ČASOPIS SAVEZNOG STRUČNOG ODBORA ZA AUTOMATIZACIJU JUGOSLAVENSKOG KOMITETA ZA ETAN

JOURNAL OF TECHNICAL BOARD FOR AUTOMATION OF THE YUGOSLAV COMMITTEE ETAN

automatika

SEPARAT

G. IX 1968 Br. 2-3/68

LITERATURA

1. J. W. Perry and Allen Kent: **Tools for Machine** Literature Searching. Interscience Publishers. Inc., New York 1958
2. B. C. Vickery, M. A.: **Classification and Indexing in Science** Butterworths Scientific Publications, London, 1959
3. Allen Kent: **Information Retrieval and Machine Translation** Interscience Publishers, Inc., New York 1960
4. Dr. Hans Rudolf Schenk: **Der Einsatz von Lochkartenmaschinen und Computers in der Codeless Scanning-Methode** Nachrichten für Documentation, 3, 1961
5. Joseph Becker **Information storage and Retrieval** John Wiley 1963, New York, London
6. B. V. Jakušin: **Jazyk klassifikacionu-deskriptornogo tipa dlja IPS, realizuemoj na sčjotno-perforacionnyh mašinah** Naučno tehničeskaja informacija, 4, 1966
7. Morris Rubinoff: **A Rapid Procedure for Launshing a Micro-thesaurus** IEEE Transaction, July 1966

AVTORJEV NASLOV:

Stanko Črepinšek, dipl. ing., Zavod za avtomatizacijo, Ljubljana, Tržaška c. 2.

ELEKTROINŠTITUT
LJUBLJANA, Hajdrihova 2
opravlja študijsko razvojno delo s področja elektriškega gospodarstva in elektroindustrije

Združeno podjetje
ISKRA KRANJ
TOVARNA ELEMENTOV ZA ELEKTRONIKO LJUBLJANA

elma **tovarna elektromateriala**
LJUBLJANA — ČRNUČE
Telefon 341-005
Telex 31-218
priporoča svoje kvalitetne izdelke!

B. Souček, K. Čuljat, V. Bonačić

Digitalna računala u mjerenju i regulaciji

DK 681.14—523.8
IFAC IA 4.5.1

U mjerenju i regulaciji naglo se širi primjena jeftinih minijaturnih računala. U članku je opisana ova primjena, sa specijalnim osvrtom na slijedeće tačke:
Malo laboratorijsko računalo. Reprezentacija brojeva i strojno programiranje. Vrste instrukcija. Binarna i simbolička reprezentacija programa. Funkcionalna organizacija digitalnog računala. Logika transfera podataka. Selekcija vanjskih jedinica tj. mjernih i kontrolnih mjesta. Programirani transfer. Transfer s prekidom programa. Jednociklusni transfer. Primjer računala u mjerenju. Primjer računala u regulaciji.

Digital Computers in Measuring and Regulation. There is rapid growth of applications of miniature digital computers for measurement and control purposes. In the paper this application is reviewed, with special attention to the following subjects:
Small laboratory computers. Representation of numbers and machine language programming. Kinds of instructions. Binary and symbolic programming. Functional organization of digital computer. Selection of peripheral units, i. e. of measurement and control points. Programmed data transfer. Interrupt transfer. Cycle stealing transfer. Example of computer in measurement. Example of computer in control.

1. Uvod

1.1. Malo laboratorijsko računalo

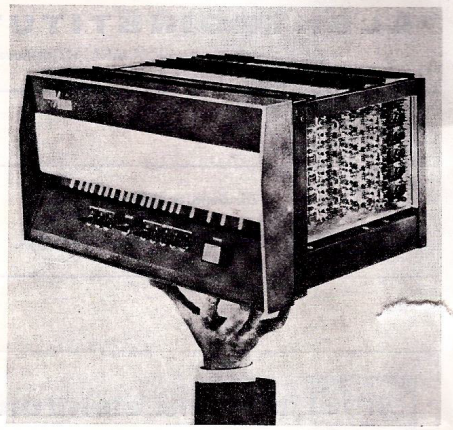
Laboratorijsko računalo je novi elektronički instrument čija primjena brzo raste. Upotrebljava se za skupljanje podataka iz procesa, za preliminarnu analizu, za praćenje eksperimenta i za kontrolu procesa i mjernih

Sl. 1. a)

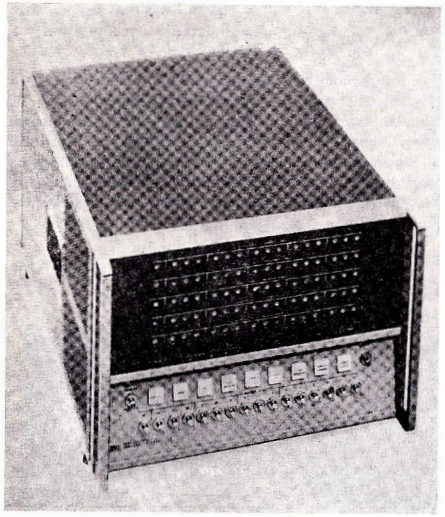
aparatura. U generalnom konceptu laboratorijsko računalo je slično ostalim digitalnim računalima. Međutim, tehničke karakteristike podešene su specifičnostima zadatka. Glavni naglasak dan je na brzinu i na mogućnost obrade podataka onako, kako dolaze s procesa (obrada u stvarnom vremenu — real time processing).

Od 1966. digitalna računala gotovo se isključivo grade od integriranih mikrokrugova, čime im je cijena i veličina bitno smanjena u odnosu na ranija tranzistorska računala. Od svih računala laboratorijska su najmanja jer se od njih ne traže kompleksne matematske operacije već jednostavne mjerne i kontrolne funkcije. Veličina im je usporediva s veličinom osciloskopa, a cijene krajem 1967. kreću se od 8000 do 16000 $ (zajedno s teleprinterom, te čitačem i bušačem trake). Time je automatsko upravljanje procesima i eksperimentima postalo mnogo pristupačnije korisnicima. Umjesto gradnje kompleksnog sistema »po mjeri« za svaki specifični zadatak, danas je jednostavnije kupiti »konfekcijski« proizvedeno jeftino laboratorijsko računalo i prilagoditi ga konkretnom zadatku. Ovakvo rješenje je ne samo znatno jeftinije već omogućuje konstrukciju sistema u mnogo kraćem vremenu. Također, u slučaju potrebe ovaj sistem je lako adaptabilan na drugi zadatak. Stoga sve češće laboratorijska računala postaju bitni dio mjernih instrumenata i kontrolnih sistema. Slika 1 prikazuje nekoliko integriranih laboratorijskih računala raspoloživih na tržištu u 1967. god.

U završnoj fazi razvoja je nova tehnologija integriranih mikrokrugova poznatih pod nazivom LSI (large scale integration). Očekuje se da će ova tehnologija još dalje bitno reducirati cijenu i veličinu digitalnih računala.



Sl. 1. b)



Sl. 1. c)

Laboratorijska računala, premda namijenjena nematematskim zadacima, mogu se upotrebljavati i za naučne i tehničke matematske proračune i redovito su opremljena i matematski orijentiranim programerskim jezicima (tipa »FORTRAN«).

1.2. Primjena laboratorijskih računala

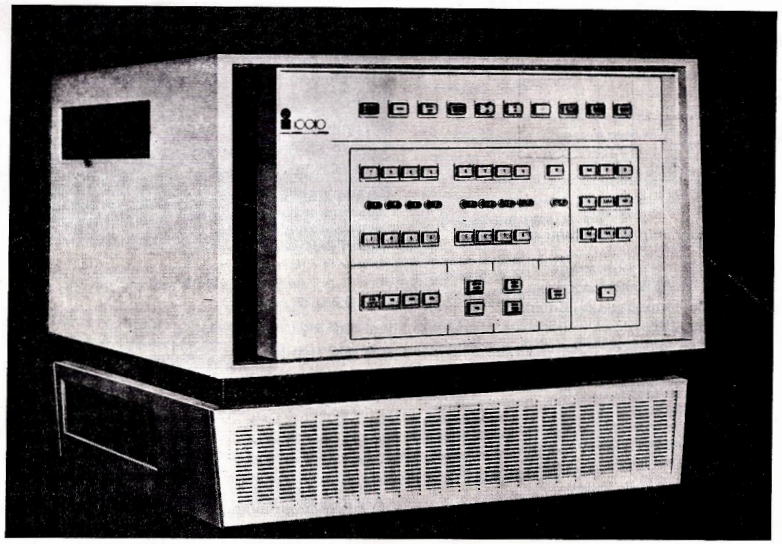
Primjena laboratorijskih računala vrlo brzo raste na širokom području, a može se podijeliti na tri osnovne kategorije: redukcija podataka, programirani kontrolni sistemi i sistemi s digitalnom povratnom vezom.

Sistemi za redukciju podataka mjere analognu informaciju iz procesa, pretvaraju je u digitalnu i pomoću laboratorijskog računala transformiraju je u sažetiji oblik podesan za direktnu upotrebu. Blok shema takvog sistema prikazana je na slici 2. Ovaj sistem ima dvije moguće varijante poznate pod nazivima »off-line« i »on-line«.

Kod »off-line« sistema izmjereni i digitalizirani podaci privremeno se memoriraju na nekom mediju (bušena traka, magnetska traka itd.). Nakon što je mjerenje gotovo podaci se iz privremene memorije prebacuju u digitalno računalo gdje se definitivno obrađuju. Prednost ovakvog načina rada je što ne zahtijeva električko spajanje procesa s računalom.

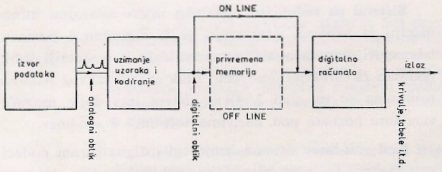
Kod »on-line« sistema nema privremenog memoriranja već podaci iz procesa direktno ulaze u laboratorijsko računalo. Zbog toga je i konstrukcija ovakvih sistema kompleksnija, ali omogućuje praćenje promjena na procesu istovremeno kako se događaju.

Programirani kontrolni sistemi počinju s laboratorijskim računalom koje sadrži potrebne numeričke podatke i program rada. Ti podaci se dekodiraju i upravljaju fizikalnim sistemom kao npr. alatnim strojevima, pokrećući ih po unaprijed zadanom programu. Blok shema ovakvog sistema prikazana je na slici 3.



Sl. 1 a), b), c), d) — Primjeri laboratorijskih digitalnih računala. Osnovne karakteristike: 4000 riječi memorija; brzina 1 — 2 μsec; FORTRAN; Cijena 8—16000 $ (računalo sa teleprinterom)

Kod **sistema s digitalnom povratnom vezom** laboratorijsko računalo nalazi se u kontrolnoj petlji povratne veze, kao što se vidi sa slike 4. Problemima kao što su nestabilnost petlje i vremenska kašnjenja u petlji povratne veze treba posvetiti posebnu pažnju. Ovi sistemi omogućuju visoku tačnost i brzi odziv.



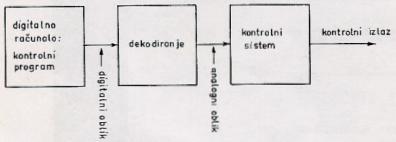
Sl. 2. Sistem za redukciju podataka

U ovom članku neće se razmatrati svi problemi koji mogu nastati u sistemima sa slike 2, 3 i 4. Pažnja će biti posvećena samo načinu, kako laboratorijsko računalo uklopiti u ovakove sisteme. Prilikom sastavljanja ovog članka uzeto je u obzir da je ova problematika kod nas nova, te su stoga opisani i neki osnovni elementi. Obrađena problematika ilustrirana je izgrađenim sistemima opisanim u referencama (1) i (2).

2. Reprezentacija brojeva i strojno programiranje

2.1. Brojni sistemi i binarna aritmetika

Digitalna računala rade s informacijom u numeričkom obliku, dakle s brojevima. Broj je veličina reprezentirana grupom znamenaka, a svaka znamenka može poprimiti razne vrijednosti. Za svakodnevni život čovjek koristi dekadski brojni sistem. Znamenke ovog sistema poprimaju



Sl. 3. Programirani kontrolni sistem

vrijednosti 0 do 9, to je sistem s bazom 10. Taj sistem čovjek je vjerojatno izabrao stoga što ima deset prstiju pomoću kojih simbolizira deset vrijednosti znamenaka. Brojeve je međutim moguće prikazati i u sistemima s drugim bazama. Broj N može se prikazati pomoću njegovih znamenaka d_i ovako

$$N = \sum_{i=0}^{n} d_i B^i = d_0 + d_1 B^1 + d_2 B^2 + \ldots + d_n B^n$$

gdje je B baza sistema. Npr. u dekadskom sistemu

$$2687 = 2 \cdot 10^3 + 6 \cdot 10^2 + 8 \cdot 10^1 + 7 \cdot 10^0$$

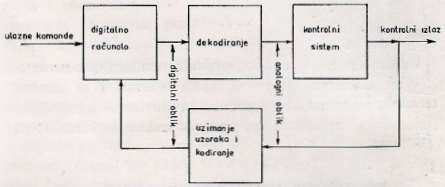
Kako je u elektrotehnici lakše tehnološki proizvesti elemente sa dva diskretna stanja nego sa deset, u digitalnoj tehnici prevladao je brojni sistem sa **bazom dva**. Binarni broj

$$b_3\, b_2\, b_1\, b_0 = b_3 \cdot 2^3 + b_2 \cdot 2^2 + b_1 \cdot 2^1 + b_0 \cdot 2^0$$

Npr. u binarnom brojnom sistemu je $1010 = 1 \cdot 2^3 + 0 \cdot 2^2 + 1 \cdot 2^1 + 0 \cdot 2^0$ što je u dekadskom sistemu $8 + 2 = 10$. Digitalni sistemi rade s binarnim brojevima. Svi binarni brojevi mogu se grupiranjem po tri binarne znamenke u jednu prikazati u oktalnom sistemu čime se dobiva na preglednosti. Tabela 1 prikazuje niz brojeva u spomenuta tri sistema. Konverzija decimalnog broja u binarni vrši se

Tabela 1

Dekadski	Binarni	Oktalni
0	0000	0
1	0001	1
2	0010	2
3	0011	3
4	0100	4
5	0101	5
6	0110	6
7	0111	7
8	1000	10
9	1001	11
10	1010	12



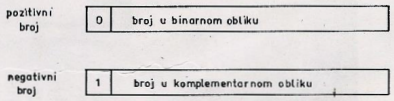
Sl. 4. Sistem s digitalnom povratnom vezom

sukcesivnom diobom decimalnog broja sa 2, pri čemu ostaci predstavljaju binarne znamenke.

Primjer: pretvorba dekadskog broja 14 u binarni

dioba	ostatak (bin. znamenke)
$14 : 2 = 7$	0
$7 : 2 = 3$	1
$3 : 2 = 1$	1
$1 : 2 = 0$	1

Dakle $14_{10} = 1110_2$



Sl. 5. Reprezentacija predznaka u računalu

Binarno zbrajanje je identično dekadskom kao što se vidi iz slijedećih primjera, s tim da se »jedan dalje« uzima kod rezultata 2, a ne 10.

1	1010
+ 1	+ 0011
10	1101

Za prikazivanje negativnih brojeva koristi se tzv. »komplementarni« oblik broja, koji se dobije zamjenom svih nula jedinicama i obrnuto. Jedino se krajnje desna jedinica i eventualne nule desno od nje ne mijenjaju.

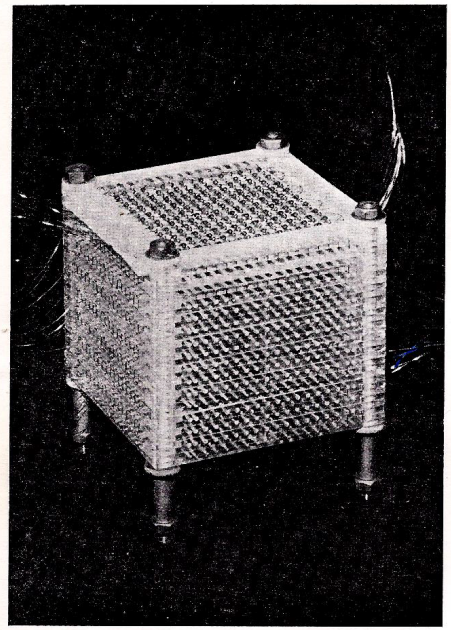
Primjer: binarni prikaz broja 6: 00110

komplement broja 6: 11010

Ovaj način prikaza negativnih brojeva omogućuje izvršavanje operacije odbijanja pomoću logike i pravila za zbrajanje.

Primjer:

9	binarna reprezentacija broja 9:	01001
— 6	komplement broja 6:	+ 11010
3		00011



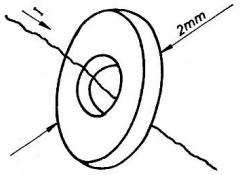
Sl. 6. a) Memorija digitalnog računala

Vidi se da komplementarna reprezentacija ima krajnje lijevu znamenku jednaku jedinici te se ona upotrebljava kao oznaka predznaka u računalima prema slici 5.

2.2. Principi programiranja digitalnih računala

Centralni dio digitalnog računala je memorija, koja služi za memoriranje informacija. Sve informacije imaju oblik binarnih brojeva. Memorija se sastoji od velikog broja riječi (word), a u svaku riječ može se smjestiti jedna informacija.

Gotovo isključivo se danas proizvode feritne memorije, slika 6.a.* Osnovni element memorije je feritni prsten ili jezgrica, slika 6.b. Jedna jezgrica može »pamtiti« jedan bit informacije, a tu funkciju omogućavaju njena magnetska svojstva. Informacija se upisuje u jezgricu tako da se pošalje strujni impuls u jednom smjeru kroz žicu, koja prolazi kroz jezgricu. Kad pobuda (strujni impuls) nestane, jezgrica ostaje u jednom definiranom magnetskom stanju, koje se zove remanentni magnetizam, npr. $+Br$. Strujni impuls suprotnog smjera ostavlja jezgricu u drugom magnetskom stanju, $-Br$. Magnetskim stanjima $+Br$ i $-Br$ mogu se pridijeliti vrijednosti 1 odnosno 0 binarne znamenke.

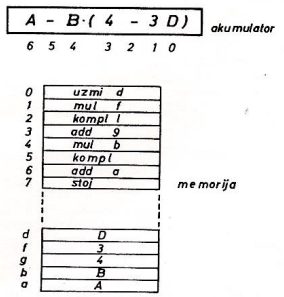


Sl. 6. b) Osnovni element memorije — feritna jezgrica

Za n bitni binarni broj odnosno n bitnu informaciju potreban je niz od n jezgrica. Takav niz jezgrica se u sklopu memorije naziva riječ, a memorije digitalnih računala imaju obično 4 do 32 tisuće riječi. Svaka riječ memorije ima svoju adresu pomoću koje se identificira. Broj jezgrica odnosno broj bita u riječi obično varira od 12 do 36.

Memorija služi za uskladištavanje dviju različitih vrsti informacija: podataka i instrukcija. Podaci su brojevi nad kojima se vrše neke računske operacije. Instrukcije su upute za rad digitalnim računalom, a niz takvih uputa kao jedna cjelina predstavljaju program. Računalo izvršava program korak po korak izvršavajući po jednu instrukciju u jednom vremenskom intervalu.

Digitalno računalo prilikom izvršavanja jedne instrukcije operira s najviše dva operanda (broja). Jedan operand nalazi se u binarnom registru koji se zove akumulator. Drugi se nalazi u memoriji stroja, a instrukcija mora specificirati njegovu adresu (adresu lokacije memorije u kojoj je operand smješten).

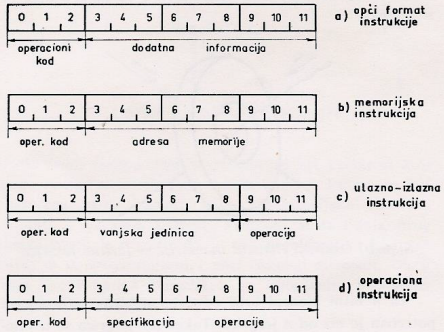


Sl. 6. c) Prikaz sadržaja memorije kod izvršavanja jednog programa. Uz akumulator je brojevima 0 do 6 označen redosljed promjene sadržaja akumulatora prilikom izvršavanja programa

* Na slici je prikazana feritna memorija izrađena na Institutu »Ruđer Bošković«.

Instrukcije imaju oblik kao npr. »uzmi *d*«, pri čemu računalo uzima sadržaj lokacije *d* memorije i prebacuje ga u akumulator. Ako nakon toga slijedi instrukcija »multipliciraj *f*«, računalo množi sadržaj akumulatora sa sadržajem lokacije *f*, i ostavlja rezultat u akumulatoru. Memorija i akumulator prikazani su na slici 6a.

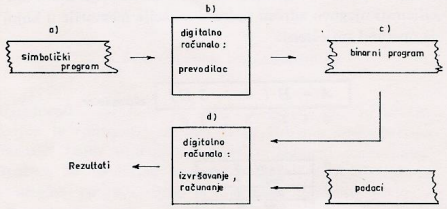
Slijedeći primjeri pokazuju neke osnovne oblike programa.



Sl. 7. Vrste instrukcija

Pravocrtni program izvršava se korak po korak u neprekinutom nizu. Primjer pokazuje programiranje izračunavanja vrijednosti izraza *A-B* (4-3D)

0/ uzmi *d*	(*d* sadrži *D*)
1/ mul *f*	(*f* sadrži 3)
2/ kompl	
3/ add *g*	(*g* sadrži 4)
4/ mul *b*	(*b* sadrži *B*)
5/ kompl	
6/ add *a*	(*a* sadrži *A*)
7/ stoj	



Sl. 8. Prevođenje simboličkog oblika strojnog programa u binarni oblik

U ovom primjeru lokacije *f* i *g* sadrže konstante 3 odnosno 4, a lokacije *a*, *b*, *d* sadrže parametre *A*, *B* i *D*. Lokacije 0 do 7 sadrže sam program, kao što je prikazano na slici 6c.

Program s petljom omogoćuje da se jedan dio operacija nekoliko puta ponovi. Slijedeći primjer prikazuje program koji računa izraz $35\,(A+B)$ za sve cjelobrojne vrijednosti od *A* počevši s nulom i uskladištuje rezultate u sukcesivne lokacije memorije počevši od adrese *h*.

Izračunavanje izraza

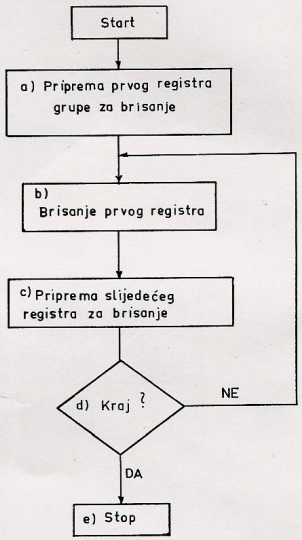
0/ uzmi *f*	(sadrži *A*, u početku *0*)
1/ add *b*	(sadržaj od $b = B$ tj. $b = B$)
2/ mul *g*	([*g*] = 35)
3/ spremi u *h*	

Priprema nove adrese

4/ uzmi 3	([3] = spremi u *h*)
5/ add *c*	([*c*] = 1)
6/ spremi u 3	([3] = spremi u $h+1$)

Priprema novog parametra A

7/ uzmi *f*	([*f*] = *A*)
10/ add *c*	
11/ spremi u *f*	([*f*] = $A+1$)
12/ skoči u *0*	



Sl. 9. Primjer dijagrama toka

Već prve 4 instrukcije izračunavaju izraz. Slijedeće tri instrukcije samo pomiću za jedinicu adresu u koju se stavlja rezultat. Daljnje tri instrukcije povećavaju za jedan parametar *A*. Zadnja »skok« instrukcija omogućuje računalu ponavljanje programa za slijedeću vrijednost parametra.

Program s ogranicima ne izvršava se uvijek istim redosljedom instrukcija, već ima mogućnost grananja. To prikazuje slijedeći primjer koji izračunava 2^{10}.

početni uvjeti

0/ uzmi *f*	([*f*] = —10)
1/ stavi u *g*	
2/ uzmi *m*	([*m*] = 1)
3/ stavi u *k*	

jednostruko potenciranje

4/ uzmi *k*	
5/ mul *h*	([*h*] = 2)
6/ stavi u *k*	

test broja petlji

7/ uzmi *g*	([*g*] = —10)
10/ add *m*	([*m*] = 1)
11/ stavi u *g*	
12/ preskoči ako je *0*	
13/ skoči u 4	
14/ nastavak	

U gornjem primjeru prve 4 instrukcije osiguravaju početne uvjete programa. Slijedeće tri instrukcije obavljaju stvarni posao jednostrukog potenciranja, a naredne 4 ispituju da li je program izvršen 10 puta. Ako nije, program se ponavlja u petlji skačući u lokaciju 4. Nakon 10 prolaza kroz petlju program preskoči u adresu 14 tj. na novi ogranak, jer je ispunjen uvjet, da je na akumulatoru nula.

2.3. Vrste instrukcija

Navedena tri primjera pokazuju kako se zadatak za digitalno računalo razbija na elementarne instrukcije. Svako računalo ima ograničeni broj elementarnih instrukcija koje se razlikuju od stroja do stroja, ali su izabrane na bazi istih principa. O kakvim tipovima instrukcija se radi bit će pokazano na konkretnom primjeru digitalnog računala PDP-8.

PDP-8 je laboratorijsko digitalno računalo koje ima memoriju od 4096 riječi, a svaka riječ dužine je 12 bita (binary digit tj. binarna znamenka). Sve instrukcije zbog toga također imaju 12 bitnu binarnu konfiguraciju.

Prva tri bita (»operacioni kod«) određuju tip instrukcije ,pa PDP-8 ima $2^3 = 8$ osnovnih instrukcija. One su označene oktalnim brojevima 0 do 7 i definirane u tabeli 2. Podijeljene su na tri klase: memorijske, ulazno-izlazne i operacione, slika 7.

Memorijske instrukcije, slika 7 b, uzimaju jedan operand iz memorije. Zbog toga moraju odrediti vrstu operacije koja će se izvršiti nad operandom i adresu lokacije memorije u kojoj se operand nalazi. Drugi operand se nalazi na akumulatoru. Memorijske instrukcije imaju operacione kodove 0 do 5.

Ulazno-izlazne instrukcije, slika 7 c, vrše prebacivanje podataka između akumulatora i perifernih uređaja. Njihov operacioni kod (prva tri bita) je 6. Preostali bitovi selektiraju periferni uređaj i određuju redosljed događaja.

Operacione instrukcije, slika 7 d, ne saobraćaju niti memorijom niti s perifernim uređajima. Jedna grupa ovih podinstrukcija vrši operacije s podatkom koji se već nalazi u akumulatoru. Druga grupa preskače ili ne instrukciju koja ih slijedi ovisno o rezultatu testiranja stanja akumulatora i time omogućuje grananje u slijedu programa. Operacioni kod im je 7. Razne kombinacije preostalih 9 bita specificiraju razne podinstrukcije.

Za detaljnije razumijevanje instrukcija treba se vratiti na tabelu 2 nakon diskusije o slici 13.

2.4. Binarna i simbolička reprezentacija programa

Upotreba navedenih instrukcija vidi se na slijedećem primjeru programa koji treba zbrojiti sadržaje lokacija 100 i 101 i spremiti rezultat u lokaciju 102. Program je smješten u memoriji u lokacijama 103 do 106, a prikazan je na tabeli 3.

Tabela 2

Instrukcije	Operacioni kod	Simbol. ime	Opis
Memorijske	0	*AND*	»logički i«
	1	*TAD*	binarno zbrajanje
	2	*ISZ*	inkrement i preskok, ako je nula
	3	*DCA*	deponiranje i brisanje *AC*
	4	*JMS*	skok u potprogram
	5	*JMP*	skok
Ulazno-izlazne	6	*IOT*	ulazno-izlazni transfer
Operacione	7	*IAC*	inkrement *AC*
		CLA	brisanje *AC*
		CLL	brisanje linka (*L*)
		CMA	komplement *AC* (ovdje je komplement međusobna zamjena nula i jedinica)
		CML	komplement *L* (ovdje je komplement međusobna zamjena nula i jedinica)
		RAL	rotiranje sadržaja *AC* i *L* za jedno mjesto u lijevo
		RTL	rotiranje sadržaja *AC* i *L* za dva mjesta u lijevo
		RAR	rotiranje sadržaja *AC* i *L* za jedno mjesto u desno
		RTR	rotiranje sadržaja *AC* i *L* za dva mjesta u desno
		OSR	»logički ili« između konzolnog [»switch«] registra i *AC*
		SKP	bezuvjetni preskok
		SNL	preskok ako je sadržaj $L \neq 0$
		SZL	preskok ako je sadržaj $L = 0$
		SNA	preskok ako je sadržaj $AC \neq 0$
		SZA	preskok ako je sadržaj $AC = 0$
		SMA	preskok ako je sadržaj $AC < 0$
		SPA	preskok ako je sadržaj $AC > 0$
		NOP	ciklus bez operacije
		HLT	zaustavljanje programa

Preskočne

Akumulator (AC)

Tabela 3

Lokacija	Sadržaj lokacije	Komentar	Simb. repr.
100	1	sadrži prvi broj, 1	*X*, 1
101	4	sadrži drugi broj, 4	*Y*, 4
102	—	mjesto za rezultat	Z, 0
103	7200	(*CLA*) očisti akumulator (*AC*), ([*AC*] = *0*)	*CLA*
104	1100	pribroji [*AC*] sadržaj lokacije 100. Rezultat ostaje u *AC*. [*AC*] = 1	*TAD X*
105	1101	pribroji sadržaju *AC*, sadržaj lokacije 101. [*AC*] = 5	*TAD Y*
106	3102	Deponiraj sadržaj *AC* u lokaciju 102. [*AC*] = *0*; [102] = 5	*DCA Z*

Stvarni program uskladišten je u memoriji u **binarnom obliku** čiji je oktalni ekvivalent dan u koloni 2 tabele 3. Ovakva numerička reprezentacija nije pogodna za korisnika. Stoga korisnik piše program u **simboličkom obliku,** gdje instrukcije imaju mnemotehničko značenje. Taj oblik vidi se u zadnjoj koloni tabele.

Binarni oblik teško je razumljiv za čovjeka, a simbolički nema nikakvog značenja za stroj. Ove teškoće svladava se upotrebom specijalnog programa za prevođenje (compiler, assembler, translator). Operacija prevođenja prikazana je na slici 8 i sastoji se od faza *a* do *d*.

a) Korisnik piše program u simboličkom obliku na specijalnom uređaju (teletype, flexowriter, card puncher). Ovaj uređaj izbuši program na papirnu traku ili kartice pogodne za čitanje u stroj.

b) U digitalno računalo stavlja se program za prevođenje. Ovaj program čita instrukciju po instrukciju simboličkog programa korisnika i prevodi ih u binarni oblik. Program za prevođenje sadrži riječnik korespondentnih simboličkih i binarnih oblika **instrukcije.** (Tako npr. simbol *TAD* prevodi u binarnu instrukciju 1). Simboličkim nazivima lokacija prevodilac pridružuje raspoložive adrese memorije u binarnom obliku.

c) Nakon završenog prevođenja korisnik dobiva program u binarnom obliku.

d) Binarni program stavlja se u digitalno računalo te može početi njegovo izvršavanje.

2.5. Dijagram toka

Kod sastavljanja programa korisno je načiniti blok dijagram događaja, koji se zove dijagram toka a prikazan je na slijedećem primjeru: napisati program koji će obrisati 100 lokacija memorije počevši od lokacije »*START*«. Dijagram toka za taj program vidi se na slici 9. Fazama *a* do *e* sa dijagrama toka odgovara slijedeći program:

lokacija	sadržaj	komentar
I/	*CLA*	*a* i *b*
J/	*DCA START*	
K/	*ISZ J*	*c*
L/	*ISZ BR*	*d*
M/	*JMP J*	
N/	*HLT*	*e*
BR/	*—100*	

Unutar programa lokacija BR služi kao brojilo broja petlji. Kad sadržaj brojila od —100 dođe do nule, program preskoči u lokaciju *N* i zaustavi se.

2.6. Indirektno adresiranje

Adresiranje u računalima može biti direktno i indirektno. Indirektno se simbolički označava slovom *I*. Kod direktnog adresiranja adresa uz instrukciju sadrži **operand.** Kod indirektnog, adresa uz instrukciju sadrži **stvarnu adresu** operanda. Razlika između dvije vrste adresiranja vidi se iz slijedećeg dva primjera

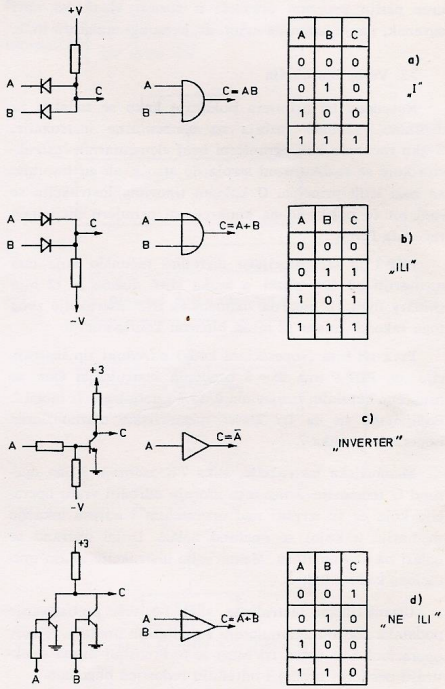
A, DCA N	/sadržaj *AC* staviti u lokaciju N
N, 7	/sadržaj lokacije *N* je 7
B, DCA I N	/sadržaj *AC* staviti indirektno u lokaciju *N*, dakle u stvari u lokaciju 7

Time je završen kratki pregled osnovnih principa programiranja potrebnog za mjerne i regulacione svrhe. Matematsko programiranje i numeričke metode predstavljaju posve zasebno područje i prelaze okvire ovog članka.

3. Digitalne operacije i funkcionalna organizacija računala

3.1. Osnovni digitalni elementi

Digitalne operacije povezuju dvije ili više diskretnih varijabli od kojih svaka može imati samo dva stanja. Ako



A	B	C
0	0	0
0	1	0
1	0	0
1	1	1

A	B	C
0	0	0
0	1	1
1	0	1
1	1	1

A	B	C
0	0	1
0	1	0
1	0	0
1	1	0

Sl. 10. Logički elementi

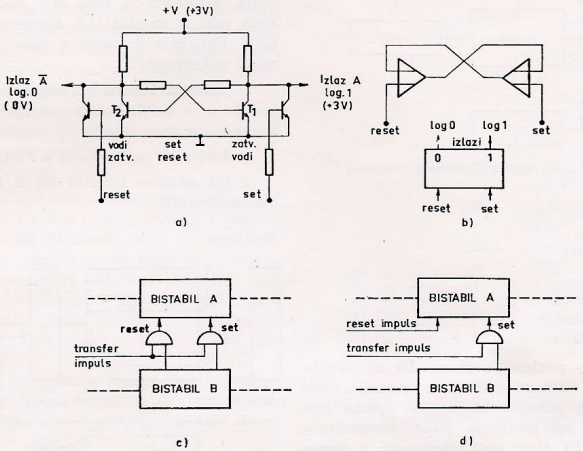
se dva stanja jedne varijable označe s A i $\overline{A}$ one mogu značiti

A logički »1«, da, »true«, npr. $+3\,V$

$\overline{A}$ logička »0«, ne, »false«, npr. $0\,V$

Operacije između binarnih varijabli mogu se vršiti osnovnim logičkim elementima kao: »i« (and), »ili« (or), »inverzija« (not), »ne ili« (nor) itd. Slika 10 prikazuje navedene logičke elemente za nivoe: logički »1« — $+3\,V$; logička »0« — $0\,V$. Za svaki element navedena je električka shema, logički simbol i tablica stanja, koja veže izlaznu varijablu s više ulaznih. Tablica stanja predstavlja rješenje pripadnih logičkih funkcija izraženih pomoću Boole-ove algebre.

Na slici 10. a prikazana su »i« vrata (and gate). Ako se naponski signal $+3\,V$ istovremeno pojavi na ulazu A i na ulazu B, na izlazu C bit će također $+3\,V$. U bilo kojem drugom slučaju, dakle ako se na nekom od ulaza ili na oba pojavi $0\,V$, na izlazu C bit će $0\,V$. Ekvivalentna logička shema ovoj elektroničkoj nalazi se u sredini slike 10. a. Nacrtani simbol »i« vrata označava logičku »i« funkciju. Kod logičkog simbola više se ne operira s naponskim nivoima nego s vrijednostima binarnih varijabla 0 i 1. Logička »i« funkcija formalno je ekvivalentna množenju, pa se i označava kao produkt. Zato se piše $C = A \,.\, B$. Ispravnost korištenja operacije množenja može se vidjeti i u tablici stanja, ako se varijablama A i B pridaju vrijednosti 0 i 1 u raznim kombinacijama i računa C. Tako npr. za $A = 0$ i $B = 1$ dobiva se $C = 0$.



Sl. 11. Bistabil i transfer

Slika 10. b prikazuje »ili« vrata (or gate). Taj sklop daje na izlazu logičku jedinicu, ako na A *ili* na B *ili* na oba ulaza dođe jedinica. I ovdje, kao i na ostalim sklopovima, vrijede zaključci i za električku shemu ako se logičkoj jedinici pridruži naponski nivo $+3\,V$, a nuli $0\,V$. Logička »ili« operacija ekvivalentna je algebarskoj operaciji zbrajanja. Zato je ovdje $C = A + B$. Izuzetak u tom formalizmu postoji kad su i A i B jedinice. Tada je $C = 1$ (a ne 2 kao što bi dala dosljedna primjena formalizma zbrajanja). Tablica stanja obuhvaća sve vrijednosti varijabli A i B.

Slika 10. c predstavlja inverter ili »ne« sklop (not). Logička jedinica na ulazu daje nulu na izlazu.

Slika 10. d je »ne ili« sklop (nor). Taj sklop ujedinjuje funkciju »ili« i funkciju »ne«. Na izlazu tog sklopa dobiva se logička nula ako barem na jedan ulaz dođe logička jedinica. Tablica stanja opet obuhvaća sve slučajeve i vidljivo je da daje rezultate upravo suprotne onima sa slike 10. b.

Elementi, koji vrše navedene logičke funkcije mogu biti izvedeni i na niz drugih načina. Od 1966. god. u pravilu se ovi logički elementi rade u integriranoj tehnici.

Vrijednosti binarne varijable »0« i »1« pamte se sklopom sa dva stabilna stanja, koji se zove bistabil. Slika 11 (a) i (b) prikazuje električnu shemu i logički simbol jednog bistabila, koji je u biti ostvaren unakrsnim spajanjem dvaju »ne ili« sklopova kao što je česta praksa u integriranoj tehnici. Bistabil je u stanju logički »1« ili »set« ako je tranzistor T_1 u zasićenja, a T_2 zatvoren. Obratna situacija označava logičko stanje »0« ili »reset«. (Ta definicija može naravno biti i drugačija). Bistabil se postavlja u ova stanja preko ulaza »set« i »reset« i ostaje u njima i nakon prestanka pobuda na ulazu. U tome i jest srž njegovog »pamćenja«.

Numerička informacija sastavljena je od n bita, a za njeno pamćenje potreban je sklop s n bistabila, koji se zove registar. Operacije s informacijama u digitalnom sistemu obično se svode na prijenos informacija između registara i transformacije informacija na registru. Slika 11 c i d prikazuje dva načina prijenosa informacije. Slika 11 c prikazuje kako transfer impuls ispituje stanje bistabila B i preslikava ga u bistabil A. Slika 11 d pokazuje kako reset impuls najprije briše bistabil A (postavlja ga u logičko stanje »0«), a zatim transfer impuls ispituje izlaz »1« bistabila B. Bistabil A bit će postavljen u stanje »1« samo ako je bistabil B bio u stanju »1«.

Ovdje je prikazan samo prijenos informacije između registara jer je to najčešća operacija kod uključivanja računala u mjerni ili regulacioni sistem. Transformacije unutar računala, pamćenje pomoću feritnih memorija i drugi detalji digitalnih računala prelaze okvire ovog članka.

3.2. Funkcionalna organizacija digitalnog računala

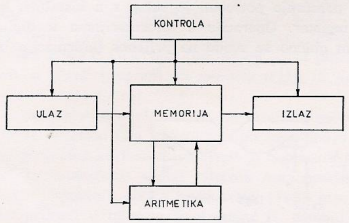
Digitalno računalo sastoji se od 5 osnovnih dijelova, slika 12. Osnovna funkcija računala je: prijem informacije iz vanjskog svijeta (ulaz), transformacija u novi oblik (memorija i aritmetička jedinica) i vraćanje transformirane informacije u vanjski svijet (izlaz). Ove funkcije koordinira kontrolna jedinica.

Funkcionalna organizacija digitalnog računala prikazana je detaljnije na slici 13 a. Ova slika prikazuje organizaciju digitalnog računala PDP-8, ali i sva ostala računala u principu su slično organizirana.

Korespondencija između slika 12 i 13 a je slijedeća:

1. Aritmetička jedinica
— akumulator
— bistabil za vezu (link)

2. Memorija
— feritna memorija 4096 riječi, 12 bita
— memorijski adresni registar
— memorijski »buffer« registar



Sl. 12. Općenita blok shema digitalnog računala

3. Kontrolna jedinica
— programsko brojilo
— instrukcijski registar

4. Ulaz/izlaz
— teleprinter
— prednja ploča (konzola)
— priključci za ulaz/izlaz

Akumulator (*AC*) predstavlja glavni dio aritmetičke jedinice. Može vršiti aritmetičke i logičke operacije, i biti testiran za donošenje jednostavnih logičkih odluka. Ima 12 bita. Spojen je na memoriju preko memorijskog »buffer« registra (*MB*). Može slati informaciju u memoriju ili primati iz memorije ili sa konzole. Većina ulazno izlaznih veza također završava na *AC*.

Veza (*V*) je jednobitni registar spojen na *AC* u petlji po kojoj se informacija može pomicati (posmak, shift). *V* može biti komplementiran, brisan i testiran za donošenje odluke.

Feritna memorija (*FM*) sadrži informaciju ili podatke koje treba obraditi i instrukcije odnosno program koji upravlja računalom tokom obrade. Sastoji se od 4096 riječi, od po 12 bita. Ciklus memorije je 1,5 μ sek.

Memorijski adresni registar (*MA*) se sastoji od 12 bita i kontrolira pristup do memorije za vrijeme izvršavanja instrukcija. U svakom ciklusu memorije sadržaj *MA* adresira jednu lokaciju memorije.

Memorijski »buffer« registar (*MB*) sadrži 12 bita i direktno je vezan na memoriju. Sve informacije koje ulaze ili izlaze iz memorije moraju proći kroz ovaj registar. Informacija se može prenositi između *MB* i *AC*, programskog brojila i instrukcijskog registra.

Programsko brojilo (*PB*) je 12 bitni registar čiji se sadržaj stalno modificira pod kontrolom programa. *PB* šalje u *MA* adresu u kojoj se nalazi ona instrukcija programa koju u slijedećem koraku treba izvršiti. Sadržaj *PB* kod normalnog rada povećava se za jedinicu nakon izvršavanja svake instrukcije. Jedino kod »skok« instrukcija *PB* dobiva novu adresu iz memorije preko *MB*. Startna adresa programa može se postaviti u *PB* i ručno preko konzole.

Instrukcijski registar (*IR*) prima iz *MB* operacioni kod (prva tri bita instrukcije). Na temelju ovog koda *IR* aktivira logičke sklopove potrebne za izvršavanje jedne od 8 mogućih instrukcija.

Teleprinter služi za komunikacije sa strojem pomoću pisanog teksta ili bušene papirne trake.

Konzola služi za unošenje podataka u računalo ručnim postavljanjem vanjskih preklopnika.

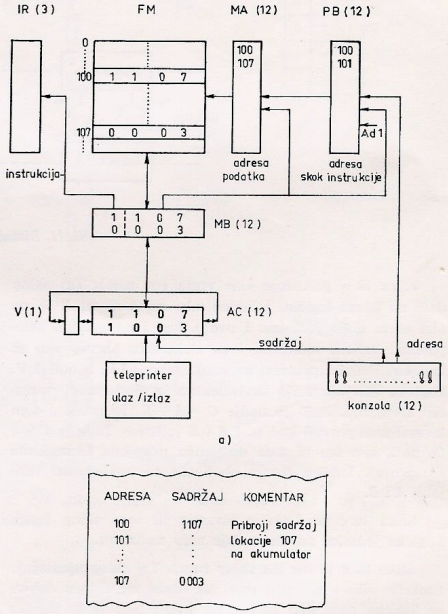
Ulazno-izlazni priključci služe za spajanje vanjskih uređaja na digitalno računalo.

Funkcioniranje računala može se pratiti na izvršavanju instrukcije sa slike 13 b. Sadržaji pojedinih registara vide se na slici 13 a. Razmatra se instrukcija 1107 (*TAD* 107), koja se nalazi u lokaciji 100. U lokaciji 107 nalazi se konstanta 3.

U zadnjem koraku prethodne instrukcije adresa 100 smještena je u *PB*. Koraci u izvršavanju instrukcije 1107 izvršavaju se kako slijedi:

1. Adresa 100 prebacuje se iz *PB* u *MA*.

2. *MA* adresira lokaciju 100 iz koje se sadržaj 1107 prebacuje u *MB*.



Sl. 13. Funkcionalna shema digitalnog računala:
a) osnovne jedinice računala
b) primjer instrukcije koja se upravo izvršava

3. Prva oktalna znamenka 1 prebacuje se iz *MB* u *IR* i pripremi logika za zbrajanje (instrukcija 1).

4. Ostali bitovi iz *MB* (107) prebacuju se u *MA*.

5. *MA* adresira lokaciju 107 čiji sadržaj 003 prelazi u *MB*.

6. Logika za zbrajanje, pripremljena u koraku 3, pribroji sadržaj *MB* postojećem sadržaju *AC*, a rezultat ostaje u *AC*.

7. Sadržaj *PB* poveća se za 1 naznačujući da će slijedeća instrukcija programa dolaziti iz lokacije 101.

Sličan tok događaja ponovio bi se kod instrukcije iz lokacije 101 i svih ostalih. Koraci 1 do 4 predstavljaju pripremni dio u kojem se instrukcija dekodira. Koraci 5 do 7 predstavljaju izvršni dio instrukcije.

4. Transfer podataka

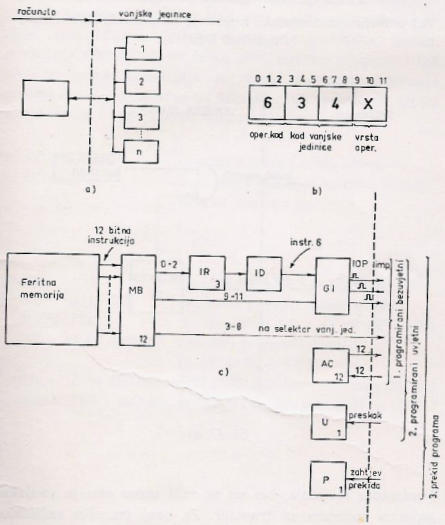
4.1. Ulaz i izlaz iz računala

Kako bi se digitalno računalo povezalo u jedan od lanaca sa slika 2, 3 i 4 potrebno je izgraditi posebnu logiku za transfer podataka između procesa i računala. Ova logika poznata pod engleskim nazivom »interface«, može realizirati transfer na 4 razna načina:

a) programirani bezuvjetni transfer
b) programirani uvjetni transfer
c) transfer s prekidom programa
d) jednociklusni transfer

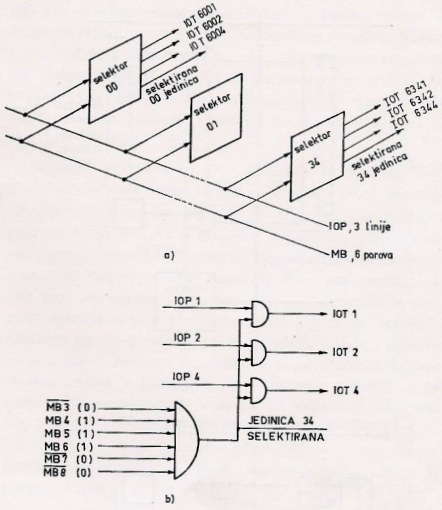
Kako je digitalno računalo vrlo brzo ono može istovremeno komunicirati s većim brojem mjernih i kontrolnih mjesta kao što se shematski vidi na slici 14 a. Instrukcija koja vrši transfer mora selektirati vanjsku jedinicu i odrediti vremenski redosljed operacija. Kod računala PDP-8 to je instrukcija s operacionim kodom 6 prikazana na slici 14 b. Logika potrebna za izvršavanje transfer instrukcije i za električku vezu s vanjskim jedinicama vidi se na slici 14 c.

Kao sastavni dio programa ,i transfer instrukcija se normalno nalazi u feritnoj memoriji. Iz memorije se instrukcija prilikom izvršavanja prebacuje u *MB*. Bitovi



Sl. 14. Logika za transfer
a) vezanje većeg broja vanjskih jedinica (mjernih i kontrolnih mjesta) na jedno računalo
b) instrukcija za transfer
c) osnovne jedinice računala, uključene u transfer

0 do 2 prelaze u *IR* i instrukcijski dekoder (*ID*). Iz *ID* izlazi osam kanala svaki za pripremu logike za pojedinu instrukciju. Na slici je nacrtan kanal za transfer instrukciju (6) koji aktivira posebni generator impulsa (*GI*). Iz *GI* izlaze tri voda na kojima mogu nastati impulsi fazno pomaknuti kao na slici. Iz *MB* bitovi 9, 10 i 11 određeni



Sl. 15. Selektori vanjskih jedinica
a) veći broj selektora priključen na jedno računalo
b) logička izvedba jednog od selektora

instrukcijom, priključeni su na *GI* i svaki od njih određuje da li će se pojaviti ili ne impuls na pojedinim izlazima voda *GI*. Bitovi 3 do 8 iz MB izlaze iz računala sa ciljem da selektiraju jednu od vanjskih jedinica.

Transfer podataka vrši se preko *AC* koji ima 12 ulaza i 12 izlaza prema vanjskom svijetu.

4.2. Selekcija vanjskih jedinica

Iz prethodne slike vidljivo je da računalo šalje komande preko tri voda iz *GI* i 6 parova vodova iz *MB* (za svaki bit i njegov komplement po jedan par). Ovi vodovi ulaze u sve vanjske jedinice, slika 15 a. Ulazni dio svake vanjske jedinice je selektor, slika 15 b. Jedan dio selektora predstavljaju »i« vrata (brana, and gate), koja primaju 6 bitni kod od *MB*. Ona daju izlaz samo onda kad se na ulazu pojavi ona kombinacija, koja je pridružena toj vanjskoj jedinici. Na slici se vidi priključak spomenute brane, koja reagira na oktalni kod 34. Sve druge vanjske jedinice ignorirat će ovaj kod. Izlaz ove brane vodi se na kontrolne ulaze triju brana selektora koje dobivaju pobudu od *GI* iz računala.

Na taj način računalo upravlja vanjskom jedinicom preko 4 izlaza iz selektora.

4.3. Transfer podataka između vanjske jedinice i akumulatora

Veze *AC* s vanjskim jedinicama prikazane na slici 14 c detaljnije se vide na slici 16 a (ulaz u računalo) i 16 b (izlaz iz računala). Vanjska jedinica mora imati registar

6. Logika za zbrajanje, pripremljena u koraku 3, pribroji sadržaj *MB* postojećem sadržaju *AC*, a rezultat ostaje u *AC*.

7. Sadržaj *PB* poveća se za 1 naznačujući da će slijedeća instrukcija programa dolaziti iz lokacije 101.

Sličan tok događaja ponovio bi se kod instrukcije iz lokacije 101 i svih ostalih. Koraci 1 do 4 predstavljaju pripremni dio u kojem se instrukcija dekodira. Koraci 5 do 7 predstavljaju izvršni dio instrukcije.

4. Transfer podataka

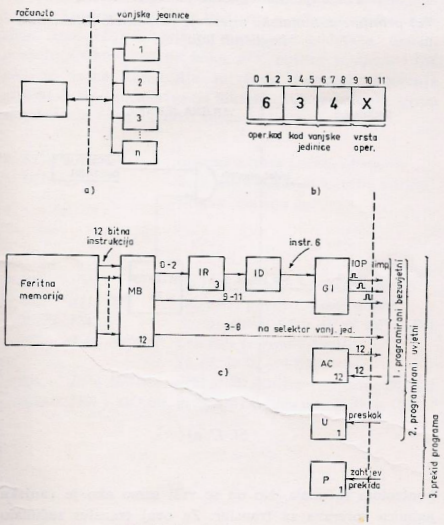
4.1. Ulaz i izlaz iz računala

Kako bi se digitalno računalo povezalo u jedan od lanaca sa slika 2, 3 i 4 potrebno je izgraditi posebnu logiku za transfer podataka između procesa i računala. Ova logika poznata pod engleskim nazivom »interface«, može realizirati transfer na 4 razna načina:

a) programirani bezuvjetni transfer
b) programirani uvjetni transfer
c) transfer s prekidom programa
d) jednociklusni transfer

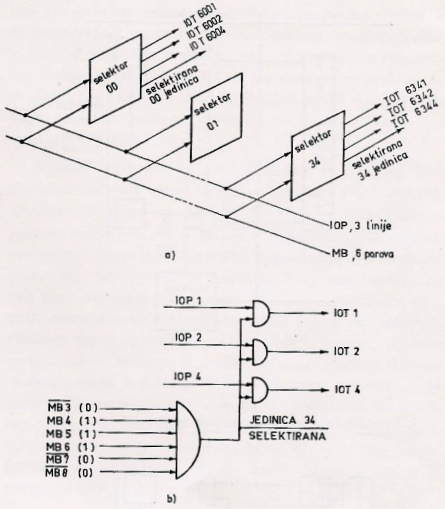
Kako je digitalno računalo vrlo brzo ono može istovremeno komunicirati s većim brojem mjernih i kontrolnih mjesta kao što se shematski vidi na slici 14 a. Instrukcija koja vrši transfer mora selektirati vanjsku jedinicu i odrediti vremenski redosljed operacija. Kod računala PDP-8 to je instrukcija s operacionim kodom 6 prikazana na slici 14 b. Logika potrebna za izvršavanje transfer instrukcije i za električku vezu s vanjskim jedinicama vidi se na slici 14 c.

Kao sastavni dio programa ,i transfer instrukcija se normalno nalazi u feritnoj memoriji. Iz memorije se instrukcija prilikom izvršavanja prebacuje u *MB*. Bitovi



Sl. 14. Logika za transfer
a) vezanje većeg broja vanjskih jedinica (mjernih i kontrolnih mjesta) na jedno računalo
b) instrukcija za transfer
c) osnovne jedinice računala, uključene u transfer

0 do 2 prelaze u *IR* i instrukcijski dekoder (*ID*). Iz *ID* izlazi osam kanala svaki za pripremu logike za pojedinu instrukciju. Na slici je nacrtan kanal za transfer instrukciju (6) koji aktivira posebni generator impulsa (*GI*). Iz *GI* izlaze tri voda na kojima mogu nastati impulsi fazno pomaknuti kao na slici. Iz *MB* bitovi 9, 10 i 11 određeni



Sl. 15. Selektori vanjskih jedinica
a) veći broj selektora priključen na jedno računalo
b) logička izvedba jednog od selektora

instrukcijom, priključeni su na *GI* i svaki od njih određuje da li će se pojaviti ili ne impuls na pojedinim izlazima voda *GI*. Bitovi 3 do 8 iz MB izlaze iz računala sa ciljem da selektiraju jednu od vanjskih jedinica.

Transfer podataka vrši se preko *AC* koji ima 12 ulaza i 12 izlaza prema vanjskom svijetu.

4.2. Selekcija vanjskih jedinica

Iz prethodne slike vidljivo je da računalo šalje komande preko tri voda iz *GI* i 6 parova vodova iz *MB* (za svaki bit i njegov komplement po jedan par). Ovi vodovi ulaze u sve vanjske jedinice, slika 15 a. Ulazni dio svake vanjske jedinice je selektor, slika 15 b. Jedan dio selektora predstavljaju »i« vrata (brana, and gate), koja primaju 6 bitni kod od *MB*. Ona daju izlaz samo onda kad se na ulazu pojavi ona kombinacija, koja je pridružena toj vanjskoj jedinici. Na slici se vidi priključak spomenute brane, koja reagira na oktalni kod 34. Sve druge vanjske jedinice ignorirat će ovaj kod. Izlaz ove brane vodi se na kontrolne ulaze triju brana selektora koje dobivaju pobudu od *GI* iz računala.

Na taj način računalo upravlja vanjskom jedinicom preko 4 izlaza iz selektora.

4.3. Transfer podataka između vanjske jedinice i akumulatora

Veze *AC* s vanjskim jedinicama prikazane na slici 14 c detaljnije se vide na slici 16 a (ulaz u računalo) i 16 b (izlaz iz računala). Vanjska jedinica mora imati registar

uvjetnog transfera prikazane su na slici 17. Vanjska jedinica, koja učestvuje u transferu, ima ugrađen jedan bistabil koji karakterizira njen status. Vanjska jedinica drži svoj »status« bistabil (S) u stanju 0 tako dugo dok priprema podatak, a prebaci ga u stanje »1« kad je spremna za transfer s računalom. Kad program odredi trenutak transfera, računalo preslika stanje status bistabila (vanjska jedinica) na uvjet bistabil *U* (računalo). Transfer instrukcija, koja vrši ovo preslikavanje, ispituje stanje bistabila *U* i donosi odluku:

— ako je »*U*« u stanju »*0*« program prelazi na slijedeću instrukciju

— ako je »*U*« u stanju »*1*« program preskače slijedeću instrukciju.

Isječak programa gdje se vrši odluka na bazi ispitivanja status bistabila, može izgledati ovako:

```
.          .          .
.          .          .
.          .          .
100    ┌ 6341          ┐  da li je jedinica 34 spremna?
101 ne │ ▶ JMP 100     │ da  ne; pitaj ponovo
102    └   JMP TRANS   ┘     da; izvrši transfer
.          .          .
.          .          .
.          .          .
```

Slika 17 b pokazuje primjer upotrebe uvjetnog transfera kod automatiziranog mjerenja promjera kugličnih ležajeva. Cijeli mjerni sistem (vanjska jedinica) ima oktalni kod 34. Želi se saznati koliki postotak kugličnih ležajeva ima dijametar d_1, d_2 itd. Mjerni sistem pretvara dijametar u analogni napon (D_a) koji se u analogno-digitalnom konverteru (*ADC*) pretvara u odgovarajući numerički iznos (D_d) i zapisuje u registar podataka. Sistem posjeduje status bistabil kojeg *ADC* prebaci u stanje 1 tek kad je jedna konverzija završena. Rezultati mjerenja zapisat će se u memoriju od lokacije 1000 nadalje. Program za izvršenje ovog zadatka može izgledati ovako:

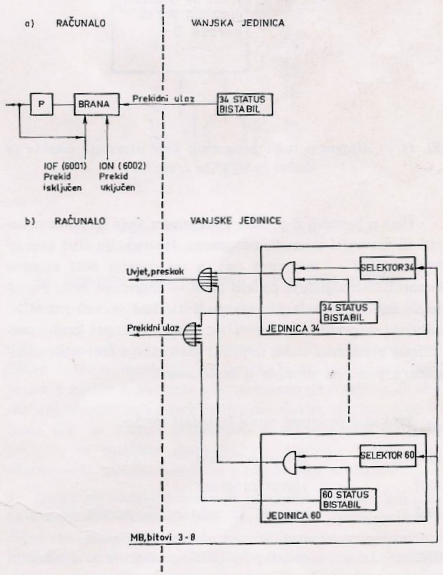
```
START,  1000      startna adresa početka liste
DIJAM,  0         lokacija u kojoj se formira adresa
  .  .            proporcionalna Dd
  .  .
  .  .
  .  .
▶ 200,  6341      da li je jedinica 34 spremna?
  201,  JMP 200   ne; pitaj ponovo (skoči u 200)
  202,  6342      da; preslikaj RP u AC (u AC je Dd)
  203,  TAD START pribroji startnu lokaciju
                  (u AC je Dd + 1000)
  204,  DCA DIJAM uskladišti dobivenu adresu u DIJAM
  205,  ISZ I DIJAM pribroji jedinicu adresi Dd + 1000
  .  .
  .  .
  .  .
  .  .
  .  .
```

Ovakav program izvršava se kod učitavanja svakog mjerenja u računalo. Rezultat ovakvog mjerenja i analize je krivulja raspodjele dijametara, slika 17 c.

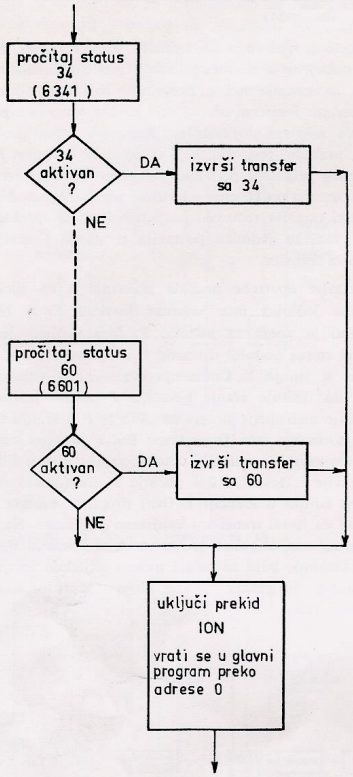
4.6. Transfer s prekidom programa (program interrupt)

Često je potrebno da transfer bude diktiran perifernim uređajem, a ne računalom. Vrijeme računala koristi se za izvršavanje nekog programa koji nije povezan s transferom. Periferni uređaj vrši prekid programa postavljanjem zahtjeva za transfer. Računalo zatim prelazi u drugi, prekidni program, koji omogućava transfer. Nakon završenog prijenosa podataka računalo nastavlja izvršavanje programa prekinutog od strane perifernog uređaja. Za prekidni transfer računalo posjeduje bistabil »prekid« (*P*) kojeg vanjska jedinica postavlja u stanje 1 onda kad zahtijeva transfer.

Primjer upotrebe prekida prikazan je na slici 18 a. Vanjska jedinica ima »status« bistabil. Ovaj bistabil direktno je vezan za bistabil *P*. Čim vanjska jedinica postavi status bistabil u stanje 1, istog trena i bistabil *P* prelazi u stanje 1. Unutarnja organizacija računala je takva da ispituje stanje bistabila *P* tokom izvršavanja bilo koje instrukcije programa. Ako je *P* u stanju *0*, program normalno teče. Onog trena kad *P* poprimi stanje 1, računalo završi instrukciju u toku, zapiše adresu slijedeće instrukcije u lokaciju 0 i počinje s programom čija je startna adresa u lokaciji 1. Ovaj program napisan je sa svrhom da izvrši transfer s vanjskom jedinicom. Na kraju tog programa osiguran je povratak u lokaciju *0* (skok instrukcijom), gdje se nalazi adresa slijedeće instrukcije prekinutog programa, nakon čega računalo nastavlja normalni rad.



Sl. 18. Transfer s prekidom programa
a) samo jedna vanjska jedinica
b) istovremeni rad s više vanjskih jedinica



Sl. 18. c) dijagram toka programa koji utvrđuje odakle je došao zahtjev za transfer

Ulaz u bistabil *P* prolazi kroz branu koja se može otvoriti ili zatvoriti pomoću programa. Instrukcija 6001 otvara branu (*ION* — interrupt on), a instrukcija 6002 zatvara branu i onemogućuje prekid (*IOF* — interrupt off). Pored toga, kad se *P* nalazi u stanju 1 tj. kad je već primljen zahtjev za prekid, brana se zatvara. Stoga pri kraju prekidnog programa treba napisati instrukciju 6001 koja omogućava ponovne prekide u budućnosti.

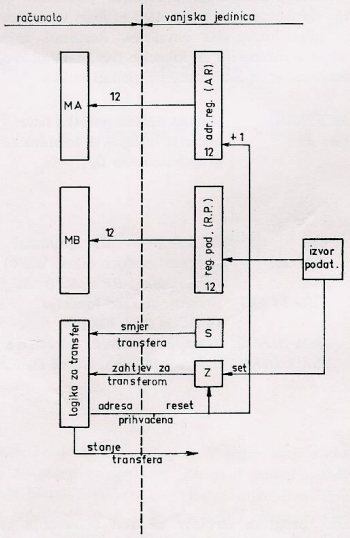
Prekidni program može izgledati ovako:

```
1000 . . . .    glavni program
1001 . . . .    glavni program
1002 . . . .    pojavio se zahtjev za prekid programa
-------------------------------------------------------
0000 . . . .    sadržaj PB (1003) prebacuje se u lokaciju
                000
0001 JMP 2000   skoči na startnu adresu prekidnog
                programa
-------------------------------------------------------
2000, . . . .
2001, . . . .   prekidni program,
  .             izvršava transfer
  .
  .
  .
2276, . . . .
2277, ION       omogući novi prekid
2300, JMP I 0000 vrati se na glavni program
-------------------------------------------------------
1003, . . . .   nastavak glavnog programa
1004, . . . .
```

Navedeni primjer pokazuje upotrebu prekida prilikom transfera samo s jednom vanjskom jedinicom. Ukoliko postoji potreba za transferom s više vanjskih jedinica simultano, koristi se kombinacija prekidnog i uvjetnog transfera kao na slici 18 b. Prekidna linija iz računala bit će aktivirana preko »ili« vrata ako bilo koji status bistabil dođe u stanje »1« (njegova vanjska jedinica zahtijeva prekidni transfer). Računalo prelazi iz glavnog programa u prekidni pomoću kojeg mora ispitati koja vanjska jedinica je izazvala prekid. Prekidni program šalje niz transfer instrukcija sa sukcesivnim adresama vanjskih jedinica. Ovako se status bistabili jedan za drugim vežu na liniju za uvjet. Kad je pronađen »aktivni« status bistabil, prekidni program izvrši transfer sa dotičnom vanjskom jedinicom. Dijagram toka ovakvog prekidnog programa prikazan je na slici 18 c.

4.7. Jednociklusni transfer (data break, cycle stealing, input autput processor, data channel)

Svi dosad opisani transferi vršili su se preko aritmetičke jedinice i pomoću programa za transfer. Svaki program sastoji se od više instrukcija što znači trošenje



Sl. 19. Jednociklusni ulazni transfer

znatnog postotka vremena računala za transfer. Stoga postoji posebna metoda koja omogućuje neprogramirani jednociklusni transfer i istovremeno izvršavati neki program koji nema nikakve veze s tim transferom. Transfer se obavlja posebnom logikom koja ima direktni i od računala posve nezavisni pristup do memorije. Takva organizacija omogućuje dva posla u isto vrijeme:

— računalo uzima instrukcije iz memorije i izvršava njime određeni program

— logika za transfer od vremena do vremena, po potrebi transfera, »ukrade« od računala jedan ciklus tokom kojeg se program ne može vršiti i u tom ciklusu kompletno izvrši transfer. Kod PDP-8, čiji ciklus traje 1,5 μ sek, ova logika omogućuje oko 500.000 transfera u sekundi, a da pri tom još uvijek oko 50 % vremena računalo ostaje na raspolaganju za drugi, programirani posao.

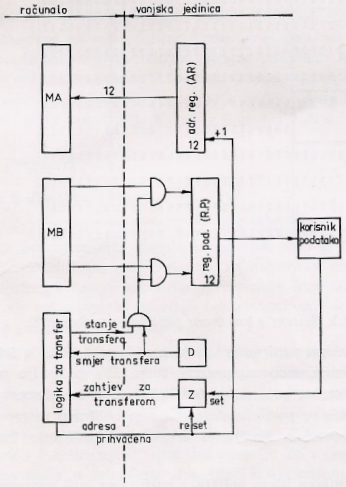
Kod jednociklusnih transfera vanjska jedinica daje računalu

— adresu memorije s kojom se vrši transfer
— podatak, ukoliko se radi o ulaznom transferu
— indikaciju smjera transfera (ulaz, izlaz)
— zahtjev za početak transfera.

Računalo daje vanjskoj jedinici signale

— »zahtjev prihvaćen«
— indikaciju da se nalazi u stanju transfera.

Slika 19 predstavlja jednociklusni ulazni transfer. Kod ovog primjera treba podatke upisati u sukcesivne lokacije memorije počevši od neke startne lokacije. Vanjska jedinica ima svoj adresni registar na kojem je zapisana startna adresa. Kad novi podatak dođe iz izvora on se uskladišti na registar podataka vanjske jedinice. Istovremeno izvor postavi bistabil Z u jedinicu i time šalje zahtjev za

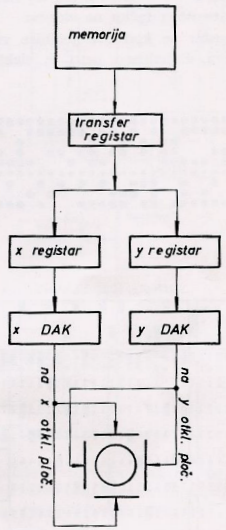


Sl. 20. Jednociklusni izlazni transfer

početak transfera. Računalo zatim prenese adresu iz adres registra vanjske jedinice na memorijski adres registar. Time je određena lokacija, u koju zatim računalo upisuje podatak sa *RP* (posredstvom *MB*). Signal »zahtjev prihvaćen« može se iskoristiti za brisanje bistabila Z i inkrement perifernog adres registra. Time je logika pripremljena za transfer slijedećeg podatka koji može doći izvana.

Slika 20 prikazuje jednociklusni izlazni transfer. Kad god je »korisnik podataka« spreman da primi slijedeći podatak iz memorije računala, on postavlja zahtjev preko bistabila *Z*. Računalo primi izvana adresu lokacije i iz nje šalje podatak korisniku. Signal, »zahtjev prihvaćen« resetira bistabil *Z* i inkrementira adresu. Tako korisnik dobiva podatke iz sukcesivnih lokacija memorije, a računalo istovremeno obavlja neki drugi posao, posve nezavisan od transfera.

Kombiniranjem spomenutih vrsta transfera računalo može riješiti široki dijapazon problema u mjerenju i regulaciji.



Sl. 21. Blok shema osciloskopskog prikaza

4.8. Transfer podataka preko standardnih jedinica

Transfer podataka vrši se između memorije digitalnog računala i vanjskih jedinica. U mjerenju vanjska jedinica vrši automatsko mjerenje parametara eksperimenta i podatke šalje u memoriju računala. U regulaciji računalo prima podatke iz procesa i istovremeno ga regulira. U oba slučaja čovjek samo nadzire izvana sistem u vidu promatrača dok se podaci automatski prenose preko vanjskih jedinica za vrijeme eksperimenta. Vanjske jedinice su obično neki specijalni mjerni i regulacioni uređaji.

Drugi tip vanjskih jedinica su standardni uređaji kao npr. sistemi s bušenom papirnom trakom, bušenim karticama i magnetskom trakom. Kod izlaznog transfera podaci se prenose iz feritne memorije računala na vanjske medije u svrhu privremenog memoriranja. Ulazni transfer prenosi s vanjskog medija u memoriju računala informacije koje su nekad prije registrirane. Taj transfer odgovara »off-line« načinu rada kao što je prikazano na slici 2.

Kod svih vanjskih jedinica vrijede principi transfera opisani u tački 4. Selektor vanjske jedinice i status bistabil vezani su za logiku digitalnog računala i omogućavaju transfer. Kod standardnih vanjskih jedinici najčešće se vrši programirani uvjetni transfer i transfer s prekidom programa, a brzi jednociklusni transfer obično se koristi u slučajevima brzog prijenosa velikih količina podataka.

4.8.1. *Osciloskopski prikaz i svjetlosno pero**

Ova vanjska jedinica omogućava vizuelno grafičku komunikaciju između čovjeka i računala u oba smjera.

Kod izlaznog transfera podaci se iz memorije vode na transfer registar računala (npr. akumulator), a zatim na x odnosno y registre i pripadne digitalno-analogne konvertere, slika 21. Dobiveni napon proporcionalan digitalnim podacima dovodi se na otklonske pločice katodne cijevi i otklanja zraku u x i y smjeru. Na taj način svaki novi podatak generira novu tačku na ekranu.

Ulazni transfer se kod ove jedinice vrši uz pomoć svjetlosnog pera. Svjetlosno pero je elektronički sklop

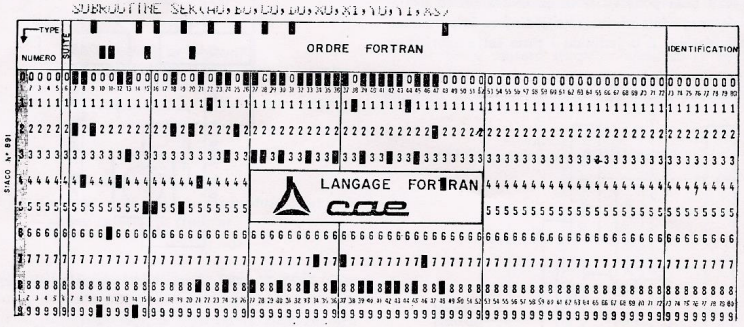
vrši prema specifičnostima konkretnog sistema. Ova jedinica se najčešće koristi za praćenje eksperimenta, koji je u toku, dakle vrši transfer u »živom vremenu«.

4.8.2. *Sistem za štampanje znakova***

Ova jedinica je vrlo slična običnom pisaćem stroju. Podaci iz memorije računala električkim putem preko releja aktiviraju sistem poluga, koje omogućuju štampanje znakova (slova, brojevi, specijalni znaci) na listu papira. Ulaz podataka vrši se preko tastature printera. Pritiskom na tipku za pojedini znak generiraju se električni impulsi u određenom kodu i zapisuju u memoriju. Kod ove jedinice potrebno je dekodirati odnosno kodirati informaciju. Kod ovdje pridružuje svakom znaku neki binarni broj.

Brzina izlaznog transfera, dakle štampanja na papir je oko 10 znakova u sekundi, a kod brzih printera je reda 1000 znakova u sekundi. Brzina ulaznog transfera ograničena je brzinom čovjeka. Programirani uvjetni transfer i transfer s prekidom programa dolaze u obzir.

Sl. 22. a) bušena traka



Sl. 22. b) bušena kartica

čiji glavni dio — fotoosjetljivi element — generira električki impuls kao odziv na svjetlosnu pobudu. Ako se svjetlosnim perom dodirne bilo koja tačka rastera, kojeg na ekranu katodne cijevi generira računalo, električki impuls iz svjetlosnog pera omogući računalu da zapamti koordinate dodirnute tačke. Na taj način računalo pamti sve krivulje, koje po ekranu ispisuje svjetlosno pero. Podatak se unosi u računalo na bazi transfera s prekidom programa.

Osciloskopski prikaz je najbrža vanjska jedinica, a brzina mu je komparabilna s brzinom modernog digitalnog računala. Sve vrste transfera dolaze u obzir, a izbor se

4.8.3. *Sistem s bušenom papirnom trakom****

Bušena papirna traka je medij na kojem je informacija privremeno zapamćena. Slika 22 a pokazuje primjer takve trake, gdje su podaci izbušeni u određenom kodu. Najčešće se podaci nalaze u istom obliku u memoriji, pa nije potrebno kodiranje odnosno dekodiranje kao kod printera.

Prilikom izlaza podataka perforator buši papirnu traku pomoću igala za bušenje. Električki signali aktiviraju magnete perforatora, a magneti osiguravaju bušenje rupa na traci tamo gdje su u binarnom broju bile jedinice.

* Detaljnije referenca (2)

** Vidi referencu (1).

*** Vidi referencu (1).

Ulazni transfer podataka s perforirane trake vrši se pomoću čitača. Kod fotočitača se preko fotoosjetljivog elementa generira električki impuls na onim mjestima, gdje u retku postoji rupa.

Zbog potrebe pokretanja tromih mehaničkih dijelova u perforatoru, izlazni transfer je obično za red veličine sporiji od ulaznog. Uobičajene brzine ulaznog transfera su 100 do 1000 znakova u sekundi, a izlaznog 50 do 100.

4.8.4. *Sistem s bušenim karticama*

Slika 22 b prikazuje primjer bušene kartice. Format podataka je nešto drugačiji od formata kod bušene trake, ali to ne čini bitnu razliku. Zato vrijede isti principi na kojima bazira ulazni i izlazni transfer podataka, a ulazno izlazne jedinice su bušač i čitač kartica. Vrlo je prikladno koristiti kartice kod diskontinuirane prirode podataka.

Tipična brzina izlaznog transfera je oko 300 znakova u sekundi, a ulaznog oko 500.

4.8.5. *Sistem s magnetskom trakom*

Magnetska traka služi za privremeno memoriranje podataka, a radi na sličnom principu kao poznata magnetofonska zraka.

Kod izlaznog transfera se podaci iz memorije pretvaraju u strujne impulse koji upravljaju glavama za snimanje. Kod ulaznog transfera magnetsko stanje trake izaziva električke impulse u glavama za čitanje.

Velike brzine ove jedinice moguće su zbog toga, što se čitanje i upisivanje na traku vrši magnetskim putem bez pokretnih mehaničkih dijelova. Ulazno-izlazne brzine su oko 3000 znakova u sekundi. Transfer podataka s magnetskom trakom vrši se u blokovima.

Kod ovog kratkog pregleda standardnih vanjskih jedinica navedene su neke tipične brzine transfera, ali samo zbog predodžbe o redu veličine. Brzine pojedinih komercijalnih jeddinica variraju u širokom rasponu. Osim nabrojenih jedinica postoje još neke, koje ovdje nisu opisane, kao npr. pisač, magnetski bubanj i magnetski disk.

5. Primjeri

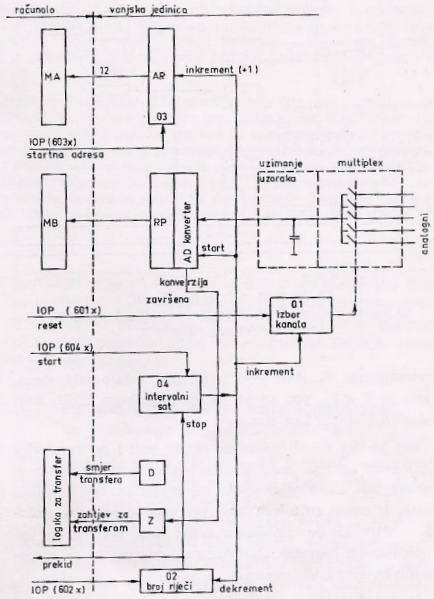
5.1. Računalo u mjerenju

Slika 21 prikazuje sistem za simultani jednociklusni transfer pet analognih signala u računalo PDP-8. Podaci se preko multipleksa redom spajaju na računalo i zapisuju u sukcesivne lokacije memorije. Vremenske intervale uzimanja uzoraka signala određuje »intervalni sat« koji se pokreće i zaustavlja iz računala.

Prije početka transfera računalo šalje transfer instrukciju 601x koja će biti prihvaćena samo od jedinice 01 (izbor kanala multipleksa). Ovom instrukcijom postavlja se logika multipleksa u početni položaj. Nakon toga računalo pomoću instrukcije 602X spaja registar 02 na akumulator (AC) i prebacuje u njega informaciju koja označava broj podataka u transferu. Iza toga instrukcijom 603X računalo spaja registar 03 na AC i prebacuje u 03 podatak koji označava startnu lokaciju memorije. Ovim operacijama čitava logika pripremljena je za početak. Računalo starta intervalni sat slanjem instrukcije 604X i transfer počinje u taktu kojeg daje sat. Od ovog momenta transfer se vrši automatski, a računalo može izvršavati neki nezavisni program.

Tokom svakog takta intervalni sat prebacuje multiplex za jedan kanal, povećava adresu u registru 03 za jedinicu, starta jednu analogno digitalnu konverziju i smanjuje broj riječi preostalih u registru 02 za jedinicu.

Kad god je analogno digitalna konverzija gotova, preko bistabila Z postavlja se zahtjev računalu da izvrši jednociklusni transfer.



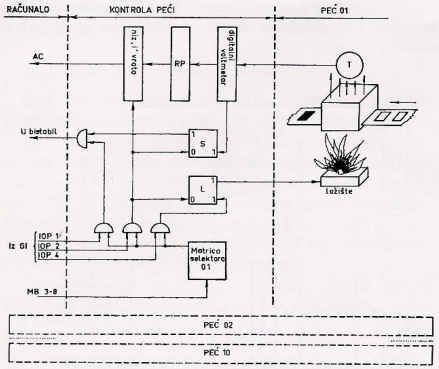
Sl. 23. Računalo u mjerenju: Automatsko skupljanje podataka s većeg broja analognih izvora, na bazi jednociklusnog transfera.

Kad broj riječi u registru 02 padne na nulu, zaustavlja se intervalni sat i pomoću »prekida« signalizira računalu da je kompletni transfer završen. Pomoću prekidnog programa računalo donosi odluku o slijedećem poslu.

5.2. Računalo u regulaciji

Slika 22 pokazuje primjer računala PDP-8 u regulaciji. U tom primjeru računalo kontrolira rad desetak peći. Predmete, koji se peku, dovodi u peć beskonačna traka (konvejer), koja se pomiče konstantnom brzinom. Kad god temperatura pređe T_0, peć se isključuje kako bi se temperatura spustila na nižu. Podatak o temperaturi termometar T pretvara u analogni napon. Digitalni voltmetar vrši analogno digitalnu konverziju, prebacuje digitalni podatak u registar podataka RP i istovremeno postavlja status bistabil S u stanje 1. Instrukcija 6011 ispituje da li je status bistabil u stanju 1 slanjem impulsa $IOP1$. Kad je S postavljen u stanje 1, $IOP1$ postavlja bistabil U u računalu u stanje 1, a računalo briše AC i prelazi na izvršavanje instrukcije 6012. Pri tom se generira impuls $IOP2$, koji

briše bistabil *S*, bistabil *L* (za kontrolu ložišta) i prebacuje podatke iz *RP* u *AC*. Peć je dakle na moment isključena jer je *L* u stanju 0. Zatim slijedi kompariranje dobivenog podatka o temperaturi *T* s maksimalnom radnom



Sl. 24. Računalo u regulaciji:
a) logika za simultanu regulaciju većeg broja peći pomoću jednog računala

temperaturom T_0. Ako je $T > T_0$, peć ostaje isključena, a ako je $T < T_0$, peć se uključuje impulsom *IOP4*, koji postavlja bistabil *L* u stanje 1.

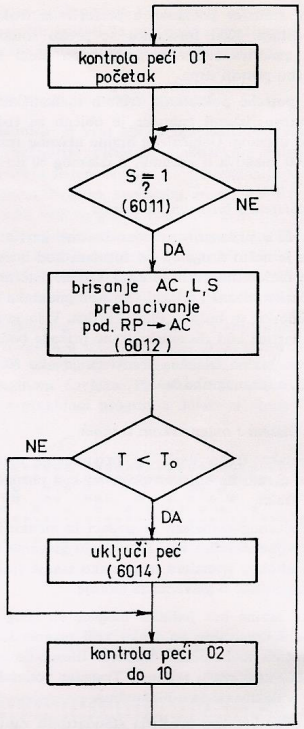
Isti proces slijedi zatim za ostale peći i po završetku se ponavlja opet za prvu. Kratko prebacivanje bistabila *L* u stanje »0« nema djelovanja na tromi sistem kontrole ložišta. U ovom primjeru kontrole sporog procesa, računalo kontrolira bez problema veliki broj peći, a da još uvijek najveći postotak vremena računala ostaje na raspolaganju za neki drugi posao.

Obzirom da je upotreba računala u mjerenju i regulaciji kod nas u začetku, u članku su opisani samo generalni principi i ispušteni mnogi tehnički detalji. Cilj je bio da se uoče oni principi koji su zajednički svim računalima. Od računala do računala postoje međutim specifičnosti i razlike.

Na primjer kod memorije razlike postoje u broju i dužini riječi i ciklusu rada. Također može postojati drugačiji repertoar instrukcija. Ima računala s više od jednog prekidnog ulaza, koji su raspoređeni u nivoe po prioritetu. Neka računala imaju osim jednociklusnog transfera još i specijalni dvo ili trociklusni transfer. Time se nešto usporava transfer ali se znatno reducira broj potrebnih vanjskih logičkih sklopova. Neka računala imaju mogućnost da u jednom ciklusu prime podatak i povećaju za jedinicu sadržaj lokacije adresirane podatkom. To pruža mogućnost upotrebe računala kao amplitudnog analizatora (mjerenje gustoće vjerojatnosti amplitude). Današnja laboratorijska računala imaju cikluse reda veličine jedne mikrosekunde što im omogućuje analizu i do milijun podataka u sekundi.

Glavni tehnološki napredak kod laboratorijskih računala odvija se u dva smjera:

— ciklus računala skraćuje se sa mikrosekundnog na nanosekundno područje;



Sl. 24. b) dijagram toka regulacionog programa računala

— brze memorije povećavaju se od nekoliko stotina ili hiljada, na desetke hiljada riječi.

Zbog ovog razvoja često nije više potrebno štediti vrijeme i prostor računala, jer ga ima u izobilju. Stoga je tendenca da se sa simboličkog programiranja prijeđe na još jednostavniji, viši oblik, sličan Fortranu, Cobolu ili Algolu u matematskom programiranju. Jedan takav jezik je CONTRAN (control command transformation), koji omogućuje inženjerima da u relativno kratko vrijeme mogu programirati ili preprogramirati računalo, za razne kontrolne i mjerne poslove. Drugi takav jezik je REAL TIME FORTRAN. Ugradnjom laboratorijskog računala, rad na konstrukciji mnogih sistema skraćuje se barem u skali jedna godina — jedan mjesec.

LITERATURA

1. K. Čuljat, V. Bonačić, B. Souček: **Redukcija i sortiranje podataka eksperimenta računalom,** isti broj ovog časopisa.
2. V. Bonačić, B. Souček, K. Čuljat: **Optička komunikacija s računalom za »on-line« kontrolu,** isti broj ovog časopisa.

NASLOV AUTORA:

B. Souček — K. Čuljat — V. Bonačić, Institut »Ruđer Bošković«, Zagreb