga, uz razumno male sklopovske i programske izmjene funkcija koncentriranja može se uobličiti u sistem za prikupljanje podataka i djelomičnu lokalnu obradu, što predstavlja kvalitativni skok u odnosu na postojeći uredjaj.

6. LITERATURA

Vrsalović,D.: "Alfanumerički terminal za primjenu u nastavi", ETF Zagreb, 1974.

UNIVAC DCT 1000 Data Communication Terminal, 1970.

Peatman,J.B.: "Microcomputer-Based Design", Mc Graw - Hill, 1977.

Fairchild Micro Systems: "Hardware Reference Manual", 1976.

Filipović,N.: "Znakovni terminal s mikroprocesorom", ETF Zagreb, 1979.

INFORMATICA 1/1979

REMOTE JOB ENTRY BETWEEN HETEROGENOUS COMPUTERS

J. KNOP
R. SPETH

UDK: 681.327.8

COMPUTING CENTRE, UNIVERSITY OF DUSSELDORF
FEDERAL REPUBLIK GERMANY

This article gives a short review of possibilities to perform remote job processing between heterogenous computers. Starting from the method of emulation for RJE station protocols within each connected host a special implementation in applying this method is described - namely the usage of a minicomputer as interface processor between heterogenous hosts to map the host specific station protocols onto each other. The applicability of such interface processors within modern network projects as "network entry processors" is outlined.

PRENOS DALJINSKIH PAKETNIH OBDELAV MED RAZLIČNIMI RAČUNALNIKI - Članek podaja kratek pregled možnosti za prenos poslov med računalniki različnih proizvajalcev. Izhajajoč iz metode emulacije protokolov paketnih terminalov na vsakem od priključenih računalnikov, je najprej opisana posebna uporaba tega principa: uporaba miniračunalnika, ki služi kot vmesni procesor med različnimi računalniki kot protokolni pretvornik. Nato je opisana možnost uporabe takšnih vmesnikov v smislu "mrežnih vstopnih procesorjev" v kontekstu modernih računalniških mrež.

1. General overview

The realisation of connections between heterogenous computers for Remote Job Entry(RJE) processing is presently done in two different ways

- emulation of the systems specific RJE interface of the host attached for remote execution at the job submitting site
- implementation of a network wide standardized RJE procedure at each system attached to the network

In case of an emulation solution RJE facilities provided by a "master" host are used by a coupled "slave" host working in a station mode of the master. Therefore job exchange in both directions can be pictured according figure 1.

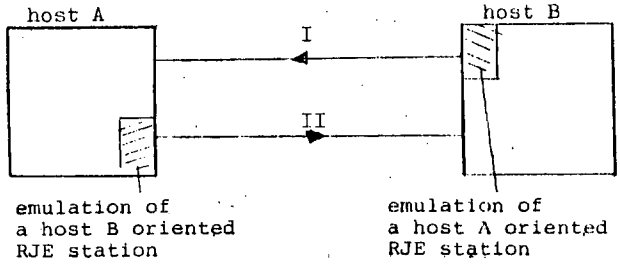


Figure 1

The inclusion of additional systems C,D,.... using the emulation technique let increase very rapidly the number and amount of emulation software needed. Consequently valuable system resources have to be reserved. Because of different line control procedures of each master system the job exchange according to I and II in figure 1 is normally bound to two seperate communication lines. A coupling of large scale computers on emulation level is for example [1].

The other kind of realisation- that is the implementation of a standardized RJE procedure - avoids these disadvantages as well in the simple case of a point to point connection , for example [2], as in the case of the existence of a network which manages the transport of data between the communicating computers. In the last case communication control procedures can be divided into "network access protocols" and "higher level protocols". The RJE protocol as a higher level protocol is normally placed on top of all communication protocols(figure 2). The GMD Network [3] is an example for an implementation of that kind.

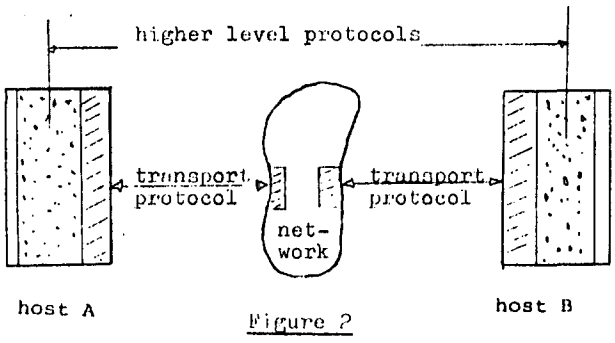


Figure 2

In order to be compatible between different communication networks generally agreed communication protocols are needed. A first good step into that direction is the recommendation X.25 by the CCITT [4] which fixes the interface to networks for a line access(HDLC) and a packet switching procedure. Additionally efforts are undertaken on national and international level to elaborate standardized higher level communication protocols [5], [6] . According to the comfort and the generality of such hierarchically ordered protocols a large expend for implementation has to be put up with them. Furthermore there partially may be problems because of the lack of suitable communication hardware at presently built computer systems to support the new control procedures which are bit oriented at least on network level.

A special method of realizing RJE exchange between two(or perhaps three) computers which is an extension of the emulation technique but can be used in general network implementations(see chapter 3) has been applied in a project at the Computing Center of the University of Düsseldorf, and can be described as

coupling of hosts via protocol mapping within a small interface processor.

This method consists in taking the emulation functions out of the host systems and put them into a small computer in between which works as a slave station to both sides and maps the host specific protocols P_1 and P_2 at an internal interface I onto each other(see figure 3).

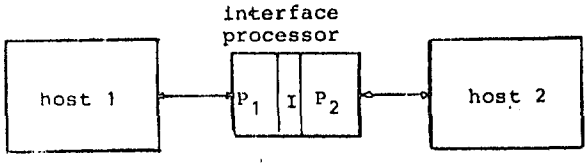


Figure 3

Main advantages of this method with respect to emulations done in the host systems themselves are for example

- cheap hardware solution
- straight forward software development on a functionally limited system
- comfortable manipulation of the small stand alone system in contrary to host system integrated communication processors which often lack I/O and software test facilities
- uncritical testing with standard RJE ports at each side without reserved test times for the whole host system

In chapter 2 the performed implementation is shortly described and in chapter 3 the extended usability of this method for general network implementations is outlined.

2. RJE coupling via an interface processor

Two large scale computer sytems, a CGK TR440 at the Computing Center of the University of Düsseldorf and a CDC CYBER 76/72 at the Computing Center of the University of Cologne, have been coupled for RJE via the minicomputer system Dietz 621(see figure 4). The two systems have extended local networks with both dialoque and remote batch stations. By this connection users at the TR440 side have accesss to the extreme calculation performance of the CYBER system, while the users at the CYBER side can use special software packages on the TR440 system (mainly for administrative purposes). The job submission can take place at any local terminal on both sides.

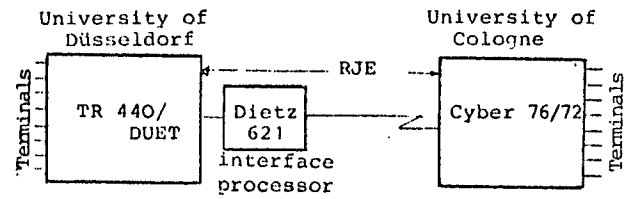


Figure 4

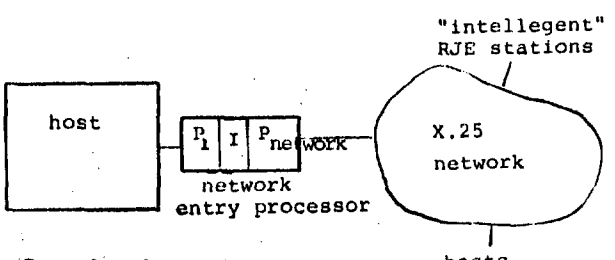
The system Dietz 621 has a main memory of 32 kB and is attached to both host systems by 4800/ 9600 baud connections. The standard station ports used are those of a UT200 at the CYBER system and of a dialoque station SIG51 at the TR440 system with the polling line procedures MODE4 and MSV1 respectively. Main functions performed by the Dietz 621 system are

- handling the two line access procedures
- automatic service for the CYBER dialoque system INTERCOM within the UT200 station
- handling the different data streams for job input, job output, operator communication and job status information
- control of data flow between the TR440 and Dietz 621 above station level by a (simple, "higher", two bytes) control protocol
- definite restart mechanism supervised by the Dietz system

3. Network entry processors in communication networks

The special coupling for RJE as described before is no extendable way for a general solution. For presently planned general networks - on national basis for example the "Pilot Project(PIX)", a joint effort of several local projects supported by the Ministry of Research and Technology or for example the "DVS NW" a network project of the Ministry for the Interior of North-Rhine Westfalia - hierarchically ordered communication protocols have been defined [7], [8].
Nevertheless the method of interface processors can effectively be used even within such general networks(see for example [9]). This usage mainly covers two aspects:

- as front/end processor in the sense of a network entry processor to support the adaption of the host system to an X.25 network(see figure 5)
- as a terminal concentrator to support the network adaption of terminals which can not run the network access protocols but have terminal internal protocols P_1, P_2, ... (see figure 6).



P_1 = local station protocol
I = Interface
P_n = network access protocol

Figure 5

terminals

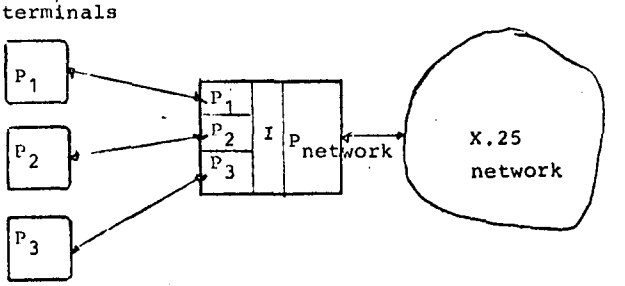


Figure 6

For both cases techniques having been worked out in the described coupling project can be taken over. Especially the problems of a suitable process to process communication and of an effective internal buffer management for communication tasks within the Dietz 621 system have been solved. They will be applied in the network project (X.25, Remote Job Entry, Virtual Terminal) of the University of Düsseldorf within the PIX group according to figures 5 and 6.

For long terms the network adaption and support of higher level protocols have to be provided by the manufacturers within the system software. For that demand a set of standardized communication protocols must exist having been agreed by standardisation bodies(for example ISO).

Main purpose of presently performed network projects within the development and research area (as for example the PIX project) is therefore to define and work out communication protocols which are implementable on computers of different make and which are suitable for standardisation.

References

[1] TIELINE ; Coupling of a CD 6400/CYBER 74 to an IBM 370/155; CDC NOS-BE TIELINE USER'S GUIDE

[2] The ALWR INTERFACE; Interface for communication between heterogenous computers for batch processing; GWDG, Göttingen 1976

[3] E.Raubold: The GMD-Net , Goals and Structure Der GMD-Spiegel 1/76

[4] CCITT ORANGE BOOK, Volume VIII, Public Data Networks, Geneva 1977

[5] DIN NI Subgroup UA16.2:Application oriented communication services

[6] ISO/TC97/SC16/Working Group 2 : Open System Interconnection - Users of Transport Service

[7] The PIX Protocol Definitions, GMD Darmstadt 1978
- The PIX RJE Protocol
- The PIX Virtual Terminal Protocol
- The PIX Message Link Protocol
- X.25 Based Process Process Communication

[8] DVS NW - Data Communication System of the Ministry for the Interior of North-Rhine Westfalia, Düsseldorf 1976

[9] J.Johnson, J.Heap: Designing an X.25 network -to-host interface ; Proceedings of the GI-Workshop, Berlin 1978

DIJALOG RAČUNSKIH SISTEMA SA PERIODIČNIM ODZIVOM

S. ĆERAMILAC
D. GLUŠAC

UDK: 681.327.8

INSTITUT „MIHAILO PUPIN", BEOGRAD

Uobičajena asinhrona komunikaciona procedura je modifikovana uvodjenjem periodičnog potvrdnog i odrečnog odziva, čime je njena postojanost na spoljašnje smetnje povećana.

Izložen je princip ovakvog dijaloga računarskih sistema, uslovi njegove realizacije i jedna praktična implementacija istog.

Standard asynchronous communication is modified by introducing periodically affirmative and negative amswering, improving resistivity on outer noise.

Principe of dialogue, conditions concerning realisation and one practical implementation, are shown.

1. UVOD

Sve češća potreba za povezivanjem udaljenih računarskih sistema u mreže podrazumeva, iz ekonomskih razloga, korišćenje manje kvalitetnih kanala i komunikacione opreme uz nesmanjene zahteve za pouzdanim prenosom. Dva su osnovna pristupa razrešenja ovih oprečnih preduslova, od kojih jedan ide na izbor komunikacione procedure, a drugi na primenu redundantnih kodova, no nezavisno ili u kombinaciji oni rezultuju u manjoj ili većoj redukciji propusne moći kanala. Opredeljujući se za prvi od ova dva pristupa, u poznatu asinhronu komunikacionu proceduru uveli smo periodični odziv prijemnog sistema. Time je propusna moć kanala umanjena za trajanje do najviše jednog odziva, ali je pritom skoro isključena mogućnost gubitka poruke (ili višekratnog prijema iste) usled smetnji.

2. KOMUNIKACIONA OPREMA

Koriste se dva programabilna sistema (u daljem tekstu "strana A" i "strana B"), čiji se stepen složenosti može kretati od inteligentnog terminala do velike računarske mreže. Strana A, kao i B, snabdevena je standardnim serijskim asinhronim medjuspojem i uredjajem za konverziju povorke binarnih signala u fizičku veličinu podobnu komunikacionom medijumu, kao u primeru na Sl. 1. Tip prenosa poruka i odziva je naizmeničan, po jednom kanalu, pri brzinama do 1200 bit/sek. Većim investiranjem u opremu, u cilju podizanja efikasnosti komuniciranja, mogu se primeniti i drugačiji tipovi prenosa i veće brzine.

3. PRINCIP KOMUNICIRANJA

Komunikaciona procedura ostvarena je programski, a kao što pokazuje Sl. 2. rešena je modularno i na strani A i na B, ostavljajući mogućnost uključenja u glavne programe u svojstvu potprograma.

Strane A i B razmenjuju četiri vrste informacija, gde strana A otprema:

"POR" - sa podacima za dalju obradu, i
"KPR" - sa obaveštenjem o završetku razmene podataka.

strana B -

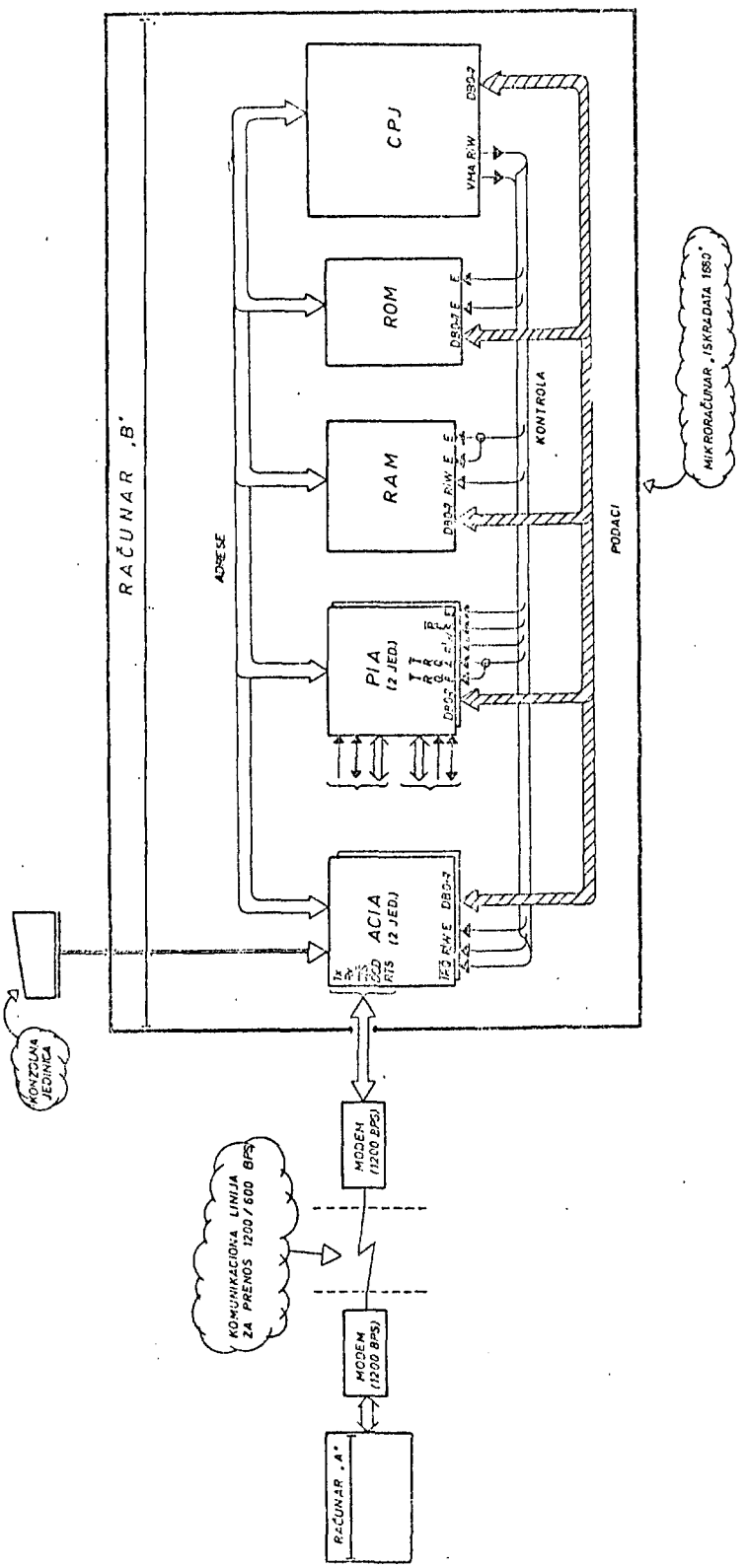
"ACK" - potvrdu ispravnog prijema (POR) i
"CAN" - obaveštenje o neispravnom prijemu (POR).

Proces (vidi Sl. 2) započinje strana A, otpremom POR, posle čega prelazi u stanje čekanja odziva ACK ili CAN od strane B. Istovremeno se na strani A aktivira generator vremena čekanja (Δtča) u kome se, ukoliko je komunikaciona oprema ispravna, taj odziv mora pojaviti. U slučaju odsustva prijave ACK ili CAN od strane B, u ovom intervalu, glavni program na strani A se obaveštava o kvaru veze. Ukoliko strana A primi odziv ACK, glavnom programu se šalje dozvola za momentalnu otpremu sledeće POR.

U slučaju da glavni program na strani A nije pripremio sledeću POR, potvrda ACK se pamti i aktivira svakim završetkom sledećeg prijema bez obzira na njegovu ispravnost. Ovo se ponavlja sve dok glavni program na strani A ne pripremi sledeću POR.

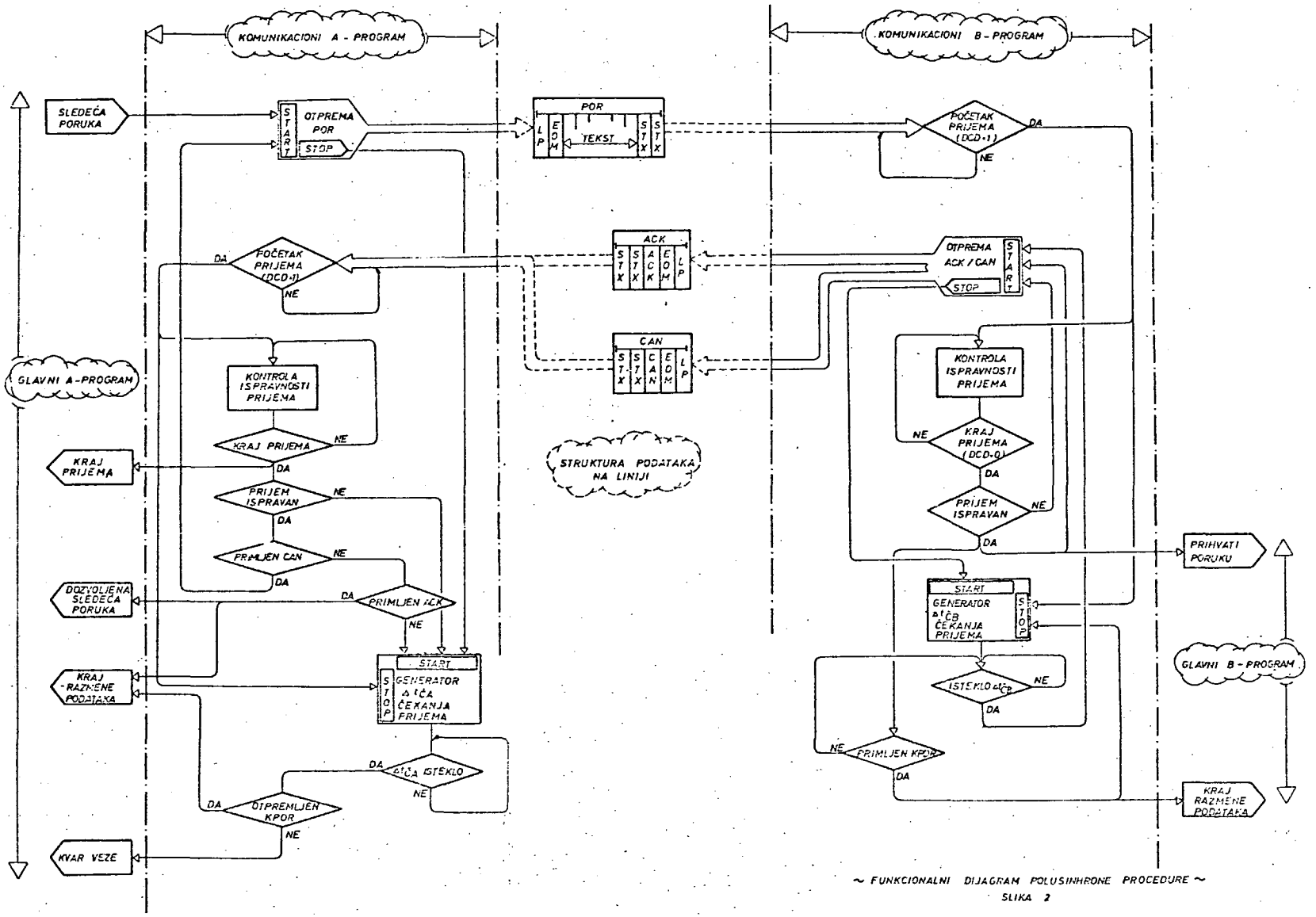
Ukoliko na primljenu POR strana B odgovori sa CAN, a strana A prepozna ovaj odziv, ponovi se otprema iste POR zapamćene u otpremnoj memoriji. Strana A ne reaguje na neispravno primljene odzive sve dok ne primi prvi ispravan odziv od strane B i zapamti ga. Generator vremena čekanja (Δtča) aktivira se krajem svake otpreme, po kriterijumu da je prijem bio neispravan, ili da primljeni odziv nije ni ACK ni CAN. Posle svake otpreme POR, strana A prelazi u stanje očekivanja odziva od strane B. Strana A diktira završetak komuniciranja predajom obaveštenja "KPR", na kojeg strana B ne odgovara nikakvim odzivom u slučaju dobrog prijema. To je znak strani A da je "KPR" ispravno otpremljen, što se konstatuje istekom vremena Δtča u kome se se odziv ne javlja, nemajući ovom prilikom značenje kvara na vezi. U slučaju da strana B ne primi dobro "KPR" ona otprema CAN, na koji se "KPR" ponovi i tako sve do ispravnog prijema istog. Poruka "KPR" pruža mogućnost, da ukoliko strane A i B raspolažu i jednim i drugim komunikacionim programom, mogu ostvariti ravnopravnu razmenu poruka tako, što bi po uspešnoj otpremi "KPR" strane A i B promenile uloge.

Strana B, kao što pokazuje Sl. 2, vrši obradu ispravno primljenih POR u svom glavnom programu, a o ispravnosti ovog prijema odlučuje njen komunikacioni program.



~ MODEL ISPITIVANOG KOMUNIKACIONOG SISTEMA SA DETALJNIJIM PRIKAZOM ORGANIZACIJE PODSISTEMA „B" ~

SLIKA 1



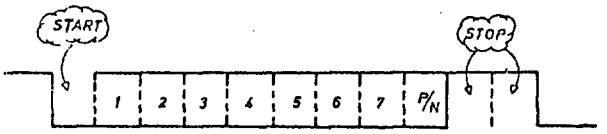
~ FUNKCIONALNI DIJAGRAM POLUSINHRONE PROCEDURE ~

SLIKA 2

Na neispravno primljenu POR (ili KPR) ona šalje odziv CAN, dok u slučaju ispravnog prijema šalje odziv ACK. Svaki od ova dva moguća odziva ponavlja se u pravilnim vremenskim razmacima, koje odredjuje generator vremena čekanja (Δtčb). Interval izmedju dva odziva izabran je tako da omogućuje strani A početak predaje, koji strana B otkriva i briše prethodno memorisani odziv, obrazujući sledeći na osnovu analize novoprimljenih podataka. U ovom intervalu strana B nalazi se u stanju očekivanja POR. Periodično ponavljanje odziva ACK (ili CAN) od strane B, sve dok se završetkom njihovog prijema na strani A ne inicira otprema nove (ili prethodne) POR, daje proceduri obeležje sinhronizma. Ovaj sinhronizam je ipak delimičan i uspostavlja se ili u intervalima pripreme sledeće POR na strani A, ili u slučaju dugotrajnih smetnji koje čine odzive ACK ili CAN nepreponatljivim.

4. FORMATI PORUKA I KODOVI

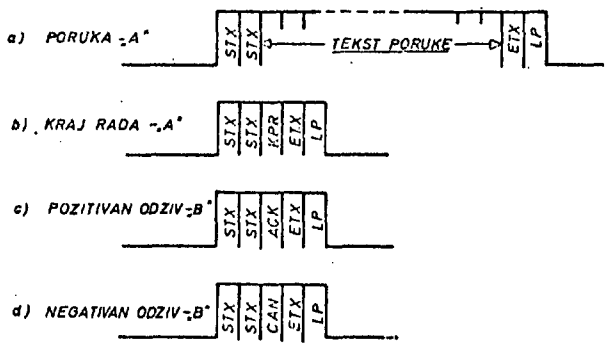
Obzirom, da se komuniciranje obavlja preko standardnih serijskih asinhronih medjuspojeva, sve poruke su segmentirane na slogove koji sadrže 10-11 bita, kao na Sl. 4. U sastav sloga ulaze: startni bit, 7 informacionih bitova, bit transverzalne parnosti, i jedan ili dva stopna bita. Dok sinhrone prenose karakteriše razmena poruka u obliku povorke bitova na fiksnoj dužini, dotle poruke u asinhronom prenosu dopuštaju proizvoljan broj slogova od 10 ili 11 bita (Sl. 3).



~ FORMAT ZNAKA ZA ASINHRONI PRENOS ~

SLIKA 3

U slučaju ovde opisanog prenosa, koji je modifikacija asinhronog, izdvojeni su neki funkcionalni znaci: stx-start poruke, etx-kraj poruke, ack-prihvaćena poruka, can-odbijena poruka: kpr-kraj razmene poruka, a ostalih 2^7 - 5 znakova su informacioni.



~ FORMATIZOVANE INFORMACIJE ~

SLIKA 4

Sve poruke su formatirane tako, da se prvo otpremaju dva znaka stx iza kojih sledi niz znakova korisne poruke, koji se završavaju znakom etx, dok se poslednji otprema znak lp-longitudinalne parnosti, formiran na osnovu sadržaja svih prethodnih znakova po principu sabiranja "MOD2" na pozicijama bitova u slogu. Ovom formatu podležu i odzivi ACK, CAN i KPR, sa jednim funkcionalnim znakom.

Mada kontrole koje smo izabrali tj. transverzalna parnost, longitudinalna parnost i format (koji definiše sinhronizaciju na nivou poruke) najčešće zadovoljavaju zahteve pouzdanog prenosa, u uslovima ekstremno velikih smetnji na linijama može se pribeći strožijim kontrolama uvodjenjem zaštitnih redundantnih kodova. Ovo uvek vodi smanjenju efikasnosti prenosa, umanjenjem udela korisne informacije u poruci.

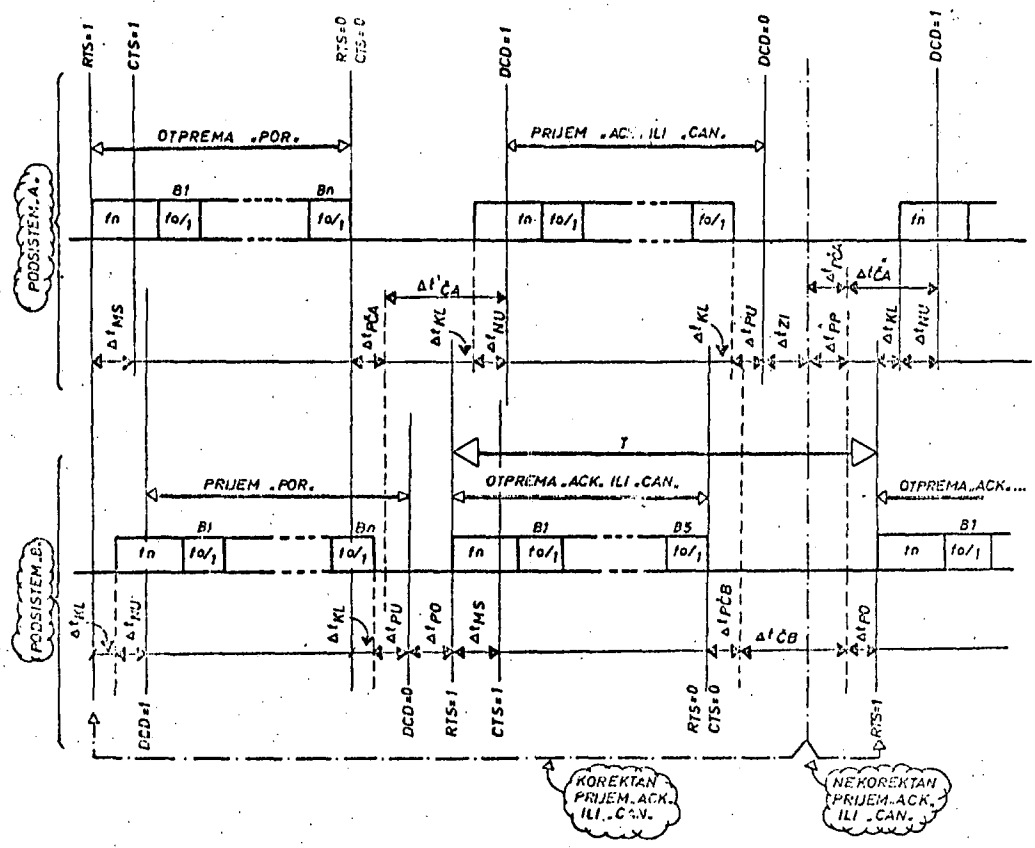
Ove greške izazvane smetnjom, iako malo verovatne, možemo smatrati katastrofalnim po opisanu proceduru komuniciranja. To su konverzija odziva ACK u CAN i obrnuto, gde u prvom slučaju strana B umesto sledeće poruke ponovo prima prethodnu (prosledjujući je u obradu), a u drugom gubi poruku, koju nije prihvatila kao ispravnu, primajući sledeću. Svako pojačavanje kontrole zaštitnim kodovima, ukoliko se isti primenjuju i na odzive, smanjuje verovatnoću ova dva nepovoljna dogadjaja.U našem slučaju može se medju raspoloživim 7-bitnim znacima izabrati par (ack, can) tako da oba (sa svojim lp) imaju maksimalnu moguću Hamming-ovu distancu, čime se ova opasnost obično drastično umanjuje bez narušavanja efikasnosti prenosa.

5. USLOVI KOMUNICIRANJA SA PERIODIČNIM ODZIVOM

Vremenski dijagram na Sl. 5 prikazuje tok razmene informacija izmedju A i B. Na njemu su označeni ključni signali RTS, CTS i DCD koje podsistemi razmenjuju sa modemima, frekventno kodovane informacije na TT-liniji (fn-noseća i f 1/0 - odgovarajuće logičkom "1" ili "0"), te vremenski intervali vezani za fizičke osobine sistema u prijemu i otpremi. Značenje ovih intervala je sledeće:

- Δt_{ms} - kašnjenje izmedju zahteva modemu za otpremom (RTS=1) i njegove spremnosti (CTS=1)
- Δt_{kl} - kašnjenje informacije s kraja na kraj linije
- Δt_{nu} - kašnjenje izmedju prispeća noseće učestanosti na prijemni kraj linije i otkrivanja iste (DCD=1)
- $\Delta t_{pča}$ - trajanje programskog prelaska u fazu čekanja prijema na strani A
- $\Delta t'_{ča}$ - trajanje programskog čekanja (na strani A) odziva od strane B na otpremljenu POR
- Δt_{po} - trajanje programske pripreme odziva ACK ili CAN na strani B
- $\Delta t_{pčb}$ - trajanje programskog prelaska u fazu čekanja prijema na strani B
- $\Delta t_{čb}$ - trajanje programskog čekanja prijema POR na strani B, koju može otpremiti stana A
- Δt_{zi} - trajanje završnih ispitivanja statusa prijema na strani A koji definišu njenu reakciju kao:
  - (a) neispravan prijem ili nespremnost nove POR → prelazak u fazu čekanja ponovnog odziva iz B
  - (b) ispravan prijem CAN → ponovna otprema prethodne POR
  - (c) ispravan prijem ACK → otprema naredne POR (ako je ova prispela u otpremni registar A)
- $\Delta t''_{ča}$ - trajanje programskog čekanja na strani A, po osnovi slučaja opisanog prethodno pod (a)
- Δt_{pp} - interval koji obezbedjuje detekciju prijema POR na strani B, ukoliko strana A reaguje na način prethodno opisane pod (b) i (c).

Uočava se da intervali: Δtčb, Δt'ča, Δt''ča, i Δtpp (podvučeni u opisu) predstavljaju zavisno promenljive veličine, a obzirom da su svi ostali neposredno vezani za fizičke osobine sistema (ili trajanje izvršenja programa), oni će se u prethodnima javiti kao nezavisno promenljive.



~ VREMENSKI DIJAGRAM POLUSINHRONOG KOMUNICIRANJA ~
SLIKA 5

Interval Δtpp, prema svom opisu, mora ispuniti sledeći uslov:

$$\Delta t_{ppmin} = \Delta t_{kl} + \Delta t_{nu} \qquad (1)$$

Koristeći se ovim uslovom, slikom 5 i mogućnošću unifikacije vremena Δt'ča i Δt"ča, svodjenjem na jedinstveno Δtča po principu većeg (Δtča ≈ Δt"ča), dolazimo do uslova koje moraju ispuniti programski generatori vremena čekanja prijema na stranama A i B, radi izbegavanja "trka" na liniji:

$$\Delta t_{ča} \geq 2\ (\Delta t_{kl} + \Delta t_{nu}) + (\Delta t_{pp} - \Delta t_{pča}) \qquad (2)$$

$$\Delta t_{čb} \geq 2\ \Delta t_{kl} + \Delta t_{nu} + \Delta t_{pu} + (\Delta t_{zi} - \Delta t_{pčb}) \qquad (3)$$

Članove (podvučene) u izrazima 2 i 3 odredjujemo prema srednjem vremenu i broju izvršenja pripadnih instrukcija u komunikacionim programima na stranama A i B.

6. OPIS PROCESNIH DIJAGRAMA SISTEMA "B"

U želji da se izbegne suvišna opširnost, a imajući u vidu sličnost ključnih funkcija komunikacionih algoritama na stranama A i B, izloženo je rešenje procesnih dijagrama samo na strani B koja generiše periodični odziv.

Ceo tok procesa komuniciranja sistema "B" dat je u dva procesna dijagrama: rutina "prijem poruke" Sl. 6 i rutina "otprema odziva" - Sl. 7.

Realizovana rutina "prijem poruke" sadrži sledeće funkcije i kontrole:

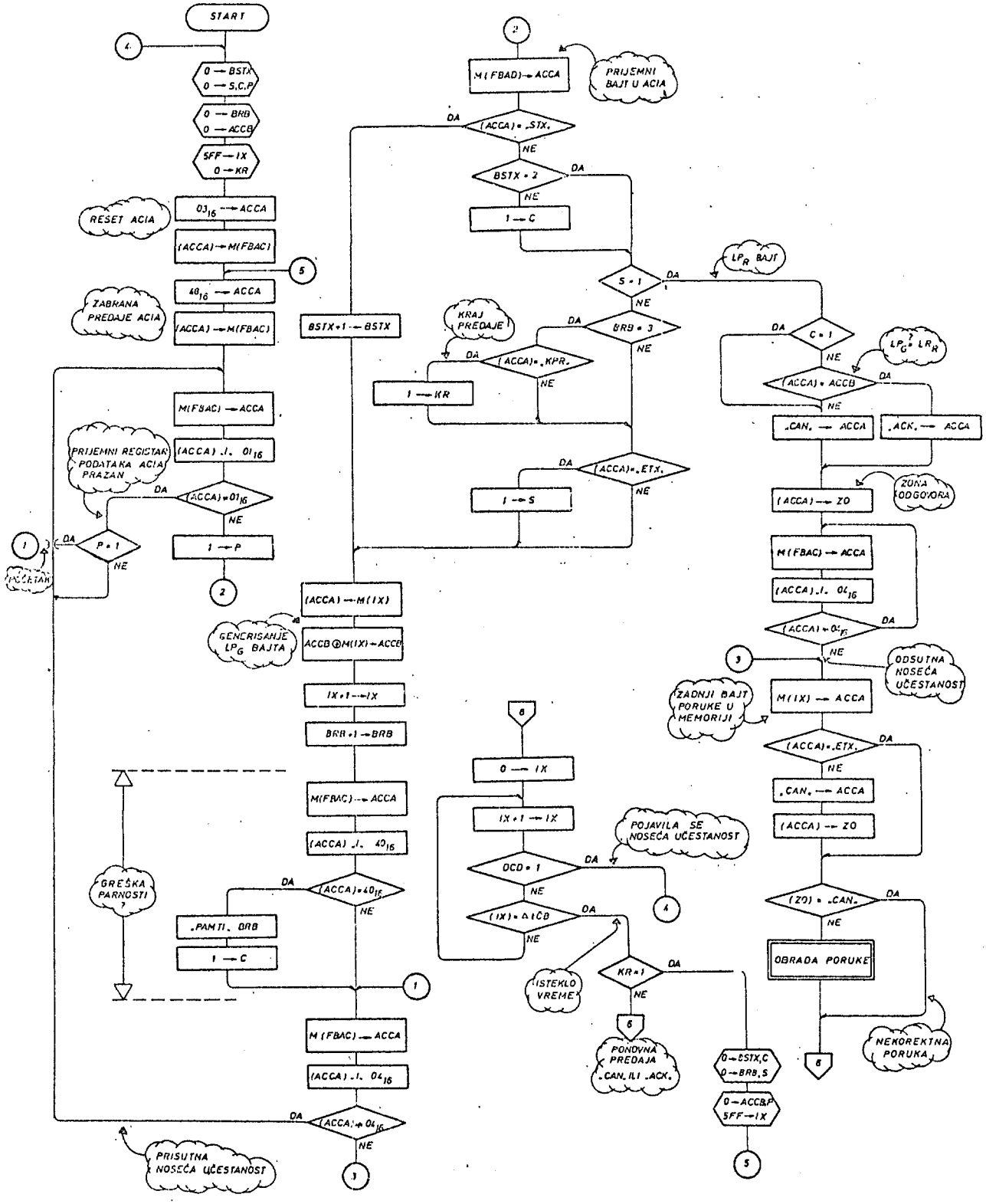
- funkcija cikliranja i čekanja završetka prijema jednog bajta, oformljenog iz povorke bitova,
- generisanje LP bajta (longitudinalne parnosti),
- formiranje poruke u RAM memoriji,
- ispitivanje kvaliteta primljene poruke,
- razlučivanje korektna/nekorektna poruka,
- formiranje pozitivnog/negativnog odziva,
- kontrola parnosti primljenog bajta i poruke,
- kontrola zaglavlja poruke,
- detektovanje nestanka noseće učestanosti.

Rutina "otpreme odziva" sastoji se od:

- ispitivanje "spremnosti za predaju",
- odredjivanje režima "predaje" u ACIA (serijski asinhroni medjuspoj),
- subrutina ispitivanja "predajni registar ACIA prazan",
- predaja odziva "ACK/CAN".

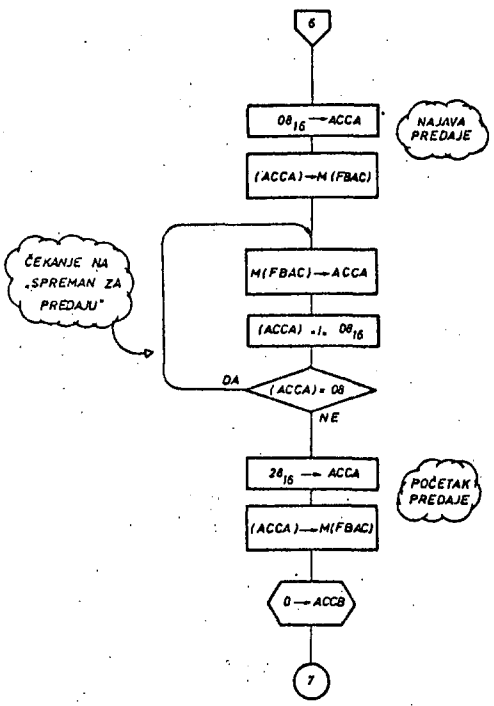
7. ZAKLJUČAK

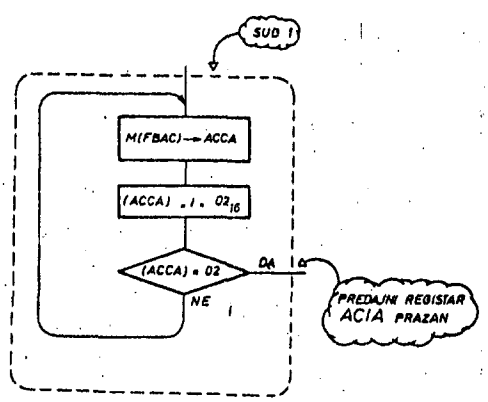
Opisana procedura sa periodičnim odzivom namenjena je automatskoj pouzdanoj daljinskoj razmeni podataka preko komunikacionih medijuma izloženih visokom nivou smetnji. Neposredno je primenjiva na sve programabilne sisteme sa standardnim serijskim asinhronim medjuspojem, kakav je najčešće u upotrebi, dopuštajući i druge mogućnosti kao na primer: paralelni medjuspoj, simultana razmena podataka preko četvorožične veze uz korišćenje povratnih kanala za odzivni signal i sl. Procedura je proverena funkcionalno na paru domaćih računara "Iskradata 1680" i "HRS-100" i paru domaćih modema "PP1200" u jednosmernom prenosu podataka (i periodičnih odziva u suprotnom smeru) brzinama 600 i 1200 bit/sek preko improvizovane TT-parice.

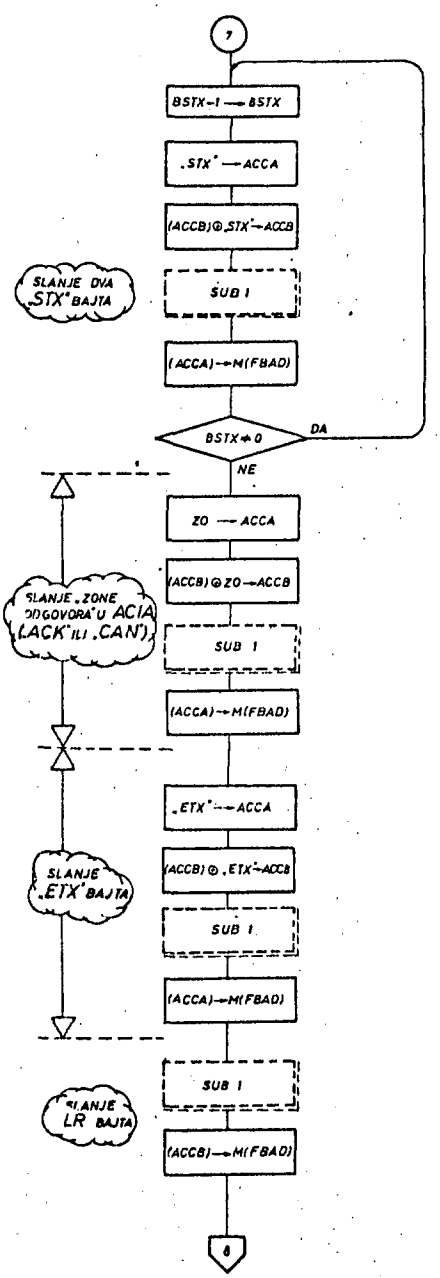


~ RUTINA „PRIJEM PORUKE" ~

SLIKA 6







~ RUTINA „ODPREMA ODZIVA" ~
SLIKA 7

8. NAPOMENA AUTORA

Koristimo priliku da se zahvalimo Dr DEJANU ŽIVKOVIĆU, dipl.ing. na recenziji ovog rada i korisnim savetima i sugestijama koji su doprineli njegovoj realizaciji.

9. LITERATURA

/1/ Wesley W. Chu "Advances in Computer Communications" Artech Hause 1974.

/2/ Saul Stimler "Real Time Data-Processing Systems" Mc Graw - Hill Book Company.

D o d a t a k

10. OPIS RADA PODSISTEMA "B"

Uloga podsistema "B" u procesu komuniciranja objašnjena u prethodnim poglavljima dobija detaljniju implementaciju u formi procesnih dijagrama sa Sl. 6. Na slici je opisana rutina "Prijem poruke" a na Sl. 7. rutina "Otprema odziva" a obe su smeštene u memoriji mikroprocesora "ISKRADATA 1680".

10.1. Opis rutine "Prijem poruke"

Prijem poruke počinjemo u početku rada sa: "Startom", "0" u BSTX; "0" u S, C, P, "0" u BRB, "0" u ACCB, "0" u KR i 5FF u IX. Sledi reset ACIA (asinhroni komunikacioni kontroler u "ISKRADATA 1680") sa: 03(hex) u ACCA, (ACCA) u M(FBAC). Pod FBAC se podrazumeva adresa

ACIA u mikroprocesoru (i to adresa kontrolnog/status registra) i ako upisujemo neki sadržaj u FBAC tada ga upisujemo u kontrolni registar, ako čitamo FBAC tada se čita status registar. Potom sledi "zabrana predaje ACIA" koja se realizuje sa: 48(hex) u ACCA, (ACCA) u M(FBAC). U sledećoj rutini ispitujemo "biti" status registra ACIA koji kad je "1" znači "prijemni registar podataka ACIA *pun*", *odnosno znači da je ACIA primila* niz bitova u rednom obliku i formirala jedan bajt koji se može preuzeti u memoriju (ovo implicitno znači, da se na liniji pojavila i noseća učestanost). Ova rutina je sledeća:M(FBAC) u ACCA, (ACCA) "I" 01(hex) da li je (ACCA) ≠ o1(hex) (da, dok se ne pojavi noseća učestanost i formira bajt), da li je P=1 (ne, dok se ne primi prvi bajt poruke), ponovo M(FBAC) u ACCA. Kad se primi prvi bajt tada je: (ACCA) = 01(hex) i ide dalje "1" u P, konektor 2, M(FBAD) u ACCA (prijemni registar podataka ACIA). U daljem niz sledi: da li (ACCA)="STX" (da, za prva dva bajta poruke koja su obavezno "STX"), BSTX+1 u BSTX (brojač "STX" - bajtova),(ACCA) u M(IX), ACCB "ekskluzivno ili" M(IX) u ACCB (na ovaj način se za svaki primljeni bajt formira kontrolni bajt - "bajt longitudinalne parnosti"), IX+1 u IX, BRB+1 u BRB.Sledi "kontrola parnosti bajta" (tzv. "vertikalna parnost") i to: M(FBAC) u ACCA, (ACCA) "I" 40(hex), da li je (ACCA) = 40(hex) (ako jeste, znači "bit 8" u status registru ACIA se nalazi u stanju "1" tj. ACIA je otkrila grešku parnosti u primljenom bajtu), "pamti" BRB (ovde je naglašena samo manja rutina koja memoriše u posebnim zonama memorije stanja (broj) BRB - brojačabajtova na kojima je nadjena greška), 1 u C (kriterijum C znači da je rezultat analize ispravnosti prijema poruke negativan i da će se poslati odziv sa "CAN"). Dalje, prolazimo petlju "ispitivanje prisustva noseće učestanosti" i to: konektor 1, M(FBAC) u ACCA, (ACCA) "I" 04(hex), da li je (ACCA) = 04(hex) (da, u ovom trenutku kad primamo ispravno prvi bajt poruke, a "bit 4" status registra ACIA u stanju "0" znači, da je prisutna noseća učestanost - signal DCD= =0), ponovo M(FBAC) u ACCA.

Dok se ne formira sledeći bajt kružimo u petlji:M(FBAC) u ACCA, (ACCA) "I" 01(hex), (ACCA) ≠ 01(hex), da li je P=1 (da, prvi bajt je već primljen), konektor 1,M(FBAC) u ACCA, (ACCA) "I" 04(hex), (ACCA) ≠ 04(hex), ponovo M(FBAC) u ACCA. Ako se tokom prijema poruke u bilo kome trenutku izgubi noseća učestanost, tada po uslovu: (ACCA) = 04(hex), konektor 3, M(IX) u ACCA, da li je: (ACCA) = "STX" (ne, u slučaju akcidentnog nestanka noseće učestanosti), "CAN" u ACCA (negativan odgovor analize), (ACCA) u ZO. Ovo je važna kontrola prijema poruke.

Drugi bajt poruke (inače drugi "STX") ide već opisanom petljom i brojač BSTX se napreduje na vrednost 2. Treći bajt poruke više nije funkcionalni bajt i idemo novim putem: M(FBAD) u ACCA, da li je (ACCA) = "STX" (NE), da li je BSTX = 2 (da, ako nije onda je greška u zaglavlju poruke i imamo: 1 u C), da li je S = 1 (NE), da li je BRB = 3 (DA), da li je (ACCA) = "KPR" (ne, jednakost se javlja samo na kraju predaje niza poruka kad se od strane "A" pošalje funkcionalna poruka "KRAJ PREDAJE" i tada ide: 1 u KR), da li je (ACCA)= = "ETX" (ne, kad je ispunjena jednakost onda je kraj jedne poruke i ide 1 u S), (ACCA) u M(IX) itd.

Po završnom prijemu poruke zaključno sa "ETX" idemo sledećim putem (sad se primi zadnji - LP_r bajt): da li je S=1 (DA), da li je C=1 (da, ako su otkrivene greške u prijemu, i tada ide: "CAN" u ACCA), da li je (ACCA) = (ACCB) (da li su LP_g - generisani i LP_r - primljeni bajt longitudinalne parnosti isti) ako jeste ide "ACK" u ACCA ako nije "CAN" u ACCA, (ACCA) u ZO (zona odgovora). Dalje, sledi rutina ispitivanja trenutka iščezavanja noseće učestanosti: M(FBAC) u ACCA,(ACCA) "I" 04(hex), da li je (ACCA) ≠ 04(hex) (da, dok je prisutna noseća učestanost), ponovo M(FBAC) u ACCA. *Kad nestane noseća učestanost završen je prijem poruke* i sledi obrada poruke i predaja odziva. Ako je poruka korektno primljena zadnji bajt u memoriji mora biti "ETX" što se ispituje na sledeći način: M(IX) u ACCA, da li (ACCA) = "ETX" (da, ako je poruka ispravno primljena) ako nije nadjena jednakost: "CAN" u ACCA, (ACCA) u ZO.

Dalje, sledi preispitivanje da li da se pristupi obradi poruke: da li je (ZO) = "CAN", ako nije vrši se obrada poruke (ovo je veća programska celina, ovde samo naglašena), ako jeste imamo kriterijum "NEKOREKTNA PORUKA", i preko medjustraničnog konektora 6 idemo na procesni dijagram "otprema odziva". Po završenoj predaji odziva preko konektora 8 dolazimo na kolo za vremensko kašnjenje: 0 u IX, IX+1 u IX, da li je DCD=1 (NE), da li je IX = Δtčb, ponovo IX+1 u IX. U ovoj petlji se kruži dok se ne ispune uslovi:

(a) DCD=1 što znači da se pojavila noseća učestanost i počinje prijem nove poruke (prethodna je završena sa "ACK") ili prijem iste poruke (prethodna je završena sa "CAN").

(b) isteklo je vreme Δtčb i tada: da li je KR=1 (NE) konektor 6 i ponovo se šalje zadnje predati odziv.

Ako je podsistem "A" poslao poruku "KRAJ RADA" tada je: KR=1 i imamo: "0" u BSTX, C, BRB, S, ACCB,P i 5FF u IX, konektor 5,48(hex) u ACCA, (ACCA) u M(FBAC), potom petlja: M(FBAC) u ACCA, (ACCA) "I" 01(hex), (ACCA) ≠ 01(hex), P=1 (NE), ponovo M(FBAC) u ACCA. U datoj petlji kružimo do ponovne pojave kriterijuma "prijemni registar podataka ACIA PUN".

10.2. Opis rutine "otprema odziva"

Preko konektora 6 idemo na: 08(hex) u ACCA, (ACCA) u M(FBAC) što predstavlja najavu predaje), M(FBAC) u ACCA, (ACCA) "I" 08(hex), da li je (ACCA) = 08(hex) (da, za vreme čekanja "spremnosti za predaju"), 28(hex) u ACCA, (ACCA) u M(FBAC), "0" u ACCB, konektor 7.

Sledi "otprema odziva" koja započinje slanjem dva "STX" bajta: BSTX-1 u BSTX,"STX" u ACCA, (ACCB) "EKSKLUZIVNO ILI" "STX" u ACCB (formira se LP bajt u predaji),(ACCA) u M(FBAD), da li je BSTX≠0 (da, za prva dva "STX" bajta).

Sledi predaja ZO kriterijuma (Zona Odgovora): ZO u ACCA, (ACCB) "EKSKLUZIVNO ILI" ZO u ACCB, SUB1, (ACCA) u M(FBAD).

Subrutina SUB1 ispituje kad je predajni registar ACIA postao prazan tj. kada se predaje sledeći bajt: M(FBAC) u ACCA, ACCA "I" 02(hex), da li je (ACCA) = 02(hex), ponovo M(FBAC) u ACCA.

Dalje se šalje "ETX" bajt: "ETX" u ACCA, (ACCB) "EKSKLUZIVNO ILI" "ETX" u ACCB, SUB1, (ACCA) u M(FBAD). *Na kraju se otprema LP bajt*: SUB1, (ACCB) u M(FBAD), konektor 8.

PREČNI PREVAJALNIK ZA PASCAL

B. BARLIČ

UDK: 519.685

KEMIJSKI INŠTITUT „BORIS KIDRIČ", LJUBLJANA

V delu je opisana predelava standardnega prevajalnika za Pascal P v prečni prevajalnik za računalnike PDP-11. Predelava je zasnovana na tem, da se generiranje ukazov za hipotetični računalnik v orginalnem prevajalniku (stack computer) spremeni v generiranje ustreznih zaporedij ali posameznih strojnih ukazov za računalnik PDP-11. Prevajalnik je v celoti napisan v Pascalu.

CROSS COMPILER FOR PASCAL. A conversion of the standard Pascal P compiler into a cross compiler for PDP-11 computers is described. The conversion was acomplished by changing the stack computer instructions generation routines to generate corresponding instructions or sequences of instructions for PDP-11 computer. The compiler is coded completely in Pascal.

UVOD

Zaradi potreb po učinkovitejšem razvijanju vedno večje uporabniške programske opreme nastajajo in se bolj ali manj uspešno uveljavljajo novi programski jeziki. Za uspešno uporabo določenega programskega jezika morajo biti na voljo ustrezni prevajalniki na čim več računalnikih.

Pascal je programski jezik, ki se v novejšem času precej hitro uveljavlja. Da bi Pascal prilagodili različnim uporabnikom in različno velikim računalnikom je nastalo več poenostavljenih ali razširjenih različic Pascala. Najbolj znani primeri so Pascal S (2), Pascal P (3) in Concurrent Pascal (4,4a).

Ker je razvoj prevajalnika še vedno obsežna naloga, se poskuša to delo poenostaviti z razvojem delno prenosljivih prevajalnikov. S tem namenom je bil razvit tudi prevajalnik Pascal P (3), ki smo ga uporabili kot osnovo za naš prevajalnik. Prvotni prevajalnik generira ukaze za nek hipotetični računalnik (Stack Computer (3)). Za prenos prevajalnika na nek drug računalnik je več poti, opisanih v literaturi (3,5).

V nadaljevanju je opisana priprava prečnega prevajalnika, ki je bil narejen na Kemijskem inštitutu "Boris Kidrič". Razvoj prečnega prevajalnika je prva faza razvoja prevajalnika za nek računalnik, če pri tem uporabljamo že obstoječ prevajalnik na kakem drugem računalniku. Ker je opisani prevajalnik v celoti napisan v Pascalu, je prenos prevajalnika dokaj preprost, seveda če oprema in velikost računalnika to dopuščata.

PREVAJALNIK

Razvoj prevajalnika obsega predvsem dopolnitve za generiranje strojnih ukazov za računalnik PDP-11, pri čemer so potrebne še nekatere dodatne spremembe in dopolnitve.

Generiranje strojnih ukazov

Prevajalnik Pascal-P generira ukaze za nek izmišljen računalnik (imenovan stack computer), ki je po naboru ukazov (P-code) precej prilagojen Pascalu. Pri opisanem prevajalniku smo spremenili generiranje ukazov tako, da se ločeno za vsak ukaz, ki ga generira prevajalnik Pascal-P, generira en ali več strojnih ukazov za računalnik PDP-11, ki izvršijo isto funkcijo. Strojne ukaze prevajalnik zapisuje na posebno datoteko (LGO) v obliki tabel, kot jih zahteva povezovalnih programov v operacijskem sistemu RSX-11M (6).

Prevajalnik, ki ga dobimo na ta način generira preveden program neposredno brez vmesnih stopenj (druge možnosti glej (5,3)). Predelava ni preobsežna, saj je treba dopolniti in razširiti le nekaj modulov, ki generirajo strojne ukaze, večine modulov v prevajalniku pa ni potrebno spreminjati. Pomankljivost izdelanega prevajalnika pa je v tem, da preveden program ni optimiziran, saj generiranje strojnih ukazov ni prilagojena možnostim, ki jih nudi nabor strojnih ukazov na računalniku PDP-11.

Seznam ukazov P-kode u ustrenimi ukazi računalnika PDP-11, kot jih generira prevajalnik je v dodatku.

Dopolnitve in spremembe prevajalnika

Pri razvoju prečnega prevajalnika je bilo potrebno prvotni prevajalnik dopolniti in spremeniti na nekaterih mestih. Najobsežnejša dopolnitev je dodatek modulov za disasembliranje prevedenega programa. Kljub temu, da je disasembliranje zelo poenostavljeno, je ta dopolnitev pri testiranju prevajalnika zelo koristna (7). Konstant, ki so v prevedenem programu med ukazi ne prepozna, poleg tega zaradi dokaj kompliciranе oblike tabel z generiranim prevedenim programom (6) ne najde naslovo za lokalne skoke znotraj modulov.

Kljub naštetim pomanjkljivostim in poenostavitvam se je dopolnitev v praksi pokazala kot koristna. Prevajalnik smo najprej testirali na majhnih, le nekaj vrstic dolgih programih. Pri tem se pokaže velika večina napak in skoraj vse smo našli s pomočjo disasembliranega programa. Ko je prvo testiranje končano in se začne prevajanje večjih programov, postane disasembliranje brez pomena in zaradi hudih poenostavitev tudi nezanesljivo. Zato se v takih primerih disasembliranje prevedenega programa izključi, kar se v opisanem prevajalniku doseže s posebnim komentarjem (8) v obliki (+$A-+); na začetku valja A+.

Prvotni prevajalnik in Pascal P pozna štiri datoteke: INPUT, OUTPUT, PRD in PRR. Prečni prevajalnik potrebuje le tri datoteke: INPUT, OUTPUT in LGO; datoteki INPUT in OUTPUT sta dekla-

rirani kot "FILE OF ALPHA", dototeka LGO, kamor se zapiše preveden program, pa je deklariran kot "SEGMENTED FILE OF INTEGER".

Pomembni sta še dve spremembi, ki sta potrebni zaradi različnega obsega celih števil (velikost besede) in različnega kodiranja znakov na obeh računalnikih. Prevajalnik generira kode strojnih ukazov kot celoštevilčne vrednosti in pri tem potrebuje vseh 16 bitov v besedi. Pri celoštevilčni aritmetiki na računalniku PDP-11 pomeni zgornji bit (bit 15) v besedi predznak števila. Na računalniku Cyber 72 je obseg celih števil mnogo večji, zato pri prečnem prevajalniku načelno to ne predstavlja problema. Ker smo želeli prevajalnik pripraviti tako, da bi bilo pri nadaljevanju dela s prevajalnikom (tj. prenos prevajalnika v celoti na računalnik PDP-11) čim manj dela, smo generiranje prevedenega programa pripravili tako, kot da dela prevajalnik na računalniku s 16-bitno besedo. Seveda je potrebno obliko, v kateri je zapisan preveden program, upoštevati pri prenosu prevedenih programov na računalnik PDP-11.

Problem različnega kodiranja znakov na obeh računalnikih se da rešiti s preprosto tabelo za pretvarjanje znakov iz ene kode v drugo. Pretvarjanje znakov je potrebno le tedaj, ko prevajalnik v prevedenem programu generira znak ali niz znakov kot kontanto. Prenos imen modulov je rešen na drug način, ker morajo biti imena modulov v prevedenem programu kodirana v posebni obliki (Radix 50 format) (6).

PRENOS PREVEDENIH PROGRAMOV

Ker prevajalnik teče na računalniku Cyber 72, prevedeni programi pa na PDP-11, je potrebno prevedene programe prenašati iz računalnika Cyber 72 na PDP-11. Za to uporabljamo dva načina: prenos po komunikacijski zvezi med računalnikoma in prenos na magnetnem traku.

Pri prvem načinu je računalnik PDP-11 priključen kot terminal na Cyber 72. Ker obstoječa komunikacijska programska oprema dopušča le prenos tekstov (kodiranih datotek), je treba preveden program kodirati in ga nato prenesti po komunikacijski zvezi. Na računalniku PDP-11 je treba datoteko dekodirati in jo zapisati v obliki, kot jo zahteva povezovalnik (6). Za kodiranje in dekodiranje smo napisali dva majhna neodvisna programa. Ta način je primeren zlasti za prenašanje manjših programov, ker je počasen zaradi majhne hitrosti prenosa podatkov po komunikacijski zvezi.

Pri drugem načinu prevedeni program zapišemo na magnetni trak. Na računalniku PDP-11 trak prečitamo in prevedeni program zapišemo v zahtevani obliki (6). Tudi za pisanje in črtanje traku smo napisali dva kratka neodvisna programa. Ta način prenosa ima v primerjavi s prejšnjimi to slabost, da je potrebno trak fizično prenesti od enega računalnika k drugemu, vendar je to pri prenosu večjih programov najlažje izvedljivo.

KNJIŽNICA PODPROGRAMOV

Vse komplicirane funkcije, ki jih programski jezik zahteva, so sprogramirane kot podprogrami, ki so shranjeni v posebni knjižici. Prevajalniku takih funkcij torej ni treba v celoti prevajati, temveč generira le strojni ukaz za klic ustreznega podprograma. Prevajalnik je zaradi tega precej preprostejši in hitrejši, pa tudi prevedeni programi so krajši. Podprogrami v knjižnici so napisani v zbirnem jeziku računalnika, na katerem tečejo prevedeni programi, v našem primeru je t MACRO-11(9).

Standardne funkcije

Prva skupina podprogramov v knjižnici so pascalove standardne funkcije. V fazi testiranja smo nekatere standardne funkcije zelo poenostavili. Vhodno-izhodne operacije so omejene na preprosto čitanje in pisanje na terminalu in delajo le za datoteki INPUT in OUTPUT. Aritmetične funkcije z realnimi števili (kot so transcendentne funkcije) so izvedene s pomočjo ustreznih fortranskih podprogramov.

Klice standardnih funkcij prevajalnik prevaja tako, da v sklad naloži vse potrebne parametre in pokliče ustrezni podprogram, nato šele pobere rezultate, ki jih mora podprogram pustiti v skladu. Kateri podatki so za posamezne funkcije v skladu in katere rezultate prevajalnik od njih pričakuje, je opisano v literaturi (3), npr. funkciji EOLN in EOF dobita v skladu naslov, kjer so shranjeni parametri datoteke, v skladu pa pustita logični rezultat (true ali false).

KOMPLICIRANI UKAZI

Kompliciranih ukazov prevajalnik ne prevaja direktno, temveč generira ukaze za klice ustreznih podprogramov. Podatke, ki jih podprogrami potrebujejo, spravi v sklad ali pa jih pusti kot konstante med ukazi prevedenega programa, včasih jih prenese tudi v registrih. Prevajalnik ali podprogram poskrbita za to, da takšnih konstant računalnik ne poskuša izvajati kot ukaze. Kateri od ukazov se prevedejo v klice podprogramov je razvidno iz tabele v dodatku ; to so predvsem ukazi za operacije z množicami in nizi (indeksiranimi spremenljivkami, zapisi in nizi znakov).

Prevajalnik za Pascal, ki teče na računalniku Cyber 72, shranjuje vsako množico v eno samo besedo (8), ker so besede na tem računalniku dovolj velike (60 bitov). Besede na računalniku PDP-11 so za tak način shranjevanja veliko premajhne (16 bitov), saj Pascal zahteva možnost uporabe črk in številk (1,8) (36 elementov). Poleg tega so ASCII kode črk med 65 za črko A in 90 za črko Z. Zato smo pri našem prevajalniku predvideli za shranjevanje množice 8 besed (128 bitov), kar zadostuje tudi za množico malih črk (ASCII kode med 97 in 122) in celo za množice vseh znakov, česar Pascal sicer ne zahteva.

OSTALI PODPROGRAMI

Poleg doslej opisanih podprogramov je v knjižnici še več pomožnih podprogramov, ki ostalim podprogramom opravljajo določene pomožne skupne funkcije in podprogrami za vzdrževanje sklada.

Vse podatke prevajalnik shranjuje v skladu in tudi za shranjevanje vmesnih rezultatov pri izvajanju operacij uporablja sklad.

Ko se začne izvajanje kakega modula, ta najprej dobi v skladu blok za shranjevanje lokalnih spremenljivk. V bloku je rezerviran prostor za prenos rezultata funkcije (potreben pri funkcijskih podprogramih), prostor za podatke o modulu, ki mu blok pripada, prostor za kazalce na sosednja bloka in prostor za lokalne in začasne spremenljivke, ki jih modul potrebuje. Prostor na vrhu sklada se uporablja za shranjevanje vmesnih rezultatov pri izvajanju operacij.

Sklad vzdržujejo sledeči štirje podprogrami:

INPR.. je podprogram, ki (poleg ostalih funkcij na začetku izvajanja programa), zgradi začetek sklada. Takoj na začetku glavnega programa prevajalnik generira klic tega podprograma. INPR., najprej zahteva od operacijskega sistema dodaten prostor v spominu, zgradi začetek sklada in blok glavnega programa ter postavi začetne vrednosti potrebnih parametrov za začetek izvajanja vhodno-izhodnih operacij. Ves prostor, ki je potreben za izvajanje vhodno-izhodnih operacij je rezerviran v bloku glavnega programa.

INIT.. je podprogram, ki v sklad vstavi blok za

nek modul. Prevajalnik generira klic tega podprograma na začetku vsakega modula, razen glavnega programa. Ta podprogram v skladu zgradi nov blok in v ustrezni prostor spravi kazalce, ki povezujejo sosednje bloke med seboj.

CLØS.. je podprogram, ki iz sklada umakne zadnji blok in vzpostavi stanje, kakršno je bilo pred klicem podprograma INIT.., pri tem seveda potrebuje kazalce, ki povezujejo med seboj sosednje bloke. Klic tega podprograma generira prevajalnik na koncu vsakega modula, razen glavnega programa.

.LØCA. je podprogram, ki v prevedenem modulu omogoča dostop do globalnih spremenljivk. Ta podprogram potrebuje kot vhodni podatek razliko nivojev do modula, v katerem je globalna spremenljivka deklarirana. Kot rezultat vrne naslov začetka bloka v skladu, ki pripada temu modulu. S pomočjo tega naslova in lokalnega naslova spremenljivke v tem bloku prevajalnik lahko generira ukaze za operacije z globalno spremenljivko.

ZAKLJUČEK

Prečni prevajalniki so koristni tedaj, ko omogočajo razvoj programov v višjem jeziku za računalnike, za katere ni na voljo ustrezne razvojne programske opreme. To je pomembno zlasti pri uporabi mikroračunalnikov, ki imajo premajhno kapaciteto za uporabo obsežnejše programske opreme in so namenjeni predvsem specializiranim nalogam.

Razvoj prečnega prevajalnika je lahko tudi prva faza prenosa kakega prevajalnika v celoti na nek računalnik. To pride v poštev tudi v našem primeru, saj so računalniki PDP-11 dovolj zmogljivi tudi za uporabo nekoliko obsežnejše programske opreme. Zato seveda prečni prevajalnik ne sme biti preveč okrnjen. To je bil glavni razlog, zakaj smo v našem prevajalniku predvideli uporabo množic do 128 elementov. S tem so izpolnjeni vsi pogoji za prevajanje prevajalnika samega, ki je v celoti napisan v Pascalu. Prečni prevajalnik je seveda uporaben tudi za prenos kateregakoli prevajalnika, ki je napisan v Pascalu, npr. prevajalnik za Pascal-S (2).

Razvoj lastnega prevajalnika je sicer dokaj zahteven, vendar pa ima tudi precej prednosti, ki se pokažejo pri vzdrževanju in izpopolnjevanju prevajalnika v skladu s potrebami. Pri kupljenih prevajalnikih največkrat ne dobimo izvornega programa, kar pomeni, da prevajalnik lahko dopolnjuje le proizvajalec. Kot kaže opisani primer, pa samostojni razvoj ni predrag in je izvedljiv tudi v naših razmerah.

Ocenjujemo, da smo za pripravo tega prevajalnika potrebovali 4 do 5 mesecev dela enega programerja. V primerjavi z razvojem kakega prevajalnika v celoti to ni veliko, kar prav gotovo opravičuje razvoj in uporabo prenosljivih prevajalnikov, kakršen je Pascal-P (5).

ZAHVALA

Zahvaljujem se dr. E. Zakrajšku za vsestransko pomoč, diskusije in nasvete pri delu, dipl.ing. R. Rojku za testni program in mgr. J. Roškarju za pomoč pri računalniški obdelavi. Nalogo je finansirala RS Slovenije.

LITERATURA

(1) K. Jensen, N. Wirth, Pascal User manual and report, Springer-Verlag, Berlin, 1978.

(2) N. Wirth, Pascal-S: A subset and its implementation, ETH, Zürich, 1975.

(3) U. Amman,et al., The Pascal P Compiler: Implementation notes , ETH, Zürich, 1976.

(4) B. Hansen, The programming language Concurrent Pascal,IEEE Transactions on Software Engineering 1, 2 (June 1978), 199-207.

(4a) A.C. Hartmann, A Concurrent Pascal Compiler for Minicomputers, Lecture notes in Computer Science, No. 50, Springer-Verlag, Berlin, 1977.

(5) R.E. Berry, Experience with the Pascal-P Compiler, Software-Practice and Experience, 8, 617-627 (1978).

(6) RSX-11M Task Builder Reference Manual, Order No. AA-2588D-TC, Digital Equipment Corporation.

(7) E. Zakrajšek, osebne informacije.

(8) E. Zakrajšek, Programski jezik Pascal, Društvo matematikov, fizikov in astronomov SR Slovenije, Ljubljana, 1976.

(9) IAS/RSX-11 MACRO-11, Reference Manual, Order No. DEC-11-OIMRA-B-D, Digital Equipment Corporation.

DODATEK

SEZNAM STROJNIH UKAZOV

SC UKAZI(3)	PDP-11 UKAZI	
ABI	BPL	.+2
	NEG	-2(R5)
ABR	JSR	PC,ABR...
ADI	ADD	-(R5),-2(R5)
ADR	JSR	PC,ADR...
AND	BIT	-(R5),-(R5)
	BEQ	.+3
	MOV	#1,(R5)+
	BR	.+1
	CLR	(R5)+
DIF	JSR	PC,DIF...
DVI	JSR	PC,DVI...
DVR	JSR	PC,DVR...
EOF	JSR	PC,EOF...
FLO	JSR	PC,FLO...
FLT	JSR	PC,FLT...
INN	JSR	PC,INN...
INT	JSR	PC,INT...
IOR	BIS	-(R5),-2(R5)
MOD	JSR	PC,MOD...
MPI	MOV	-(R5),R0
	MUL	R0,-(R5)
	MOV	R1,(R5)+
MPR	JSR	PC,MPR...
NGI	NEG	-2(R5)
NGR	JSR	PC,NGR...
NOT	MOV	#1,R0
	SUB	-(R5),R0
	MOV	R0,(R5)+
ODD	JSR	PC,ODD...
SBI	SUB	-(R5),-2(R5)
SBR	JSR	PC,SBR...

```
SGS             JSR   PC,SGS...

SQI             MOV   -(R5),R0
                MUL   R0,(R5)
                MOV   R1,(R5)+

SQR             JSR   PC,SQR...

STOC, STOB      MOV   -(R5),R0
                MOVB  R0,@-(R5)

STOI, STOA      MOV   -(R5),@-(R5)

STO*            JSR   PC,STO*..

(KJER JE * = R, S)


STOM            JSR   PC,STOM..
                DOLZINA

TRC             JSR   PC,TRC...

UNI             JSR   PC,UNI...

STP             JSR   PC,STOP..

CSP             JSR   PC,XXX$$$

(KJER JE XXX IME STANDARDNE FUNKCIJE)


DEC             SUB   #VREDNOST,-2(R5)

ENT             MOV   #DOLZ,R0
                JSR   PC,INIT..

(KJER JE DOLZ DOLZINA LOKALNEGA BLOKA
V SKLADU)


FJP             TST   -(R5)
                BNE   .+2
                JMP   @#NASLOV

(NASLOV JE NASLOV ZNOTRAJ MODULA)


INC             ADD   #VREDNOST,-2(R5)

INDC, INDB      ADD   #ODMIK,-2(R5)
                CLR   R0
                MOVB  @-(R5),R0
                MOV   R0,(R5)+

INDI, INDA      ADD   #ODMIK,-2(R5)
                MOV   @-(R5),(R5)+

IND*            ADD   #ODMIK,-2(R5)
                JSR   PC,IND*..

(KJER JE * = R, S)


INDM            ADD   #ODMIK,-2(R5)
                JSR   PC,INDM..
                DOLZINA

(UKAZ           ADD   #ODMIK,-2(R5)
PREVAJALNIK GENERIRA LE TEDAJ, KO JE ODMIK
RAZLICEN OD NIC)


IXA             MOV   -(R5),R0
                MUL   R0,#DOLZINA
                ADD   R1,-2(R5)

LAO             MOV   #NASLOV,(R5)
                ADD   MAXM$$,(R5)+

(MAXM$$ VSEBUJE NASLOV ZACETKA SKLADA)


LCA             BR    .+(DOLZ/2)
        NSL:    NIZ
                (DOLZ/2 BESED)
                MOV   #NSL,(R5)+

(NIZ JE KONSTANTA (NIZ ZNAKOV) MED UKAZI)


LDOC, LDOB      CLR   R0
                MOVB  NASLOV(R2),R0
                MOV   R0,(R5)+

LDOI, LDOA      MOV   NASLOV(R2),(R5)+

LDO*            JSR   PC,LDO*..

(KJER JE * = R, S)


LDOM            JSR   PC,LDOM..
                DOLZINA

(CE JE POTREBNO, JE PRED LDO- UKAZOM SE
UKAZ            MOV   MAXM$$,R2         )

MOV             JSR   PC,MOV...
                DOLZINA NIZA

(DOLZINA NIZA JE KONSTANTA, KI SLEDI
UKAZU ZA KLIC PODPROGRAMA)


MST             MOV   #RN,R3

(RN  JE RAZLIKA NIVOJEV)


RET*            MOV   #DOLZINA,R1
                JSR   PC,CLOS..
                RTS   PC

(KJER JE * ZNAK, KI OZNACUJE TIP FUNKCIJE.
DOLZINA JE PRI FUNKCIJAH DOLZINA REZULTATA,
PRI PODPROGRAMIH PA JE ENAKA NIC)


SROC, SROB      MOV   -(R5),R0
                MOVB  R0,NASLOV(R2)

SROI, SROA      MOV   -(R5),NASLOV(R2)

SRO*            MOV   #NASLOV,R1
                JSR   PC,SRO*..

(KJER JE * = R, S)


SROM            MOV   #NASLOV,R1
                JSR   PC,SROM..
                DOLZINA

(CE JE POTREBNO, JE PRED SRO- UKAZOM SE
UKAZ            MOV   MAXM$$,R2         )


XJP             MOV   -(R5),R1
                JMP   NASLOV(R1)

CHK*            JSR   PC,CHK*..
                SPODNJA MEJA
                ZGORNJA MEJA

(KJER JE * = I, A)


CUP             MOV   #DOLZ,R1
                JSR   PC,NASLOV

(KJER JE DOLZ DOLZINA PARAMETROV)
```

```
XXX*            MOV   -(R5),R0
                SUB   -(R5),R0
                B&&   .+3
                MOV   #0,(R5)+
                BR    .+2
                MOV   #1,(R5)+

(KJER JE * = I, A, C, B)


XXX*            JSR   PC,DIF*..

(KJER JE * = R, S)


XXXM            JSR   PC,DIFM..
                DOLZINA NIZA

( PRI XXX = EQU  JE   && = EQ,
            GEQ           LE,
            GRT           LT,
            LEQ           GE,
            LES           GT,
            NEQ           NE)


LDA             MOV   #NASLOV,(R5)
                ADD   R&,(R5)+

(KJER JE & = 4 ZA LOKALNE ALI 2 ZA
GLOBALNE SPREMENLJIVKE.
CE JE POTREBNO, JE PRED UKAZOM LDA-
 UKAZ           JSR   PC,.LOCA.
                RAZLIKA NIVOJEV   )


LDCI,LDCB,LDCC  MOV   #VREDNOST,(R5)+

LDCR            MOV   #PRVA,(R5)+
                MOV   #DRUGA,(R5)+

(KJER STA PRVA IN DRUGA OBE BESEDI
REALNE VREDNOSTI)


LDCS            BR    .+8
                MNOZICA
                JSR   PC,SAVS..

(MNOZICO PREVAJALNIK GENERIRA KOT 8 BESED
DOLGO KONSTANTO)
```

```
LODC, LODB      CLR   R0
                MOVB  NASLOV(R&),R0
                MOV   R0,(R5)+

LODI, LODA      MOV   NASLOV(R&),(R5)+

LOD*            MOV   R&,(R0)+
                MOV   #NASLOV,R1
                JSR   PC,LOD*..

(KJER JE * = R, S)


LODM            MOV   R&,R0
                MOV   #NASLOV,R1
                JSR   PC,LODM..
                DOLZINA

(& = 4 ZA LOKALNE SPREMENLJIVKE, 2 ZA
GLOBALNE SPREMENLJIVKE. PRED  UKAZOM
LDO-  JE PO POTREBI SE UKAZ
                JSR   PC,.LOCA.
                RAZLIKA NIVOJEV      )


STRC, STRB      MOV   -(R5),R0
                MOVB  R0,NASLOV(R&)

STRI, STRA      MOV   -(R5),NASLOV(R&)

STR*            MOV   R&,R0
                MOV   #NASLOV,R1
                JSR   PC,STR*..

STRM            MOV   R&,R0
                MOV   #NASLOV,R1
                JSR   PC,STRM..
                DOLZINA

(ZNAKA & IN * POMENITA ISTO, KOT PRI
IKAZU  LOD- )


UJP             JMP   @#NASLOV

UJC             JSR   PC,UJC...

UKAZA CHR IN ORD PREVAJALNIK IZPUSTI.
```

SISTEM OBRADE PODATAKA NA SVETSKOM PRVENSTVU U SKIJAŠKIM SKOKOVIMA NA PLANICI 1979. GODINE

M. MILETIĆ
M. KOMUNJER

UDK: 681.3.06

ELEKTROTEHNA, LJUBLJANA
ELEKTROTEHNA, ZAGREB

Članak ukratko opisuje strojnu i programsku opremu upotrebljenu pri obradi podataka na V svetskom prvenstvu u skijaškim skokovima na Planici marta 1979 godine. Dat je prikaz mikroračunarskog sistema za prikupljanje podataka, specijalnih pretvarača za DELTA 340/5 minirąčunar, generatora video karaktera te potrebnih programa za zadovoljenje dvojnih zahteva obrade podataka.

In this article brief description is given of hardware and software used for data procesing at V world championship in ski jumping at Planica in march 1979. Microcomputer sistem for data aquisition, special interfaces for DELTA 340/5 (PDP 11 compatible) minicomputer, video character generator and required programs for dual purposes are given.

1. U V O D

Organizacija V svetskog prvenstva u skijaškim skokovima drugog na Planici, poverena je Organizacionom komitetu koji se sastoji od grupe entuzijasta koji već niz godina rade na razvoju ovog interesantnog sporta. Početkom januara 1979 godine kontaktirana je "Elektrotehna"-OOUR za računare DIGITAL sa željom da se domaći računari DELTA programa iskoriste za obradu podataka. Koordiniranom akcijom sa Elektrotehnoćkim fakultetom u Ljubljani i Radiotelevizijom Ljubljana definirani su zahtevi za informacijama koje će se obradivati preko Delta računara. Mikroprocesorski sistem za prikupljanje podataka izradila je ekipa prof. dr. A. Wedama sa Elektrotehničkog fakulteta, sistem za generisanje i sinhronizaciju video signala ekipa dipl. ing. J.Biteżnika sa RTV Ljubljana.

2. S I S T E M Z A P R I K U P L J A N J E P O D A T A K A

Ovaj sistem sastoji se od centralne jedinice i do 12 perifernih jedinica koje služe za unošenje sledećih podataka:

1. startni broj takmičara
2. brzina takmičara na mostu
3. brzina vetra
4. dužine skoka
5. pet sudijskih ocena

Sistem je baziran na univerzalnom mikroračunarskom sistemu sa "Motorola" 6800 procesorom i različitim ROM za specifične funkcije.

Sve ulazne jedinice vezane su paralelno preko simetričnih linij sa centralnom jedinicom, dok su tastature i LED pokazivači vezane preko 40-linijskog paralelnog adaptera.

Veza centralne jedinice sa Delta računarom ostvarena je takođe sa simetričnom linijom na udaljenosti oko 200 metara brzinom prenosa od 300 baud-a.

Kako mikroračunarski sistem za unošenje brzina vetra i takmičara nije bio pravovremeno pripravljen, ovi podaci unosili su se u Delta računar preko video terminala po 20 mA strujnom krugu brzinom od 9600 baud-a.

Poruke mikroračunara su uvek bile dugačke 4 bajta sa jedinstvenim, indetifikacionim prvim bajtom i ponavljane su pet puta radi veće pouzdanosti u prenosu pod uslovima niza smetnji od radio uredjaja, TV aparatura, niskonaponske mreže i ekstermnih temperatura.

3. R A Č U N A R S K I S I S T E M D E L T A 340/5

Specijalan sistem tipa Delta 340/5 je bio pripravljen za Planicu sa maksimalnim naglaskom na pouzdanost rada. Centralni procesor je baziran na PDP 11/34, obezbeđeno je 160 kbajta MOS memorije sa baterijskim napajanjem za slučaj nestanka struje, dva diska tipa RK05 sa po 2,5 Mbajta, dva terminala sa papirnim otiskom tipa LA36 i 8 videoterminala KOPA 700.

Ulaz podataka sa mikroračunara išao je preko asinhronog pretvarača tipa DL11A sa dodatnim pretvaračem sa simetrične linije na TTL nivo.

Izlaz podataka u formi 16 bitne reči išao je preko pretvarača tipa DR11K. Oba ova specijalna pretvarača bila su duplirana i stalno aktivna sa mogućnošću programskog prelaženja na željenu jedinicu.

Zbog temperaturnih varijacija od -15 do +25°C sistem nije isključivan pet dana, a preko noći temperatura je donekle povišavana radijatorima u RTV centru. Kopa terminali su pak bili izloženi i najminimalnijim temperaturama jer su se nalazili na sudijskom stupu bez ikakvog zagrevanja, te isključenim napajanjem preko noći!

4. SISTEM ZA GENERISANJE VIDEO SIGNALA

Pri ranijim upotrebama računara na skijaškim skokovima, televizija se zadovoljavala snimanjem ekrana videoterminala. Kvalitet slike je bio na granici prihvatljivog te je odlučeno da se izvrši elektronsko povezivanje između računara i TV mešačkog pulta.
Odabran je hibridni (MOS,TTL) TV kontroler kanadske firme "Matrox". MTX-1632SL je organiziran kao 512X8 RAM, prikazuje 16X32 karaktera sa 7X9 ASCII matricom, interno se osvežava, spolja sinhronizuje, TTL je kompatibilan sa vremenom pristupa manjim od 550 ns i malom potrošnjom sa 5V.
Pozicija karaktera na ekranu je određena sa 9 bita, ostalih 7 se koriste kao standardan ASCII set.
Pristup memoriji je moguć u vremenu horizontalnog ili vertikalnog impulsa blankiranja, znači 12 mikrosekundi svakih 60 mikrosekundi ili 3,3 ms svakih 20 ms.
Radi što bržeg prenosa iskorišćene su obe mogućnosti tako da se kompletan ekran uvek ispiše za vreme jedne poluslike.
Sinhronizacija je izvršena korišćenjem kontrolnih signala DATA READY i DATA ACCPDT koji signaliziraju prisustvo važećih podataka na 16 linija i potvrdu memorisanja istih u VRAM (video RAM).
Impuls DR se pamti u VRAM adapteru, DA pamti DR11K pretvarač i šalje procesoru kao potvrdu okončanog prenosa. Ovo omugočava prenos jednog karaktera za oko 10 mikrosekundi te nije potrebno koristiti prekide programa (interup).

5. PROGRAMSKI ZAHTEVI SISTEMA

Da bi mogli opisati programski sistem koji je bio izradjen za svjetsko prvenstvo u skijaškim letovima "PLANICA 79", potrebno je da u kratko opišemo samu strukturu takmičenja. Pri tome je neobično važno obratiti pažnju na same kanale i protok ključnih informacija.
Iz opisa same računarske konfiguracije vide se točno punktovi sa kojih su dolazile informacije u sistem.
Važno je napomenuti da je sistem morao zadovoljiti dvije grupacije: TV i organizacioni komitet, što se je pokazalo tokom takmičenja kao određeno protivurječje.
Sama dinamika takmičenja mogla bi se opisati slijedećim fazama:

1. Takmičar na startu..... vrijeme 0
2. Takmičar na mostu..... vrijeme 10
3. Takmičar doskočio..... vrijeme 20
4. Poznata dužina skoka .....vrijeme 25
5. Poznata brzina vjetra i brzina na mostu .....vrijeme 20-40
6. Poznate sve ocjene sudaca .....vrijeme 60
7. Ponovo prva faza.....vrijeme 65

Vremena uz pojedine faze dana su prosječno u sekundama, a bila su ustanovljena tek za vrijeme pravog takmičenja.
Na raspolaganju smo imali slijedeće ulazne uredjaje:

A.8 terminala ekranskog tipa (2+2 na sudačkom tornju, 4 u računarskoj sali)
B.2 terminala sa štampačima (u računarskoj sali)
C.2 veze sa mikroprocesorskim sistemom za prikupljanje informacija o broju takmičara, dužini skoka i ocjenama sudaca)
D.2 TV monitora koja su prikazivala u internom TV sistemu izlaz iz procesorskog sistema
E. Telefonski kanal izmedju tornja i startnog mjesta
F. Interfonski kanal sa TV režijom

Uredjaji od A-C bili su vezani direktno na računar dok su kanali D-F bili namijenjeni ljudskoj komunikaciji.

Zadaci računarskog praćenja i obrade takmičenja bili su slijedeći:

1. Prikazati za fazu 1 na TV ekranu startni broj, prezime i inicijale imena te državu takmičara, a u slučaju skoka 2 ili 3 prethodne rezultate
2. Nakon doskoka prikazati sve kao pod 1, plus dužinu skoka, brzinu na mostu i brzinu vjetra
3. Prikazati ocjene sudaca i trenutni poredak
4. Nakon završene serije izraditi rezultate serije prema ocjenama sudaca i dužinama skoka, štampati tu listu te nakon toga prikazivati listu na TV ekranu.

Ovo ponoviti za sve tri serije, te na kraju dana izraditi listu za pobjednike dana. Svaki dan trebalo je izraditi iste obrade, a na kraju takmičenja trebalo je izraditi listu konačnog pobjednika.
Dok se sistem planirao i analizirao, ustanovljeni su slijedeći kanali informacija:

A. Za dobivanje startnog broja, dužine skoka, ocjene sudaca komunikacioni kanal iz mikroprocesorskog sistema sa sudačkog tornja.
B. Za dobivanje brzine vjetra, telefonski kanal i operatorsko unošenje u sistem.

Sistem programa izradjen ne temelju gornjih informacionih kanala pokazao se kao potpuno neadekvatan prvog dana probnih skokova, a razlog je bila dinamika dotoka informacija, koja je bila sasvim drugačija od dinamike koja nam je bila prezentirana dok se sistem projektirao.
Situacija je izgledala slijedeće:
Takmičar na zaletištu, za TV je potrebno prikazati ime, prezime i startni broj.
Na mikroprocesorskom kanalu sa sudačkog tornja ne stiže informacija o startnom broju. Takmičar je doskočio, TV kamere ga prate, informacije o skoku sa kanala A još nema.

Napokon dolazi informacija o dužini skoka, on se prikazuje na ekranu zajedno sa brzinom vjetra i brzinom na mostu. Sada dolazi najkritičnija razlika izmedju predvidjenog modela i stvarne situacije. TV prikazuje slijedećeg tamičara na mostu, a suci su još daleko od toga da dadu svoje ocjene po kanalu A.
TV zahtijeva prikaz imena novog takmičara, iako ocjene za prethodnog još nisu poznate, a sistem je izgradjen tako da obradjuje i prati takmičara po takmičara, a nikako simultano dvojicu.
Nakon ovakvog početka ekipa zadužena za programsku podršku, odlučuje da prepravi sistem, tako da odgovara stvarnom modelu takmičenja. Vrijeme koje smo imali na raspolaganju bilo je 1 dan i sve raspoložive noći.
Samo meteorološka situacija omogućila nam je dodatni dan. Nakon ponovnog programiranja sistem je imao slijedeću funkcionalnost:

1.Takmičar na mostu

U trenutku pojave takmičara na mostu kroz službeni telefonski kanal doznaje se startni broj prije pojave istog kroz kanal A. Operater u sali unaša broj preko terminala i prikazuje ga na TV monitoru. Podatak se sa strane evidentiranja ne smatra službenim. Preko službenog kanala A dolazi broj takmičara, on se uspoređuje sa onim koji je ručno unesen i ukoliko su isti, upisuje se u banku podataka, a u koliko su različiti obavještava se operator o razlici.

2. Takmičar doskočio

Dužina skoka prenesena kroz telefonski kanal dolazi u sistem oko 10 sekundi ranij nego kroz službeni kanal, te je operater momentalno unosi i prikazuje na TV ekranu. Asinhrono ovoj operaciji sa sudačkog tornja unosi se preko terminala brzina vjetra i brzina na mostu za takmičare, i u slučaju da je startni broj takmičara koji se prikazuje na ekranu i onog na kojeg se odnose mjerenja isti, ti podaci se ulažu u banku podataka i prikazuju se na ekranu. Pod istim uvjetom nakon dolaska dužine skoka kroz službeni kanal vrši se memoriranje u banku podataka i prikaz službene dužine na ekranu.
(Tokom takmičenja ove dvije vrijednosti nisu se nikad razilazile).

3. Čekanje na ocjene

Od starta takmičara do ovog momenta proteklo je otprilike 30 sekundi.
Sada se TV sistem razilazi od sistema prikupljanja podataka. TV je u ovom momentu obično počela pokazivati slijedećeg takmičara na mostu i tražila je od računarskog sistema da prikaže njegove podatke. Pošto je sistem sada bio dizajniran da radi u paraleli sam sa sobom, mogao je operater inicirati prikazivanje karakteristika novog takmičara.
Program za skupljanje oficijelnih informacija i dalje je čekao na službene podatke iz kanala A. U trenutku prispjeća svih podataka po kanalu A, oni su bili prikazani operateru i zatraženo je da ih on potvrdi prema slici na posebnom TV kanalu. Ovo je bilo nužno iz nekoliko razloga: komunikacija od sudačkog tornja prema računaru bila je jednosmjerna, pa je bilo nemoguće da računar diktira bilo kakve korekcije na sudački toranj, ocjene sudaca bile su često pogrešno utipkane, pa je bilo potrebno da se cijeli mikroprocesorski sistem resetira i nanovo u njega upišu ocjene. Nakon što je računaru odobreno da prihvati ocjene, on ih još jednom provjerava po logičkoj ispravnosti i za svaku ocjenu koja nije odgovarala nužno je ispraviti vrijednost. Tek nakon svih tih provjera ocjena je bila upisana u banku podataka, a u slučaju da su karakteristike tog takmičara još uvjek bile na ekranu, ocjene su se tada prikazivale i na ekranu. Svaka kompletna informacija bila je zapisana i na vanjski medij diska.

Po završetku serije zapis rezultata serije bio je izračunat i sortiran iz disk datoteke i štampan za organizacioni komitet, a za TV je bila prikazana lista na TV ekranu.

6. P R O G R A M S K A I Z V E D B A S I S T E M A

Za realizaciju sistema bili su upotrebljeni programski jezici MACRO I FORTRAN.
Centrala karakteristike programskog paketa, bila je memorijsko rezidentna banka podataka u koju su imali simultano pristup svi programi na sistemu.
Ovo je bilo moguće izvesti zbog osobine DELTA-M operativnog sistema, koji dozvoljava "zajedničku memoriju" za više programa. Pojednostavljeno rečeno jedna lista u FORTRANU okupirala je isto fizičko mjesto u memoriji sa listom u MACRO programu i dugom FORTRAN programu. Sam sistem je multiprogramski pa je tako otpala briga oko organizacije više simultano aktivnih programa u sistemu.
Paket programa sastojao se iz slijedećih programa:

- VRAM

Program koji je bio startan odredjenim ugadjanjem u sistemu i kao ulaz je uzimao polje alfanumeričkih znakova, a kao izlaz je generirao sliku na TV monitoru (JEZIK MACRO).

- TV 1

Program koji je interaktivno komunicirao sa operaterom za terminalom, tražio od njega broj takmičara, potom je pretraživao banku podataka za informacije o tom startnom broju, te je nakon toga selektirao program VRAM, startao VRAM i vratio se da dobije nova uputstva od operatera. Ovaj program nije modificirao banku podataka, već je služio samo za prikaz informacija iz banke podataka. (JEZIK MACRO).

- TV 2

Program koji je interaktivno unosio i preračunavao brzine takmičara na mostu. Rezultate je unosio u banku podataka, a u koliko su se ti rezultati odnosili na takmičara koji je bio pokazivan na TV monitoru, onda je on zahtijevao i startanje VRAMA sa istim podacima.
(JEZIK FORTRAN).

- AMI

Program koji je prikupljao podatke sa mikroprocesorskog sistema sa sudačkog tornja, filtrirao neispravne podatke, uklanjao moguć šum na liniji, te nakon ispravno dobivenih podataka, startao program CALC. Ovaj program modificirao je privremenu banku podataka.
(JEZIK MACRO).

- CALC

Program za obradu, ispitivanje logičnosti, ažuriranje i prikaz podataka.

Na temelju informacija dobivenih po kanalu A, program je izvršio provjeru logičke ispravnosti podataka (dužina veća od 200 m, ocjena završena 5 ili 0, sve manje od 20 i slično). Nakon dobivenog kompletnog seta podataka program je tražio prihvaćanje istih od operatera, u slučaju negativnog prihvaćanja mogli su se neprihvatljivi unositi ručno. Program je potom ažurirao banku podataka i ujedno upisao podatke za vanjski medij.

U koliko je na ekranu bio prikazan takmičar za kojeg su mu stigli podaci program je inicirao VRAM i prikaz istih podataka. (JEZIK FORTRAN).

- SORT 1

Program za sortiranje rezulatata jedne serije skokova, štampanje rezulatata na linijski štampač, ujedno je i stvorena datoteka za isti prikaz na TV ekranu.
(JEZIK FORTRAN).

- SORT TV

Program koji je sortiranu listu prikazivao na TV ekranu. (JEZIK FORTRAN).

- SORT 2

Program za sortiranje rezultata jednog dana, štampanje istih i stvaranje filea za prikaz na TV ekranu.
(JEZIK FORTRAN).

- SORT 3

Program za sortiranje rezultata ukupnog pobjednika. Ovaj program nikad nije dovršen za sistem koji je bio ponovno dizajniran. Razlog je bio jednostavna nestašica vremena. Stvarno je šteta obzirom da je ovaj program bio nužan za organizacioni komitet i sa tog stanovišta je ostatak obrade bio gotovo nevažan.

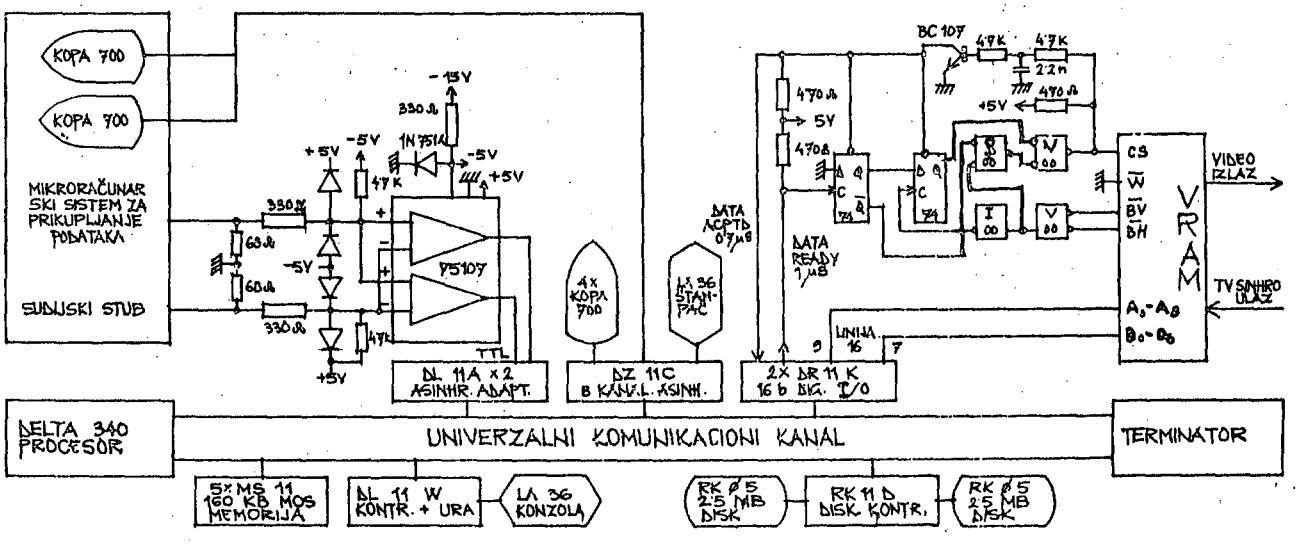
Usudjujem se reći da je sa programskog i projektnog stanovišta ovaj program nosio manje od 10% kompleksnosti cijelog sistema. Važno je napomenuti da su programi SORT bili konstantno modificirani zbog česte izmjene većine računanja pojedinih rang lista.

7. Z A K L J U Č A K

Računarska obrada podataka na svjetskom prvenstvu u skijaškim letovima "PLANICA 79" pokazala je nekoliko stvari:

- Sagradili smo u veoma kratkom roku (100 sati) sistem koji je uspješno pratio dinamiku takmičenja.
- U konceptu i dizajnu sistema nije bilo bitnih propusta (drugi sistem).
- Propustili smo da sa sredstvima javnog informiranja (novinari i TV komentatori) održimo sastanak i objasnimo im što se vidi na TV monitorima.
- Dozvolili smo da se neke specifikacije računanja definiraju u toku samog takmičenja što je uzrokovalo neizvršenje zadnje obrade na sistemu.
- Stekli smo dragocjena iskustva za rješavanje problema slične naravi, pa stoga mogu reći da je za nas zadatak izvršen uspješno.

Nadamo se da će slijedeća priredba u kojoj će DELTA sistem imati ulogu računarske obrade dokazati da smo na temelju ovog projekta u stanju da na visokom profesionalnom nivou rješimo sve probleme i zadatke vezane za bilo koji takav projekt.



SIMULACIJA STREŽNIH SISTEMOV Z GPSS

N. GUID

UDK: 681.3 : 519.682.6

VISOKA TEHNIŠKA ŠOLA, MARIBOR

Članek obravnava reševanje najrazličnejših osnovnih tipov strežnih sistemov s simulacijo z jezikom GPSS. Podani so problemi pri vrednotenju rezultatov in priporočila za izboljšanje točnosti simulacije.

QUEUEING SYSTEMS SIMULATION WITH GPSS. This paper deals resolution of the most different fundamental types of queueing systems by the simulation with GPSS. The problems in validating simulation results and recommendations in improvement of simulation precision are considered.

I. UVOD

Problemi čakanja, ki jih uspešno rešujemo z metodami množične strežbe, pogosto obstojajo tudi v računalniških sistemih, kjer se pojavljata neregularnost dolžine strežbe in neregularnost prihoda zahtev ali kjer mnogo zahtev uporablja eno ali več računalniških strežnih naprav. Pomen strežnih sistemov in njihove analize je zlasti narasel s povečujočo uporabo računalniških sistemov z delovanjem v realnem času (procesni računalniki, računalniki s časovnim dodeljevanjem). Z metodami množične strežbe je možno oceniti zakasnitve pri obdelavi vseh vrst (programska ali elektronska oprema), velikost potrebnega pomnilnika, izkoriščenost pomnilnika in procesorja, dalje nam pomaga določiti najučinkovitejše strežno pravilo itd.

Teorija množične strežbe je razvila eksaktne analitične metode, s katerimi pa je mogoče rešiti le nekatere idealizirane strežne sisteme. Tako smo prisiljeni razvijati aproksimativno analizo iz kombinacij nekaj idealiziranih sistemov. Omenjene direktne analitične metode lahko nadomestijo drage simulacijske postopke, toda samo do neke meje, ko je potrebno zaradi večjega obsega in natančnosti uporabiti simulacijo, ki jo izvedemo z računalniškimi jeziki za simulacijo diskretnih sistemov.

Prednost simulacije pred analitičnimi metodami je v tem, da lahko obravnava nek zapleten detajl v logiki delovanja sistema in izračuna učinek sprememb tega detajla. Dalje lahko s simulacijo ocenimo medsebojni vpliv med več podsistemi.

Simulacija tudi večinoma ni direktno orodje za sintezo. V splošnem je simulacija omejena na določitev, kako posebna konfiguracija reagira na posebno okolje. Še vedno ostane načrtovalcu funkcija analize rezultatov in odločitve, kje in kako izboljšati sistem, ali z drugimi besedami, načrtovalec ostaja povratni element v načrtovalni zanki.

II. SIMULACIJSKI JEZIKI

Simulacijske jezike (simulatorje) delimo v [1]:
1. uporabnikove simulatorje,
2. posebno namenske simulatorje in
3. splošno namenske simulatorje.

Kdorkoli hoče napisati svoj lastni simulacijski program (t.i. uporabnikov simulator), mora imeti za to naslednje razloge:
- želene karakteristike modela niso niti primerne, niti možne v obstoječih simulacijskih jezikih,
- zahtevane so posebne vhodne in izhodne lastnosti ter
- program se bo uporabljal dostikrat.

Glavna prednost uporabnikovih simulatorjev je učinkovitost. Cena za njihov razvoj mora biti poplačana v večkratni uporabi. Njihova slabost je nefleksibilnost. Pisani so seveda v višjih programskih jezikih.

Posebno namenski simulator ima vgrajeno logiko posebnega strežnega sistema, tako da uporabniku ostane samo vstavljanje nekaterih parametrov. Število posebno namenskih simulatorjev je ogromno. Dobimo jih lahko pri prodajalcih računalnikov in konzultanskih organizacijah. Dokumentacija javnosti žal ni dostopna. Omenil bi samo tri jezike:

1. GERTS III QR je razširjega verzija GERT (Graphical Evaluation and Review Technique, [2]). Razvili so ga za analizo sistemov s strežnimi napravami, ki zahtevajo večkratne izvore za izvršitev strežne aktivnosti.
2. QAL (Queuing Analysis Language, [3]) je visoko nivojski jezik za poenostavitev reševanja tako enostavnih kot kompleksnih modelov množične strežbe. Z njim zelo učinkovito simuliramo sodobne računalniške sisteme, ki imajo vedno več paralelnosti v delovanju.
3. CSS (Computer System Simulator, [4]) je manj splošen od GPSS. Uporablja precej tehnik iz GPSS, namenjen je za simulacijo računalniškega programskega sistema, predvsem operacijskega sistema.

Splošno namenski simulator je namenjen za simulacijo najrazličnejših strežnih sistemov. Najbolj razširjeni jeziki so GPSS, SIMSCRIPT in SIMULA.

Sam sem izbral GPSS, saj je bil to edini jezik, ki mi je bil dostopen. GPSS (General Purpose Simulation System) so razvili pri IBM. Obstoja več verzij in je instaliran na večini računalniških sistemov. Ima naslednje pomembne

lastnosti:
- je lahek za učenje,
- zahteva velik hitri pomnilnik,
- zahteva daljši čas simulacije in
- je primeren za simulacijo strežnih sistemov vseh vrst.

III. VREDNOTENJE REZULTATOV SIMULACIJE

Vprašanje, ali je simulacijski rezultat pravilen, zahteva obravnavo dveh neodvisnih zahtev [1]:

1. prva se nanaša na točnost modela
2. druga pa na preciznost statičnih rezultatov.

Ad. 1: Visoka stopnja natančnosti zahteva, da mnogo poenostavljenih ali izpuščenih detajlov bistveno ne vpliva na sistemsko učinkovitost. S primerjavo simulacijskih tekov z ali brez nekega detajla lahko izmerimo relativni efekt spremembe. Zmeraj spreminjamo samo en parameter ali spremenljivko, da lahko ugotovimo njen vpliv. To je zlasti pomembno pri kompleksnih modelih, saj se le tako izognemo logičnim, t.i. sistematskim napakam.

Ad. 2: Ta problem je posledica narave procesa v tehniki simulacije, ki temelji na odbirkih iz neke porazdelitve (statistike). Problem je določiti točko ali trenutek, ko je dosežena iskana preciznost. To dosežemo z nivojem zaupanja, kar pa je težko izvesti.

Simulacijski rezultati navadno niso neodvisni, temveč imajo celo visoko korelacijo, kar ima za posledico težko določitev variance, ki je potrebna pri izračunu intervala zaupanja [5]. V praksi opažujemo nekaj zaporednih kumulativnih rezultatov in jih primerjamo med seboj. Če ti rezultati fluktuirajo v ozkem pasu, je preciznost dosežena, če opazimo njihov trend, povečamo število simulacijskih tekov.

Potem je tu še nestacionarnost porazdelitve odbirkov oziroma odvisnost od začetnih pogojev. Vpliv prehodnega pojava odpravimo na več načinov:

a) Postavimo tipične začetne pogoje, kar pomeni, da moramo sistem že delno poznati, preden sploh začnemo s simulacijo.

b) Simuliramo daljši čas.

c) Z metodo paketne srednje vrednosti simuliramo krajši čas.

   N tekov razdelimo v p paketov dolžine n tekov (N=np). Učinek je enak učinku eksperimenta dolžine n tekov, ki ga ponovimo p-krat, tako da končno stanje enega teka postane začetno stanje naslednjega simulacijskega teka. Prednost te metode je v tem, da ni potrebno odstraniti začetno periodo pri vsakem paketu in da zahteva majhen pomnilnik, saj shranjuje samo vsoto paketnih srednjih vrednosti in vsoto njihovih kvadratov ter števila za tvorbo tekoče paketne srednje vrednosti.

d) Simuliramo nekaj tekov z netipičnimi začetnimi pogoji, nato samo zbrišemo statistiko, ne da bi spremenili stanje sistema. Nato nadaljujemo s simulacijskimi teki ter izračunamo novo statistiko. Toda žal ne eksistira preprosto pravilo, ki bi povedalo, koliko tekov moramo zbrisati. Metoda za določitev števila izbrisanih tekov temelji na poznavanju funkcije standardne deviacije v odvisnosti od števila tekov pri simulaciji brez brisanja začetnih tekov.

   Pri GPSS je najlaže uporabiti zadnji način (stavek RESET za brisanje statistike).

   Rezultat je nadalje odvisen od zaporedja naključnih števil [6]. Izkaže se, da je fluktuacija še zmeraj opazna tudi pri tisočem teku (cca 5 %), no vseeno akumulirana povprečna vrednost teži k stacionarni povprečni vrednosti. Temu bi se izognili s p-kratno ponovitvijo serije n simulacijskih tekov z različnimi zaporedji naključnih števil, kar pa s standardnimi programskimi instrukcijami pri GPSS ni mogoče.

IV. SIMULACIJA Z JEZIKOM GPSS

Narava in osnovni koncepti GPSS so taki, da omogočajo enostavno simulacijo vseh vrst strežnih sistemov in mrež. GPSS ima vgrajen program za izpis določenih statističnih izhodnih rezultatov (izkoriščenost strežnika, intenzivnost prometa, povprečno in maksimalno število čakajočih enot v vrsti, delež enot, ki jim ni treba čakati v vrsti, povprečna dolžina strežbe, povprečna dolžina čakanja v vrsti itd.). To so merila, ki nastopajo tudi pri analizi strežnih sistemov in jih lahko dobimo zato, ker GPSS vsebuje generatorje psevdonaključnih števil. S pomočjo le teh in najrazličnejših funkcijskih odvisnosti je možno predstaviti še tako zapleteno verjetnostno porazdelitev presledkov med zaporednima prihodoma enot in verjetnostno porazdelitev dolžine strežbe [7].

Moje izkušnje pri delu z GPSS

Edini večji problem predstavljajo generatorji naključnih števil, saj nobeden od številnih učbenikov ne zajema kompleksno analizo njihovega delovanja.

Zaporedje, ki ga generira eden izmed 8 enakih generatorjev naključnih števil (RN1 do RN8), spremenimo le s spremembo začetne vrednosti v algoritmu za generiranje naključnih števil (imenovano "seed").

1. Če imamo v nekem sistemu dva ali več naključnih procesov, tedaj ne smemo uporabiti različne generatorje z isto začetno vrednostjo, saj pride do neke vrste "resonance". Posledica tega je, da rezultati tudi pri mnogo simulacijskih tekov (cca 10.000) odstopajo od povprečnih vrednosti za cca 50 %.

2. Sprememba enega ali več začetnih vrednosti naključnih generatorjev spremeni rezultate pri cca 10.000 simulacijskih tekov do 5 %, pri cca 20 simulacijskih tekov pa do 40 %. To pomeni, da je simulacija z malo teki nenatančna.

3. Bloki z enakomerno porazdelitvijo (n.pr. ADVANCE 80, 40; GENERATE 3,1 itd.) uporabljajo vrednosti samo enega generatorja naključnih števil, t.j. RN1.

4. Če več naključnih procesov v strežnem sistemu uporablja isti generator naključnih števil, ne moremo študirati vpliva prioritete, saj tedaj nastopi drugi vrstni red zaseganja števil iz generatorja naključnih števil kot pri sistemu brez prioritet [7]. To pomeni, da se hkrati z vplivom prioritete vrine še drug vpliv, ki pokvari rezultate. Temu se izognemo tako, da vsakemu naključnemu procesu priredimo en generator naključnih števil z različno začetno vrednostjo. Če imamo samo eno enakomerno porazdelitev, tedaj ji pustimo generator RN1, kar pa seveda nikjer v programu eksplicitno ne zapišemo (za druge na-

ključne procese tedaj ne smemo uporabiti RN1).

5. Primerjava analitično izračunanih vrednosti ter rezultatov simulacije pri različnih številih tekov je pokazala, da je potrebno simulirati 10.000 tekov oz. enot, ki gredo skozi strežni sistem. Večina primerjav je dala odstopanje manjše od 5 %, edino v nekaj primerih sem dobil 8 + 12 % odstopanja od analitično dobljene vrednosti. Primerjavo sem izvršil tudi z različnimi sekvencami naključnih števil (različne začetne vrednosti generatorjev) in vsa odstopanja so bila v gornjih mejah.

6. Primerjal sem tudi vrednosti, dobljene pri različno dolgih eliminiranih začetnih periodah (odprava prehodnega pojava) in nisem opazil hitrejše konvergence k analitičnim povprečnim vrednostim.

Moje priporočilo je sledeče:
Upoštevati je treba podana priporočila v zvezi z uporabo generatorjev naključnih števil. Simulirati moramo tako, da bo šlo skozi strežni sistem cca 10.000 enot enega tipa, ne da bi upoštevali prehodni pojav. Rezultat simulacije bo z veliko verjetnostjo odstopal od analitično dobljene vrednosti največ za 5 %.

V. SIMULACIJA RAZLIČNIH TIPOV STREŽNIH SISTEMOV Z GPSS

GPSS je primeren za simulacijo strežnih sistemov vseh vrst. Z njim lahko enostavno simuliramo najbolj splošno naključno porazdelitev. Izvor zahtev in porazdelitev vhodnega toka omogoči blok GENERATE, porazdelitev dolžine strežbe blok ADVANCE. Enega strežnika predstavimo z blokoma SEIZE in RELEASE, več paralelnih strežnikov pa z blokoma ENTER in LEAVE. Vso statistiko čakajoče vrste dobimo s pomočjo blokov QUEUE in DEPART, medtem ko se statistika strežnih naprav izvrši avtomatsko.

Pri simulaciji nehomogenih strežnih sistemov uporabimo več enakih segmentov modela s samo različnimi bloki GENERATE.

Pri cikličnih strežnih sistemih koristimo blok TRANSFER.

Ne da bi bilo potrebno posebej specificirati, GPSS simulira strežni sistem z neskončno kapaciteto in pravilom FIFO.

Sisteme z omejeno kapaciteto simuliramo s pomočjo blokov ENTER, ADVANCE Ø (s časovno zakasnitvijo Ø) in LEAVE (poslednji blok LEAVE moramo postaviti za blok, ki označuje vstop enote v proces strežbe).

Strežne sisteme s poljubnim neprioritetnim strežnim pravilom (n.pr. LIFO, RSS) simuliramo s pomočjo blokov LINK in UNLINK.

Pri sistemih z neprekinjevalnim prioritetnim strežnim pravilom je potrebno samo prirediti ustrezni prioritetni indeks določenemu tipu enot (uporabimo samo operand E v bloku GENERATE).

Za simulacijo sistemov s prekinjevalnim prioritetnim strežnim pravilom z nadaljevanjem uporabimo poleg operanda E v blokih GENERATE še PREEMPT in RETURN namesto SEIZE in RELEASE v segmentih modela s prekinjevalnimi enotami.

Pri simulaciji sistemov z drugimi prioritetnimi pravili moramo uporabiti še logične, tesťne in selektivne operacije.

Sisteme z grupno strežbo simuliramo s pomočjo operanda B v blokih SEIZE in RELEASE oziroma ENTER in LEAVE.

Pri simulaciji strežnih sistemov z grupnim vhodnim tokom uporabimo blok SPLIT.

Simulacija mreže strežnih sistemov je preprosta in ne zahteva kakih posebnih navodil, saj je glavna značilnost GPSS, da je bločni diagram nekega sistema ekvivalenten tako diagramu poteka kot glavnemu programu.

VI. PRIMER SIMULACIJE Z GPSS

V center za obdelavo sporočil prihajata dva tipa (razreda) sporočil s poissonsko porazdelitvijo (λ_1=0,5 s^{-1}, λ_2=0,1 s^{-1}). Prvi tip zahteva krajšo in konstantno dolžino obdelave (b_{11}=0,2 s), drugi tip pa daljšo dolžino obdelave z eksponentno porazdelitvijo. Primerjal sem delovanje sistema:
1. brez prioritet
2. če uvedemo neprekinjevalno prioriteto sporočilu tipa 1 in
3. če uvedemo prekinjevalno prioriteto sporočilu tipa 1.

Problem sem rešil analitično in s pomočjo simulacije. Naš primer predstavlja strežni sistem, ki ga označimo z razširjeno Kendallovo oznako M_1, M_2/D_1, $M_2/1/\infty$, ∞/∞, ∞/FIFO (PRI).

Za vse tri variante sem izračunal analitično povprečno dolžino zadrževanja za posamezen tip sporočila (W_1, W_2) in za oba tipa skupaj (W). Analitično žal ni mogoče določiti povprečnih dolžin čakanj za posamezen tip sporočila pri sistemu brez prioritet.

Da se izognemo vplivu zaradi različnega zaseganja po zaporedju naključnih števil, potem ko uvedemo prioriteto (glej poglavje IV), priredimo vsaki porazdelitvi samostojni generator naključnih števil, ki pa ne sme startati z isto začetno vrednostjo.

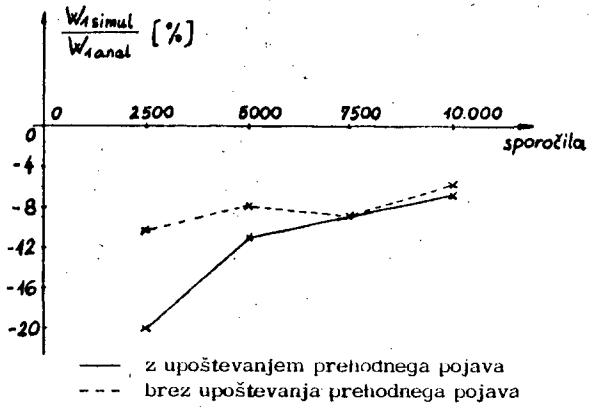
Rezultati simulacije so podani v oklepaju zraven ustreznih analitičnih rešitev (tabela 1) in ustrezajo cca 10.000 sporočilom 1. razreda in cca 2.000 sporočilom 2. razreda, potem ko v statistiki nisem upošteval prehodni pojav s 500 sporočili 1. razreda in 100 sporočili 2. razreda.

Hitro opazimo, da s prioritetnimi strežnimi pravili pri enaki izkoriščenosti procesorja dosežemo krajše povprečne dolžine zadrževanja in dolžine čakanja za oba razreda skupaj (W in Wq). Prekinjevalno strežno pravilo je v tem primeru več kot 3-krat učinkovitejše od navadnega pravila FIFO.

	brez prioritet	neprek. str. prav.	prek. str. prav.
W	7,25 (7,03)	4,5 (4,30)	2,26 (2,25)
W_1	- (6,20)	3,0 (2,78)	0,211 (0,212)
W_2	- (11,18)	12,0 (11,91)	12,55 (12,45)

Tabela 1

Študiral sem še relativna odstopanja od analitične vrednosti povprečnih dolžin zadrževanj v odvisnosti od števila simulacijskih tekov (prispelih sporočil v proces streženja) pri upoštevanju in neupoštevanju prehodnega pojava (slika 1).



Slika 1

VII. POVZETEK

Današnja razvojna stopnja teorije množične strežbe ne dopušča detajlne analize kompletnega sistema, ampak le analizo podsistemov. Pri slednjih je teorija razvila vrsto kriterijev, ki jih lahko koristno uporabljamo pri aproksimativni analizi in tudi pri simulaciji kompleksnejših sistemov množične strežbe.

Bilo bi napak, če bi trdil, da je najprimernejši jezik GPSS, saj je bil to edini jezik, ki mi je bil dostopen. Res pa je v ZDA to najbolj razširjen jezik za simulacijo diskretnih sistemov. Z njim je mogoče z lahkoto simulirati še tako kompleksen strežni sistem ali mrežo strežnih sistemov. Njegovo moč veča tudi možnost klicanja subrutin v jeziku FORTRAN. Problematična je hitrost simulacije, ki bi jo lahko izboljšali z metodo paketne srednje vrednosti z različnimi neodvisnimi zaporedji naključnih števil.

Preseneča tudi relativno veliko odstopanje od analitično dobljenih vrednosti, za kar upravičeno sumim generatorje naključnih števil, ki ne uporabljajo ravno najboljšega algoritma za generacijo naključnih števil.

LITERATURA

1. P.H. Seaman: " On teleprocessing system design. Part VI: The role of digital simulation", IBM Systems Journal, Vol. 5, No. 3, 1966

2. M.J. Maggard and others: "GERTS III QR: A multiple resource constrained network simulation model", Management Datamatics, Vol. 5, No. 1, 1976.

3. D.V. Foster and others: "A lanquage for analysis of queuing models", Modeling & Simulation, Vol. 5, Pittsburgh (USA), 24. + 26. april 1974.

4. P.H. Seaman, R.C. Soney: "Simulating operating systems" IBM Systems Journal, Vol. 8, No.4, 1969.

5. G. Gordon: "System Simulation", Prentice-Hall, Inc., New Jersey, 1969.

6. N. Guid: "Uporaba metod množične strežbe pri analizi in načrtovanju računalniških in telekomunikacijskih mrež in sistemov", magistrsko delo, Fakulteta za elektrotehniko, Ljubljana, 1977.

7. T.J. Schriber: "Simulation using GPSS", John Wiley & Sons, New York, 1974.

TRETJE REPUBLIŠKO TEKMOVANJE SREDNJEŠOLCEV S PODROČJA RAČUNALNIŠTVA

R. REINHARDT
M. MARTINEC
R. DORN

UDK: 371.3 : 681.3

SLOVENSKO DRUŠTVO INFORMATIKA, LJUBLJANA

Povzetek. Prispevek predstavlja poročilo o tretjem republiškem tekmovanju srednješolcev iz področja računalništva, ki ga je organiziralo Slovensko društvo Informatika v aprilu 1979. V prispevku so vse naloge z rešitvami in pregled rezultatov tekmovanja.

THIRD COMPUTER SCIENCE CONTEST FOR HIGH-SCHOOL STUDENTS. The article represents a report on third Computer Science Contest. It includes the complete set of problems with their solutions and a short overwiev of contest results.

I. Uvod

Ena od rednih dejavnosti Slovenskega društva Informatika je tudi popularizacija računalništva in informatike med srednješolsko mladino. Komisija za popularizacijo računalništva je zato organizirala že tretje republiško tekmovanje srednješolcev iz področja računalništva.

Tekmovanje je bilo 21. aprila na Fakulteti za elektrotehniko v Ljubljani, udeležilo pa se ga je rekordno število tekmovalcev: 56 po prvem letu pouka in 36 po drugem letu pouka računalništva.

Pri organizaciji letošnjega tekmovanja so poleg Društva in Fakultete za elektrotehniko sodelovali še Inštitut Jožef Stefan, sodelavci koordinativne delovne skupine za izvedbo projekta Pouk računalništva v usmerjenem izobraževanju, finančno pa so tekmovanje podprli Elektrotehna - TOZD Digital, Iskra - TOZD Računalniki, Intertrade, Republiški računski center in Hotel Lev.

Tekmovanje je otvoril rektor Univerze E. Kardelja v Ljubljani prof. dr. Slavko Hodžar, tekmovalce pa so pozdravili: predsednik Slovenskega društva Informatika prof. dr. Anton P. Železnikar, dekan Fakultete za elektrotehniko prof. dr. Jernej Virant in predsednik komisije za popularizacijo računalništva Roman Dorn.

Primarni cilj tekmovanja je popularizacija računalništva, obenem pa se učenci srednjih šol seznanijo z možnostmi študija na področju računalništva. Ker večino tekmovalcev spremljajo učitelji računalništva, je tekmovanje tudi priložnost za izmenjavo izkušenj in mnenj. Zato je potekal vzporedno s tekmovanjem tudi pogovor o pouku računalništva v usmerjenem izobraževanju, računalniških poklicih, računalniški opremi za srednje šole in kadrovskih potrebah. Na tem pogovoru so se srečali predstavniki višjih in visokih šol, predstavniki izobraževalne skupnosti in uporabniki iz različnih delovnih organizacij.

Po tekmovanju so si udeleženci organizirano ogledali bližnje računalniške centre, predstavniki višjih in visokih šol iz obeh slovenskih univerz pa so jih podrobneje seznanili s študijem in učnimi načrti svojih organizacij. Sledila je razglasitev rezultatov, na kateri so prvouvrščeni tekmovalci prejeli plakete in knjižne nagrade.

II. Naloge za učence po prvem letu pouka računalništva.

Čas reševanja je 2 uri in 30 minut. Dovoljena je uporaba vse literature. Ena naloga od petih je neobvezna.

1. Napiši program, ki izpiše naslednjo tabelo števil:

```
... 0 0 0 1 0 0 0 ...
... 0 0 1 1 1 0 0 ...
... 0 1 2 3 2 1 0 ...
... 1 3 6 7 6 3 1 ...
```

V tabeli je prva vrstica sestavljena iz samih ničel, le srednje število je 1, vsako število v naslednjih vrsticah pa je enako vsoti treh nad njim ležečih števil iz prejšnje vrstice. Tabela naj se izpiše v 21 stolpcih, izpis pa se naj konča, ko so vsa izpisana števila v zadnji vrsti različna od nič.

2. Imamo tako ozek most, da se na njem dva avtomobila ne moreta srečati. Na vsakem koncu mostu je postavljen semafor in tipalo, ki pove, če pred mostom čaka kak avtomobil. Tudi sam most je opremljen z instrumentom, ki pove, če je na mostu kak avtomobil. Napiši postopek za krmiljenje semaforjev, ki bo skrbel, da nihče ne čaka po nepotrebnem in da se promet v konicah odvija izmenično (most je zelo kratek). Naprave ob mostu krmilimo z naslednjima podprogramoma in podprogramsko funkcijo:

ODPRI(str)	str je oznaka ene izmed strani mostu. ODPRI povzroči, da se na imenovani strani odpre semafor;
ZAPRI(str)	str je spet oznaka strani mostu. ZAPRI zapre semafor na imenovani strani;
TIPALO(t)	t označuje, katero tipalo bi radi vprašali, če vidi kak avto. t lahko označuje bodisi tipalo na eni izmed strani mostu, ali pa tipalo na mostu. TIPALO je funkcija, ki pove, če imenovano tipalo zaznava kakšen avtomobil. Če ga zaznava, je njena vrednost DA, sicer pa NE.

Imeni strani mostu sta A in B, ime tipala na mostu pa je M. A, B, M, DA in NE so vnaprej definirane konstante.

Primeri:

ODPRI(A)	Odpre semafor na strani A.
ZAPRI(B)	Zapre semafor na strani B.
TIPALO(M)	Ima vrednost DA, če je na mostu kak avto.
TIPALO(B)	Ima vrednost NE, če na strani B ni vozil.

3. Definirani imamo naslednji funkciji (n je naravno število):

$$s(n) = \begin{cases} n & \text{če } 0<n<10; \\ s(n \,/\, 10) + n \bmod 10 & \text{če } n>9; \end{cases}$$

Deljenje je celoštevilčno, $n \bmod 10$ pa pomeni ostanek pri deljenju n z 10.

$$p(n) = \begin{cases} n & \text{če } 0<n<9; \\ 0 & \text{če } n=9; \\ p(s(n)) & \text{če } n>9; \end{cases}$$

a) Izračunaj s(15324) in p(15324).
b) Kaj računa funkcija s (razloži).
Neobvezno:
c) *Dokaži, da za vsako naravno število n velja p(n) je ostanek pri deljenju n z 9.*

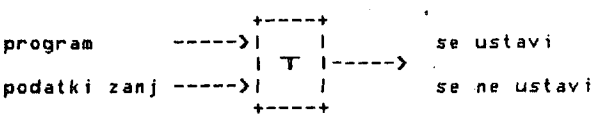
4. Neki programer sumljivih kvalitet nam je prinesel naslednji program:

```
C     program obrne podatke
      INTEGER T(100),Z
      INTEGER I,J,N
      READ(2,1)N
1     FORMAT(I5)
      READ(2,2)(T(I),I=1,N)
2     FORMAT(16I5)
      J=N
      DO 3 I=1,N
        Z=T(I)
        T(I)=T(J)
        T(J)=Z
        J=J-1
3     CONTINUE
      WRITE(3,4)(T(I),I=1,N)
4     FORMAT(1X,10I10)
      CALL EXIT
      END
```

```
program o(input,output);
  var n,i,j,z:integer;
      t:array[1..100]of
                integer;
  begin {obrni podatke}
    readln(n);
    for i:=1 to n do
      read(t[i]);
    j:=n;
    for i:=1 to n do
      begin z:=t[i];
        t[i]:=t[j];
        t[ ]:=z;
        j:=j-1
      end;
    for i:=1 to n do
      write(t[i])
  end.
```

Napiši, kaj program izpiše, če dobi naslednje podatke:
10
9 8 7 6 5 4 3 2 1 0
Ali lahko uganeš, kaj je programer hotel napisati in program popraviš?

5. Programi, ki so zapisani v nekem programskem jeziku (fortran ali pascal) se lahko pri danih podatkih ustavijo, ali pa tudi ne. Dokaži, da ne obstoja program (imenujmo ga T) zapisan v istem programskem jeziku, ki bi za vsak program in njegove podatke izračunal, ali se program ustavi ali ne.



Nasveta:

A) Predpostavi, da bi imeli tak program T. S pretvorbami, ki so izvedljive, ga spremeni v drugačen program, ki gotovo ne obstoja. Če so pretvorbe zanesljivo izvedljive, potem T ne obstoja.

B) Program, ki naj bi se po enem razmisleku ustavil in se po drugačnem razmisleku ne bi, zanesljivo ne obstoja.

III. Naloge za učence po drugem letu pouka računalništva.

(Pogoji so isti kot za tekmovelce po prvem letu pouka.)

1. n otrok se hoče loviti. Poznajo izštevanko z m besedami. Napiši program, ki pove, kateri od otrok lovi. Otroke oštevilčimo s številkami od 1 do n in začnemo izštevati pri prvem otroku. Lovi tisti, ki zadnji ostane v krogu.

2. Neko informacijo imamo natisnjeno na papirju v posebni kodi. Na papirju so izmenično črni in beli pasovi. Tanki pasovi (tako črni kot beli) pomenijo ničlo, debeli pa enico. Debeli pasovi so dvakrat debelejši od tankih. Napiši postopek, ki bo izpisal zaporedje ničel in enic, *ki je zakodirano na papirju.* S čitalnikom se premikamo po papirju s konstantno hitrostjo. Za ugotovitev hitrosti imamo pred informacijo na papirju en tanek črn pas. Za čitanje imamo na voljo naslednji funkciji:

BARVA	pove, kakšna barva je trenutno pod glavo čitalnika.
ČAS	*nam pove čas v milisekundah, ki je* pretekel od začetka programa.

Primer:

Kakšen bi bil algoritem, ki bi se prilagajal spremenljivi realni hitrosti odčitavanja?

3. Za nenegativna števila n imamo definirane funkcije f, g in h takole:

$$f(n) = \begin{cases} 0 & \text{če } n=0 \\ 1 & \text{če } n=1 \\ f(n-1)+f(n-2) & \text{če } n>1 \end{cases}$$

$$h(a,b,n) = \begin{cases} a & \text{če } n=0 \\ b & \text{če } n=1 \\ h(b,a+b,n-1) & \text{če } n>1 \end{cases}$$

$$g(n) = h(0,1,n)$$

a) Izračunaj f(5) in g(5).
b) Pokaži, da za vsak a, b, c, d in e velja: h(a,b,e)+h(c,d,e)=h(a+c,b+d,e) .
c) Pokaži, da za vsak nenegativen cel n velja f(n)=g(n).

4. Neki programer sumljivih kvalitet nam je prinesel naslednji program:

```
      INTEGER A(10,10),I,J,Z
      READ(2,1)((A(I,J),
     -       J=1,10),I=1,10)
1     FORMAT(10I5)
      DO 3 I=1,10
        DO 2 J=1,10
          Z=A(I,J)
          A(I,J)=A(J,I)
          A(J,I)=Z
2       CONTINUE
3     CONTINUE
      WRITE(3,4)((A(I,J),
     -       J=1,10),I=1,10)
4     FORMAT(1X,10I10)
      CALL EXIT
      END
```

```
program t(input,output);
  var i,j,z:integer;
      a:array[1..10,1..10]
        of integer;
  begin
    for i:=1 to 10 do
      for j:=1 to 10 do
        read(a[i,j]);
    for i:=1 to 10 do
      for j:=1 to 10 do
        begin z:=a[i,j];
              a[i,j]:=a[j,i];
              a[j,i]:=z
        end;
    for i:=1 to 10 do
      begin
        for j:=1 to 10 do
          write(a[i,j]);
        writeln
      end
  end.
```

Izmisli si primerne podatke za ta program in zapiši podatke in rezultate. Ali lahko uganeš, kaj je imel programer v mislih in popraviš program tako, kot misliš, da bi moral delovati?

5. Ista naloga kot 5. naloga za tekmovalce po 1. letu pouka.

IV. Rezultati prvih petnajstih tekmovalcev v vsaki skupini

po prvem letu pouka računalništva

Mesto	št. točk	Tekmovalec
01	111	Mirjam Lešnik, I. gimnazija Ljubljana - Bežigrad
02	104	Anton Verbovšek, I. gimnazija Ljubljana - Bežigrad
03	103	Uroš Kunaver, I. gimnazija Ljubljana - Bežigrad
04	089	Marjan Horvatič, Gimnazija Novo mesto
05	075	Ester Zimic, Šc "Vojvodina" - gimnazija Tolmin
06	074	Bojan Cestnik, I. gimnazija Ljubljana - Bežigrad
07	073	Andrej Brodnik, I. gimnazija Ljubljana - Bežigrad
07	073	Tomi Dolenc, I. gimnazija Ljubljana - Bežigrad
09	072	Dario Medoš, Gimnazija Koper
10	069	Nada Žagar, Gimnazija in ekonomska šola, Trbovlje
11	065	Gorazd Planinšič, I. gimnazija Ljubljana - Bežigrad
12	061	Jana Padežnik, Gimnazija Miloša Zidanška - Maribor
13	059	Miloš Požar, Gimnazija Nova Gorica
14	057	Simona Jaklič, I. gimnazija Ljubljana - Bežigrad
14	057	Metod Purgar, Center srednjih šol - Jesenice

po drugem letu pouka računalništva

Mesto	št. točk	Tekmovalec
01	083	Mark Pleško, VII. gimnazija Vič - Ljubljana
02	081	Kazimir Gomilšek, Gimnazija Miloša Zidanška - Maribor
03	077	Matjaž Lampe, I. gimnazija Ljubljana - Bežigrad
04	069	Cveto Gregorc, I. gimnazija Ljubljana - Bežigrad
05	064	Milan Bizant, Gimnazija Ljubljana - Šentvid
06	054	Darko Hanžel, I. gimnazija Ljubljana - Bežigrad
07	053	Janez Bonča, I. gimnazija Ljubljana - Bežigrad
08	051	Srečko Starič, Gimnazija Novo mesto
09	047	Rado Juvan, Gimnazija-ekonomska šola, Trbovlje
10	046	Marko Ahčan, VII. gimnazija Vič - Ljubljana
10	046	Branko Premzel, Tehniška elektro, strojna in tekstilna šola Maribor
12	043	Cveto Brkič, Gimnazija Novo mesto
13	042	Borut Stariha, Prva gimnazija, Maribor
14	039	Nada Liben, Šolski center Idrija - gimnazija Jurija Vege
14	039	Miran Ulbin, Gimnazija Miloša Zidanška - Maribor

V. Rešitve nalog za učence po prvem letu pouka računalništva

1. Program za izpis tabele najprej v fortranu, nato pa v pascalu:

```
C     program tab
        INTEGER X(21),Y(21),I,P
C       Sestavimo prvo vrstico
        DO 1 I=1,21
          X(I)=0
1       CONTINUE
        X(11)=1
C       Dokler ni prvo število v vrsti različno
C       od 0 ponavljamo
```

```
2       CONTINUE
C         Izpis vrstice
          WRITE(3,3)(X(I),I=1,21)
3         FORMAT(2116)
C         P je prvo število v izpisani vrsti
          P=X(1)
C         Izračunamo novo vrsto ...
          Y(1)=X(1)+X(2)
          Y(21)=X(20)+X(21)
          DO 4 I=2,20
            Y(I)=X(I-1)+X(I)+X(I+1)
4         CONTINUE
C         ... in jo prepišemo v staro
          DO 5 I=1,21
            X(I)=Y(I)
5         CONTINUE
        IF(P.EQ.0)GO TO 2
      CALL EXIT
      END
```

```
program tab(output);
  const mx=21;
  var x,y:array[1..mx]of integer;
      i:integer;
  begin
    { sestavimo prvo vrstico }
    for i:=1 to mx do
      x[i]:=0;
    x[mx div 2 + 1]:=1;
    { Dokler ni prvo število v vrsti različno
       od 0 ponavljamo }
    page(output);
    repeat
      { izpisemo vrsto  }
      for i:=1 to mx do
        write(x[i]:6);
      writeln;
      { p je prvo stevilo v ze izpisani vrsti }
      p:=x[1];
      { izracunamo novo vrsto ... }
      y[1]:=x[1]+x[2];
      y[mx]:=x[mx-1]+x[mx];
      for i:=2 to mx-1 do
        y[i]:=x[i-1]+x[i]+x[i+1];
      { ... in jo prepisemo v staro }
      for i:=1 to mx do
        x[i]:=y[i];
    until p<>0;
  end.
```

2. Rešitev zapišemo (skoraj) v pascalu.
   Iz pascala se postopek tako jasno vidi, da je zapis v slovenščini nepotreben.

```
program most(output);
  type prisoten=(DA,NE);
       točka=(A,B,M);

  { ukazi za krmiljenje naprav na mostu }
  procedure odpri(x: točka); extern;
  procedure zapri(x: točka); extern;
  function tipalo(x: točka): prisoten; extern;

  procedure prehod(x,y: točka);
    { če je na strani x kakšen avto, potem
       enega spusti čez most. }
    begin
      repeat until tipalo(M)=NE;
      if tipalo(x)=DA then
        begin
          zapri(y);
          odpri(x);
          repeat until tipalo(M)=DA
        end
    end;

  begin {most}
    zapri(A); zapri(B);
    repeat
      prehod(A,B);
      prehod(B,A)
    until false
  end {most}.
```

3.

a) $s(15324) = s(1532)+4 = s(153)+2+4 = s(15)+3+2+4 = s(1)+5+3+2+4 = 1+5+3+2+4 = 15$

   $p(15324) = p(s(15324)) = p(15) = p(s(15)) = p(6) = 6$

b) s izračuna vsoto cifer (v desetiškem zapisu) svojega argumenta. Utemeljitev: vsota cifer enomestnega števila (t. j. števila, ki je manjše od 10) je to število samo.

   Vsoto cifer večmestnega števila pa dobimo tako, da zadnji cifri prištejemo vsoto cifer tega števila brez zadnje cifre. n mod 10 je očitno zadnja cifra števila n v desetiškem sestavu, n / 10 (celoštevilčno) pa je število n brez zadnje cifre.

c) Trditev očitno velja za n<10. Vzemimo poljuben k>9. Naj trditev velja za vse n<k. Poglejmo, če velja tudi za n=k. Ker je k>9, velja

$$p(k) = p(s(k)).$$

Ker je k>9, je s(k) (vsota cifer v številu k) gotovo manjša od k, zato lahko uporabimo hipotezo, da trditev velja za vse n<k. p(s(k)) je torej ostanek pri deljenju s(k) z 9. Pokažimo, da imata s(k) in k pri deljenju z 9 isti ostanek. Če so a[i], i=0, 1, ... m cifre števila k, potem velja

$$k = a[0]*10^0 + a[1]*10^1 + \ldots + a[m]*10^m$$

in

$$s(k) = a[0] + a[1] + \ldots + a[m].$$

k in s(k) imata enak ostanek pri deljenju z 9, če je njuna razlika deljiva z 9.

$$k-s(k) = a[0]*(10^0-1) + a[1]*(10^1-1) + \ldots + a[m-1]*(10^{m-1}-1) + a[m]*(10^m-1)$$

Vsa števila oblike (10^i-1) so očitno deljiva z 9, torej je tudi razlika k-s(k) deljiva z 9, zato pa imata k in s(k) isti ostanek pri deljenju z 9. Indukcija opravi svoje in trditev je dokazana.

4. Program izpiše tisto, kar je prečital:

9 8 7 6 5 4 3 2 1 0

Iz strukture programa se vidi, da je programer verjetno hotel izpisati zaporedje v obrnjenem vrstnem redu. V ta namen bi morala teči zanka, ki elemente zaporedja menja, le do polovice zaporedja. Popravljen program bi bil:

```
. . .                  |    . . .
M=N/2                  |    for i:=1 to n div 2 do
DO 3 I=1,M             |      begin z:=t[i];
  Z=T(I)               |        t[i]:=t[j];
  T(I)=T(J)            |        t[j]:=z;
. . .                  |    . . .
```

(Druga ugibanja so seveda prav tako dobra rešitev, le program moramo pravilno popraviti.)

5. Predpostavimo, da T obstaja. Naj bo T funkcija, njena argumenta sta program in podatki zanj, njena vrednost pa je true, če se program ustavi in false, če se ne ustavi. Sestavimo s T naslednji program:

```
procedure Q(prog,podat);
  begin
    while T(prog,podat) do;
    write("OK");
  end;
```

Vprašajmo se, kaj se zgodi, če poskušamo izračunati

```
Q(Q,p1),
```

kjer je p1 nek program, ki ne potrebuje podatkov. Če bi se Q ustavil, bi se to lahko zgodilo samo če je T(Q,p1) = false, kar pomeni, da bi moral T trditi, da se Q ne ustavi. Če pa se Q ne bi ustavil, se to lahko zgodi samo, če je T(Q,p1) = true, kar pomeni, da bi T moral trditi, da se Q ne ustavi. Niti prvo niti drugo se ne more zgoditi, zato tak Q ne more obstajati. Ker pa v Q vse razen T obstaja, T ne obstaja. Q. E. D.

VI. Rešitve nalog za učence po drugem letu pouka računalništva

1. Program za izštevanje najprej v fortranu nato pa v pascalu.

```
C     Program izštevanka
      INTEGER OTROCI(30),N,M,I,J,P
C     Čitanje
      READ(2,1)N,M
1     FORMAT(2I2)
C     vsi otroci so v krogu
C     predpostavljamo 1<N<31, M>0
      DO 2 I=1,N
        OTROCI(I)=1
2     CONTINUE
C     začnemo s prvim
      P=0
C     N-1 jih mora izpasti
      DO 5 I=2,N
C       vsakokrat moramo šteti do M
        DO 4 J=1,M
C         izpadlih otrok ne smemo šteti
3         CONTINUE
            P=P+1
C           štejemo v krogu (za n-tim sledi
C                            prvi otrok)
            IF(P.GT.N)P=1
          IF(OTROCI(P).EQ.0)GO TO 3
4       CONTINUE
C       otrok P izpade
        OTROCI(P)=0
5     CONTINUE
C     poiščemo edinega neizpadlega otroka
      P=1
6     IF(OTROCI(P).NE.0)GO TO 7
        P=P+1
      GO TO 6
7     CONTINUE
C     izpis
      WRITE(3,8)N,M,P
8     FORMAT('1Število otrok',I10/
     -       ' dolžina izštevanke',I5/
     -       'Olovi otrok',I13)
      CALL EXIT
      END
```

```
program izstevanka(input,output);
  const mx=30;
  var m,n,i,j,p:integer;
      otroci:array[1..mx]of boolean;
  begin
    readln(n,m);  { čitanje }
    { vsi otroci so v krogu }
    { predpostavljamo 1<n<=mx, m>0
    for i:=1 to n do otroci[i]:=true;
    { začnemo s prvim }
    p:=0;
    { n-1 otrok mora izpasti }
    for i:=1 to n-1 do
      begin
        { vsakokrat moramo šteti do m }
        for j:=1 to m do
          { izpadlih otrok ne smemo šteti }
          repeat
            p:=p+1;
            { štejemo v krogu,
                 n-temu sledi prvi }
            if p>n then p:=1;
          until otroci[p];
        { otrok p izpade }
        otroci[p]:=false
      end;
    { poiščemo edinega neizpadlega otroka }
    p:=1;
    while not otroci[p] do p:=p+1;
    { izpis }
    page(output);
    writeln(' Število otrok',n:10);
    writeln(' dolžina izštevanke',m:5);
    writeln;
    writeln(' lovi otrok',p:13)
  end.
```

2. Postopek zapišemo (skoraj) v pascalu, kar ne more škoditi preglednosti.

```
program barcode(output);
  type čb=(črna,bela);
  var t,t0,zakas: integer;
      b: čb;
      podat: 0..1;

  function barva: čb; extern;
  function čas: integer; extern;

  begin
    { ignoriramo belino vse do začetka }
    repeat until barva=črna;
    { izmerimo širino prvega črnega pasu }
    t:=čas;
    repeat until barva=bela;
    t0:=čas-t; { t0 je čas za prelet ničle }
    { pravo čitanje se začenja }
    repeat
      t:=čas;
      b:=barva;
      { počakamo, da čitalnik pride do
         spremembe barve }
      repeat until barva<>b;
      zakas:=čas-t;
      { zakas je čas potovanja čez zadnji pas }
      { odločimo se, ali je to 0 ali 1 }
      if zakas>1.5*t0 then
        begin
          podat:=1;
          { popravimo vzorčni čas }
          t0:=zakas/2
        end
      else
        begin
          podat:=0;
          { popravimo vzorčni čas }
          t0:=zakas
        end;
      write(podat)
    until false
  end.
```

Osnovni postopek, ki bi deloval le pri konstantni hitrosti, ne potrebuje popravljanja vzorčnega časa. V program bi lahko vgradili še test za konec podatkov (npr. če se barva zelo dolgo ne spremeni), vendar tega naloga ne zahteva.

3.

a) f(5) = f(4)+f(3) = f(3)+f(2)+f(3) =
f(2)+f(1)+f(2)+f(3) =
f(1)+f(0)+f(1)+f(2)+f(3) =
1+0+1+f(1)+f(0)+f(3) = 1+0+1+1+0+f(2)+f(1) =
1+0+1+1+0+f(1)+f(0)+f(1) = 1+0+1+1+0+1+0+1 =
5,

kar se seveda lažje izračuna, če zapišemo tabelo:

n	0	1	2	3	4	5	6	7...
f(n)	0	1	1	2	3	5	8	13...

g(5) = h(0,1,5) = h(1,1,4) = h(1,2,3) = h(2,3,2) = h(3,5,1) = 5

b) Če so a, b, c in d, e poljubna nenegativna cela števila, je treba dokazati, da velja

$$f(a,b,e)+f(c,d,e) = f(a+c,b+d,e).$$

Trditev bomo dokazali z indukcijo:
- Pri e=0 in e=1 trditev očitno velja.
- Naj bo k poljubno naravno število večje od 1. Vzemimo, da trditev velja za vse e<k. Dokažimo, da tedaj velja tudi pri e=k.

h(a,b,k)+h(c,d,k) = h(b,a+b,k-1)+h(d,c+d,k-1) = h(b+d,a+b+c+d,k-1),

h(a+c,b+d,k) = h(b+d,a+c+b+d,k-1).

S tem je trditev dokazana.

c) Dokazati moramo, da za vsak nenegativen cel n velja

$$f(n) = g(n).$$

Tudi to trditev bomo dokazovali z indukcijo. Veljavnost je očitna za n med 0 in 2. Spet naj bo k neko naravno število večje od 2 in predpostavimo, da trditev velja za vse n<k. Za dokaz, da trditev velja tudi pri n=k, bomo potrebovali trditev pod točko b):

f(k) = f(k-1)+f(k-2) = h(0,1,k-1)+h(0,1,k-2) = h(1,1,k-2)+h(0,1,k-2) = h(1,2,k-2)

g(k) = h(0,1,k) = h(1,1,k-1) = h(1,2,k-2)

Prepričali smo se, da tako f kot g računata fibonaccijeva števila. Opazimo lahko, da je pri tem g v primerjavi z f mnogo učinkovitejša.

4. Program prečita celoštevilčno matriko velikosti 10x10. Izpiše jo nespremenjeno. Programer je hotel verjetno matriko transponirati, kar bi dosegel, če bi program popravil takole:

```
. . .                          |    . . .
DO 3 I=1,10                    |    for i:=1 to 10 do
  DO 2 J=1,I                   |      for j:=1 to i-1 do
    Z=A(I,J)                   |        begin
. . .                          |    . . .
```

Še lažje pa je odstraniti zanke in zamenjati indekse pri izpisu:

```
  . . .                        | . . .
  READ(2,1)((A(I,J),J=... |read(a[i,j])|
1 FORMAT ...                   |{ zanki za izpis }
  WRITE(3,4)((A(J,I),J=...|      write(a[j,i]);
  . . .                        | . . .
```

(Veljavno rešitev dobimo tudi, če si mislimo, da je programer hotel kaj drugega, le program je treba pravilno popraviti.)

5. Glej 5. nalogo pri nalogah za učence po enem letu pouka.

TABELA 1: Število udeležencev na vseh treh tekmovanjih

	1977	1978	1979
po 1. letu	?	52	56
po 2. letu	?	27	36
skupaj	47	79	92

Opomba: V letu 1977 tekmovanje ni bilo ločeno na dve skupini.

TABELA 2: Število tekmovalcev in povprečen uspeh po šolah

Šola	Št. tekmovalcev in povprečno št. točk po 1. letu		po 2. letu		Št. prijavljenih tekmovalcev
Center srednjih šol - Jesenice	2	51.00	2	7.50	6
CSŠ Črnomelj - gimnazija splošne smeri	1	18.00	0	---	3
Ekonomska srednja šola Črnomelj	2	21.50	0	---	2
Elektrotehniška srednja šola Krško	1	19.00	0	---	2
Elektrotehniška šola v Ljubljani	7	24.29	0	---	20
Gimnazija "Boris Ziherl" Škofja loka	4	9.50	4	14.50	10
Gimnazija Kočevje	2	26.50	0	---	3
Gimnazija Koper	5	39.00	1	34.00	12
Gimnazija Ljubljana - Šentvid	2	35.00	2	41.50	8
Gimnazija Miloša Zidanška - Maribor	2	51.50	3	48.00	6
Gimnazija Nova Gorica	4	45.00	0	---	5
Gimnazija Novo mesto	3	54.33	4	33.50	10
Gimnazija Trbovlje	4	42.25	1	47.00	7
I. gimnazija Ljubljana - Bežigrad	8	82.50	5	57.40	13
Prva gimnazija Maribor	0	---	6	26.50	6
Šc Idrija - gimnazija Jurija Vege	0	---	1	23.00	1
Šc "Vojvodina" - gimnazija Tolmin	1	75.00	1	39.00	2
Tehniška el. str. in tekstilna šola Maribor	0	---	3	23.33	3
Tehniške strojna in elektro šola, Trbovlje	7	20.00	0	---	8
Tehniška tekstilna šola, Kranj	1	40.00	0	---	1
VII. gimnazija Ljubljana Vič	0	---	3	49.67	3
Skupaj	56	40.00	36	34.50	137
Število šol	17		13		21
Število gimnazij	11		11		14
Število tehniških srednjih šol	6		2		7

GRAFIČKI TERMINAL S PRIKAZOM SLIKE NA TV MONITORU

A. BARIĆ

UDK: 681.327.11

ZAVOD ZA ELEKTRONIKU
ELEKTROTEHNIČKI FAKULTET, ZAGREB

U članku su prikazane mogućnosti konstrukcije grafičkog terminala. Opisana je struktura grafičkog terminala s iscrtavanjem slike pomoću niza točaka. Analizirane su neke mogućnosti uštede memorije za smještaj prikazne datoteke. Opisan je grafički terminal izradjen na Zavodu za elektroniku Elektrotehničkog fakulteta u Zagrebu.

GRAPHICS TERMINAL WITH A TV MONITOR The article reviews the possibilities in the graphics terminal design. The structure of the terminal using raster-scan technique is presented. Some possibilities for minimizing the display file are discussed. The graphics terminal which was designed in Institute for Electronics, Faculty of Electrical Engineering in Zagreb is also described.

1. UVOD

Grafički prikaz rezultata dobivenih izvodjenjem programa često je pregledniji i zato korisniji od ispisa s dugim nizovima brojčanih vrijednosti. To je jedan od glavnih razloga upotrebe grafičkih stanica kao ulazno - izlaznih jedinica računala. Osnovno funkcionalno obilježje grafičkih terminala je pretvorba digitalnih podataka primljenih od računala u odgovarajuću sliku na zaslonu. Prema načinu prikaza slike grafički se terminali dijele na repetitivne i one koji pamte sliku na zaslonu optičke stanice. Kod repetitivnih stanica pamte se podaci o slici u dodatnoj memoriji smještenoj u terminalu, dok kod stanica s pamćenjem ulogu memorije ima posebno izradjena katodna cijev. Razvoj tehnologije omogućio je izradu jeftinih memorijskih elemenata što predstavlja osnovni preduvjet za ekonomski opravdanu izgradnju repetitivnih grafičkih terminala za široku primjenu. Glavne prednosti, radi kojih se danas preferira upotreba repetitivnih grafičkih stanica, su: mogućnost selektivnog mijenjanja sadržaja dijelova zaslona, generiranje dinamičke slike i jednostavnost realizacije optičkih ulaznih naprava kao što je npr. svjetlosno pero.

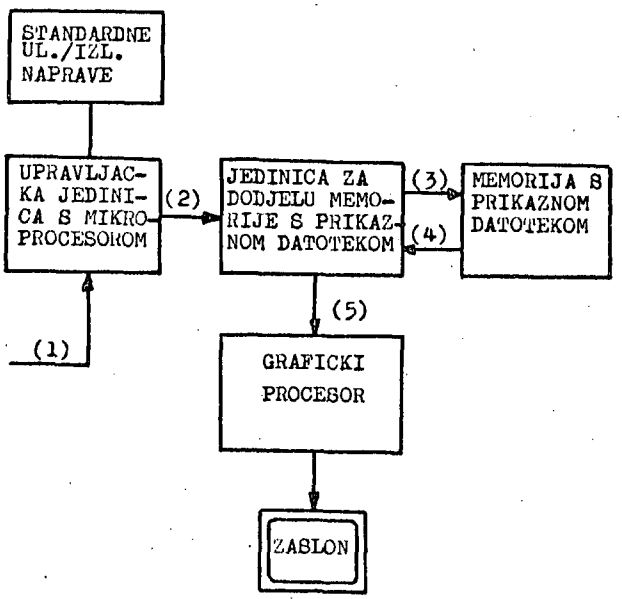
S obzirom na zahtjeve za niskom proizvodnom cijenom i jednostavnošću konstrukcije, danas se često kao prikazna naprava grafičkog terminala upotrebljava TV monitor. To uvjetuje poseban način pamćenja i prikaza slike. Konstrukcija ovakvog grafičkog terminala je predmet razmatranja u ovome članku.

Kod starijih tipova repetitivnih grafičkih stanica slika se iscrtava tako da se u slijedu iscrtavaju vektori od kojih se slika sastoji. Učestalost iscrtavanja cijele slike je u tom slučaju takva da se, s obzirom na tromost oka, dobije prividno mirna slika. Upotreba TV monitora kao prikazne naprave uvjetuje drugačiji način iscrtavanja slike. Da bi se sliku moglo prikazati na zaslonu TV monitora, potrebno ju je rastaviti u nizove točaka od kojih se pojedini redak slike sastoji. Cijelu sliku se iscrtava ukupno 25 puta u sekundi, tj. prikazuje se kao pedeset poluslika što odgovara CCIR propisima. Pri tome se podaci o slici pamte tako da je svakoj pojedinoj točki na zaslonu pridijeljen jedan ili više bitova u prikaznoj datoteci, ovisno o tome s koliko razina intenziteta ili s koliko se boja želi crtati. Jednostavnim se računom dobiva da je za crtanje na zaslonu dimenzija 512 x 512

točaka potrebno 32 K memorije s 8 - bitnim riječima, pri čemu pojedina točka može biti ili osvijetljena ili zatamnjena. Za crtanje s tri osnovne boje potrebno je tri puta više memorije što iznosi 96 K.

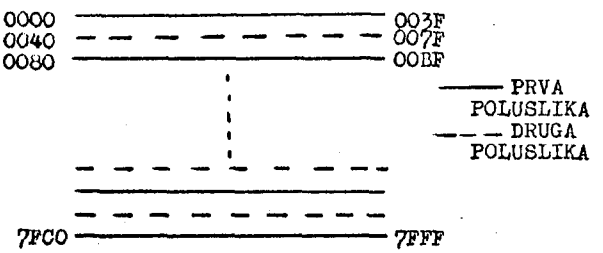
2. STRUKTURA GRAFICKOG TERMINALA

Na slici 1. prikazana je načelna struktura grafičkog terminala.



Sl. 1.

Formatizirani grafički podaci kojima je opisan vektor dolaze od računala (1). Upravljačka ih jedinica obradjuje i pretvara u oktete koje je potrebno upisati u prikaznu datoteku. Jedinica za dodjelu memorije s prikaznom datotekom omogućuje tada prolaz podataka u smjerovima označenim s (2) i (3). U sve preostalo vrijeme podatke iz prikazne datoteke čita grafički procesor i prikazuje ih kao sliku na zaslonu (smjerovi (4) i (5)). Na slici 2. prikazana je korespondencija memorijskih adresa prikazne datoteke i položaja pripadnih točaka na zaslonu dimenzija 512 x 512 točaka. Vidljivi dio slike smješten je u sredini zaslona monitora koji teoretski ima dimenzije 830 x 625 točaka. Izbor površine za prikaz od 512 x 512 točaka pojednostavljuje adresiranje



Sl. 2.

memorije s prikaznom datotekom. Petnaest bitni adresni registar grafičkog procesora sastavljen je na način prikazan na slici 3.

ADRESNI BITOVI

0 do 5	.....	izlazi 3 do 8 horizontalnog brojila
6	.....	izlaz bistabila poluslike
7 do 14	.....	izlazi 0 do 7 vertikalnog brojila

Sl. 3.

Otipkavanje jednog retka na zaslonu izvodi se tako da se nakon svakih osam impulsa ritma sinhrono paralelno napuni posmačni registar, koji se zatim serijski prazni za vrijeme slijedećih osam impulsa frekvencije 13 MHz. Frekvencija ritma od 13 MHz daje ukupno 25 cijelih, odnosno 50 poluslika, što zadovoljava propise za generiranje televizijske slike. Sinhronizacija vertikalnog i horizontalnog oscilatora, te generiranje potisnih impulsa izvodi se odgovarajućim dekodiranjima na izlazima horizontalnog i vertikalnog brojila.

Opisani zadaci jedinice za dodjelu memorije te grafičkog procesora definiraju i njihovu sklopovsku izvedbu. U svrhu projektiranja upravljačke jedinice grafičkog terminala, potrebno je razmotriti daljnje zahtjeve koji se postavljaju na njezine funkcije.

Za povezivanje računala s udaljenim terminalom najčešće se upotrebljava serijski sinhroni prijenos podataka. Poruke koje izmjenjuju računalo i grafički terminal sadrže, uz

standardne dijelove, i grafičke podatke. Ovi se podaci sastoje od grafičkih kontrolnih znakova i pripadnih podataka kojima se opisuju koordinate ili geometrijska svojstva vektora koje treba nacrtati. Standardni dijelovi poruka formiraju se prema skupu pravila koja čine komunikacijski protokol, pa stoga terminal mora imati mogućnost analize i sinteze poruka. Dodatni zahtjev koji se postavlja na grafički terminal jest pretvorba grafičkih podataka u prikladan zapis u prikaznoj datoteci. Kad terminal kao prikaznu napravu koristi TV monitor, onda zapis u prikaznoj datoteci mora omogućiti iscrtavanje slike rastavljene u nizove točaka. U načelu bi računalo moglo slati podatke o slici koja je već rastavljena u nizove točaka, ali bi to onda značilo dodatno opterećenje računala i znatno opterećenje komunikacijskog kanala prema grafičkom terminalu. Navedeni zadaci grafičkog terminala sugeriraju ugradnju mikroprocesora u upravljačku jedinicu, čime se posao konverzije grafičkih podataka prepušta terminalu, a mikroprocesor takodjer obavlja nadzor nad razmjenom poruka u skladu s komunikacijskim protokolom.

3. MOGUĆNOSTI UŠTEDE MEMORIJE ZA POHRANJIVANJE PRIKAZNE DATOTEKE

Izradi eksperimentalne grafičke stanice prethodila je analiza mogućnosti uštede memorije za pohranjivanje prikazne datoteke. Osnovna premisa pri razmatranju mogućnosti smanjenja prikazne datoteke jest činjenica da slika često sadrži neiscrtane površine, što odgovara velikim područjima koja su u prikaznoj datoteci ispunjena samo nulama. Prikladnim rastavljanjem slike u dijelove,mogu se uvodjenjem posebnih kodova pamtiti položaji zatamnjenih dijelova zaslona uz minimalni utrošak memorije. Rastavljanje slike može se provesti na razne načine. Iscrtavanje slike pomoću niza točaka sugerira rastavljanje slike u retke i kodiranje unutar retka. Kodiranje se obavlja tako da prikazna datoteka sadrži podatke o tome koliko se zatamnjenih točaka nalazi u retku prije prve osvijetljene, odnosno koliko se osvijetljenih točaka nalazi u slijedu. Druga ideja za rastavljanje slike potiče od načina prikaza na zaslonu monitora kod alfanumeričkih terminala, gdje je zaslon podijeljen na pravokutne ili kvadratne dijelove. Kodovi znakova koji se prikazuju na zaslonu pamte se u ovom slučaju u radnoj memoriji, a pripadni podaci za iscrtavanje smješteni su u ispisnoj memoriji, kojoj jedan dio adresne riječi predstavlja kod znaka, a drugi dio položaj na zaslonu. Ako se u ispisnu memoriju postave segmenti vektora, može se metoda za prikaz alfanumeričkih znakova koristiti i za grafički prikaz. Pri tome se prije iscrtavanja mora dopustiti modifikacija i kombiniranje podataka s osnovnim segmentima, tj. omogućiti maskiranje, pomak, rotaciju i preklapanje osnovnih segmenata.

Analizirajući sklopovsku složenost grafičkog procesora i algoritme nadopune prikaznih datoteka kod raznih metoda uštede memorije, došlo se do zaključka da su te metode ekonomski neopravdane za izgradnju grafičkog terminala za široku primjenu. Osim činjenice da metode uštede memorije funkcioniraju samo za relativno jednostavne slike, cijena i izrada grafičkog procesora kod metoda uštede premašuju današnju cijenu memorijskih elemenata od kojih je načinjena memorija s prikaznom datotekom, kreiranom po načelu jedan bit za jednu točku.

4. OPIS IZGRADJENE GRAFICKE STANICE

U svrhu verifikacije provedenih razmatranja izgradjena je na Zavodu za elektroniku Elektrotehničkog fakulteta u Zagrebu pokusna grafička stanica s iscrtavanjem slike pomoću niza točaka. Stanica posjeduje mogućnost promjene rastera na zaslonu, tako da jednom bitu u prikaznoj datoteci pripadaju 1, 4, 16,ili 64 točke zaslona. Promjenljiva koordinatna mreža na zaslonu omogućuje da cijena ugradjene memorije prati zahtjeve za kvalitetom prikaza te da odgovara veličini upotrebljenog zaslona. Osim toga pokazalo se da promjenljivost rastera znatno olakšava ispitivanje uredjaja. Naime, uz najfiniji raster teško je uočiti eventualne nepravilnosti pri iscrtavanju. Nadalje je kod ove grafičke stanice realizirana mogućnost rada s TV monitorom u boji, odnosno s tri odvojena crno-bijela monitora. U tu svrhu prikazna datoteka sastoji se od tri memorijske cjeline koje se nalaze na istim adresama. Mikroprocesor Fairchild F8, koji je ugradjen u grafičku stanicu, pomoću dva bita ulazno - izlaznog registra odredjuje na koju će memoriju doći signal za pisanje, čime je riješen problem demultipleksiranja pri pisanju u

prikaznu datoteku. Problem čitanja riješen je pomoću multipleksora koji je upravljan istim bitovima ulazno - izlaznog registra. Prilikom iscrtavanja, grafički procesor adresira istovremeno sve tri prikazne datoteke iz kojih se izlazni podaci dovode odvojeno na ulaze posmačnih registara. Izlazi tih registara vode se, nakon miješanja s potisnim impulsima, na upravljanje elektronskim topovima. Složeni sinhronizacijski impuls dobiva se kombiniranjem vertikalnog i horizontalnih sinhroimpulsa i dovodi se na odgovarajući ulaz monitora.

Trenutno implementirana programska podrška sadrži rutine za kontrolu grafičkih podataka i njihovu konverziju, pri čemu se koristi algoritam za generiranje dužina napravljen prema načelima rada sklopa BRM (binary rate multiplier).

Cijeli se upravljački uredjaj grafičke stanice sastoji od 64 integrirana sklopa niskog i srednjeg stupnja integracije, te sklopova Fairchild 3850, 3851 i 3853 i memorijskih modula s integriranim sklopovima 2102A. Za ispitivanje uredjaja korišteni su modificirani TV prijemnici tvornice ISKRA (crno-bijeli) i RCA (kolor).

5. ZAKLJUCAK

Upotreba grafičkih terminala nailazi na sve šire primjene, što rezultira povećanim interesom za njihovu jeftinu i jednostavnu izgradnju. Analize mogućnosti izgradnje grafičkog terminala s iscrtavanjem slike pomoću niza točaka, te zapažanja kod rada s pokusnom izvedbom terminala pokazala su da se uz danas dostupne integrirane komponente isplati konstruirati grafičke stanice koje kao izlaznu napravu koriste TV monitor. Daljnju prednost kod izgradnje ovakvih terminala predstavlja i pojava integriranih grafičkih procesora, što pojednostavljuje konstrukciju i smanjuje njihovu cijenu.

6. LITERATURA

Newman,W.M.: "Trends in Graphic Display Design", IEEE Transaction on Computers, Vol. C25, No. 12. December 1976.

IPC Science and Technology Press; "Raster-Scan Colour Graphic Display", Reprinted from Computer Aided Design, Vol. 9., No. 4., October 1977.

Jordan,B.W.,Barett,R.C.: "A Cell Organised Raster Display for Line Drawings", Communications of the ACM, No. 2., Vol. 17., February 1974.

Laws,B.A.:"A Gray-Scale Graphic Processor Using Run-Length Encoding", Proceedings of the Conference on Computer Graphics, Pattern Recognition & Data Structure, May 14-16,1975.

Baskett,F.,Shustek,L.: "The Design of a Low Cost Video Graphics Terminal", SLAC PUB-1715, STAN-CS-75-546,February 1976.

RAZISKOVALNE NALOGE, PRIJAVLJENE NA RSS V LETU 1979

V tej rubriki objavljamo kratke povzetke raziskovalnih nalog, ki jih financira Področna raziskovalna skupnost za avtomatiko, računalništvo in informatiko, ki so s področja računalništva in informatike.

Naslov naloge: Adaptivni digitalni sistemi

Projekt: Računalniška tehnika in proizvodnja

Nosilec naloge: Jernej Virant, Fakulteta za elektrotehniko, Ljubljana

Program raziskave:

- analiza adaptivnih algoritmov
- analiza modelov učenja
- teorija modularnosti in univerzalni logični moduli
- iskanje modela učenja, katerega osnova je v univerzalnih logičnih modulih
- snovanje adaptivnih digitalnih sistemov
- ilustrirani primeri.

Naslov naloge: Metode umetne inteligence v informacijskih sistemih

Projekt: Informacijski sistemi

Nosilec naloge: Ivan Bratko, Institut "Jožef Stefan", Ljubljana

Program raziskave:

- sinteza metode za računalniško predstavljanje kompleksnega znanja, ki bi bila primerna za uporabo v logično kompleksnih informacijskih sistemih
- razvoj algoritmov za računalniško manipuliranje s strukturiranim znanjem, izvajanjem implicitno vsebovanih informacij ter diskretno optimizacijo
- dopolnjevanje in vzdrževanje programske opreme za logično zahtevno nenumerično programiranje: konfiguriranje operacijskega sistema UNIX za konfiguracijo računalnika PDP-11 na IJS, adaptacija in instalacija prevajalnikov za jezike LISP in POP-2 na PDP-11, adaptacija prevajalnika za jezik PROLOG na novem računalniku ljubljanske univerze, če bo izbrani računalnik to omogočal.

Naslov naloge: Metode množične strežbe pri analizi in načrtovanju teleinformacijskega sistema

Projekt: Sistem daljinskega nadzora in upravljanja

Nosilec naloge: Niko Guid, Iskra-Industrija za avtomatiko TOZD Procesna tehnika, Ljubljana

Program raziskave:

- študij teorije množične strezbe in metod, ki izhajajo iz nje,
- študij simulacijskega jezika (Odločili smo se za simulacijski jezik GPSS, ki je instaliran na računalniku CDC - Cyber na RRC. To je najbolj razširjen jezik za simulacijo diskretnih sistemov v ZDA. Z njim je mogoče z lahkoto simulirati še tako kompleksen strežni sistem ali mrežo strežnih sistemov),
- aplikacija metod množične strežbe na različnih primerih strežnih problemov v komunikacijskih in računalniških sistemih in mrežah.

Naslov naloge: Analiza sistemov za upravljanje baz podatkov (Analiza uporabnosti sistemov za upravljanje baz podatkov na računalnikih PDP 11/34 in Delta 340 za potrebe NC obdelovalnih sistemov).

Projekt: Informacijski sistemi

Nosilec naloge: Pavel Šmarčan, Visoka tehniška šola, VTO strojništvo, Maribor

Program raziskave:

V okviru realizacije naloge je predvidena izdelava modela baze podatkov in aplikativnega softwara, ki bo služil za testiranje različnih sistemov za upravljanje baz podatkov. Model bo izdelan za potrebe TSN Maribor. S tem bo zajamčena realna uporabnost podatkov raziskave.

Naslov naloge: Večnivojski sistem upravljanja z mini in mikro računalniki III

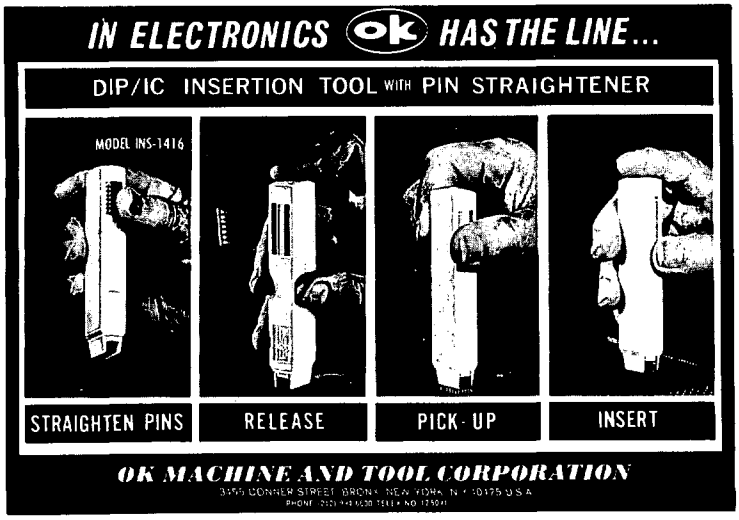
Projekt: Računalniška avtomatizacija industrijskih procesov

Nosilec naloge: Jurij Tasič, Institut "Jožef Stefan", Ljubljana

Program raziskave:

- teoretski aspekti večnivojskega upravljanja s stališča optimizacije velikih sistemov (modelna koordinacijska metoda, metoda za koordinacije ciljev),
- programska oprema za omenjeni metodi
- upravljanje z upoštevanjem omejitev na spremenljivkah stanja (nelinearne ali linearne omejitvene enačbe ali neenačbe)
- prenos metod statične optimizacije na dinamične probleme
- uporaba negradientnih metod (Powell) pri optimalnem upravljanju dinamičnih sistemov
- dopolnitev mikroračunalniškega sistema DARTA 80 v večnivojski sistem, razvoj potrebne krmilne logike in krmilne programske opreme za nadzorni računalniški sistem
- uporaba domačega visokonivojskega jezika FORMIC na mikroračunalniku DARTA 80
- poenostavljanje velikih sistemov in njih razstavitev na podsisteme, primerne za večnivojsko upravljanje.

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

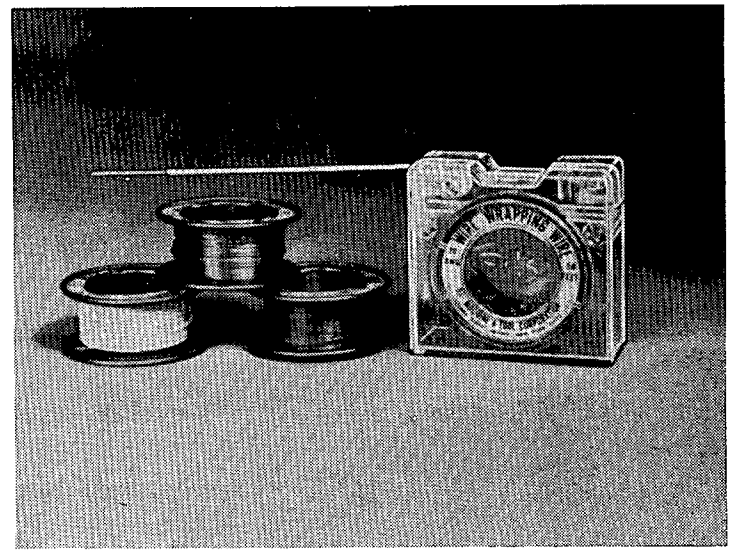


INS 1416 je orodje za vstavljanje 16- ali 14-kontaktnih integriranih vezij v podnožja ali izvrtane luknje tiskanega vezja. Posebnost je zoženi profil, ki omogoča vstavljanje vezij, ki so na plošči tesno skupaj. V držalo sta vrezani vodili za ravnanje deformiranih kontaktov integriranega vezja. Vezje 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.



LITERATURA IN SREČANJA

17-21 sept. Paris, Francija

CONVENTION INFORMATIQUE
Informacije: Convention Informatique, 6 place de Valois, 75001 Paris, France.

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.

18-20 sept. Marianske Lazne, Češkoslovaška

1st IMEKO SYMPOSIUM OF THE TC 10 TECHNICAL DIAGNOSTICS 79
Informacije: CSVTS - House of Technics, Ing. J. Karl, Gorkeho nam. 23 -112 82 Praha 1.

19-20 sept. Toulouse, Francija

CONGRESS AFCET "RECONNAISSANCE DES FORMES"
Informacije: AFCET, Boulevard Pereire 156, Paris.

24-28 sept. Garmisch-Partenkirchen, Avstrija

6th INTERNET CONGRESS
Informacije: Internet-Kongres, Organisations Komitee, Uhdestrasse 11a, D-8000 München 71, BDR

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 Regelunstechnik, Postfach 1139, D-4000 Düsseldorf 1, BDR.

25-28 sept. London, Velika Britanija

EURO-IFIP 79
Organizator : IFIP
Informacije: IFIP Secretariat, 3 rue du Marché, CH-1204 Geneva, Switzerland

10-12 okt. Bari, Italija

AICA: CONGRESSO ANNUALE
Organizator: Associazione Italiana per il Calcolo Automatico
Informacije: Maria Tereza Pazienza, Corso di Laurea in Scienze dell'Informazione, Istituto di Fisica, via Amendolá 1973, Bari, Italia.

25 sept. - 5 okt. Bonn, ZRN

GI '79
Informacije: P.P. Spies, Institut für Informatik, Universität Bonn, Weglerstrasse 6, D-5300 Bonn, BDR.

26-29 sept. Montreal, Quebec, Kanada

9th INTERNATIONAL SYMPOSIUM AND EXHIBITION ON MINI AND MICROCOMPUTERS - MIMI '79 MONTREAL
Informacije: The Secretary, MIMI '79 Montreal, PO Box 2481, Anaheim, CA 92804, USA.

1-4 okt. Huntsville, Alabama, ZDA

1st INTERNATIONAL CONFERENCE ON DISTRIBUTED COMPUTING SYSTEMS
Informacije: Gérard Le Lann, IRIA, Domaine da Valuceau, Rocquencourt, BP105, 78150 Le Chesnay, France.

3-5 okt. Rio de Janeiro, Brazilija

5th INTERNATIONAL CONFERENCE ON VERY LARGE DATA BASES
Informacije: Mr. R.J. Líbero (Gen. Conf. Chm), IBM do Brasil, Caixa Postal 1830-ZC-00, Rio de Janeiro-RJ-20.000 Brazil.

8-9 okt. Palo Alto, Kalifornija, ZDA

1979 INTERNATIONAL SYMPOSIUM ON COMPUTER HARDWARE DESCRIPTION LANGUAGES AND THEIR APPLICATIONS
Organizator: ACM SIGDA, IEEE-CS
Informacije: Donald L. Dietmeyer, Dept. of Electrical and Computer Engineering, University of Wisconsin, 1425 Johnson Drive, Madison, WI 53706, USA.

2nd IFAC SYMPOSIUM ON OPTIMIZATION METHODS
Informacije: IFAC 1979, Bulgarian Scientific Technical Unions, 108 Rakovski Street, Sofia, Bulgaria.

17-19 okt. Versailles, Francija

2nd INTERNATIONAL SYMPOSIUM ON DATA ANALYSIS AND INFORMATICS
Organizator: IRIA
Informacije: Secrétariát des Journées, Service des Relations Extérieures, IRIA, Domaine de Valuceau, Rocquencourt, BP5, 78150 Le Chesnay, France.

22-24 okt. Los Angeles, Kalifornija, ZDA

COMPUTER IN AEROSPACE CONFERENCE II
Informacije: Richard R. Erkeneff, Data Controls and Processing Systems, McDonnell Douglas Astronautics Co., Dept. 236, Building 13-3, 5301 Bolsa Avenue, Huntington Beach, CA 92644, USA.

22-24 okt. Stuttgart, ZRN

2nd IFAC/IFIP SYMPOSIUM ON INFORMATION CONTROL PROBLEMS IN MANUFACTURING TECHNOLOGY
Organizator: VDI/VDE-Gesellschaft Mess- und Regelungstechnik
Informacije: VDI/VDE-Gesellschaft Mess- und Regelungstechnik, Attn: H. Wiefels, Postfach 1139, D-4000 Düsseldorf 1, BDR

23-25 okt. Berlin, ZRN

1st EUROPEAN SYMPOSIUM ON REAL-TIME DATAHANDLING AND PROCESS CONTROL
Organizator: Purdue Europe, ESONE Committee, European CAMAC Association (ECA)
Informacije: Real-Time Data 79, Congress Organization Company, Kongress-Zentrale, John-Foster-Dulles-Allee 10, D-1000 Berlin 21, BDR.

24-25 okt. München, ZRN

WORKSHOP ON MICROCOMPUTING
Informacije: Werner Remmele, Siemens AG, B AT MCS, Balanstr. 73, D-8000 Munich 80, BDR.

25-27 okt. Bologna, Italija

EUROGRAPHICS 79
Organizator: Italian Chapter of ACM in cooperation with ECI
Informacije: Umberto Cugini (Org.Com.Chm.), Istituto di Disegno di Machine, Politechnico di Milano, Piazza Leonardo da Vinci 32, 20133 Milano, Italia.

29-31 okt. Detroit, Michigan, ZDA

ACM 1979 ANNUAL CONFERENCE
Informacije: Mayford L. Roark, Ford Motor Company, The American Road, Room 895 WHQ, Dearborn, MI 48121, USA; ali James L. Elshoff, *Computer Science Department* General Motors Research Lab., Warren, MI 48090, USA.

29-31 okt. San Juan, Puerto Rico

20th ANNUAL SYMPOSIUM ON FOUNDATIONS OF COMPUTER SCIENCES
Informacije: Ronald V. Book, Dept. of Mathematics, Univ. of California, Santa Brabara, CA 93106.

november, Versailles, Francija

AFCET CONGRESS ON SMALL GROUPS AND LARGE SYSTEMS
Informacije: AFCET, 156 boulevard de Péreire, 75017 Paris, France.

5-8 nov. Chicago, ZDA

COMPSAC 79 - 3rd INTERNATIONAL COMPUTER SOFTWARE AND APPLICATIONS CONFERENCE
Informacije: COMPSAC 79, Box 639, Silver Spring, MD 20901, USA

13-16 nov, Tokyo, Japonska

SYMPOSIUM ON FLOW MEASUREMENT AND CONTROL IN INDUSTRY
Organizator: The Society of Instruments and Control Engineers, Japan
Informacije: IMEKO Tokyo Flow Symposium 1979, The Society of Instrument and Control Engineers, Kotohira Annex, Toranomon 1-15-5, Minato-ku, Tokyo 105, Japan;

14-26 nov. Houston, Texas, ZDA

FIRST INTERNATIONAL MICRO AND MINICOMPUTER CONFERENCE
Informacije: Samuel C. Lee, Dept. of Electrical Engineering and Computing Sciences, University of Oklahoma, 202 W Boyd, Norman, OK 73019, USA.

26-28 nov. Tokyo Japonska

THIRD IFAC/IFIP SYMPOSIUM ON SHIP OPERATION AUTOMATION
Informacije: prof. Y. Lijima, Secretary General ISSOA-79 Tokyo University of Mercantile Marine, 2-1 Etchujima, Koto-ku, Tokyo 135, Japan.

26-30 nov. Budapest, Madžarska

AUDITING AND SECURITY OF COMPUTING SYSTEMS
Informacije: SZAMOK, H-1502 Budapest 112, POB 146, Hungary

27-29 nov. Pacific Grove, California, ZDA

SIGCOMM SIXTH DATA COMMUNICATIONS SYMPOSIUM
Informacije: Franklin F. Kuo, Dept. of Electrical Eng. University of Hawaii, Honolulu, HI 96822, USA.

10-12 dec. Pacific Grove, California, ZDA

7th SYMPOSIUM ON OPERATING SYSTEM PRINCIPLES
Informacije: Conf. chm. Michael D. Schroeder, Xerox Palo Alto Research Centre, 3333 Coyote Hill Road, Palo Alto, CA 94304, USA.

leto 1980

9-11 jan. Oxford, Velika Britanija

1980 CONFERENCE ON PATTERN RECOGNITION
Informacije: Josef Kittler, Nuclear Physics Laboratory, *Keble Road*, Oxford OX1 3RH, *England*

30 jan.-1 feb. Monterey, California, ZDA

INTERNATIONAL SYMPOSIUM ON MICROCOMPUTERS AND THEIR APPLICATION
Informacije: Secretary, MIMI-80 (Monterey), Box 2481, Anaheim, CA 92804.

12-14 feb. Kansas City, ZDA

ACM COMPUTER SCIENCE CONFERENCE
Informacije: Conf. Chm. Earl J. Schweppe, Dept. of Computer Science, University of Kansas, Lawrence, KS 66044.

4-6 marec Zurich, Švica

1980 INTERNATIONAL ZURICH SEMINAR ON DIGITAL COMMUNICATIONS
Informacije: Secretariat 1980 International Zurich Seminar, D. Hug, Dept. ENF, BBC Brown, Boveri and Co. Ltd., CH-5401 Baden, Switzerland.

12-14 marec Kiel, ZRN

GI-NTG CONFERENCE ON COMPUTER ARCHITECTURE AND OPERATING SYSTEMS
Informacije: Prog. chm., G. Zimmermann, Institut für Informatik und Prakt. Math., Universität Kiel, D-2300 Kiel, Germany.

31 marec - 2 april Brighton, Velika Britanija

CAD 80 - 4th INTERNATIONAL CONFERENCE ON COMPUTERS IN ENGINEERING AND BUILDING DESIGN
Informacije: Conf.Chm. Gareth Jones, IPC Science and Technology Press Ltd., 32 High st., Guildford, Surrey, England GU1 3EW.

28-30 maj Shiraz, Iran

IFAC/IFIP CONFERENCE ON SYSTEM APPROACH AND COMPUTER APPLICATIONS FOR DEVELOPMENT
Organizator: Iran Society of Automatic Control Engineers
Informacije: Secretary of IFAC/IFIP Conference, Iran 1980, PO Box 737, Shiraz, Iran

23-27 junij, Brussels, Belgija

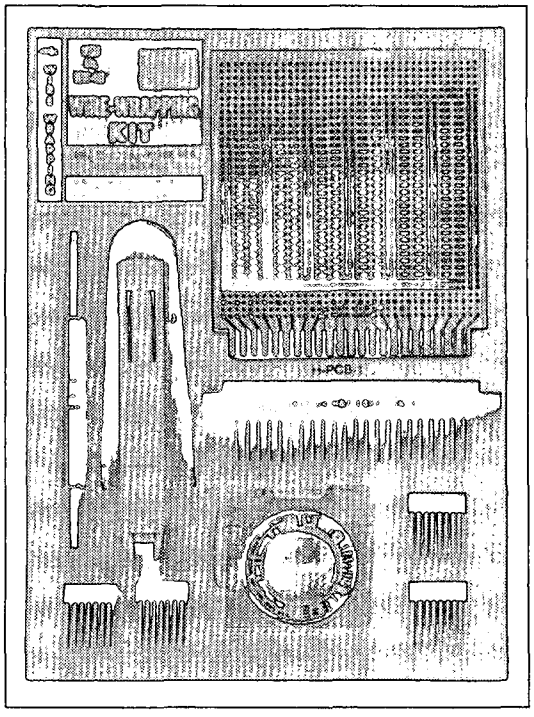
WORLD FORUM OF INTERNATIONAL TRANSNATIONAL ASSOCIATIONS
Organizator: Union of International Associations (UAI)
Informacije: UAI, rue aux Laines 1, 1000 Brussels, Belgium

oktober, Kyoto, Japan

CONFERENCE ON MAN-MACHINE COMMUNICATIONS IN CAD AND CAM
Organizator: IFIP WG5.2, 5.3
Informacije: IFIP Secretariat, 3 rue du Marché, CH-1204 Geneva, Switzerland

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

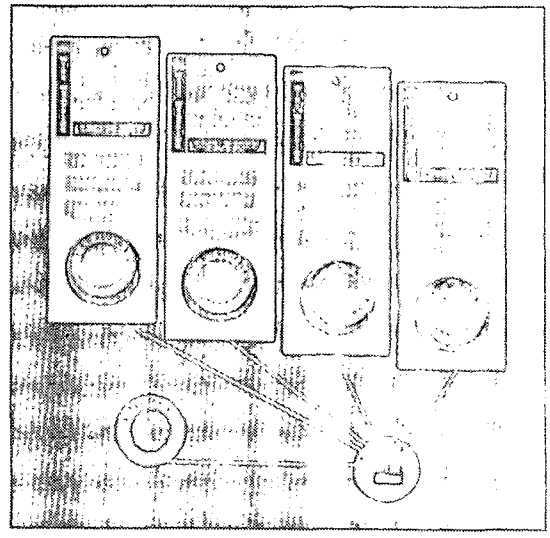


OK MACHINE & TOOL CORPORATION
3455 CONNER ST., BRONX, N.Y. 10475 U.S.A.
TELEX 125091

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.

WIRE-WRAPPING WIRE



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

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.

Ko pišete proizvajalcu, omenite časopis INFORMATICA.

SIMPOZIJ „COMPUTER APPLICATIONS IN CHEMICAL ENGINEERING"

Letošnji simpozij v Montereux-u (Švica), od 8.-11. aprila , je predstavljal dvanajsto tovrstno strokovno srečanje po vrsti in obenem 214 srečanje v organizaciji Evropske federacije za kemijsko inženirstvo (European Federation of Chemical Engineering). Simpozij je potekal ob odlični organizaciji švicarskih strokovnjakov s tega področja in pod pokroviteljstvom naslednjih združenj:

- S.I.A. (Fachgruppe Verfahrens und Chemieinge nieurtechnik)
- American Institute of Chemical Engineers (AIChE)
- Society of Chemical Engineers in Japan
- 143rd Research Committee of Process Systems Engineering, Japan.

To srečanje je nadaljevalo tradicijo simpozijev z enako tematiko, kakor so bili leta 1973 v Parizu, 1975 v Karlovih Varih, 1977 v Visokih Tatrah in 1978 zopet v Parizu.

Simpozija se je udeležilo 250 strokovnjakov, pretežno iz evropskih držav (Švica, Avstrija, Italija, Francija, Španija, Portugalska, Belgija, ZR Nemčija, Danska, Velika Britanija, Nizozemska, Švedska, Finska, Norveška in Madžarska) ter iz ZDA in Japonske (nekaj udeležencev iz izvenevropskih držav), Turčije, Izraela, Saudske Arabije, Avstralije, Brazilije in Mehike. Nekoliko preseneča odsotnost udeležencev iz socialističnih držav (Sovjetska zveza, Češkoslovaška, Nemška demokratična republika), ki so sicer tudi tokrat imeli prijavljenih nekaj referatov. Vseh prijavljenih referatov je bilo okrog 120.

Iz Jugoslavije sta bila prijavljena dva referata; od teh je bil prezentiran samo eden, in sicer iz Instituta "Jožef Stefan", Ljubljana.

Prispevki na simpoziju so bili razdeljeni na trinajst sekcij z naslednjimi temami oz. naslovi:

1. Integracija razvojnega procesa z uporabo računalnikov
2. Novosti v matematičnem modeliranju
3. Načrtovanje in obratovanje šaržnih procesov
4. Izobraževanje kemijskih inženirjev v uporabi računalnikov
5. Sinteza procesa
6. Flowsheeting
7. Baze podatkov in programske knjižnice
8. Ocenjevanje projektov
9. Varnost in zanesljivost
10. Osnovne operacije
11. Upravljanje procesov z računalnikom
12. Optimiranje
13. Matematične metode in nove tehnike.

Delo na simpoziju je potekalo dnevno od 8,30 do 18,30 v treh vzporednih sejah. Vsaka seja je obravnavala po tri do pet referatov. Štiri seje so bile namenjene preglednim referatom, na katerih so vabljeni strokovnjaki podali pregled in strokovno mnenje o prijavljenih prispevkih, kar je dalo več pobud za kvalitetne diskusije.

Splošni vtis s simpozija lahko povzamemo v naslednjem:

- od leta 1973 je uporaba računalnikov v kemijskem inženirstvu dosegla precejšen napredek;
- kemijsko inženirstvo kot disciplina nujno potrebuje računalnike in računalniške metode za svoj nadaljni razvoj;
- uporaba računalnikov v nekaterih ožjih usmeritvah kemijskega inženirstva postaja vsakdanja praksa;
- kemijska in procesna industrija v tujini potrebuje tovrstna znanja in zato tudi izdatno finansira ustrezne raziskave oziroma razvoj te stroke;
- velik poudarek se daje interdisciplinarnosti: kemijski inženirji se intenzivno poglabljajo v uporabo računalniških metod za dosego konkretnih in uporabnih inženirskih ciljev (kakor so npr. načrtovanje procesov in naprav, analiza obratovanja, izboljevanje obratovanja, izboljšano upravljanje procesov in podobno). K reševanju inženirskih problemov pristopajo strokovnjaki s področij uporabne matematike, računalništva, avtomatike in informatike;
- v skladu z dejanskimi potrebami procesne industrije vlagajo univerze precejšnje napore in sredstva v izobraževanje kemijskih inženirjev v uporabi računalniških metod za dosego inženirskih ciljev;
- tako v uporabi in razvoju metod kakor tudi v izobraževanju je čutiti močan vpliv napredka na področju računalništva v zadnjih letih predvsem v zvezi z zmogljivejšo in dostopnejšo računalniško opremo , tako da postaja interaktivna uporaba računalnikov preko razmeroma cenenih videoterminalov, ki so priključeni na visokozmogljivo mednarodno računalniško omrežje vse bližja realnim potrebam in možnostim širšega kroga inženirjev pri njihovih vsakdanjih opravilih, ne da bi jim bilo za to nujno potrebno podrobnejše poznavanje računalništva;

JČ

NOVICE IN ZANIMIVOSTI

UPORABA MIKRO RAČUNALNIKOV V RADIOAMATERSTVU

RADIOAMATERSTVO JE PODROČJE, KJER JE UPORABA MIKRO RAČUNALNIKOV IZREDNO ZANIMIVA, POTREBNA, VSESTRANSKA IN TUDI DOKAJ ZAHTEVNA. MIKRO PROCESORJI PA SI UTIRAJO POT TUDI V RADIOAMATERSKE APARATURE, KO ZVIŠUJEJO ZANESLJIVOST, NATANČNOST IN FUNKCIONALNOST NAPRAV ZA SPREJEM IN ODDAJO SIGNALOV. DOSEŽKI VESOLJSKE TEHNIKE SE HITRO PRENAŠAJO TUDI V APARATE ZA RADIO AMATERJE. OGLEJMO SI NA KRATKO, KAKO SO POSAMEZNI MIKROPROCESORSKI ELEMENTI UPORABLJENI V APARATIH IN KAKŠNE SO MOŽNOSTI UPORABE MIKRO RAČUNALNIKA V RADIOAMATERSKIH PROCESIH.

RAČUNALNIK LAHKO PODATKE SPREJEMA, JIH PREDELUJE IN ODDAJA. PREDELAVA PODATKOV V RAČUNALNIKU JE VOBČE NEOMEJENA IN ODVISNA OD INVENTIVNOSTI PROGRAMERJA OZIROMA RAČUNALNISKEGA UPORABNIKA. RADIO AMATER JE LAHKO ZELO ZAHTEVEN RAČUNALNISKI UPORABNIK, KI IŠČE VEČKRAT TUDI IZVIRNE REŠITVE ZA SVOJE KOMUNIKACIJSKE PROBLEME. SODOBNA TEHNOLOGIJA NJEGOVIH NAPRAV MU TUDI OMOGOCA, DA TE NAPRAVE SIGNALNO IN PROGRAMSKO POVEZE Z MIKRO RAČUNALNIKOM.

GLAVNA PODROČJA UPORABE MIKRO RAČUNALNIKOV IN MIKRO PROCESORJEV V RADIJSKI TEHNIKI SO DANES NA NASLEDNJIH TEMELJNIH PODROČJIH:

-- TELEPRINTERSKI PROMET PREKO BREZŽIČNIH ZVEZ;
-- POČASEN (ZVOČNI) PRENOS TELEVIZIJSKIH SLIK;
-- SPREJEM IN ODDAJA MORSEJEVIH TELEGRAFSKIH ZNAKOV;
-- OBDELAVA PODATKOV V POVEZAVI S SATELITSKIMI ZVEZAMI (AMERIŠKI IN SOVJETSKI RADIOAMATERSKI SATELITI TIPA OSCAR IN RS);
-- VODENJE ANTENSKIH USMERJENIH NAPRAV V ZVEZAH PREKO NARAVNEGA, UMETNIH SATELITOV IN METEORITSKIH ROJEV;
-- AVTOMATIČNO VODENJE TEKMOVALNIH DNEVNIKOV TER PISANJE (IZPIS) RADIOAMATERSKE DOKUMENTACIJE;
-- UPORABA RAČUNALNIKOV PRI VREDNOTENJU EKSPERIMENTALNIH PODATKOV TER PRI LABORATORIJSKEM DELU.

PODJETJA, KI IZDELUJEJO RADIOAMATERSKO OPREMO, UPORABLJAJO MIKRO PROCESORJE PREDVSEM PRI FREKVENČNI SINTEZI (NPR. SPREJEMNE/ODDAJNE NAPRAVE ICOM 701 IN 211, KENWOOD/TRIO), PRI AVTOMATIČNEM PREISKOVANJU FREKVENČNIH PODROČIJ TER ZA DIAGNOSTICIRANJE NAPAK V NAPRAVAH. TI PROCESORJI KRMILIJO HKRATI TUDI PREHOD IZ ENEGA FREKVENČNEGA OBSEGA NA DRUGI OBSEG TER VZDRŽUJEJO NORMALNO STANJE NAPRAV (PREPREČUJEJO TERMIČNE PREOBREMENITVE KONČNIH OJAČEVALNIKOV, SLABO PRILAGODITEV ANTENE NA ODDAJNIK ITN.). S TEM PA POHOD MIKRO PROCESORJEV V RADIOAMATERSKE NAPRAVE SE NI KONČAN.

RADIJSKI AMATERJI RAZVIJAJO V PROSTEM ČASU NOVE MIKRORAČUNALNISKE KONFIGURACIJE, SISTEME IN PROGRAME. TUDI RAČUNALNIK TRS-80 (RADIO SHACK) JE ŽE DOKAJ DOBRO USPOSOBLJEN ZA RADIOAMATERSKO UPORABO. SEVEDA PA SODIJO V TO DRUŽINO TUDI DRUGI MAJHNI SISTEMI, KOT SO PET, KIM IN VRSTA NOVIH PROIZVODOV. V LJUBLJANI DELUJE V OKVIRU RADIOKLUBA LJUBLJANA, DRENIKOVA 32, POSEBNA, T.I. MIKRORAČUNALNISKA SEKCIJA, KI GRADI SISTEME, RAZVIJA PROGRAME IN PRIREJA PREDAVANJA ZA ZAČETNIKE IN INŽENIRJE.

APŽ

286-MEGABITNO POMNILNO VEZJE Z MAGNETNIMI MEHURČKI

PODJETJE TEXAS INSTRUMENTS ŽE PROIZVAJA 92 KILOBITNE POMNILNIKE Z OZNAKO TIB 0203 (CENA US$ 125). PROIZVAJA PA TUDI 286524-BITNI MEHURČNI POMNILNIK S POSEBNIM V/I IN Z BLOČNO PODATKOVNO ZANČNO ARHITEKTURO. POVPREČEN ČAS DOSTOPA JE 7,3 MS, DISIPACIJA JE 0,9 W IN CENA US$ 500. NASLOV PROIZVAJALCA JE: TEXAS INSTRUMENTS, INC., 13500 NORTH CENTRAL EXPWY., DALLAS, TX. OZNAKA TEGA SUPER INTEGRIRANEGA VEZJA JE TI B 0303.

APŽ

MEHANIČNI VMESNIK ZA ELEKTRIČNI PISALNI STROJ

PODJETJE ROCHESTER DATA, INC. JE DALO NA TRŽIŠČE ELKTROMEHANIČNI VMESNIK ZA ELEKTRIČNE PISALNE STROJE, S POMOČJO KATEREGA JE MOČ SPREMENITI POLJUBEN ELEKTRIČNI PISALNI STROJ V RAČUNALNISKO KRMILJENI TISKALNIK. TA NAPRAVA JE KORISTNA ZLASTI TEDAJ, KO ŽELIMO SVOJ PISALNI STROJ SPREMENITI V TISKALNIK, NPR. V SISTEMU ZA PROCESIRANJE TEKSTOV. TAKO OPREMLJEN PISALNI STROJ LAHKO TEDAJ UPORABIMO ZA MEHANIČNO PISANJE NAJRAZLIČNEJŠIH TEKSTOV. NAPRAVA SE ENOSTAVNO PRITRDI PREKO PISALNE TASTATURE IN MAJHNI MAGNETI AKTIVIRAJO POSAMEZNE TIPKE. CENA TEGA VMESNIKA JE US$ 395, NASLOV PROIZVAJALCA JE: ROCHESTER DATA, INC., 3100 MONROE AVE.,ROCHESTER, NY 14618.

APŽ

PRVI KRMILNIK ZA TISKALNIK V OBLIKI LSI INTEGRIRANEGA VEZJA

TO NOVO INTEGRIRANO VEZJE JE CENENO, SAJ STANE EN KOMAD LE US$ 35. OZNAKA KRMILNIKA JE CY-480 IN NAMENJEN JE TISKALNIKOM, KI DELUJEJO DO HITROSTI 200 ZNAKOV NA SEKUNDO. VEZJE JE 40-NOŽIČNO, KI KRMILI IN POVEZUJE POLJUBNI STANDARDNI TISKALNIK Z MATRIKO 5 KRAT 7 TOČK. VEZJE SE NAPAJA Z ENO SAMO NAPETOSTJO 5 V IN POVEZUJE TISKALNIK Z MIKRO RAČUNALNIKOM PREKO STANDARDNEGA 8-BITNEGA PORTA. VEZJE CY-480 SPREJEMA SERIJSKI RS 232 ALI PARALELNI ASCII KOD.

NA TA NAČIN NADOMESTI TO INTEGRIRANO VEZJE DOKAJ OBSEŽNE IN DRAGE KRMILNIKE. ZASTOPANIH JE 96 ZNAKOV (VELIKE IN MALE ČRKE), VEZJE PA IMA TUDI NOTRANJI POMNILNIK ZA 48 ZNAKOV. OBSTAJA TUDI VEČ UKAZOV ZA NASTAVITEV PISALNE GOSTOTE, IN SICER ZA 10, 12 IN 16 KARAKTERJEV NA COLO, NADALJE UKAZ ZA DVOBARVNO PISANJE, ZA PISANJE NAPREJ IN NAZAJ TER TUDI ZA PISANJE PO VERTIKALI GOR IN DOL. VEZJE IMA TUDI GRAFIČNE SPOSOBNOSTI IN VSEBUJE T.I. "FLIP-PRINT". IMA TUDI LINIJE ZA ASINHRONO KOMUNICIRANJE TER ZA MEDSEBOJNO ROKOVANJE. NASLOV PROIZVAJALCA JE: CYBERNETIC MICRO SYSTEMS, 2378-B WALSH AVE., SANTA CLARA, CA 95050.

APŽ

MESEČNIK ZA UPORABNIKE MIKRORAČUNALNIŠKIH SISTEMOV TIPA TRS-80

ZALOŽBA COMPUTRONICS, INC., BOX 149RB, NEW CITY, NY 10956 IZDAJA MESEČNIK "TRS-80 MONTHLY NEWSLETTER", KI JE NAJOBSEŽNEJŠA PUBLIKACIJA, NAMENJENA UPORABNIKOM SISTEMA TRS-80. ČASOPIS IMA TELE RUBRIKE: POSLOVANJE, OSEBNE FINANCE, PRAKTIČNA UPORABA, IZMENJAVA PROGRAMOV, TRŽNI PODATKI, VPRAŠANJA IN ODGOVORI, PROGRAMSKE LISTE, ZADNJI RAZVOJ PROIZVAJALCA SISTEMOV TRS-80, TJ. RADIO SHACK. LETNA NAROČNINA ZNAŠA US$ 24. ZALOŽNIK RAZPOLAGA TUDI Z BREZPLAČNIM KATALOGOM PROGRAMSKE OPREME, KI VSEBUJE VEČ STO PROGRAMOV, KATERE JE MOČ DOBITI NA KASETAH IN DISKETAH.

APŽ

HITRI KRMILNIK ZA PRENOS PODATKOV

NA TRŽIŠČU SE JE POJAVIL DOSLEJ NAJHITREJŠI KOMUNIKACIJSKI KRMILNIK, KI LAHKO PROCESIRA 2 MILIJONA BITOV NA SEKUNDO V T.I. SDLC NAČINU (SYNCHRONOUS DATA LINK CONTROL) ALI V OBLIKI DRUGIH SINHRONIH KOMUNIKACIJSKIH PROTOKOLOV. INTEGRIRANO VEZJE 2652-1 SE LAHKO UPORABI KOT KRMILNIK V RAČUNALNIŠKIH MREŽAH IN V SISTEMIH Z VEČ TERMINALI, DO RAZDALJ, KI NE PRESEGAJO DVA KILOMETRA. KRMILNIK JE UPORABLJIV TUDI ZA T.I. NEPOSREDEN POMNILNIŠKI DOSTOP (DMA). NASLOV PROIZVAJALCA JE: SIGNETICS, 811 E. ARQUES AVE., SUNNYVALE, CA 94086.

APŽ

12-BITNI D/A KRMILNIK

INTEGRIRANO VEZJE IMA ČAS NASTAVITVE LE PLUS/MINUS 0,01 PROCENTA ALI 60 NANOSEKUND (TIPIČNA VREDNOST) IN NOSI OZNAKO 4065. VOJAŠKA IZVEDBA TEGA DIGITALNO/ ANALOGNEGA PRETVORNIKA IMA SUFIKS 83 IN DELUJE V TEMPERATURNEM OBMOČJU OD -36 DO +125 STOPINJ CELZIJA. VEZJE IMA STANDARDNO OBLIKO S 24 NOŽICAMI. CENA PRVE IN DRUGE (VOJAŠKE) IZVEDBE JE US$ 160 IN 220. NASLOV PROIZVAJALCA JE: TELEDYNE PHILBRICK, ALLIED DR., RTE. 128, DEDHAM, MA 02026.

APŽ

MIKRO RAČUNALNIK NA ENI PLOŠČI

PODJETJE DOSC, INC., 500 FIFTH AVE., NEW YORK, NY 10036 JE IZDELALO PRVI OEM MIKRO RACUNALNIK NA ENI PLOSCI, KI VSEBUJE TAKO KRMILNIK ZA GIBKI DISK KOT KRMILNIK ZA KATODNO ELEKTRONKO (CRT). IME TE PLOŠČE JE TCB-85 IN RAČUNALNIK JE PODPRT Z MONITORJEM ZA CRT, FD (GIBKI DISK), KB (TASTATURA) IN TISKALNIK. V OKVIRU TE KONFIGURACIJE NA ENI PLOŠČI JE NA VOLJO ŠE TALE PROGRAMSKA OPREMA: CP/M DOS, BASIC, COBOL, FORTRAN, UREJEVALNIK TEKSTA IN ZBIRNIK. CENA (PRI 10 KOMADIH) ZNAŠA US$ 1495.

APŽ

MALOPRODAJNE CENE TERMINALOV

MALOPRODAJNE CENE NEKATERIH ZNANIH TERMINALOV V VELIKI BRITANIJI (TOREJ NA EVROPSKEM TRŽIŠČU) SO NPR. TELE: ZNANI VIDEO TERMINAL TIPA VT 100, KI UPORABLJA MIKRO PROCESOR IN SE BO V NASLEDNJIH MESECIH POJAVIL TUDI NA DOMAČEM TRŽIŠČU, IMA CENO £ 1195. CENA T.I. VT 52 EMULATORJEV (TO SO TIPI 7000, 7002 IN 7009) ZNAŠA £ 845. CENA PISALNIKOV LA 36 DEC WRITER II IN LA 120 DEC WRITER III (SLEDNJI JE DVOSMEREN IN IMA HITROST 180 ZNAKOV NA SEKUNDO) JE £ 865 IN £ 1695. TISKALNIK LA 180 DEC PRINTER I PA ZAČENJA S CENO £ 1575. CENA TEH NAPRAV JE ZANIMIVA ZARADI PRIMERJAVE CEN V OKVIRU DOMAČIH KONFIGURACIJ TIPA DELTA (TOZD DIGITAL-ELEKTROTEHNA).

APŽ

CENEN BARVNI VIDEO TERMINAL IZ VELIKE BRITANIJE

VIDEO TERMINAL INTERCOLOR 8001 JE INTELIGENTNI, OSEMBARVNI OSNOVNI TERMINAL, KI UPORABLJA MIKRO PROCESOR 8080 TER JE UPORABLJIV TUDI KOT SAMOSTOJNA, MIKRORAČUNALNIŠKO RAZŠIRLJIVA ENOTA. DODATI JE MOGOČE 32K RAMA TER LASTNOSTI Z BARVO OZADJA, SVETLOBNO PERO (PISALO), 48 VRSTIC Z 80 ZNAKI IN 64 POSEBNIH GRAFIČNIH ZNAKOV.

RAZŠIRLJIVOST TERMINALA V MIKRO RAČUNALNIK PA JE TALE: INTERCOLOR 8051 JE RAČUNALNIK Z JEZIKOM BASIC IN DRUGIMI 8080 JEZIKI, Z VSEMI BARVNIMI DODATKI, VELIKIMI IN MALIMI ČRKAMI, GIBKIMA DISKOMA ITN. CENA OSNOVNEGA TERMINALA JE SAMO £ 910, CENA OPISANE KONFIGURACIJE PA £ 2800. NASLOV PROIZVAJALCA JE: TECHEX LTD., BRAIDLEY HOUSE, ST. PAUL'S LANE, BOURNEMOUTH, BH8 8HN, VELIKA BRITANIJA.

APŽ

MODUL Z MEHURČNIM POMNILNIKOM

ROCKWELLOW MODUL RBM 256 SHRANI 266500 BITOV TER JE SESTAVLJEN IZ 282 ZANK, OD KATERIH IMA VSAKA 1025 MEHURČNIH POZICIJ. NAPRAVA UPORABLJA 260-BITNI PODATKOVNI BLOK, KJER SO BINARNI PODATKI SHRANJENI V 256 ZANKAH; ŠTIRI ZANKE SO REZERVIRANE ZA POMOŽNE (KRMILNE) BITE. PODATKI SE PRENAŠAJO S 150 KHZ, KO JE POVPREČNI ČAS DOSTOPA 4 MS ZA PRVI BIT BLOKA. TA PRETOK (PRETOČNOST) JE OHRANJEN TUDI MED ČITANJEM S POSEBNO TEHNIKO. MODUL IMA 18 NOŽIC, DISIPACIJA JE 820 MW. PRODAJALEC: PELCO LTD., ENTERPRICE HOUSE, 83-85 WESTERN ROAD, HOVE, SUSSEX BN3 1JB.

APŽ

IBM ŠE VEDNO RASTE

KO SO SE POJAVILI MIKRO RAČUNALNIKI IN DOSEGLI SVOJ ZENIT POPULARNOSTI, SO SE POJAVILE NAPOVEDI, DA JE NASTOPIL KONEC DOMINACIJE IBM NA PODROČJU RAČUNALNIŠKIH POSLOV. VENDAR SE TO NI ZGODILO. DANES IMA IBM VEČJO ZBIRKO NAROČIL, KOT JO JE IMEL KDAJKOLI POPREJ. TRENUTNA NAROČILA PRESEGAJO KAR ŠTIRIKRAT RAČUNALNIŠKO MOČ, KI JO JE IBM PLASIRAL V VSEJ SVOJI ZGODOVINI. DOBAVNI ČAS SE JE ZA RAČUNALNIKE 303X POVEČAL NA VEČ KOT DVE LETI IN IBM IMA NAROČENIH 13000 TEH SISTEMOV, KI BODO ZAMENJALI SISTEME 370.

APŽ

TANDY NAČRTUJE NOV RAČUNALNIK

PODJETJE TANDY IMA V RAZVOJU RAČUNALNIK NASLEDNJE GENERACIJE TIPA RADIO SHACK TRS-80, KI BO IMEL BARVNI TERMINAL. TANDY, KI ZAPOSLUJE V PROIZVODNJI RAČUNALNIKA TRS-80 LE 700 LJUDI (NA MESEC IZDELAJO ŽE VEČ DESETTISOČ KOSOV TEGA RAČUNALNIKA), IŠČE ZUNANJEGA PROIZVAJALCA ZA NOVI SISTEM. TANDY BO LETOS PRODAJAL TUDI NOVE PROGRAMSKE PAKETE ZA SVOJ RAČUNALNIK.

APŽ

VELIK PORAST PROIZVODNJE OSEBNIH RAČUNALNIKOV ŽE V TEM LETU

PROIZVODNJA OSEBNIH RAČUNALNIKOV SE BO LETOS POVEČALA ZA ŠTIRI DO ŠESTKRAT V PRIMERJAVI Z LANSKO PROIZVODNJO. SAMO V ZDA BO PRODANIH LETOS 200000 DO 300000 SISTEMOV OSEBNIH RAČUNALNIKOV, LANI PA SO JIH PRODALI LE 50000. ZLASTI SE BO DVIGNILA KVALITETA IN FUNKCIONALNOST NOVIH KONZOL, KI SO BILE DOSLEJ ENOSTAVNE IN CENENE. FAIRCHILD C&I NAPOVEDUJE PRODAJO 4,6 MILIJONOV VIDEO IGER, KAR JE DVAKRAT IN VEČ KOT V LETU 1978 (2,1 MILIJONA), PRODAL BO PA TUDI 18 MILIJONOV TRAČNIH POGONOV (LANI 5,7 MILIJONOV). IZREDNO BO PORASLA TUDI PRODAJA OSTALIH IGER (NEVIDEO IGRE), IN SICER IZ 290 NA 500 MILIJONOV DOLARJEV.

APŽ

NA VIDIKU SO 32-BITNI MIKRO PROCESORJI

MEDTEM KO SI ZILOG IN MOTOROLA PRIZADEVATA, DA BI KONČNO PROIZVEDLA SVOJA

16-BITNA MIKRO PROCESORJA, JU TEXAS INSTRUMENTS IN INTEL ŽE NEKAJ ČASA PRODAJATA. SEDAJ PA SO SE POJAVILE GOVORICE, DA PRIPRAVLJATA TI IN INTEL OB KONCU TEGA LETA NOVO PRESENEČENJE, IN SICER 32-BITNI MIKRO PROCESOR. PROIZVODNJA TEH PROCESORJEV NAJ BI STEKLA V ZAČETKU LETA 1981.

APŽ

ČASOPISNI PAPIR IN PISEMSKA POŠTA BOSTA KMALU LE ŠE PRETEKLOST

OSNOVNI RAZVOJ ZA DIGITALNI ELEKTRONSKI POŠTNI PROMET JE V POLNEM TEKU. PREDSTAVLJAJTE SI, DA SE VAM VAŠI ČASOPISI, ZABAVNIKI, RAČUNI, POLOŽNICE ITN. DOSTAVLJAJO NEPOSREDNO PREKO VAŠEGA OSEBNEGA RAČUNALNIŠKEGA SISTEMA IN DA LAHKO PIŠETE PISMA (SEVEDA Z UPORABO UREJEVALNIKOV IN PROCESORJEV TEKSTA) TER JIH ODPOŠLJETE S PREPROSTIM PRITISKOM NA GUMB. VSE TO NI VEČ FANTAZIJA IN NEKATERA VELIKA AMERIŠKA PODJETJA IN VLADNE AGENCIJE ŽE IMAJO TAKŠNE SISTEME. V LETU 1980 BO TA TEHNOLOGIJA PRESENETILA POSLOVNI SVET IN TUDI DOMAČO UPORABO. NOBENA SKRIVNOST NI VEČ, DA PODJETJA TEXAS INSTRUMENTS, HEWLETT-PACKARD IN IBM RAZVIJAJO OSEBNE RAČUNALNIŠKE SISTEME, NAMENJENE PREJ OPISANIM APLIKACIJAM.

V DECEMBRU PRETEKLEGA LETA JE PODJETJE XEROX NASLOVILO NA ZVEZNO KOMISIJO ZA KOMUNIKACIJE (FCC) ZDA PETICIJO, V KATERI SE POTEGUJE ZA RAZVOJ DIGITALNEGA PISEMSKEGA/KOMUNIKACIJSKEGA SISTEMA Z UPORABO MIKRO VALOV. TA SISTEM NAJ BI RABIL ZA DISTRIBUCIJO DOKUMENTOV, PODATKOVNE KOMUNIKACIJE ITN. Z OBSEGOM DO 256000 ZLOGOV, KAR ZNAŠA PRECEJ VEČ, KOT DOPUŠČAJO DANAŠNJI TELEFONSKI SISTEMI IN VEČ KOT DOPUŠČA NOVI BELLOV SISTEM S T-NOSILNIM VALOM, KI GA PRAVKAR INŠTALIRAJO. XEROX TRDI, DA BODO DOKUMENTI LAHKO POŠILJANI PO NIŽJI CENI, KOT ZNAŠA CENA PISEMSKE POŠTE V ZDA. VSAKO PISALNO MESTO NAJ BI BILO OPREMLJENO S TASTATURO, VIDEO PRIKAZOVALNIKOM, DISKOM IN PROCESORJEM IN NAJ BI BILO OSPOSOBLJENO ZA PROCESIRANJE TEKSTOV, SORTIRANJE ITN. IN SEVEDA S POŠTNO OBDELAVO. POŠTNA OBDELAVA BO VODENA Z RAČUNALNIKOM TER BO VSEBOVALA AVTOMATIČNO NASLAVLJANJE, UPOŠTEVANJE PRIORITETE, DOSTAVO DOKUMENTA NA VEČ MEST, AVTOMATIČNO POŠILJANJE PREDHODNO SHRANJENIH SPOROČIL, PREISKOVANJE SPOROČIL ITN.

PODJETJE GT&E IMA SKUPINO ZA RAZISKAVO TRŽIŠČA V ZVEZI S SISTEMOM, KI BO POŠILJAL PODATKE PREKO TELEFONSKIH VODOV DO MODIFICIRANIH TELEVIZIJSKIH SPREJEMNIKOV DOMA IN NA DELOVNIH MESTIH. TA SISTEM BO PODOBEN SISTEMU VIEWDATA, KI SE TRENUTNO PREIZKUŠA V OKVIRU BRITANSKE POŠTE. GT&E SE ŽE DOGOVARJA ZA LICENCO Z VIEWDATA. PODOBNA POGAJANJA SO V TEKU S STRANI PODJETIJ ITT, TI IN RCA. TEXAS INSTRUMENS PREIZKUŠA TUDI HIŠNI (DOMAČI) INFORMACIJSKI SISTEM, KI POŠILJA PODATKE PREKO REGULARNIH RADIJSKIH KANALOV.

V OKVIRU POŠTE ZDA JE ŽE PRIPRAVLJEN ZA UPORABO LINIJSKI SISTEM Z OZNAKO ECOM (ELECTRONIC COMPUTER ORIGINATED MAIL). ECOM BO ZAČEL DELOVATI ŽE LETOS. POŠILJATELJ BO NAPISAL PISMO NA TERMINALU IN GA BO POSLAL PREKO TELEFONA DO SVOJE POŠTE, TA GA BO POSLALA DO NASLOVNE POŠTE, KJER SE BO PISMO IZPISALO IN ODNESLO NASLOVNIKU. TA NAČIN KOMUNIKACIJE NAJ BI SE UPORABLJAL ZA POŠILJANJE RAČUNOV, VAŽNIH SPOROČIL ITN.

SKUPINE UPORABNIKOV OSEBNIH RAČUNALNIKOV SO USTANOVILE ENOSTAVNI SISTEM Z IMENOM PCNET (MREŽA OSEBNIH RAČUNALNIKOV). PCNET UPORABLJA MODEME IN TELEFONSKE LINIJE ZA KOMUNICIRANJE. TUDI RADIOAMATERJI SO USTANOVILI SVOJ BILTEN, KI GA IZMENJUJEJO PREKO TELEFONA IN RADIJSKIH ZVEZ NA 2 METRIH (SISTEM AMRAD, AMATEUR RADIO RESEARCH AND DEVELOPMENT).

APŽ

NAJVEČJA RAČUNALNIŠKA RAZSTAVA

TAKO IMENOVANA NCC (NATIONAL COMPUTER CONFERENCE) JE BILA LETOS OD 6. DO 9. JUNIJA. LANI SE JE TE KONFERENCE UDELEŽILO SKORAJ 60000 STROKOVNJAKOV. TO ZBOROVANJE JE LETOS SPREMLJALA NAJVEČJA RAČUNALNIŠKA RAZSTAVA DOSLEJ, KO JE 400 RAČUNALNIŠKIH PODJETIJ RAZSTAVLJALO V 1700 LOŽAH. POSEBEN POUDAREK JE BIL TUDI NA UPORABI OSEBNIH RAČUNALNIKOV. RAZSTAVA IN KONFERENCA STA BILI V NEW YORKU.

APŽ

NOVI OSEBNI RAČUNALNIKI

PODJETJI TEXAS INSTRUMENTS IN HEWLETT-PACKARD PRIPRAVLJATA NOVE OSEBNE RAČUNALNIŠKE SISTEME. GRE ZA NOVE SISTEME, KI NAJ BI BISTVENO SPREMENILI TRŽIŠČE OSEBNIH RAČUNALNIKOV. NOVA SISTEMA OBEH PODJETIJ NAJ BI SE POJAVILA NA TRŽIŠČU NAJKASNEJE DO NOVEGA LETA, CENA OSNOVNE KONFIGURACIJE PA NAJ BI NE PRESEGLA $ 500. DISTRIBUCIJSKA MREŽA ZA OBA NOVA SISTEMA JE ŽE POSTAVLJENA.

APŽ

NOVE MIKROPROCESORSKE KOMPONENTE

INTEL UVAJA PROIZVODNJO NOVIH ANALOGNIH MIKRO PROCESORJEV IN 8-BITNIH SUPER MIKRO PROCESORJEV. PROCESIRANJE V REALNEM ČASU JE BILO DOSLEJ OMEJENO ZARADI NIZKE HITROSTI MIKRO PROCESORJEV. S TAKTOM 2 MHZ PROCESORJA 8080 JE MOGOČE OBDELATI SIGNALE LE DO FREKVENC 2 KHZ. V JESENI BO INTEL PROIZVAJAL MIKRO PROCESORJE Z A/D IN D/A PRETVORBO V ENEM VEZJU IN OMOGOČENO BO PROCESIRANJE SIGNALOV DO 13 KHZ. PROCESOR 2920 BO IMEL 9-BITNI KONVERZIJSKI REGISTER IN GA BO MOGOČE UPORABITI V KONFIGURACIJI S PROCESORJEM 8080. RAZEN TEGA JE INTEL RAZDELIL PROCESOR 8086 (16 BITOV) V DVA PROCESORJA Z 8 BITI IN 16-BITNIM NASLOVNIM VODILOM (SEVEDA V ENEM VEZJU); TO VEZJE NOSI OZNAKO 8088.

APŽ

QUIP NAJ BI ZAMENJAL DIP

PRI NOVIH 16- IN 32-BITNIH MIKRO PROCESORJIH, KI SE NAČRTUJEJO, SO NASTALI PROBLEMI ZARADI PREMAJHNEGA ŠTEVILA SPONK NA INTEGRIRANEM VEZJU. PREVELIKO JE ŠTEVILO NASLOVNIH, PODATKOVNIH, KRMILNIH IN V/I VODOV, DA BI LAHKO OBDRŽALI STARO RAZPOREDITEV IN ŠTEVILO PRIKLJUČNIH NOŽIC OZIROMA SPONK. TI PROBLEMI SO ZNANI ŽE PRI 40-NOŽIČNEM VEZJU S SISTEMOM DIP (DUAL-IN-LINE PACKAGE). INTEL IN 3M STA RAZVILA NOVI, 64-NOŽIČNI SISTEM QUIP (QUAD-IN-LINE PACKAGE), KI IMA PO DVE VRSTICI S 16 NOŽICAMI NA OBEH STRANEH VEZJA. S TEM SISTEMOM SE ZMANJŠA DOLŽINA NOTRANJIH VODOV, KAPACITIVNOST, INDUKTIVNOST IN UPORNOST IN SE DOSEŽEJO VEČJE OPERATIVNE HITROSTI VEZJA. TUDI CENA SE BO Z UVEDBO QUIP SISTEMA ZMANJŠALA ZA 15 ODSTOTKOV V PRIMERJAVI Z DIP SISTEMOM.

APŽ

16-BITNI MIKRO PROCESORJI

ZILOG IN MOTOROLA STA ZAČELA DOBAVLJATI SVOJA 16-BITNA MIKRO PROCESORJA Z-8000 IN 68000. VEČJE KOLIČINE TEH PROCESORJEV BODO DOBLJIVE ŠELE LETOS V JESENI. MEDTEM PA JE INTELOV PROCESOR 8086 V PRODAJI ŽE CELO LETO IN NJEGOVA CENA SE JE SPUSTILA NA VREDNOST $ 65 ZA 4 MHZ PROCESOR IN NA $ 76 ZA 5 MHZ PROCESOR (KOLIČINE DO 500 KOMADOV). SEVEDA PA STA PROCESORJA Z-8000 IN 68000 MOČNEJŠA OD 8086, VENDAR SE JE SLEDNJI ŽE ZNATNO UTRDIL NA TRŽIŠČU. NIZKA CENA INTELOVEGA PROCESORJA JE POSLEDICA POSEBNE TRŽNE STRATEGIJE.

APŽ

LABORATORIJ ZA MIKRO RAČUNALNIKE
Institut Jožef Stefan
Jamova 39, Ljubljana

TEMATIKA DIPLOMSKIH NALOG ZA DOMAČO RAČUNALNIŠKO PROIZVODNJO IN V OKVIRU RAZISKOVALNIH NALOG LABORATORIJA

Laboratorij za mikro računalnike izvaja skupaj z industrijo ter v okviru svojega raziskovalnega programa naloge z naslednjo temeljno problematiko:

- mikroračunalniške konfiguracije z 8-, 16- in 32-bitnimi mikro procesorji (najnovejša, strateška tehnologija) za avtomatizacijo postopkov in poslovno informatiko;
- osebni mikroračunalniški sistemi: tehnologija, konfiguracije, programska oprema, tržna strategija, uporaba;
- mikroračunalniški operacijski sistemi, monitorji, nalagalniki, urejevalniki, periferni procesorji, multiprocesorski sistemi;
- razvoj in raziskave razširljivih programov za mikro računalnike z uporabo metodologije informacijskih sistemov, tehniške umetne inteligence, strukturiranega programiranja;
- uporaba mikro računalnikov v telekomunikacijah ter pri kripciji (šifriranju) podatkov;
- vodenje majhnih robotov z mikro računalniki (inženirska robotika).

Pri izdelavi diplomske naloge je zagotovljena strokovna pomoč sodelavcev laboratorija, uspešna rešitev diplomske naloge, nagrajevanje po delu in objava diplomske naloge oziroma njene problematike v strokovnem tisku. Kandidati naj se javijo v sobi S-15 (2. nadstropje glavne stavbe IJS). Razpis velja do zasedbe razpoložljivih mest.

Prof.dr.Anton P. Železnikar, dipl. ing.

KRIPTOGRAFSKA LITERATURA

O KRIPTOGRAFIJI JE BILO OBJAVLJENIH VEČ PRISPEVKOV TUDI V STROKOVNIH GLASILIH NA PODROČJU RAČUNALNIŠTVA. PODJETJE IBM JE KOT PRVO UVEDLO KRIPTOGRAFSKE STANDARDE IN ALGORITME ZA KRIPTIRANJE (ŠIFRIRANJE IN DEŠIFRIRANJE TEKSTOV). EDEN TAKIH PRISPEVKOV JE ČLANEK "A CRYPTOGRAPHIC MANAGEMENT SCHEME FOR IMPLEMENTING THE DATA ENCRYPTION STANDARD" (EHRSAM), KI JE BIL OBJAVLJEN V IBM SYSTEMS JOURNAL, VOL. 17, NR. 2 IN GA JE MOGOČE NAROČITI PRI IBM ZASTOPSTVU POD ŠTEVILKO G321-5066 (Z DOLOČENO ODŠKODNINO). IBM IMA TUDI SPLOŠEN PRISPEVEK O KRIPTOGRAFIJI (CENA JE US$ 1.75), KI GA DISTRIBUIRA IBM SYSTEMS JOURNAL REPRINTS, ARMONK, NY 10504. DRUG PRISPEVEK JE FIPS PUBLICATION 46, KI GA JE MOČ DOBITI PRI US DEPARTMENT OF COMMERCE, KO JE CENA US$ 4.

APŽ

RAČUNARI PROJEKTUJU RAČUNARE

Kako se sve više kompjuterske arhitekture stavlja na poluprovodnička integrisana kola, potrebne su mnogo rafiniranije razvojne tehnike.
Za većinu kompjuterskih kompanija to znači projektovanje uz pomoč računara.

Skoro sve kompanije danas imaju neku vrstu programa za automatski i stoga kraći projektni proces. Možda je još važnije smanjivanje grešaka korišćenjem automatskog projektovanja, jer kad se jednom napravi kolo, teško ga je menjati.
IBM izgleda da ima najsavršeniji sistem automatskog projektovanja. Kompleksan sistem se koristi za projektovanje metalnih spajajućih slojeva na osnovnim podlogama integrisanih kola sa nizovima logičkih vrata (master slice gate array chip).
Šemom pakovanja postavlja se do 9 ovih logičkih kola na jedan keramički nosač 23 sloja, koji se tad stavlja na osmoslojnu karticu koja se ubada u 16-slojno povezanu spojnu ploču.
Automatski proces projektovanja se koristi takode za štampane veze na svim nivoima.
Ovakav sistem je instaliran u nekih 25 IBM organizacija širom sveta i omogučava prenos projekta sa jedne lokacije na drugu radi integracije u veći sistem. Kad je jednom projekt gotov, planovi se šalju svim proizvodnim pogonima gde se kompjuterski kontrolirani alati sa elektronskim mlazom koriste za ispisivanje maski na silikonu.
Prvi korak u tom procesu je unošenje logičkih funkcija u računar u formi lista preko alfanumeričkog terminala ili direktno diagrama preko grafičkih terminala. Projektanti tad porazdele funkcije između integrisanih kola. Računar sa nizom programa simulira logiku da proveri da li ona ustvari izvršava sve namjeravane funkcije kao što treba.
Zatim se automatski prevodi logika u fizički razpored kola. Kad dode do ove tačke, računar proverava da li kola zadovoljavaju sve aplikacione kriterijume i ograničenja.
Drugi program odlučuje o tačnom položaju svakog kola sa ciljem minimizira međusobne veze i maksimizira verovatnoču pronalaženja puteva za sve međusobne veze.
Poseban program spajanja tad polaže sve stvarne međusobne veze. Taj program takode vrši sva električna proverovanja o dozvoljenim padovima napona. U koliko program nije u stanju da realizira sve veze, projektant može sam intervenirati.
Dodatna proveravanja se vrše radi usagalašenosti raspo-reda sa geometrijskim parametrima tehnologije fabrikacije.
Kad je najzad fizički raspored finaliziran, računar generiše proceduru za ispitivanje. Određuju se ulazni stimulansi i očekivan izlaz.
Posle finalne verifikacije projektovanog integralnog kola, potrebni podaci se šalju proizvodnim pogonima.

Obzirom na svakodnevna usavršavanja tehnologije poluprovodnika, IBM projktant preporočuje sledeču strategiju: Predpostavite najagresivniju projekciju mogučnosti i cena najsavremenijih tehnologija o kojima možete sanjati i tad planirajte produkt koji će biti konkurentan!

M.M.M.

AVTORJI IN SODELAVCI

Mikroračunalnik EMZ 1001

ISKRA - Mikroelektronika je v sodelovanju z Laboratorijem za mikroelektroniko Fakultete za elektrotehniko Univerze Edvarda Kardelja v Ljubljani razvila mikroračunalnik v enem ohišju z oznako EMZ 1001. Mikroračunalnik ima izdelane operacije:

- vgrajena ura in časovne odločitve
- izpis segmentnih časovnih prikazov
- vpisovanje podatkov preko tastature
- aritmetične operacije itd.

Razvoj vezja se je pričel leta 1975 in je potekal v sodelovanju z ameriško firmo AMI. AMI, ki je po dogovoru z ISKRO solastnica vezja ga prodaja pod oznako S 2000. ISKRA je razvila programsko opremo za tri aplikacije v domačih izdelkih, ki bodo prišli na tržišče že letos.

Vezje je namenjeno predvsem uporabi v velikoserijskih izdelkih , ki potrebujejo sposobno nadzorno enoto na majhnem prostoru in za nisko ceno. Predvidena je uporaba v industrijskih avtomatih, instrumentih, registratorjih gogotkov, v blagajnah, elektronskih igrah itd.

Osnovni tehnični podatki mikroračunalnika EMZ 1001 so :

- tehnologija NMOS
- 40 izvodov
- vgrajen ROM, 1K 8 bit
- vgrajen RAM, 64 4 bit
- pretvorba 50 Hz/60 Hz v sekunde
- vgrajen dekoder za krmiljene 7 segmentov
- 13 izhodov, 8 vhodov, 8 dvosmernih I/O
- 51 ukazov
- ukazni cikel 4.5 s
- 3 nivoji klicanja subrutin
- možnost razširitve ROM z zunanjimi pomnilniki.

S.J.K.

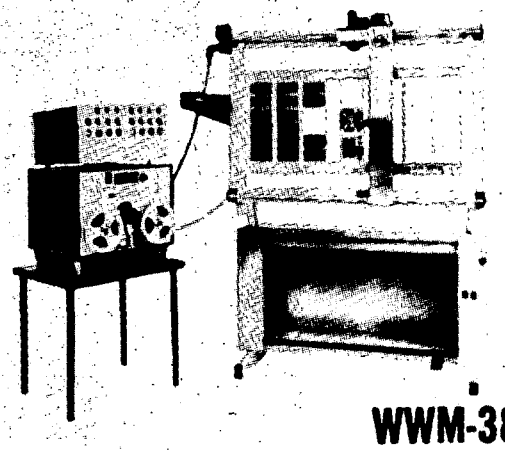
Bojan BARLIČ je diplomiral 1970. leta na Fakulteti za naravoslovje in tehnologijo, oddelek za kemijo, na Univerzi v Ljubljani. Po diplomi se je zaposlil na Odseku za uporabno matematiko na Institutu Jožef Stefan v Ljubljani, kasneje je delal na Republiškem računskem centru, sedaj pa je zaposlen na Kemijskem institutu Boris Kidrič. Podiplomski študij je končal leta 1973, prav tako na FNT, oddelku za kemijo Univerze v Ljubljani. Ukvarja se z različnimi problemi iz računalništva, največ z razvojem aplikativnih programov za reševanje problemov v kemiji.

Dušan RAIČ (1944), diplomiral na Fakulteti za elektrotehniko v Ljubljani leta 1968, magistriral 1972 in doktoriral leta 1977. Zaposlen je bil najprej na Zavodu za avtomatizacijo v Ljubljani, kjer je delal na uporabi mikro računalnikov v industriji. Od 1975 dela v Laboratoriju za mikroelektroniko na Fakulteti za elektrotehniko v Ljubljani, kjer se ukvarja z načrtovanjem digitalnih vezij LSI in z uporabo mikroračunalnikov.

Niko GUID (1951), diplomiral leta 1974 na Fakulteti za elektrotehniko v Ljubljani. Magistriral 1977 s področja teorije množične strežbe in njene aplikacije v računalništvu in telekomunikacijah. Zaposlen je kot asistent na Visoki tehniški šoli v Mariboru od marca 1979. Pred tem je delal v Iskri, kjer se je ukvarjal s testiranjem računalniške procesne periferije.

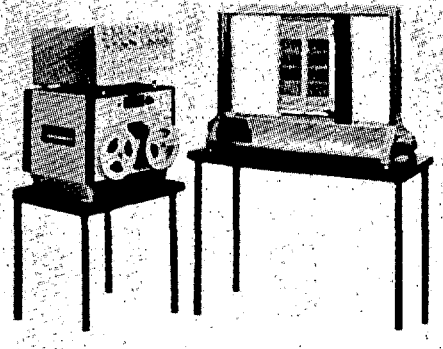
Borut KASTELIC (1955), diplomiral leta 1979 na Fakulteti za elektrotehniko v Ljubljani, smer avtomatika, s tematiko s področja mikroračunalništva. Zaposlen je na Institutu Jožef Stefan v Ljubljani, Oddelek za elektroniko. Ukvarja se s problematiko mikroračunalniških operacijskih sistemov.

WIRE WRAPPING SYSTEMS



WWM-380

Newest model features 18 character alphanumeric display for operator instructions under program tape control. Also features "absolute" locating system. 24 x 40 inch (61 x 102 cm) wiring area.

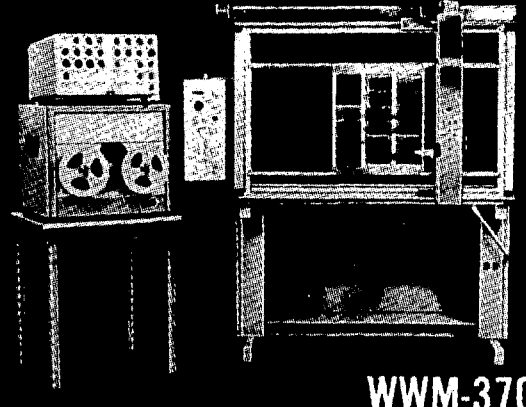


WWM-600

Lowest cost with full industrial features, fastest speed, highest accuracy. Completely portable. 17 x 24 inch (43 x 61 cm) wiring area.

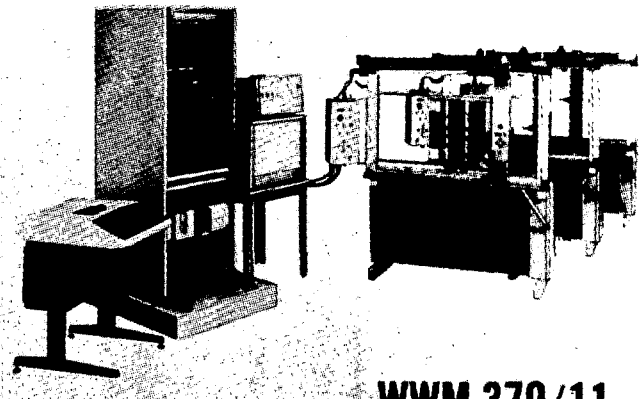
WWM-360

Same quality features as WWM-600 plus floor mounted 24 x 40 inch (61 x 102 cm) wiring area with adjustable height and tilt.



WWM-370

Bio-engineered control layout. Extra routing direction and pin displays. 24 x 40 inch (61 x 102 cm) wiring area. May be upgraded to computer control.



WWM-370/11

All features of WWM-370 plus operation under direct computer control instead of perforated tape. Permits multiple program management, program preparation and editing, and production monitoring.

OK MACHINE & TOOL CORPORATION

3455 Conner St Bronx N Y 10475 ■ (212) 994-6600 ■ Telex 125091

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 apaque , white paper and pasting it nearly over errors

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

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

Please enter my subscription to INFORMATICA and *send me the bill*.

Annual subscription price: companies 300,00 din (for abroad US $ 18), individuals 100,00 din (for abroad US $ 6)

Send journal to my ☐ home address ☐ company's address.

Surname.....

Name.....

Home address

Street.....

Postal code _____ City.....

Company address

Company.....

.....

Street.....

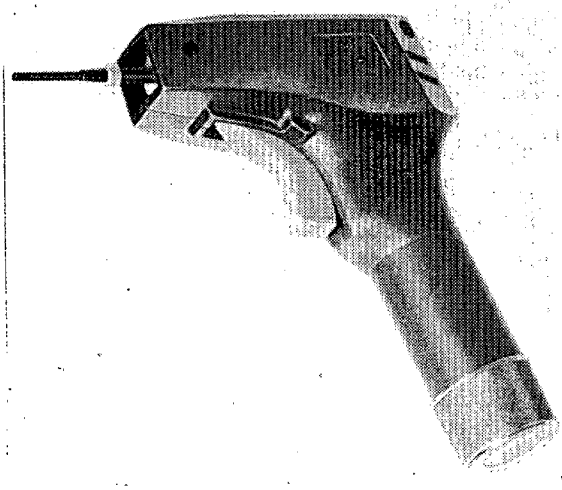
Postal code _____ City.....

Date..... Signature

.....

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

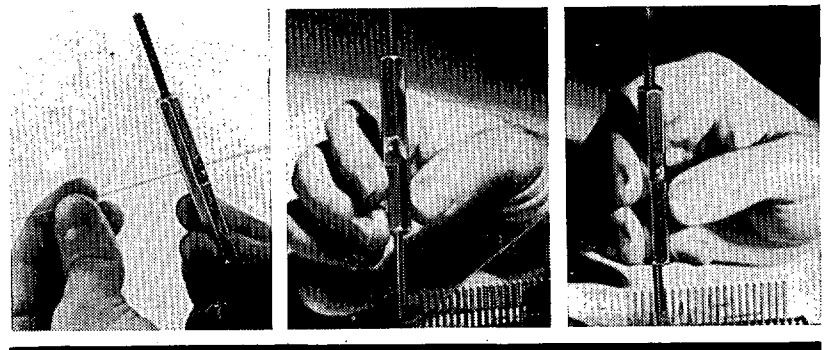
AMATERSKO ORODJE ZA OŽIČEVANJE

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

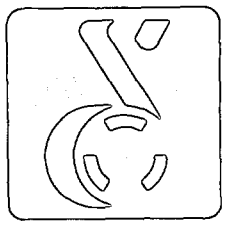
Ceneno orodje, ki opravlja funkcijo treh orodij, s podobno ceno. Z orodjem je mogoče ožičevati, odvijati in snemati izolacijo, s posebnim rezilom, vgrajenim v ročaj. Orodje primerno za delo z žico tipa 30 AWG (0,25 mm), katero se ovije na standardne (0,6 mm) trne podnožij za integrirana vezja. Uporabe 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



OK MACHINE & TOOL CORPORATION

Ko pišete proizvajalcu, omenite časopis INFORMATICA.



delta sistemi

ELEKTROTEHNA LJUBLJANA, TOZD za računalništvo Digital proizvaja in prodaja naslednje standardne računalniške konfiguracije:

DELTA 700/80

- DELTA 700 centralna procesna enota
- 512 KByte centralni pomnilnik s paritetno kontrolo,ki se lahko razširi do 4 MBytov
- 2 KByte vmesni pomnilnik spomina (cache)
- ura realnega časa
- konzolni terminal s kontrolno enoto
- dve diskovni enoti s kapaciteto po 80 MByte s kontrolno enoto
- dve magnetni tračni enoti 800/1600 b/i, 45 i/s, 9 kanalni zapis s kontrolno enoto
- asinhroni komunikacijski vmesnik
  (8 linij: EIA/CCITT modemski izhod)
  (8 linij: 20 mA tokovna zanka)
- 600 linijski tiskalnik
- KOPA 1000 alfanumerični video display terminal (2 kom)

DELTA 340/80

- DELTA 340 centralna procesna enota
- 256 KByte centralni pomnilnik s paritetno kontrolo
- 2 KByte vmesni pomnilnik (cache)
- ura realnega časa
- konzolni terminal s kontrolno enoto
- enota za baterijsko napajanje pomnilnika
- procesor s plavajočo vejico (floating point processor)
- dve diskovni enoti s kapaciteto po 80 MByte s kontrolno enoto
- dve magnetni tračni enoti (1600 b/i, 75 i/s, 9 kanalni zapis), s kontrolno enoto
- asinhroni komunikacijski vmesnik
  (8 linij EIA/CCITT modemski izhod)
  (8 linij 20 mA tokovne zanke)
- 600 linijski tiskalnik
- KOPA 1000 alafanumerični video display terminal
  (2 kom.)

DELTA 340/5

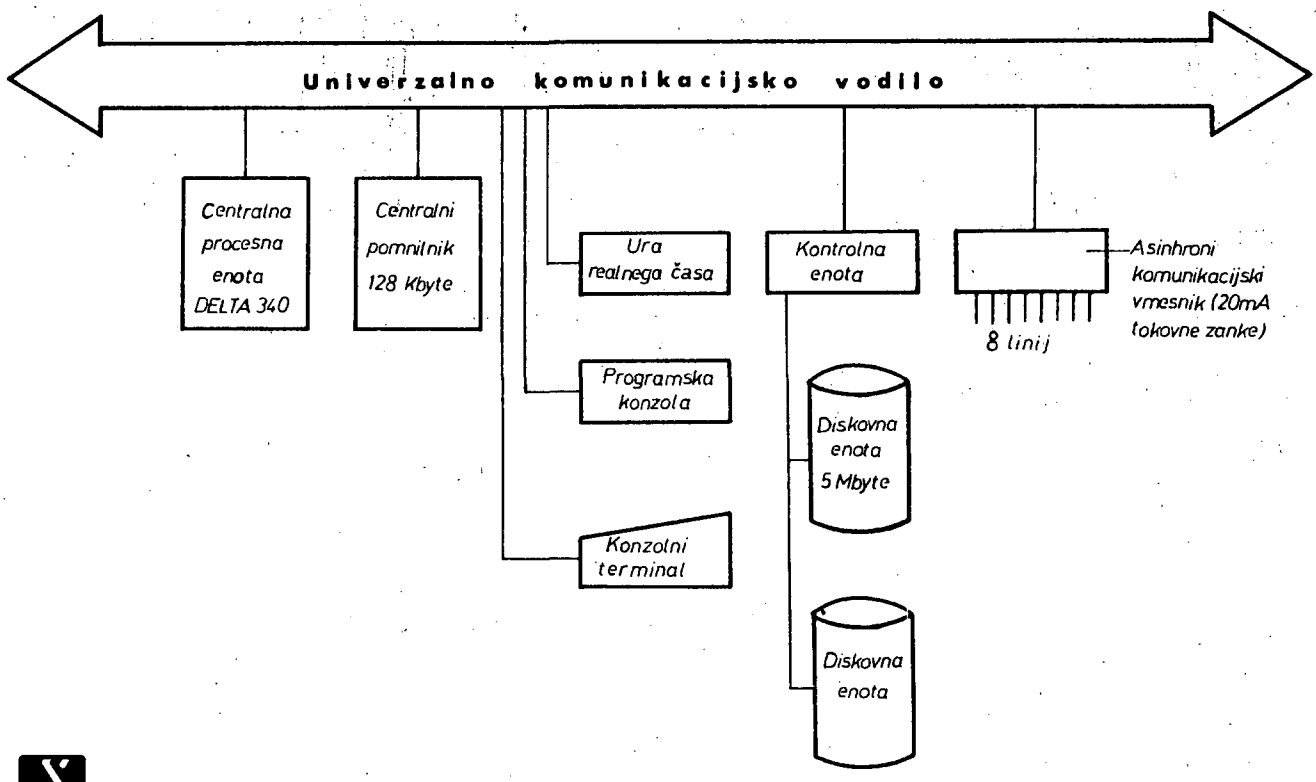
- DELTA 340 centralna procesna enota
- 128 KByte centralni pomnilnik s paritetno kontrolo, ki se lahko razširi do 256 KBytov
- ure realnega časa
- konzolni terminal s kontrolno enoto
- dve diskovni enoti s kapaciteto po 5 MByte s kontrolno enoto
- asinhroni komunikacijski vmesnik
  (8 linij: 20 mA tokovne zanke)

DELTA 340/40

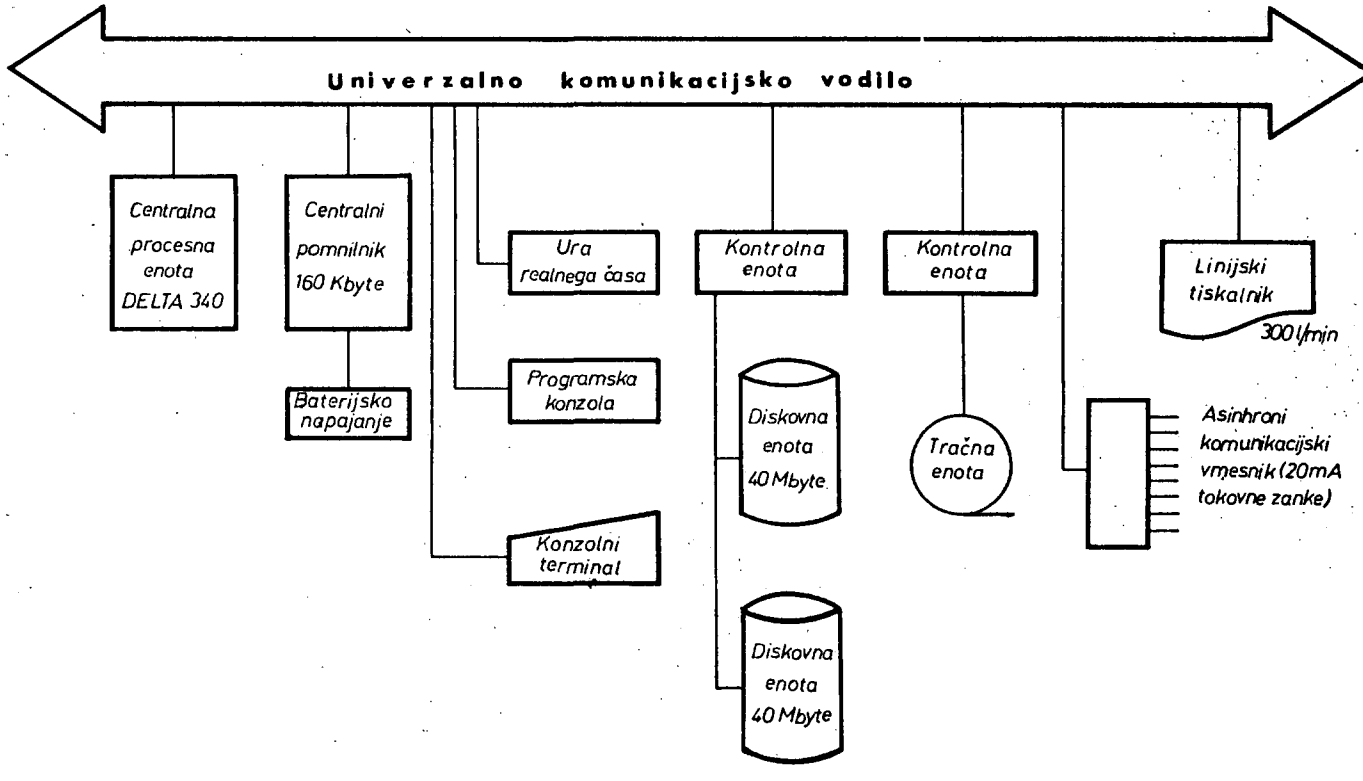
- DELTA 340 centralna procesna enota
- 160 KByte centralni pomnilnik s paritetno kontrolo do 256 KBytov
- ure realnega časa
- konzolni terminal s kontrolno enoto
- enota za baterijsko napajanje pomnilnika
- dve diskovni enoti s kapaciteto po 40 MByte s kontrolno enoto
- ena magnetna tračna enota (1600 b/i, 75 i/s, 9 kanalni zapis) s kontrolno enoto
- asinhroni kumunikacijski vmesnik
  (8 linij: 20 mA tokovne zanke)
- 300 linijski tiskalnik

NAŠTETE STANDARDNE KONFIGURACIJE LAHKO RAZŠIRITE S PRIKLJUČEVANJEM NOVIH VHODNO-IZHODNO ENOT, POVEČANJEM POMNILNIKA IPD.

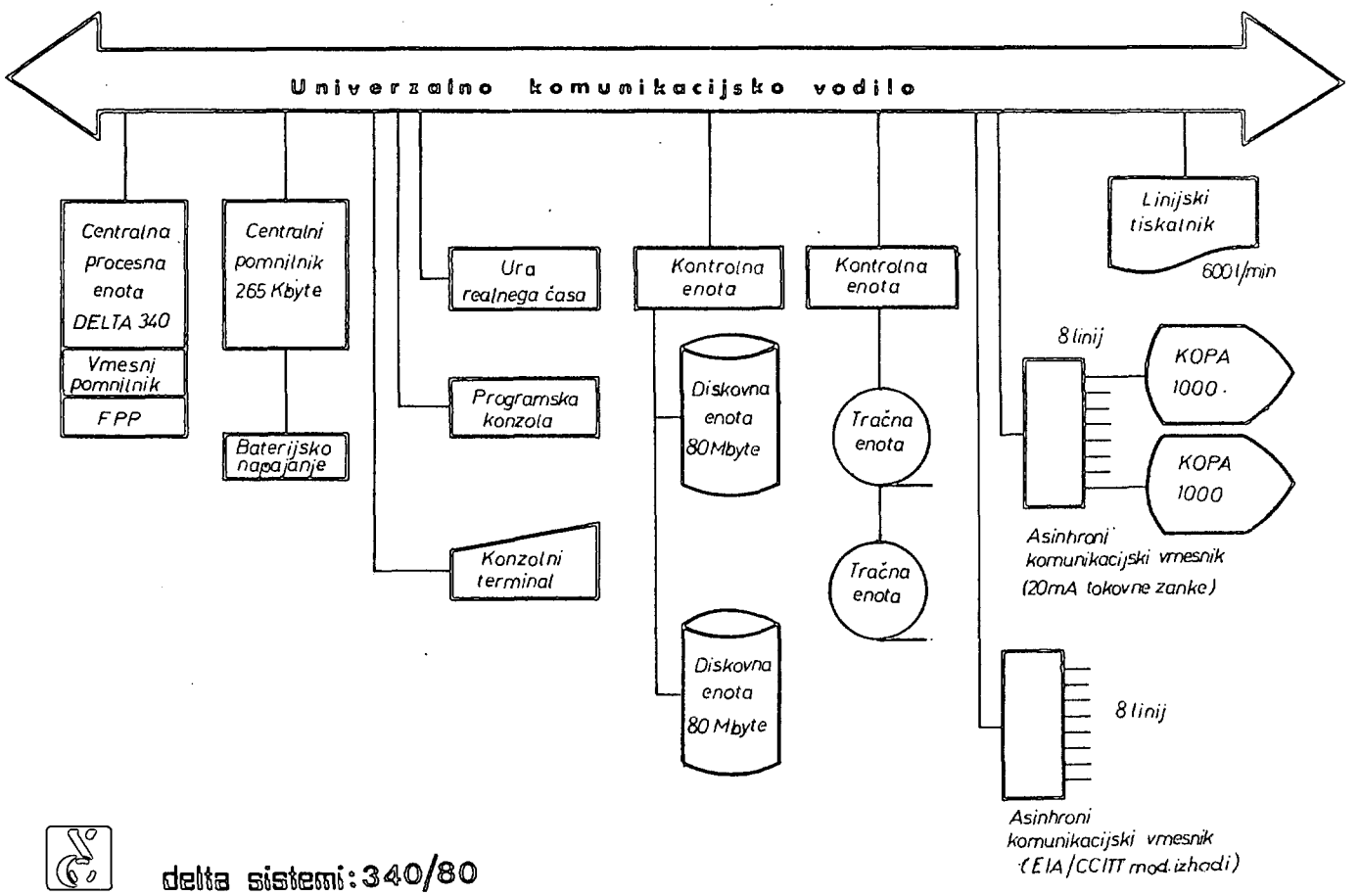
SISTEMSKI PAKETI DELTA 700/80, 340/80, 340/40 IN 340/5 VKLJUČUJEJO TUDI: OPERACIJSKI SISTEM DELTA/M S PREVAJALNIKI IN APLIKATIVNIMI PROGRAMI, ŠOLANJE V LASTNEM IZOBRAŽEVALNEM CENTRU, POMOČ PRI UVAJANJU PROGRAMSKE OPREME, INSTALACIJO RAČUNALNIŠKEGA SISTEMA IN ENOLETNO GARANCIJO ZA STROJNO IN PROGRAMSKO OPREMO.



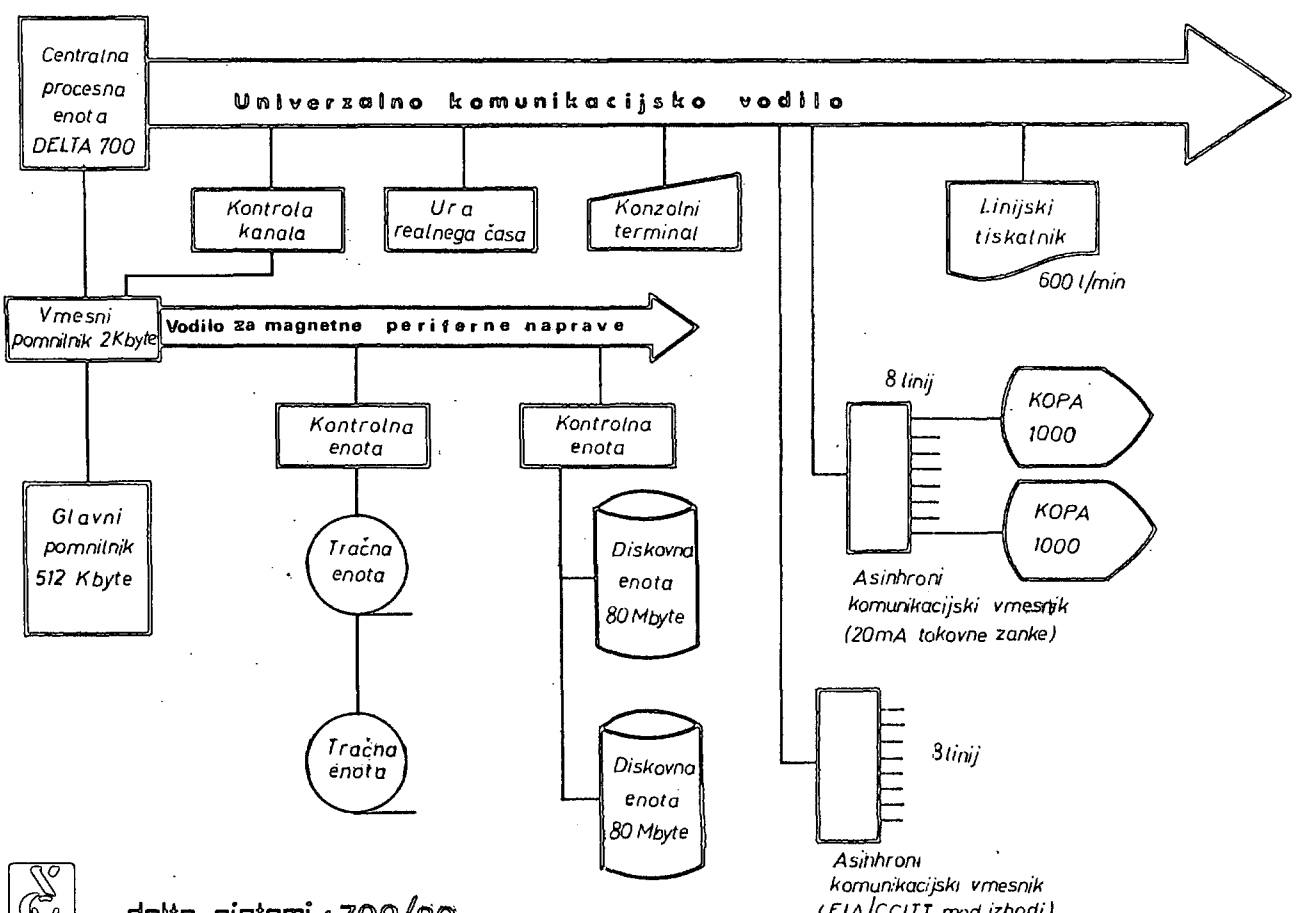
delta sistemi : 340/5



delta sistemi: 340/40



delta sistemi: 340/80



delta sistemi: 700/80

PROGRAMSKA OPREMA DELTA SISTEMOV

Osnovo sistemske programske opreme predstavlja DELTA/M operacijski sistem, ki je namenjen za delo v realnem času in časovno dodeljevanje resursov do 256 uporabnikom, ki lahko istočasno uporabljajo sistem. Glavna karakteristika DELTA/M sistema je interaktivnost. Človek in sistem komunicirata preko posebne enote, ki je običajno video terminal. Vsak monitorski ukaz se lahko vnese preko poljubnega terminala, če le uporabnikovo geslo zadošča ustrezni stopnji tajnosti. To pomeni z vidika uporabnika enake možnosti, kot da bi delal sam na sistemu.

Večuporabniško okolje zahteva zaščito med uporabniki samimi, saj bi lahko napaka enega uporabnika povzročila težave vsem drugim. Zaradi tega obstaja med uporabniki zaščita na nivoju programske opreme in na nivoju strojne opreme. Vsak disk je razdeljen v več logičnih področij, od katerih jih vsak uporabnik lahko nekaj uporablja . Praktično to pomeni, da lahko briše samo svoje nize in bere nize drugih uporabnikov, če mu le-ti to dovolijo. Elektronsko pa je zaščiten adresni prostor programov in uporaba instrukcij, ki bi lahko porušile integriteto sistema. Te lahko uporablja samo izvajalni sistem.

Multiprogramiranje je realizirano na nivoju sistema kot celote in na nivoju posameznega terminala. Tako ima lahko vsak uporabnik lastni multiprograming. To je važno predvsem za programerje, saj lahko istočasno razvijajo (prevajalnik, povezovalnik) in testirajo (izvajajo) programe.

Velika hitrost procesorja in perifernih enot ter učinkovito oblikovana programska oprema omogočata gospodarno uporabo vseh komponent DELTA računalnika.

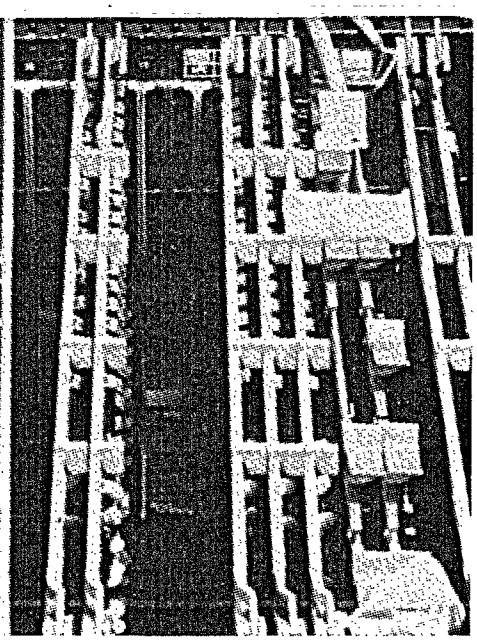
Sistem lahko istočasno upravlja industrijski proces(visoka prioriteta - realni čas), interaktivne poslovne aplikacije(srednja prioriteta), razvoj novih programov v poljubnih programskih jezikih(standardna prioriteta) in paketne obdelave (nizka prioriteta).

Aplikacijski programi se lahko pišejo v MACRO zbirnem ali enem od višjih programskih jezikov:

- FORTRAN IV
- FORTRAN IV PLUS
- BASIC 11
- RPG II
- COBOL (ANSI 74 standard)
- BASIC-PLUS-2
- PASCAL
- DATATRIEVE 11

Na tržišču ugotavljamo velike potrebe po kvalitetni komunikacijski opremi, zato posvečamo veliko pozornost prav temu področju. Komunikacijska programska oprema na DELTA/M je eden od poslov, ki se odvija v multiprogramingu in omogoča povezavo z računalniki: DELTA, PDP-11, VAX, DEC-10, DEC-20, CDC-6600, IBM 360/370, UNIVAC-11.

DELTA sistemi so namenjeni splošni uporabi. Zato je v osnovni paket vedno vključena samo tista programska oprema, ki je potrebna vsem uporabnikom. Vsak pa si lahko izbere dodatno sistemsko ali aplikativno programsko opremo. DELTA/M namreč ohranja popolno kompatibilnost navzdol z RSX-11/M operacijskim sistemom firme DEC. Ta operacijski sistem je zelo razširjen, zato je tudi ponudba ELEKTROTEHNE, DEC-a in drugih proizvajalcev zelo velika.



PODROBNE INFORMACIJE O NAKUPU DELTA SISTEMOV NUDI ELEKTROTEHNA LJUBLJANA, TOZD ZA RAČUNALNIŠTVO DIGITAL:

LJUBLJANA
Linhartova 62a
tel. (061) 323-585

ZAGREB
Aleja Borisa Kidriča 2
tel. (041) 516-690

BEOGRAD
Karađorđev trg 13
tel. (011) 694-537