

2. POGLAVLJE

VON NEUMANNOV MODEL RAČUNALA

2.1. UVOD

Jedan od najznačajnijih članaka na području arhitekture računala "Uvodna rasprava o logičkom oblikovanju elektroničkog računskog uređaja" (izvorni naslov: "Preliminary Discussion of the Logical Design of an Electronic Computing Instrument") autora A. W. Burksa, H. H. Goldsteina i von J. Neumanna, objavljen je 1946., dakle petnaestak godina prije pojave izraza "arhitektura računala". Taj je članak imao dalekosežne posljedice i utjecao je na arhitekturu sljedećih četiriju generacija računala. Naravno, i prije 1946. postojala su ozbiljna nastojanja u realizaciji stroja za računanje, odnosno računala. Njihov povijesni razvoj može se predočiti generacijama računala – od *nulte generacije* koja odgovara mehaničkim računalima pa sve do *četvrte generacije* koja se temelji na tehnologiji vrlo visokog stupnja integracije VLSI (*Very Large Scale Integration*). (Opaska: neki autori navode i petu generaciju računala koja se temelji na konceptima umjetne inteligencije (engl. *artificial intelligence*), ali tehnološka podloga i tih računala je VLSI). Opišimo, ukratko po generacijama računala, samo jedan mali isječak iz burne povijesti razvoja strojeva za računanje, odnosno računala.

2.1.1. NULTA GENERACIJA – GENERACIJA MEHANIČKIH STROJEVA ZA RAČUNANJE (1644. – 1945.)

B. Pascal je u razdoblju između 1642. i 1644. godine izgradio stroj koji je mogao obavljati računske operacije zbrajanja i oduzimanja. Godine 1673. G. W. Leibniz oblikovao je mehanički računski stroj koji je, uz zbrajanje i oduzimanje, mogao obavljati i računske operacije množenja i dijeljenja. Bio je to ekvivalent "džepnim kalkulatorima" sa četiri funkcije tri stoljeća kasnije.

C. Babbage je 1822. izgradio *diferencijski stroj* (engl. *difference engine*) koji se temeljio na računu konačnih diferencija. C. Babbage je, također, 1834. započeo rad na *analitičkom stroju* (engl. *analytical engine*) koji je obavljao četiri osnovne računske operacije i korjenovanje te je trebao biti *računski stroj opće namjene*. On je zamislio stroj koji se sastoji od četiriju jedinica: memorije (ili spremnika), jedinice za računanje (izvorno nazvane *mill* – što na engleskom znači pogon ili mlin), ulazne jedinice (čitača bušenih kartica) i izlazne jedinice (pisača i bušača kartica). Analitički je stroj bio programirljiv – čitao je instrukcije s bušenih kartica i izvršavao ih. Iz memorije je dohvaćao dva broja predočena u dekadskom brojevnom sustavu, obavljao operaciju u jedinici za računanje te pohranjivao rezultat natrag u memoriju.

Slijed bušenih kartica određivao je program koji se mogao upotrijebiti za više skupova podataka. Na žalost, analitički stroj nije bio nikad dovršen zbog problema s mehaničkom izradom koja je zahtijevala tisuće i tisuće vrlo precizno izrađenih zupčanika i osovina što je u 19. stoljeću bilo tehnološki neizvodivo.

Računski stroj koji se može promatrati kao prekretnica u tehnološkom smislu sa potpuno mehaničkih strojeva na elektromehaničke računске strojeve bio je stroj nazvan Z1, nje-mačkog istraživača K. Zusea (1934. – 1936.) koji se temeljio na elektromehaničkim relejima.

H. Aiken je 1944. u Harvardu, Sjedinjene Američke Države, izgradio elektromehaničko računalo Mark I koje se smatra prvim američkim računalom opće namjene. Ono je pohranjivalo 72 riječi od kojih je svaka bila predstavljena s 23 dekadске znamenke, a instrukcije su se izvršavale za 6 sekundi.

2.1.2. PRVA GENERACIJA – ELEKTRONIČKA RAČUNALA S ELEKTRONSKIM CIJEVIMA (1945. – 1955.)

Tijekom ratne 1943. za razbijanje šifriranih poruka, odnosno dekriptiranje njemačkih poruka u Engleskoj je izgrađeno računalo COLOSSUS koje se smatra prvim elektroničkim računalom. Ono je imalo oko dvije tisuće elektronskih cijevi, a u razvoju računala sudjelovao je A. Turing.

Iste su godine J. Mauchley i J. P. Eckert započeli s izgradnjom računala ENIAC (*Electronic Numerical Integrator And Computer*). Ono je imalo oko 18000 elektronskih cijevi i 1500 releja, težilo je oko 30 tona, a za žarenje elektronskih cijevi, anodne izvore i ventilatore za hlađenje bila mu je potrebna snaga od 140 kW! Računalo je bilo završeno 1946. Mnogi smatraju da je povijest suvremenih računala započela upravo s ENIAC-om.

M. Wilkes je na Sveučilištu u Cambridgeu 1949. izgradio računalo EDSAC (*Electronic Delay Storage Automatic Computer*) koje se smatra prvim elektroničkim računalom s pohranjivanjem programa.

Dok su J. Mauchley i J. P. Eckert radili na EDVAC-u (*Electronic Discrete Variable Automatic Computer*) koji je trebao biti nasljednik ENIAC-a, J. von Neumann, koji je inače sudjelovao u projektu ENIAC, došao je u Institut za napredna istraživanja (Institute for Advanced Study) u Princeton, SAD, te započeo projekt izgradnje računala IAS. Računalo IAS završeno je 1952. i svojim konceptima predstavlja osnovne temelje današnjih računala. Osnovni model računala, opisan u članku koji smo spomenuli na početku ovog poglavlja, poznat je kao *von Neumannov model računala*. Skoro istodobno dok je von Neumann radio na oblikovanju IAS računala, na MIT-u su razvijali 16-bitno računalo Whirlwind I koje je prvo računalo za upravljanje u stvarnom vremenu (engl. *real-time control*). U sklopu Whirlwind projekta izumljena je memorija s magnetskim jezgicama (engl. *magnetic core memory*).

Godine 1953. tada mala tvrtka IBM započinje s proizvodnjom računala IBM 701.

2.1.3. DRUGA GENERACIJA RAČUNALA – TRANZISTOR KAO GRAĐEVNA KOMPONENTA (1955. – 1965.)

Poluvodičku elektroničku komponentu *tranzistor* izumili su 1948. trojica istraživača J. Bardeen, W. Brattain i W. Shockley. Tranzistor možemo pojednostavljeno predočiti kao sklopku s dva stanja (isključeno/uključeno; engl. *off/on*) koja je električki upravljana.

Izum tranzistora predstavljao je revoluciju na području računarske tehnologije i u kasnim je pedesetim godinama u potpunosti potisnuo elektronske cijevi. Prvo računalo izgrađeno na temelju tranzistora bilo je TX-0 (*Transistorized eXperimental computer 0*) (u MIT Lincoln Laboratoryju). Godine 1960. tvrtka DEC (Digital Equipment Corporation) na tržište plasira prvo malo računalo (miniračunalo) PDP-1 čija je cijena bila oko 120 tisuća dolara. Godine 1965. DEC proizvodi 12-bitno miniračunalo PDP-8 čija je cijena bila samo 16 tisuća dolara. Tvrtka DEC prodala je preko 50 tisuća računala PDP-8. Tvrtka IBM je 1961. proizvela vrlo popularno malo poslovno računalo IBM 1401, a odmah sljedeće godine IBM 7094 koje je bilo jedno od vodećih računala za uporabu na znanstvenom području (engl. *scientific computing*).

Tvrtka CDC (Control Data Corporation) je 1964. izgradila prvo superračunalo CDC 6600 za znanstvenu primjenu. Vodeći istraživač na projektu bio je Seymour Cray koji će kasnije osnovati svoju tvrtku i graditi superračunala Cray I, II, ..., Cray-XMP itd. Općenito, pod *superračunalom* se podrazumijeva računarski sustav koji svojim računskim sposobnostima, odnosno performansom, udovoljava zahtjevima obrade na području vodećih istraživanja u znanosti i inženjerstvu.

2.1.4. TREĆA GENERACIJA – INTEGRIRANI SKLOPOVI (1965. – 1980.)

Izum postupka kojim se deseci tranzistora mogu integrirati na komadiću silicija i oblikovati u integrirani sklop ili čip (R. Noyce, 1958.) najavio je još jednu prekretnicu u izgradnji računala. Zahvaljujući integriranim sklopovima bilo je moguće graditi manja, brža i jeftinija računala. U trećoj generaciji računala poluvodičke memorije zamjenjuju memorije s magnetskim jezgricama, a brzina poluvodičkih memorija dopušta uporabu mikroprogramiranja u izvedbi upravljačkih jedinica. Nadalje, integrirani sklopovi zbog svoje niske cijene dopuštaju gradnju sustava s naglašenim stupnjem paralelnosti (uvišestručenje jedinica za obradu, izvedba protočnih instrukcijskih i aritmetičkih struktura) te višeprogramski rad (engl. *multiprogramming*) pri kojem se u memoriji računala istodobno nalazi više korisničkih programa, a pritom operacijski sustav omogućuje istodobno izvođenje dijelova pojedinih programa).

Spomenimo neka računala i porodice računala iz treće generacije: IBM System/360 Model 30, 40, 50 i 65, UNIVAC 1100 te DEC-ove PDP-11 i VAX 11 porodice računala.

Prvo se vektorsko superračunalo Cray I pojavilo na tržištu 1974.

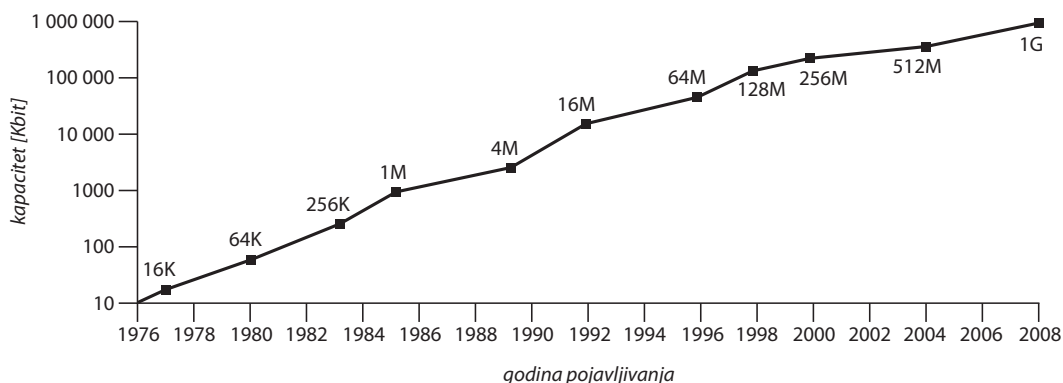
U drugoj polovini 1971. pojavljuje se na tržištu prvi 4-bitni mikroprocesor Intel 4004 koji je bio predviđen kao kalkulatorski čip. Prvi 8-bitni mikroprocesor opće namjene Intel 8008 pojavit će samo godinu dana kasnije. Godine 1974. pojavljuje se druga generacija 8-bitnih mikroprocesora čiji su tipični predstavnici Motorola 6800 i Intel 8080 te nagovještavaju revoluciju na području računala.

2.1.5. ČETVRTA GENERACIJA RAČUNALA – SKLOPOVI VRLO VISOKOG STUPNJA INTEGRACIJE (1980. – ?)

Zahvaljujući razvoju tehnologije vrlo visokog stupnja integracije VLSI (*Very Large Scale Integration*), koja omogućuje, uz nisku cijenu, realizaciju integriranih sklopova koji su početkom 80-ih imali desetke tisuća pa onda stotine tisuća, a danas stotine milijuna tranzi-

stora, računarski sustavi postaju dostupni vrlo širokom spektru korisnika i koriste se u svim sferama ljudske djelatnosti.

Povećanje broja tranzistora integriranih na čipu opisuje Mooreov zakon koji govori da se broj tranzistora na čipu udvostručuje svakih 18 – 24 mjeseca. Slika 2.1 ilustrira razvoj tehnologije VLSI na primjeru dinamičke poluvodičke memorije (DRAM – *Dinamic Random Access Memory*). Na y-osi označen je kapacitet jednog memorijskog čipa izražen u K bitovima (Kb) ($K = 1024$, odnosno 2^{10}) a na x-osi razdoblje od 1976. do 2008. godine. Kapacitet DRAM-a približno se učetverostručivao svake tri godine, tako se, na primjer, kapacitet DRAM čipa sa 16 Kb (1977.) povećao na 1 Gb ($G = 2^{30}$) (2008.). (Opaska: u daljnjem tekstu s b označavat ćemo bit a s B bajt (osam bita).)



Sl. 2.1 Povećanje kapaciteta DRAM čipa

Četvrtu generaciju računala označila je industrija *osobnih računala*. IBM-ova osobna računala temeljena na Intelovom mikroprocesoru Intel 8088, koja su se pojavila na tržištu 1981., postala su najprodavanija računala u povijesti. Osim tvrtke IBM, pojavili su se proizvođači osobnih računala kao što su Commodore, Apple, Amiga i Atari koji su temeljili dizajn na tzv. *non-Intel CPU*, tj. mikroprocesorima drugih proizvođača.

Da bi se dobila predodžba o broju mikroprocesora koji se rabe u različitim računarskim sustavima – osobnim računalima, poslužiteljima (engl. *server*), radnim stanicama, ali i u ugrađenim računalnim sustavima – možemo navesti da je samo 1998. bilo prodano 120 milijuna Intelovih procesora porodice 80x86, 74 milijuna Motorolinih procesora MC 68000, 54 milijuna procesora MIPS, 50 milijuna procesora ARM i 13 milijuna procesora PowerPC. Prema nekim izvorima u 2004. bilo je na svijetu oko 6,4 milijarde stanovnika i 0,8 milijarde osobnih računala pa je u prosjeku svaki osmi stanovnik Zemlje imao osobno računalo.

Naglasimo još jednom, mikroprocesori se ne koriste samo kao građevne sastavnice računala opće namjene, oni se rabe za izgradnju ugrađenih računalnih sustava koji se kao sastavnice ugrađuju u proizvode kao što su videoigre, kućanske naprave, laserski pisači, mobilni telefoni, automobili itd. Na primjer, od 54 milijuna isporučenih procesora MIPS 1998., samo je 1% korišten za računarske sustave opće namjene, dok je preostali dio korišten u ugrađenim računalnim sustavima. Četvrta generacija računala, zahvaljujući razvoju tehnologije, ali i arhitekture računala, obilježena je procesorima vrlo velikih performansi, na primjer 64-bitni procesor Intel Xeon (2005.) ima za faktor 6500 veću performansu u odnosu na VAX 11/780 (1978.).

Godine 1980. javlja se i novi pristup arhitekturi računala, nazvan RISC (*Reduced Instruction Set Computer*), koji se za razliku od tradicionalnog pristupa arhitekturi CISC (*Complex Instruction Set Computer*) temelji na jednostavnijoj, ali bržoj izvedbi procesora.

Navedimo neke od značajki procesora i računarskih sustava četvrte generacije: paralelizam na razini instrukcija – procesori koji izvršavaju istodobno veći broj instrukcija (superskalarani RISC i CISC procesori), višeprosorski sustavi na čipu, odnosno višezegreni procesori (engl. *multicore microprocessor*), višedretveni procesori (engl. *multithread processor*), povećani kapacitet priručne memorije (engl. *cache memory*), procesori za multimedijску primjenu temeljeni na vrlo dugim instrukcijskim riječima – VLIW (*Very Long Instruction Word*).

Razvoj tehnologije sklopovskih sastavnica snažno je utjecao na brzinu računala. Tablica 2.1. prikazuje broj osnovnih operacija u sekundi ovisno o tehnologiji koja se rabi u izvedbi sklopova. Osnovna operacija je ona koja je *izravno podržana sklopovljem*, npr. operacija zbrajanja dvaju brojeva.

| Tehnologija – sastavnica | Godina | Broj osnovnih operacija u sekundi |
|--|--------|-----------------------------------|
| elektromehanika – releji | 1940. | 10 |
| elektronika – elektronske cijevi | 1945. | 103 |
| elektronika – tranzistor | 1950. | 104 |
| mikroelektronika – sklopovi niskog stupnja integracije | 1960. | 105 |
| mikroelektronika – sklopovi srednjeg stupnja integracije | 1980. | 106 |
| mikroelektronika – sklopovi vrlo visokog stupnja integracije | 2000. | 109 |

Tablica 2.1. Broj osnovnih operacija u sekundi ovisno o tehnologiji

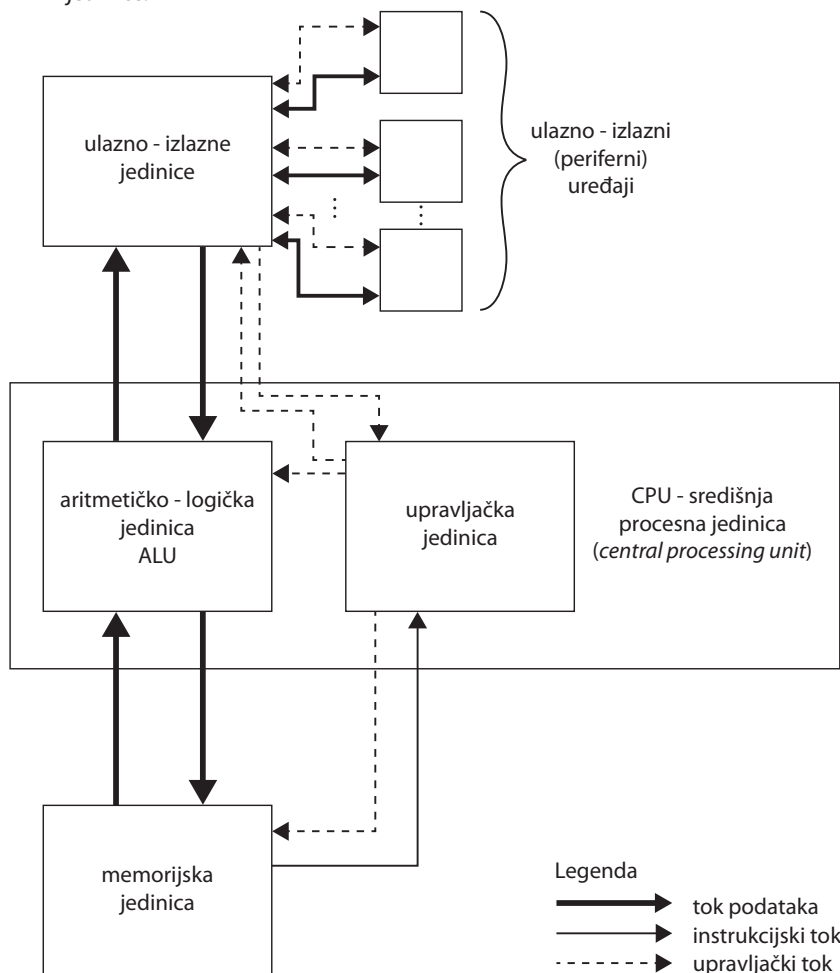
2.2. FUNKCIJSKE JEDINICE VON NEUMANNOVOG MODELA RAČUNALA

Von Neumann, Burks i Goldstine u već spomenutom su članku iznijeli osnovne zahtjeve koji su poslužili kao ishodište za određivanje arhitekture računala:

- i) računalo treba imati opću namjenu i potpuno automatsko izvođenje programa. Pod *potpuno automatskim izvođenjem programa* podrazumijevali su potpunu neovisnost računala o operateru od trenutka započinjanja izvođenja programa, tj. tijekom izvođenja programa ne zahtijevaju se intervencije operatera;
- ii) računalo treba, osim podataka potrebnih za računanje (ulazne vrijednosti, granične vrijednosti, tablice funkcija), pohranjivati međurezultate i rezultate računanja;
- iii) računalo treba imati i sposobnost pohranjivanja programa u obliku slijeda instrukcija.

Neke od izravnih posljedica ishodišnih zahtjeva jesu:

- instrukcije su u računalu svedene na numerički kod, tako da se podaci i instrukcije pohranjuju u jednakom obliku i na jednaki način u istoj jedinici. Ta se jedinica naziva *memorija* ili spremnik (engl. *memory, storage*), odnosno *memorijska jedinica*;
- budući da je računalno prvenstveno *stroj za računanje*, mora imati jedinicu koja obavlja aritmetičke operacije. Ta se jedinica naziva *aritmetička jedinica*. No stroj treba izvoditi i logičke operacije (logičko I, ILI, NE, ISKLJUČIVO ILI) koje trebaju biti podržane u jedinici za računanje pa je aritmetička jedinica nazvana *aritmetičko-logička jedinica*;
- računalno mora imati jedinicu koja tumači i razumije instrukcije svedene na numerički kod, a uz to upravlja slijedom izvršavanja instrukcija (osigurava potpuno automatsko izvođenje programa). Taj je zadatak povjeren upravljačkoj jedinici (engl. *control unit*);
- računalno treba komunicirati s vanjskim svijetom (korisnikom, procesom, drugim računalom). Jedinice koje omogućuju takvu komunikaciju nazivaju se *ulazno-izlazne jedinice*.



Sl. 2.2 Model von Neumannovog računala

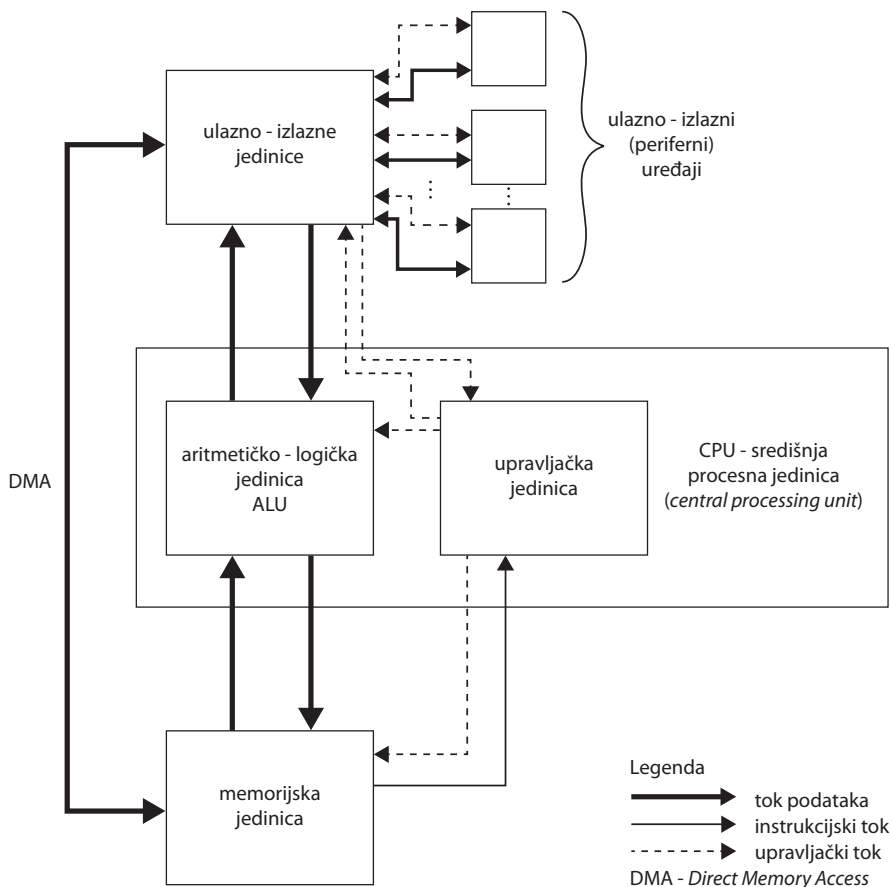
Slika 2.2 prikazuje model von Neumannovog računala. Ono se sastoji od četiri osnovne funkcijske jedinice:

- aritmetičko-logičke,
- upravljačke,
- memorijske i
- ulazno-izlazne jedinice.

2.2.1. UPRAVLJAČKI TOK, INSTRUKCIJSKI TOK I TOK PODATAKA U NEUMANNOVOM MODELU RAČUNALA

Funkcijske jedinice računala povezane su tokom podataka, instrukcijskim tokom i tokom upravljačkih signala. Većinu upravljačkih signala generira upravljačka jedinica na temelju tumačenja instrukcije. Na slici 2.2 debljom punom debljom linijom označen je tok podataka, instrukcijski tok označen je tanjom punom linijom, a crtkanom linijom tok upravljačkih signala.

Promotrimo tok podataka i instrukcijski tok. Vidimo da oba izvire iz memorijske jedinice jer su podaci i instrukcije pohranjeni u toj jedinici. Između memorijske jedinice i aritmetičko-logičke jedinice uspostavljen je dvosmjerni tok podataka: podaci (operandi) koji sudjeluju u aritmetičkim ili logičkim operacijama dohvaćaju se iz memorije, a nakon obavljene operacije podaci koji predstavljaju rezultat obrade pohranjuju se natrag u memorijsku jedinicu. Potrebno je napomenuti da memorijska jedinica *nema* sposobnost obrade, odnosno ne može obavljati niti aritmetičke niti logičke operacije nad operandima. Uočavamo (slika 2.2) da nema izravnog toka podataka između memorijske jedinice i ulazno-izlazne jedinice. Izmjena podataka između memorijske jedinice i ulazno-izlazne jedinice (a time i veza s vanjskim svijetom) u von Neumannovom modelu računala ostvaruje se neizravno: podaci (rezultat obrade) iz memorijske jedinice upućeni ulazno-izlaznoj jedinici moraju proći kroz aritmetičko-logičku jedinicu. Jednako tako, podaci iz ulazno-izlazne jedinice (ulazni podaci) upućeni memoriji moraju proći kroz aritmetičko-logičku jedinicu. To je ujedno i razlog postojanju dvosmjernog toka podataka između aritmetičko-logičke jedinice i ulazno-izlazne jedinice. Prethodno opisani smjerovi podataka odgovaraju dvjema operacijama: *izlaznoj operaciji* (smjer podataka od memorijske jedinice prema ulazno-izlaznoj jedinici) i *ulaznoj operaciji* (smjer podataka od ulazno-izlazne jedinice prema memorijskoj jedinici). Obje su operacije određene strojnim instrukcijama. Vidimo da aritmetičko-logička jedinica nepotrebno sudjeluje u izmjeni podataka između memorijske i ulazno-izlazne jedinice. To ujedno znači da tijekom izmjene podataka između memorije i ulazno-izlazne jedinice ona ne može obavljati svoj osnovni zadatak – aritmetičke ili logičke operacije. Da bi se to izbjeglo, von Neumannov model računala modificiran je tako da je uspostavljen izravan tok podataka između memorijske i ulazno-izlazne jedinice (slika 2.3).



Sl. 2.3 Model von Neumannovog računala s izravnim pristupom memoriji DMA

Izravna veza između memorijske i ulazno-izlazne jedinice naziva se *izravan pristup memoriji* (engl. DMA – *Direct Memory Access*). Prijenosom podataka na tom putu upravlja poseban DMA upravljački sklop pa je omogućen istodobni prijenos podataka i obrada u aritmetičko-logičkoj jedinici. Tok podataka uspostavljen je i između ulazno-izlazne jedinice koja obično predstavlja sučelje (engl. *interface*) s ulazno-izlaznim (perifernim) uređajima (prikazna jedinica, zaslon, pisač, miš, tipkovnica i sl.).

Instrukcijski tok usmjeren je od memorijske prema upravljačkoj jedinici. U skladu s ishodišnim zahtjevima (instrukcije svedene na numerički kod i pohranjene u istoj memorijskoj jedinici kao i podaci) nema razlike u obliku prikaza podataka i instrukcija. Jedino usmjerenost toka između memorijske jedinice i upravljačke jedinice određuje da se na tom spojnem putu nalaze instrukcije. Instrukcije, odnosno numerički kodirane instrukcije tumače se (dekodiraju) u upravljačkoj jedinici i na temelju njihova dekodiranja upravljačka jedinica generira sljedove upravljačkih signala kojima pobuđuje sklopove u aritmetičko-logičkoj jedinici, ali i operacije u ostalim funkcijskim jedinicama. Na primjer, ako je dekodirana instrukcija takva da određuje operaciju dohвата podatka iz memorijske jedinice, upravljačka će jedinica generirati upravljački signal ČITAJ (engl. *READ*) i uputiti ga memorijskoj jedinici.

Na temelju zadatka upravljačke jedinice očekujemo da svi upravljački signali izvire iz nje. No na slici vidimo da postoje i upravljački signali usmjereni prema upravljačkoj jedinici. Istina, u prvotnom von Neumannovom modelu tako usmjereni signali nisu postojali, ali vrlo brzo se pokazala potreba za njima jer se posebni upravljački signali, generirani od ulazno-izlazne jedinice, koriste za prekid izvođenja tekućeg programa, odnosno izvedbu prekidnog sustava (engl. *interrupt system*) kojim se ostvaruje jedan od osnovnih načina izmjena podataka između vanjskog svijeta i računala. Obično se signalne linije kojima se prenose ti upravljački signali nazivaju *prekidne linije*, a signali *zahtjev za prekid* (engl. *Interrupt Request – IRQ*).

Sa slika 2.2 i 2.3 uočavamo da su aritmetičko-logička i upravljačka jedinica prikazani u zajedničkom okviru. Razlog tome je što se objedinjenje aritmetičko-logičke jedinice s najnužnijom memorijom (radnim registrima) i upravljačkom jedinicom naziva *središnja (centralna) procesna jedinica* (engl. CPU – *Central Processing Unit; Central Processor Unit*) ili samo *procesor*.

(Opaska: kada procesoru pridružimo preostale funkcijske jedinice, periferne uređaje i odgovarajuću programsku opremu, govorimo o računalu, odnosno računarskom sustavu.)

2.2.2. ARITMETIČKO-LOGIČKA JEDINICA

Aritmetičko-logička jedinica sastoji se od sklopova koji obavljaju aritmetičke i logičke operacije na podacima (ti se podaci nazivaju *operandi*) te registara za privremeno pohranjivanje operanada i rezultata.

Unatoč dugoj povijesnoj tradiciji uporabe dekadskog brojevnog sustava u računskim strojevima, i tome da je dekadski brojevni sustav čovjeku najprikladniji, von Neumann, Burks i Goldstine odlučili su se za osnovu digitalnog računala uzeti *binarni brojevni sustav*. Osim jednostavnije tehnološke izvedbe sklopova (očito je jednostavnije realizirati sklop s dva diskretna stanja (0 i 1) negoli sklop s deset diskretnih stanja (0, 1, 2, ..., 9)) i ekonomičnijeg prikazivanja brojeva, razlog tome bio je i to što "računalo nije samo aritmetički računski stroj, već po svojoj prirodi treba biti i logički. Logički sustavi su sustavi koji barataju s dva stanja *istinito – lažno*, odnosno 0 ili 1".

Primjer 2.1.

Pokušajmo ocijeniti ekonomičnost brojevnog sustava koji će poslužiti kao podloga za izgradnju sklopova te naći optimalni brojevni sustav. Drugim riječima, zanima nas koju bazu brojevnog sustava trebamo izabrati da bi cijena sklopova za prikaz brojeva u zadanom opsegu od 0 do $N-1$ bila najniža. Potpuno opravdano pretpostavljamo da će cijena sklopova biti proporcionalna broju potrebnih brojevnih mjesta i broju diskretnih stanja koje svaka znamenka na brojevnom mjestu može poprimiti:

$$c = k n B,$$

pri čemu je c cijena (npr. u \$), k je pretvorbena konstanta koja omogućuje da se dobije cijena u pretpostavljenim novčanim jedinicama, n je broj potrebnih brojevnih mjesta i B je baza brojevnog sustava, odnosno broj diskretnih stanja koje znamenka može poprimiti.

Pretpostavimo da želimo predočiti brojeve samo u opsegu od 0 do 999 u dekadskom brojevnom sustavu. Cijena sklopova iznosi:

$$c = k \times 3 \times 10 = 30k$$

gdje je 3 broj potrebnih mjesta, a 10 broj diskretnih stanja koje svaka znamenka može zauzeti.

Ako se pak odlučimo za binarni brojevni sustav ($B = 2$), tada cijena iznosi:

$$c = k \times 10 \times 2 = 20k$$

pri čemu 10 odgovara broju potrebnih brojevni mjesta za prikaz brojeva u opsegu 0 – 999, a 2 je broj diskretnih stanja. Na primjer, 999 (dekadno) predloženo je s deset binarnih znamenki: 1111100111. Vidimo da je binarni brojevni sustav ekonomičniji – zahtijeva nižu cijenu izvedbe sklopova. Koji je brojevni sustav optimalan?

Za ishodište će nam poslužiti izraz za cijenu

$$c = k \times n \times B \quad (2.1)$$

pri čemu za pozicijske brojevne sustave vrijedi da je:

$$N = B^n \quad (2.2)$$

Logaritmirajmo lijevu i desnu stranu izraza (2.2):

$$N = B^n / \log$$

dobivamo:

$$\log N = n \log B \quad (2.3)$$

$$n = \log N / \log B \quad (2.4)$$

Uvrstimo izraz (2.4) u jednadžbu (2.1):

$$c = k (\log N / \log B) B \quad (2.5)$$

Tražimo vrijednost za B za koju će funkcija c postići svoj minimum:

$$\frac{dc}{dB} = 0$$

$$\frac{dc}{dB} = k \log N \frac{\log B - B(1/B) \log e}{\log^2 B} = 0$$

slijedi

$$\log B - \log e = 0,$$

dakle $B = e$.

Rezultat je pomalo čudan – baza optimalnog brojevnog sustava bi trebala biti $e = 2.71828\dots$ (baza prirodnog logaritma \ln)!

Budući da B treba biti cijeli broj, trebamo izabrati između broja 2 i 3. Zbog lakše tehnološke izvedbe sklopova odlučujemo se za bazu $B = 2$ i smatramo je optimalnom.

Aritmetičko-logička jedinica von Neumannovog računala IAS imala je sklop za zbrajanje (zbrajalo) i sklop za posmak (engl. *shifter*) kojim se podatak posmiče ulijevo ili udesno za jedno ili veći broj mjesta. Osim toga, imala je i dva 40-bitna registra za privremeno pohranjivanje operanada i rezultata: registar AC (koji se naziva *akumulator*) i registar MQ koji je upotrebljavao kao "proširenje" akumulatora AC za potrebe pohrane rezultata operacija množenja i dijeljenja. Naime, te operacije daju rezultat dvostruke duljine operanada (80 bitova) pa se 40 značajnijih bitova rezultata smješta u AC, a 40 manje značajnih bitova u MQ. Budući da je aritmetičko-logička jedinica računala IAS imala samo zbrajalo i sklop za

posmak, operacija oduzimanja izvršavala se zbrajanjem umanjnika (minuenda) i potpunog komplementa (engl. *two's complement*) odbitnika (suptrahenda). Množenje i dijeljenje nije bilo izvedeno sklopovljem, već se izvršavalo pod programskim upravljanjem izvođenjem uzastopnih operacija zbrajanja, odnosno oduzimanja i posmaka (podsjetimo se *dualizma sklopovske i programske opreme*; poglavlje 1.)

Operandi su u IAS računalu imali duljinu od 40 bita: 39 bitova bilo je namijenjeno za znamenke, a jedan bit za predznak. Zašto je izabrana duljina operanda od 40 bita? Pozornost von Neumanna i ostalih autora bila je usmjerena na oblikovanje računala koje će rješavati numeričke zadatke (npr. numeričko rješavanje parcijalnih diferencijalnih jednadžbi). Na temelju analize tadašnjih matematičkih problema (1946.) i stvarnih tehnoloških ograničenja došli su do potrebnog kapaciteta radne memorije i duljine riječi, odnosno duljine operanda: 4096 riječi duljine 40 bita. Naime, duljina od 40 bita omogućuje točnost računanja na dvanaest decimala (2^{-40} je približno $0.9 \cdot 10^{-12}$). Brojevi su bili prikazani u obliku predznačnih razlomljenih brojeva s čvrstim pomičnim zarezom (engl. *fixed-point*) u rasponu od +0.999999999998 do -0.999999999998. Brojevi koji su se nalazili izvan tog raspona morali su biti skalirani, odnosno normalizirani.

2.2.3. UPRAVLJAČKA JEDINICA

Upravljačka jedinica na temelju dekodiranja strojne instrukcije generira sve potrebne upravljačke signale za vremensko vođenje i upravljanje ostalim jedinicama računala. Ti se signali dovode u tzv. upravljačke točke i njima se aktiviraju sklopovi u pojedinim funkcijskim jedinicama. Uz to, upravljačka jedinica zadužena je za automatsko izvršavanje programa – upravljanje slijedom izvršavanja instrukcija kojima je predočen algoritam obrade. Svaki je korak algoritma predstavljen jednom strojnom instrukcijom ili slijedom strojnih instrukcija. One određuju elementarne operacije koje sklopovlje može izvesti. Slika 2.4 prikazuje format strojne instrukcije IAS računala. Pod *formatom strojne instrukcije* razumijeva se oblik (organizacija) strojne instrukcije s označenim poljima (nizovima binarnih znamenki 0 i 1) kojima je naznačena funkcija.



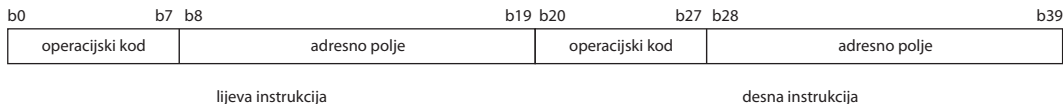
Sl. 2.4 Format strojne instrukcije IAS računala

Strojna instrukcija je duljine 20 bitova (b0 – b19) i organizirana je u dva polja: osam bitova (b0 – b7) predstavljaju polje *operacijskog koda* (*op*) (engl. *opcode – operation code*). Ono određuje operaciju koja će se izvršiti. Svakoj strojnoj instrukciji iz skupa instrukcija *jednoznačno* je pridružen 8-bitni operacijski kod. To znači da IAS može imati skup strojnih instrukcija koji se sastoji od maksimalno 256 instrukcija ($2^8 = 256$).

Drugo, 12-bitno polje (b8 – b19) je *adresno polje* koje sadržava adresu memorijske lokacije na kojoj se nalazi podatak (operand). Svakoj memorijskoj lokaciji u memorijskoj jedinici *jednoznačno* je pridružena adresa što znači da se binarnim slijedom iz adresnog polja može izravno adresirati 4096 memorijskih lokacija ($2^{12} = 4096$). 12-bitna duljina adresnog polja ujedno je i određivala ukupni kapacitet memorije: 4096 40-bitnih riječi.

Strojne instrukcije koje imaju samo jedno adresno polje nazivaju se *jednoadresne strojne instrukcije*.

Vidimo da je duljina memorijske riječi (40 bita) prilagođena duljini riječi podataka (40 bita), dok je duljina strojne instrukcije 20 bitova. To je omogućilo da se dvije strojne instrukcije smještaju u jednu memorijsku lokaciju: lijeva strojna instrukcija i desna strojna instrukcija (slika 2.5).



Sl. 2.5 Lijeve i desne strojne instrukcije IAS računala

Program se izvršava tako da upravljačka jedinica pribavlja (engl. *fetch*) instrukcije u kodiranom obliku iz memorijske jedinice (u IAS računalu istodobno se pribavljaju dvije strojne instrukcije: lijeva i desna), dekodira ih i u skladu s njihovom funkcijom generira upravljačke signale na temelju kojih funkcijske jedinice (aritmetičko-logička jedinica, memorijska jedinica, ulazno-izlazna jedinica) izvode potrebne operacije. Podsjetimo se da je algoritam obrade u obliku slijeda strojnih instrukcija pohranjen u memorijskoj jedinici u kojoj su pohranjeni i podaci. Računalo s takvom značajkom naziva se *računalo s pohranjivanjem programa*. Jedno se pitanje samo od sebe nameće: ako su podaci i strojne instrukcije predložene binarnim kodovima i ako su pohranjeni u istoj memorijskoj jedinici, kako se zna što je podatak, a što instrukcija i razlikuju li se oni na temelju slijeda bitova? Odgovor je pomalo neočekivan: na temelju slijeda bitova nije moguće razlikovati podatak od strojne instrukcije. Za detaljnije objašnjenje ovog odgovora molim čitatelja za malo strpljenja – naći ćemo ga već u ovom poglavlju.

Za IAS računalo kažemo da je *akumulatorsko orijentiran stroj* zato što središnju ulogu u izvođenju aritmetičkih operacija ima spremnik – registar: akumulator AC. Razmotrimo kako će se izvoditi aritmetičke operacije koje zahtijevaju dva operanda, npr. $C = A + B$, gdje su A i B operandi. Općenito, možemo napisati $C = f(A, B)$. Ako ovaj izraz promatramo u svjetlu strojnih instrukcija, onda bismo mogli reći da funkciji f odgovara polje operacijskog koda, a da A i B predstavljaju adrese izvorišta operanada, dok je C adresa odredišta rezultata. U skladu s time, mogli bismo očekivati da strojne instrukcije budu *troadresne*, odnosno da imaju sljedeći format: *op kod, adresno polje1, adresno polje 2, adresno polje 3*, pri čemu adresna polja 1 i 2 predstavljaju adrese izvorišta, a adresno polje 3 adresu odredišta. No strojna instrukcija IAS računala je jednoadresna. Kako se jednoadresnom instrukcijom može podržati operacija koja zahtijeva tri adrese? To je riješeno na sljedeći način. U akumulatoru AC nalazi se jedan od operanada (AC je izvorište jednog operanda), drugi se operand nalazi u memorijskoj jedinici i on je određen 12-bitnom adresom iz *adresnog polja* strojne instrukcije. Odredište rezultata je akumulator AC. Dakle, umjesto $C = f(A, B)$ možemo napisati $A = f(A, M)$, gdje A odgovara akumulatoru AC, a M odgovara adresi memorijske lokacije. Jasno, nakon obavljene specificirane operacije operand u AC je "izgubljen" jer se u akumulatoru AC sada nalazi rezultat.

Računalo IAS imalo je pet osnovnih tipova strojnih instrukcija koji su mogu razvrstati na:

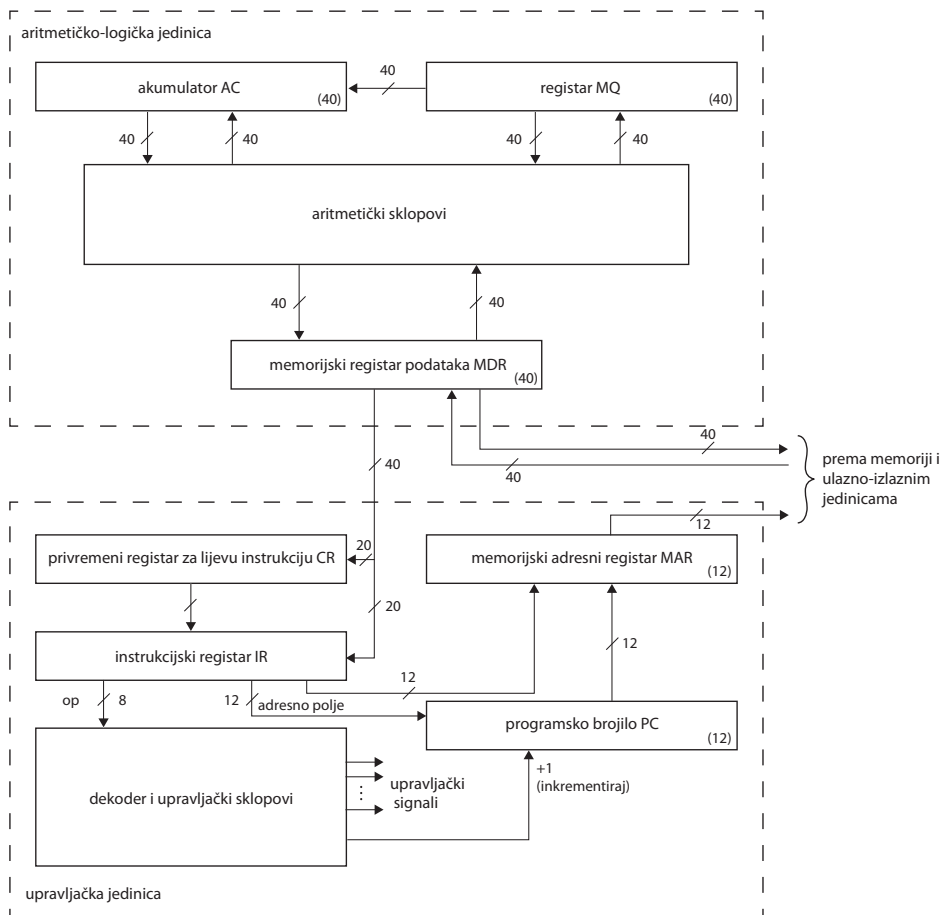
1. instrukcije za prijenos (premještanje) podataka (engl. *data transfer*),
2. instrukcije za obradu podataka (engl. *data processing*)
3. ulazno-izlazne instrukcije,
4. instrukcije za upravljanje izvršavanjem programa (engl. *program control*),
5. instrukcije s djelomičnom zamjenom (engl. *partial substitution*).

Prva četiri tipa instrukcija karakteristična su i za današnja računala. Instrukcije s djelomičnom zamjenom bile su strojne instrukcije koje su mijenjale samo adresno polje strojnih instrukcija. Pomoću njih su programi mogli preoblikovati svoje strojne instrukcije tijekom izvođenja i onda se ista instrukcija mogla primjenjivati na drugom skupu podataka. Ovakva se mogućnost, da program mijenja sam sebe, vrlo brzo pokazala kao neprikladna jer otežava ispitivanje ispravnosti rada programa i otkrivanja pogrešaka u programu te se početkom šezdesetih godina prošlog stoljeća napušta uporaba instrukcija za modifikacije adresnog dijela strojnih instrukcija. Tablica 2.2. prikazuje neke od osnovnih tipova strojnih instrukcija IAS računala.

| Tip instrukcije | Instrukcija | Opis |
|-----------------------|--|--|
| Prijenos podataka | $AC := MQ$ $AC := M(X)$ $M(X) := AC$ $MQ := M(X)$ $AC := - M(X)$ $AC := M(X) $ $AC := - M(X) $ | Prenesi sadržaj registra MQ u registar AC Prenesi sadržaj memorijske lokacije X u AC Prenesi sadržaj AC u memorijsku lokaciju X Prenesi M(X) u MQ Prenesi - M(X) u AC Prenesi apsolutnu vrijednost sadržaja memorijske lokacije X u AC Prenesi minus M(X) u AC |
| Obrada podataka | $AC := AC + M(X)$ $AC := AC + M(X) $ $AC := AC - M(X)$ $AC := AC - M(X) $ $AC.MQ := MQ \times M(X)$ $MQ.AC := AC / M(X)$ $AC := AC \times 2$ $AC := AC / 2$ | Zbroji sadržaje memorijske lokacije X i AC te rezultat pohrani u AC Zbroji apsolutnu vrijednost M(X) i sadržaj AC Oduzmi M(X) od AC Oduzmi M(X) od AC Pomnoži MQ s M(X) i pohrani produkt u obliku riječi dvostruke duljine u AC i MQ Podijeli AC s M(X) i kvocijent pohrani u AC a ostatak u MQ Pomnoži AC s dva (posmak ulijevo za jedno mjesto) Podijeli AC s dva (posmak udesno za jedno mjesto) |
| Upravljanje programom | $go\ to\ M(X, 0:19)$ $go\ to\ M(X, 20:39)$ $if\ AC \geq 0\ then\ go\ to\ M(X, 0:19)$ $if\ AC \geq 0\ then\ go\ to\ M(X, 20:39)$ $M(X, 8:19) := AC(28:39)$ $M(X, 28:39) := AC(8:19)$ | Uzmi sljedeću instrukciju iz lijeve polovine riječi M(X) Uzmi sljedeću instrukciju iz desne polovine riječi M(X) Ako AC sadržava nenegativni broj uzmi sljedeću instrukciju iz lijeve polovine M(X) Ako AC sadržava nenegativni broj, uzmi sljedeću instrukciju iz desne polovine M(X) Zamijeni adresno polje lijeve instrukcije u M(X) s 12 krajnje desnih bitova AC Zamijeni adresno polje desne instrukcije u M(X) s 12 krajnje desnih bitova AC |

Tablica 2.2. Neki od osnovnih tipova strojnih instrukcija IAS računala

Slika 2.6 prikazuje organizaciju središnje procesne jedinice, tj. procesora računala IAS. Dvije se strojne instrukcije smještene u jednoj 40-bitnoj riječi, koja je pribavljena iz memorijske jedinice, privremeno smještaju u *memorijski registar podataka* MDR. Iz MDR registra desna se strojna instrukcija smješta u *instrukcijski registar* IR. Lijeva se strojna instrukcija pohranjuje u *privremeni registar* CR. Operacijski kod (8 bita) desne instrukcije prosljeđuje se sklopu za dekodiranje (izvorno se taj sklop naziva *funkcijska tablica*), a preostalih se 12 bitova (adresno polje) prenosi u *memorijski adresni registar* MAR. MAR sadržava adresu operanda koji će se dohvatiti iz memorijske jedinice i sudjelovati u zadanoj operaciji. Nakon što se desna strojna instrukcija izvede, iz privremenog se registra CR premješta lijeva instrukcija u instrukcijski registar IR i izvodi se na prethodno opisan način. Sadržaj 12-bitnog registra *programsko brojilo* PC povećan za jedan pokazuje na sljedeću memorijsku riječ koja sadržava sljedeće dvije strojne instrukcije programa. Ako je bila dekodirana strojna instrukcija grananja ili skoka (sljedeća strojna instrukcija nije ona koja je uzastopna u programu), onda se njezino adresno polje, umjesto u registar MAR, premješta u programsko brojilo PC. Na taj je način pripremljena adresa memorijske riječi koja sadržava dvije instrukcije koje će se sljedeće pribaviti.



Sl. 2.6 Organizacija središnje procesne jedinice računala IAS

Zapamtimo, programsko brojilo PC sadržava adresu sljedeće strojne instrukcije (odnosno, u računalu IAS adresu memorijske lokacije koja sadržava instrukcijski par), a instrukcijski registar IR sadržava instrukciju čije je izvođenje upravo u tijeku. U računalu IAS programsko se brojilo nazivalo upravljačko brojilo CC (Control Counter), a instrukcijski se registar nazivao registar funkcijske tablice FR (Function Table Register).

Izvođenje strojne instrukcije odvija se u dvije faze (obje se faze nazivaju *instrukcijski ciklus*): PRIBAVI (engl. *fetch*) i IZVRŠI (engl. *execute*). Tijekom faze PRIBAVI upravljačka jedinica pribavlja strojnu instrukciju iz memorijske jedinice te se događa sljedeće:

1. korak: $M(PC) \rightarrow IR$; iz memorije se pribavlja (čita se) strojna instrukcija i smješta u instrukcijski registar IR (Opaska: u računalu IAS pribavljaju se dvije strojne instrukcije smještene u 40-bitnoj riječi.); adresa strojne instrukcije nalazi se u programskom brojilu PC;
2. korak: $PC + 1 \rightarrow PC$; sadržaj programskog brojila PC povećava se za jedan i time određuje strojnu instrukciju koja neposredno slijedi za instrukcijom koja je upravo pribavljena; u skladu s opaskom iz 1. koraka u računalu IAS određuje se par strojnih instrukcija iz slijeda;
3. korak: dekodira se operacijski kod strojne instrukcije koja je bila pribavljena u 1. koraku.

Trećim se korakom završava faza PRIBAVI. Upravljačka jedinica prelazi u fazu IZVRŠI. Za vrijeme faze IZVRŠI, upravljačka jedinica, ovisno o ishodu dekodiranja operacijskog koda, generira sljedove upravljačkih signala kojima pobuđuje operacije izravno podržane sklopovljem, na primjer, prijenos podataka iz memorijske jedinice, prijenos podataka između registra i aritmetičko-logičkih sklopova, aktiviranje aritmetičko-logičkih sklopova, prijenos podataka memorijskoj jedinici, promjena vrijednosti programskog brojila (ako je riječ o strojnoj instrukciji grananja ili skoka) i sl.

Na primjer, faza IZVRŠI, za aritmetičku operaciju *zbroji* (*add*) može biti predočena sljedećim koracima:

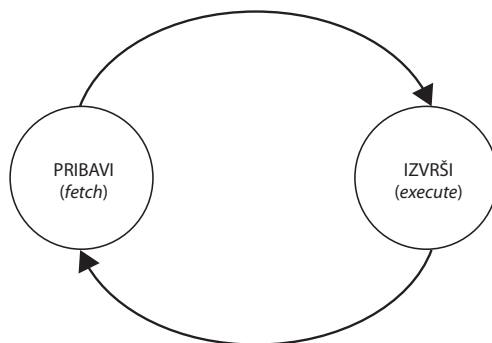
4. korak: $M(MAR) \rightarrow MDR$; dohvati podatak (operand) iz memorijske jedinice i smjesti ga u memorijski registar podataka MDR; adresa memorijske lokacije na kojoj se nalazi operand određena je sadržajem memorijskog adresnog registra MAR;
5. korak: $AC + MDR \rightarrow AC$; izvedi operaciju zbrajanja; rezultat operacije smješta se u akumulator AC.

Izvođenjem posljednjeg koraka u fazi IZVRŠI, upravljačka se jedinica vraća na 1. korak faze PRIBAVI i nastavlja se faza PRIBAVI. Ritam izmjene PRIBAVI – IZVRŠI nastavlja se sve dok se ne izvede strojna instrukcija za zaustavljanje rada (Halt). Slika 2.7 prikazuje dijagram stanja instrukcijskog ciklusa za von Neumannov model računala. Napomenimo da se u suvremenim procesorima promjene stanja PRIBAVI – IZVRŠI događaju i nekoliko stotina milijuna puta u sekundi.

(Opaska: u literaturi (L. Budin i sur.) koriste se i nazivi “dohvat instrukcije” za fazu PRIBAVI te “obavljanje operacije” za fazu IZVRŠI.)

U memorijskoj su jedinici pohranjeni podaci i strojne instrukcije u istom obliku – kao nizovi 0 i 1. Kako upravljačka jedinica razlikuje strojnu instrukciju i podatak? Na temelju binarnog niza nije ih moguće razlikovati. Strojna instrukcija i podatak mogu se razlikovati jedino na temelju *stanja upravljačke jedinice*. Ako se upravljačka jedinica nalazi u fazi PRIBAVI, tada se riječ pribavljena iz memorijske jedinice smatra strojnom instrukcijom (ili parom strojnih

instrukcija u računalu IAS) i smješta se u instrukcijski registar IR, odnosno u slučaju računala IAS – desna strojna instrukcija smješta se u IR, a lijeva u registar CR. Ako se, međutim, upravljačka jedinica nalazi u fazi IZVRŠI, riječ koja se dohvaća iz memorijske jedinice tumači se kao podatak (operand). Jednako tako, ako se tijekom faze IZVRŠI riječ pohranjuje u memorijsku jedinicu, onda je to podatak ili rezultat neke aritmetičke ili logičke operacije.



Sl. 2.7 Dijagram stanja instrukcijskog ciklusa za von Neumannov model računala

2.2.4. MEMORIJSKA JEDINICA

Memorijska jedinica IAS računala bila je, za današnje stanje razvoja tehnologije, vrlo skromnog kapaciteta: 4096 40-bitnih riječi, dakle 4 K 40-bitnih riječi ($1 \text{ K} = 2^{10}$). No izvedba 163840 bistabila, tj. $4096 \cdot 40$, bila je zbog tadašnjih tehnoloških ograničenja ocijenjena kao neizvediva. Priklonili su se rješenju u kojem je memorijska jedinica bila realizirana uporabom elektronskih cijevi sličnih katodnoj cijevi koje su bile razvijene u tvrtki RCA, Princeton. Te su se elektronske cijevi nazivale *Selectron* i za izvedbu memorijske jedinice uporabljeno je 40 takvih *Selectrona* – svaki od njih bio je kapaciteta 4096 bita. Svaka je elektronska cijev pohranjivala bitove u obliku tamnih (0) i svijetlih polja (1), a zaslon cijevi osvježavao se elektronskim snopom. Elektronski se snop rabio i za čitanje i upisivanje podataka u *Selectron*, tako da je to bila memorija s izravnim pristupom do željenog položaja (engl. *random access memory*). Naime, za memorije s izravnim pristupom vrijedi da je vrijeme pristupa memorijskoj lokaciji neovisno o položaju te lokaciji u memorijskoj jedinici. Brzina memorijske jedinice tj. vrijeme pristupa (engl. *memory access time*) bilo je 50 μs .

Memorijska jedinica nema sposobnosti obrade podatka, već može izvoditi dvije vrlo važne operacije – pohranu podatka i dohvaćanje prethodno pohranjenog podatka. Dohvaćanje podatka opisano je operacijom *čitanja* (engl. *Read*), a pohrana podatka operacijom *pisanja* (engl. *Write*).

Riječ iz memorije dohvaća se tako da se adresa memorijske lokacije čiji se sadržaj želi dohvatiti (pročitati) smješta u memorijski adresni registar MAR i time adresa postaje raspoloživa na adresnoj sabirnici, a zatim upravljačka jedinica generira upravljački signal Čitaj koji se šalje memorijskoj jedinici. Izabrana se riječ, nakon isteka vremena pristupa memoriji, npr. 50 μs (za računalo IAS), prenosi preko puta podataka (sabitnice podataka) procesoru i smješta u memorijski registar podataka MDR. Napomenimo da operacija čitanja *nije* destruktivna operacija, odnosno ona ne “ruši” ili ne mijenja sadržaj pročitane memorijske lokacije.

Operacija pisanja izvodi se na sljedeći način: podatak koji se želi pohraniti u memorijskoj jedinici smješta se u memorijski registar podataka MDR i time postaje raspoloživ na putu podataka (sabinici podataka) između procesora i memorije, adresa memorijske lokacije na kojoj se želi pohraniti podatak postavlja se u memorijski adresni registar MAR i time adresa postaje raspoloživa na adresnoj sabinici. Upravljačka jedinica generira upravljački signal Piši i šalje ga memorijskoj jedinici. Nakon isteka vremena pristupa memoriji, podatak biva pohranjen na izabranoj memorijskoj lokaciji. Operacija pisanja je, uvjetno rečeno, destruktivna u smislu da je novi podatak koji se pohranjuje na željenu memorijsku lokaciju, "prebrisao" stari sadržaj na toj memorijskoj lokaciji.

Von Neumann, Burks i Goldstine, svjesni da postoje mnogi važni razredi problema koji zahtijevaju veći kapacitet memorijske jedinice od 4096 riječi, razmatraju *hijerarhijsku organizaciju memorije*. Ona se sastoji od radne (primarne ili glavne) memorije, sekundarne memorije i treće razine – tzv. neaktivne memorije (engl. *dead store*). Radna je memorija bila u IAS računalu ostvarena Selectronima i izravno je podržavala rad procesora. Ona je ujedno bila i najbrža memorija u predloženoj hijerarhijskoj organizaciji. Sekundarnom memorijom, koja je većeg kapaciteta od radne memorije, jeftinija (cijena po bitu je manja), ali sporija, upravlja računalo i ona predstavlja njegov sastavni dio. Sekundarna je memorija kao medij za pohranu podataka koristila magnetsku žicu ili svjetlosno osjetljiv film.

Treća razina memorije jest neaktivna memorija i ona nije sastavni dio računala. Ona se po potrebi uključuje u računalo. Od memorije u drugoj razini razlikuje se samo po raspoloživosti (zahtijeva ručni zahvat operatera da bi se uključila u računalo).

2.2.5. ULAZNO-IZLAZNA JEDINICA

Računalo IAS razmjenjivalo je podatke s okolinom, odnosno operaterom. Za to su bile predviđene ove jedinice: grafička izlazna jedinica i teleprinter s pomoćnom magnetskom žicom. Kao grafička izlazna jedinica poslužile su elektronske cijevi *Selectron* koje su imale svjetla polja na pozicijama na kojima su bile pohranjene jedinice (1) i tamna polja koja su odgovala nulama (0).

Teleprinter s pomoćnom magnetskom žicom upotrebljavao se kao ulazno-izlazna jedinica. Oprema teleprintera bila je preinačena tako da je omogućavala i upis podataka s bušene papirne vrpce na magnetsku žicu, i obratno.

Strojna instrukcija INPUT(X, N) specificirala je prijenos N podataka od ulazne jedinice procesoru i onda pohranu na N slijednih memorijskih lokacija, započevši s memorijskom lokacijom s adresom X. Slično, strojna instrukcija OUT(X, N) određivala je prijenos N slijednih podataka iz memorijske jedinice, započevši s memorijskom lokacijom s adresom X, prema izlaznoj jedinici. I u ovom je slučaju na putu podataka bio i procesor.

Računalo IAS bilo je jednokorisničko računalo (engl. *single-user oriented*) – dopuštalo je istodobni rad samo jednom korisniku i zato se izmjena podataka s vanjskim svijetom obavljala pod izravnim upravljanjem procesora. Autori računala IAS razmatraju mogućnost istodobnog rada ulazno-izlazne jedinice i procesora, međutim, zbog tehnoloških ograničenja odustaju od takve izvedbe.

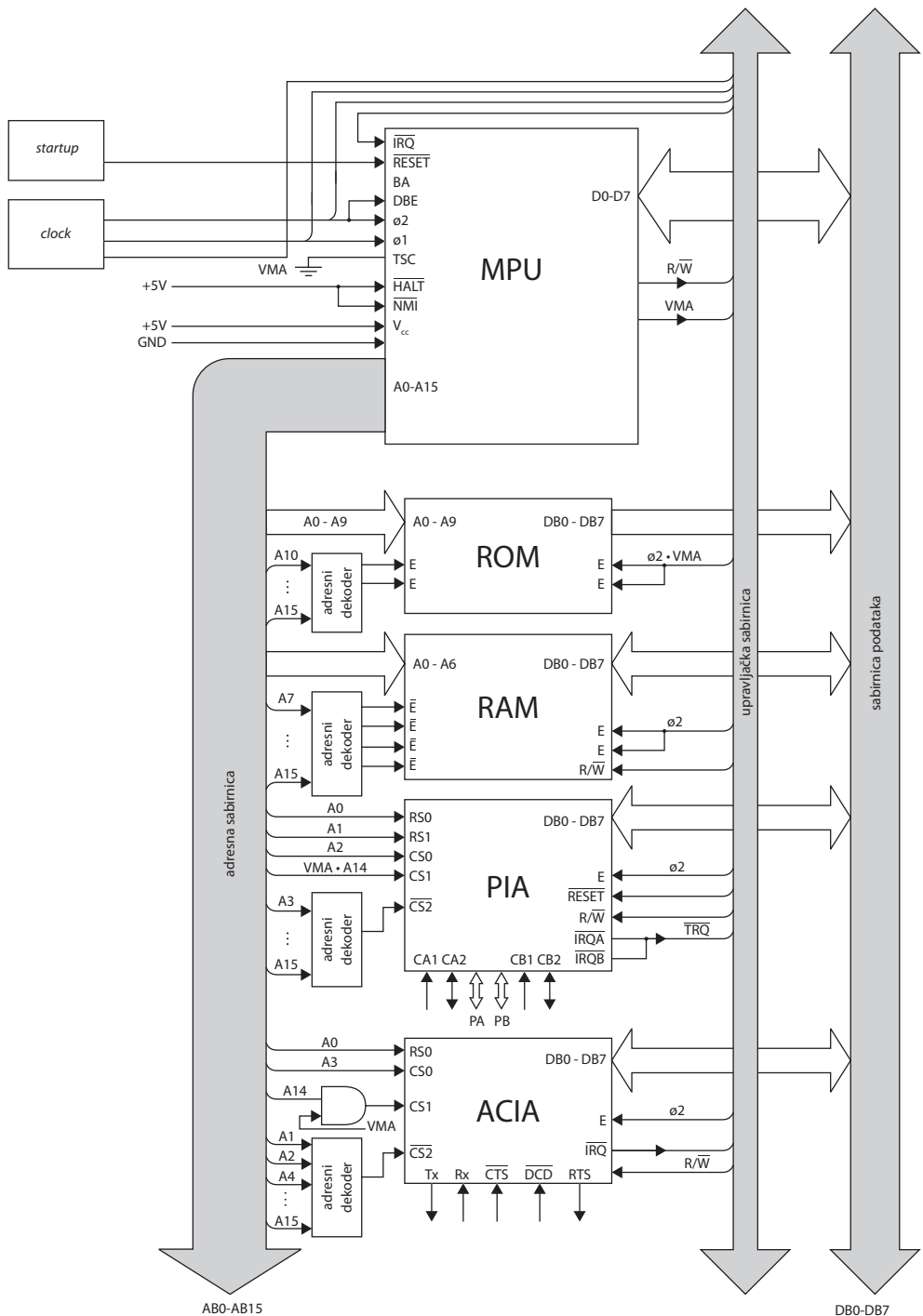
2.3. JEDNOSTAVNO MIKRORAČUNALO – VON NEUMANNOV MODEL RAČUNALA

U ovom smo poglavlju veliku pozornost posvetili von Neumannovom računskom modelu i IAS računalu. Osnovni razlog za to je što je taj model, tijekom skoro pola stoljeća, odredio osnovne zamisli u oblikovanju računala i snažno utjecao na arhitektonske značajke suvremenih računala. Zamisli kao što je računalo opće namjene, svođenje podataka i instrukcija na numerički (binarni) kod, njihovo pohranjivanje u istoj memorijskoj jedinici, četiri osnovne funkcijske jedinice (aritmetičko-logičko jedinica, upravljačka jedinica, memorijska jedinica, ulazno-izlazna jedinica) i četiri osnovna razreda strojnih instrukcija (instrukcije za prijenos (premještanje) podataka, instrukcije za obradu podataka, ulazno-izlazne instrukcije, instrukcije za upravljanje izvršavanjem programa) određuju značajke koje nalazimo u suvremenim računalima. Naravno, za više od pet desetljeća von Neumannov model računala doživio je brojne promjene i poboljšanja koja su se snažno odrazila na njihovoj performansi, pouzdanosti, raspoloživosti, prilagodljivosti i dramatičnom padu cijene računala. Veliku je zaslugu za sve to imao strahoviti tehnološki napredak, posebno mikroelektronike, odnosno tehnologije visokog stupnja integracije. Da bismo potkrijepili tvrdnju o snažnom utjecaju von Neumannovog modela na generacije računala koje su mu slijedile, razmotrimo mikroračunalo na slici 2.8. Taj jednostavan sustav temeljen je na 8-bitnom mikroprocesoru MC 6800 (procesoru ostvarenom u tehnologiji visokog stupnja integracije LSI – *Large Scale Integration*). Sve su ostale njegove funkcijske jedinice također realizirane u tehnologiji LSI tako da je tiskana pločica na kojoj se nalazi cijeli sustav veličine dlana. Je li računalo sa slike 2.8. takvo da se zaista temelji na von Neumannovom modelu?

Pokušajmo identificirati njegove osnovne funkcijske jedinice. Blok (slika 2.8) s oznakom MPU što je kratica *Microprocessor Unit* očito predstavlja procesor realiziran u LSI tehnologiji (čip) koji objedinjuje dvije funkcijske jedinice – upravljačku jedinicu i aritmetičko-logičku jedinicu. Blok s oznakom ROM (*Read Only Memory*) predstavlja memorijski čip koji zbog svoje izvedbe trajno pohranjuje sadržaj (on ostaje sačuvan i ne briše se nakon prestanka napajanja). Osim toga, sadržaj ROM-a ne može se mijenjati strojnim instrukcijama tijekom izvođenja programa. Naravno, sadržaj ROM-a može se strojnim instrukcijama čitati i zato takvu vrstu memorije nazivamo *ispisna memorija*. Zbog takvih se njezinih značajki u ispisnoj memoriji pohranjuju podaci koji se ne mijenjaju te stalni programi (npr. programi koji čine jezgru operacijskog sustava).

Čip s oznakom RAM (*Random Access Memory*) predstavlja memorijsku jedinicu, odnosno *memoriju s izravnim pristupom*. Ona tijekom izvođenja programa dopušta i upisivanje i čitanje podataka. Prestankom napajanja njezin se sadržaj briše. Zaključimo, čipovi ROM i RAM predstavljaju treću funkcijsku jedinicu – memorijsku jedinicu.

Čipovi označeni s PIA (*Peripheral Interface Adapter*) i ACIA (*Asynchronous Communication Interface Adapter*), gdje su PIA i ACIA komercijalni nazivi sklopova, predstavljaju ulazno-izlazne jedinice mikroračunala. PIA omogućuje *paralelni prijenos* podataka između mikroračunala i perifernih uređaja (i/ili vanjskog svijeta). Preko linija PA0-PA7 i PB0-PB7 (slika 2.8; označene samo s PA i PB) prenose se 8-bitni podaci (2 puta po 8 bita). Podsjetimo se, paralelni je prijenos podataka takav prijenos pri kojem je svakom bitu podatka dodijeljena jedna linija (vodič) za prijenos i cijeli se podatak prenosi istodobno. Smjer prijenosa podataka preko linija PA0 – PA7 i PB0 – PB7 određuje se tijekom inicijalizacije PIA koja se obavlja programom. Kažemo da je PIA *programirljiv sklop* jer mu se početna konfiguracija i način rada određuje programom.



Sl. 2.8 Jednostavno mikroračunalo temeljeno na mikroprocesoru MC 6800 (tvrtka Motorola)

PIA ima još dva para linija kojima je povezana s vanjskim svijetom: CA1 i CA2, te CB1 i CB2. Te se linije nazivaju *linije za rukovanje* (engl. *handshaking*) i njima se ostvaruje izmjena upravljačkih signala između računala i vanjskog svijeta. Ti signali služe za ostvarivanje jednostavnih *protokola* koji se koriste u izmjeni podataka i omogućuju pouzdani prijenos podataka.

Čip ACIA je ulazno-izlazna jedinica koja podržava *serijski prijenos* podataka između mikroručunala i vanjskog svijeta. Serijski prijenos podataka je takav da se jednom signalnom linijom prijenosi bit po bit podatka (kažemo u obliku "vlak impulsa"). Linija označena s Rx (Receive) (slika 2.8) je prijemna signalna linija kojom se podaci iz vanjskog svijeta prenose u računalo. Linija Tx (Transmit) (slika 2.8) jest predajna linija kojom se podaci šalju u vanjski svijet. ACIA je također *programirljiv sklop* i u fazi inicijalizacije programom se određuju detalji načina njegovog rada. Na slici 2.8. uočavate da ACIA ima još i druge signalne linije (\overline{CTS} , \overline{DCD} , RTS , ...) – one su namijenjene posebnom sklopu *modemu* koji omogućuje prijenos serijskih podataka telefonskim linijama (paricama).

Opisali smo glavne sastavnice mikroručunala sa slike 2.8: MPU, ROM, RAM PIA i ACIA. Te su građevne sastavnice osnovne funkcijske jedinice koje odgovaraju von Neumannovom modelu računala, samo što su one realizirane u LSI tehnologiji.

Na slici vidimo i blok označen s *clock* – to je čip koji ima funkciju generatora signala vremenskog vođenja odnosno takta. Naime, računalo je, poput Turingovog stroja, *diskretni vremenski stroj* u kojem je trenutak svake promjene određen signalom vremenskog vođenja ili "clockom" – kako mi to u žargonu nazivamo. Signal vremenskog vođenja ("clock") je periodički signal određene frekvencije čija se velika stabilnost frekvencije postiže uporabom kristalnog oscilatora. Zapravo, ovo mikroručunalo rabi dva signala vremenskog vođenja (kažemo da koristi "dvofazni signal" vremenskog vođenja): ϕ_1 i ϕ_2 (slika 2.8). Za mikroručunalo sa slike 2.8. frekvencija oba signala vremenskog vođenja jest 1 MHz; ($M = 10^6$). Budući da signal vremenskog vođenja određuje vremenske trenutke u kojima se mijenja stanje u sklopovlju, jasno je da frekvencija signala vremenskog vođenja utječe na brzinu izvođenja programa, odnosno na performansu računala (današnja osobna računala imaju frekvenciju 3 i više GHz ($G = 10^9$)). (Opaska: kada ne govorimo o kapacitetu memorije obično za K, M ili G podrazumijevamo sljedeće: $K = 10^3$, $M = 10^6$ i $G = 10^9$.) Sve su aktivnosti mikroručunala sinkronizirane signalom vremenskog vođenja – signal se dovodi procesoru MPU (oba signala: ϕ_1 i ϕ_2), ali i ROM-u (ϕ_2), RAM-u (ϕ_2), PIA-i (ϕ_2) i ACIA-i (ϕ_2) (slika 2.8).

Blok označen na slici 2.8 sa *start up* predstavlja relativno jednostavan sklop za početno upuštanje u rad mikroručunala, odnosno za vraćanje mikroručunala u početno stanje (u žargonu kažemo "resetiranje"). Sklop ima obično gumb označen s \overline{RESET} . Korisnik pritiskom na taj gumb, preko sklopa *start up*, aktivira ulaznu upravljačku liniju označenu s kojom se računalo postavlja u početno stanje. Aktiviranje \overline{RESET} ($\overline{RESET} \rightarrow 0$) znači za procesor iznimku, odnosno prekid vrlo visoke razine.

Slika 2.8 može nam poslužiti i za tumačenje toga kako se fizički u računalu ostvaruje tok podataka, instrukcijski tok i tok upravljačkih signala, odnosno kako se ostvaruje komunikacija među funkcijskim jedinicama. Na slici vidimo tri komunikacijska puta kojima su funkcijske jedinice povezane: put podataka (engl. *data bus*), put upravljačkih signala (engl. *control bus*) i adresni put (engl. *address bus*). Svaki od tih putova, koji se naziva *sabirnica* (engl. *bus*), fizički predstavlja skup linija (vodiča) kojim se prenose signali (koji svojim razinama odgovaraju logičkoj "0" i logičkoj "1"). S obzirom na namjenu sabirnice, govorimo o *sabirnici podataka* (ili *podatkovnoj sabirnici*), *upravljačkoj sabirnici* i *adresnoj sabirnici*.

Tok podataka i instrukcijski tok ostvareni su pomoću *sabirnice podataka*. Ona je *dvosmjerna* jer dopušta prijenos od procesora prema drugim funkcijskim jedinicama (memorijskoj i ulazno-izlaznim jedinicama), ali i prijenos od navedenih funkcijskih jedinica prema procesoru. Odmah ovdje navedimo jedan važan detalj: *referentna točka* u analizi ponašanja računalnog sustava i određivanju smjerova prijenosa jest *procesor*, koji ima ulogu *vodećeg modula* (engl. *master module*) jer on upravlja cijelim računarskim sustavom. Ostale funkcijske jedinice imaju ulogu *pratećeg modula* (engl. *slave*) i one "poslušno" obavljaju zadatke koje im upućuje vodeći modul – procesor. Budući da je procesor određen kao referentna točka, onda se linije usmjerene od procesora prema ostalim jedinicama nazivaju *izlaznim*, a one u suprotnom smjeru *ulaznim linijama*. Analogno tome, ako kažemo da se izvodi operacija čitanja (engl. *read*), to znači da vodeći modul, tj. procesor "čita" podatak, odnosno pristupa nekoj memorijskoj lokaciji ili nekom registru u ulazno-izlaznim jedinicama i dohvaća taj podatak (podatak predstavlja ulazni podatak za procesor).

Sabirnica se podataka za mikroročunalo na slici 2.8. sastoji od osam linija (ili vodiča) označenih s DB0, DB1, ..., DB7. Linija DB0 služi za prijenos najmanje značajnog bita podatka, dok će linija DB7 prenositi najznačajniji bit podatka (prijenos je paralelan). Za sabirnicu podataka DB0 – DB7 kažemo da je 8-bitna, tj. da je "širine" od osam bita.

Kako se njome ostvaruje instrukcijski tok i tok podataka? Kad je upravljačka jedinica u stanju PRIBAVI, iz memorijske se jedinice pribavlja podatak koji se tumači kao strojna instrukcija, odnosno sastavnica strojne instrukcije (ako se ona sastoji od više 8-bitnih riječi). Podatak – strojna instrukcija – preko sabirnice podataka prenosi se procesoru. Tijekom faze IZVRŠI podaci koji se dohvaćaju iz memorijske jedinice ili se šalju memorijskoj jedinici, preko sabirnice podataka predstavljaju elemente toka podataka. Zaključimo, dvosmjerna sabirnica podataka predstavlja komunikacijski put kojim se ostvaruje instrukcijski tok i tok podataka.

Upravljački signali koje generira upravljačka jedinica šalju se ostalim funkcijskim jedinicama pomoću *upravljačke sabirnice*. Ona se sastoji od pojedinačnih linija (vodiča) kojima se prenose upravljački signali. Na primjer, izlazna upravljačka linija označena kao R/\overline{W} prenosi upravljački signal "Čitaj" ($R/\overline{W} = 1$) ili "Piši" ($R/\overline{W} = 0$) memorijskoj jedinici, odnosno čipu RAM i time određuje operaciju koju treba obaviti memorijska jedinica. Pažljivi će čitatelj primijetiti da se taj signal dovodi i do ulazno-izlaznih jedinica (PIA i ACIA; slika 2.8), a o razlozima za to bit će riječi u poglavlju o ulazno-izlaznim jedinicama. Pažljivi će čitatelj također primijetiti da se upravljački signal R/\overline{W} ne dovodi memorijskom modulu ROM - razlog tome je što je ROM samo ispisna memorija tako da se već samim izborom tog modula određuje operacija čitanja.

Skup upravljačkih linija čini upravljačku sabirnicu. U slučaju mikroročunala sa slike 2.8 sabirnice čine linije označene s R/\overline{W} , ϕ_2 , VMA , $RESET$ i \overline{IRQ} .

Upravljačka linija \overline{IRQ} je ulazna linija i njome upravljaju ulazno-izlazne jedinice (na slici 2.8 vidi se da \overline{IRQ} "izvire" i iz PIA i ACIA). Ta se upravljačka linija naziva *prekidna linija*, a signal koji ona prenosi je *zahtjev za prekid*. Aktivno stanje signala \overline{IRQ} ($\overline{IRQ} = 0$) znači procesoru da je neka ulazno-izlazna jedinica zahtijevala prekid izvođenja tekućeg programa.

Na slici 2.8 vidimo još jednu sabirnicu – adresnu sabirnicu koja je širine šesnaest bita AB0 – AB15. Ona je iznimno važna jer pomoću nje procesor određuje kojoj lokaciji u memorijskoj jedinici ili kojem registru u ulazno-izlaznoj jedinici želi pristupiti. Von Neumanov model računala koristi adrese koje su jednoznačno dodijeljene svakoj sastavnici kojoj procesor

može pristupiti: *svaka memorijska lokacija ima jednoznačno pridruženu adresu, svaki registar u bilo kojoj funkcijskoj jedinici također ima jednoznačno pridruženu adresu*. Na temelju jednoznačne adrese procesor izabire sastavnicu s kojom želi komunicirati. Važno je napomenuti da se načelo *jednoznačnosti adrese* u računalnom sustavu mora strogo poštovati – ne smiju u računalu postojati npr. dvije memorijske lokacije s istom adresom, dva registra ili dvije funkcijske jedinice s istom adresom. Ako se to načelo ne poštuje, računalo neće ispravno funkcionirati, štoviše može imati za posljedicu kvar sklopovske sastavnice (npr. posebnih pogonskih sklopova; engl. *driver*, koji su sučelni sklopovi na sabirnici računala). Analogija, s poštovanjem načela *jednoznačnosti adrese*, jest dobro uređen grad u kojem ne smiju postojati dvije ili veći broj ulica s istim imenom ili, pak, više zgrada s jednakim kućnim brojem u istoj ulici. Naravno, kad se ta jednoznačnost ne bi poštovala, pisma i druge poštanske pošiljke bilo bi nemoguće uručiti pravom primatelju. Slično je i s “porukama” koje šalje procesor.

Adresna sabirnica je *jednosmjerna* – ona izvire iz procesora i njome se adresa koju generira procesor prenosi svakoj funkcijskoj jedinici. Funkcijske jedinice “oslušuju” adresnu sabirnicu i ako utvrde da je na njoj adresa koja upravo odgovara adresi koja je njima dodijeljena, aktiviraju se te obavljaju zadatak koji im je procesor odredio na temelju stanja upravljačkih signala.

Primjer 2.2.

Slika 2.9 prikazuje detalj priključenja memorijske jedinice RAM (slika 2.8) na sabirnicu podataka DB0 – DB7, upravljačku sabirnicu te adresnu sabirnicu AB0 – AB15. Adresna sabirnica je širine 16 bita i određuje ukupan adresni prostor od 65356 adresa (2^{16}): od 0000 - FFFF (heksadekadno), tj. ukupno 64 K lokacija ($K = 2^{10}$).

Čip RAM ima četiri ulazna priključka označena s \bar{E} (engl. *enable* – omogućiti) i dva ulazna priključka označena s E . Ti ulazni priključci služe za izbor čipa na temelju adrese. Čip je izabran, tj. omogućen i postaje aktivan ako je zadovoljen sljedeći uvjet: *istodobno* moraju sve logičke vrijednosti signala na priključcima označenima s \bar{E} i E poprimiti aktivno stanje, dakle redom $\bar{E} = 0, \bar{E} = 0, \bar{E} = 0, \bar{E} = 0, E = 1$ i $E = 1$. Na slici 2.9 vidimo da su adresne linije A7 - A15, preko logičkih sklopova ILI priključene na četiri ulaza označena s \bar{E} , a da je na dva priključka E doveden signal vremenskog vođenja $\phi 2$.

(Opaska: radi jednostavnosti adresne linije, označavat ćemo kraće samo s A i pripadajućim brojem, npr. A7 = AB7).

U skladu s prethodno navedenim uvjetom, RAM će biti izabran i “oživjeti” kada je stanje na adresnoj sabirnici sljedeće (slika 2.9):

A7 = 0, A8 = 0 i A9 = 0; logičko ILI A7 + A8 + A9 je tada 0,

A10 = 0 i A11 = 0; logičko ILI A10 + A11 je tada 0,

A12 = 0 i A13 = 0; logičko ILI A12 + A13 je tada 0,

A14 = 0 i A15 = 0; logičko ILI A14 + A15 je tada 0,

(odnosno kada su sve adresne linije od A7 do A15 u logičkoj nuli) i kada je istodobno zadovoljeno da su dva ulaza E u jedinici. Budući da se na priključke E dovodi signal $\phi 2$, koji je periodički, ulazi E će biti u aktivnom stanju u ritmu signala vremenskog vođenja $\phi 2$ (odnosno kada njegova amplituda poprima visoku vrijednost koja se može tumačiti kao logičko 1).

Adresni potprostor koji zauzima RAM dio je veličine od samo 128 adresa od ukupnog adresnog prostora od 64 K i definiran je adresama: 0000 – 007F. Budući da sedam najmanje značajnih adresnih linija (A0 – A6) koje se dovode na priključke čipa označene s A0 – A6 ne utječu na izbor RAM-a, već samo na izbor riječi u RAM-u (uz uvjet da su adrese linije A7 – A15 poprimile odgovarajuće vrijednosti), onda se obično adresa sklopa prikazuje u obliku:

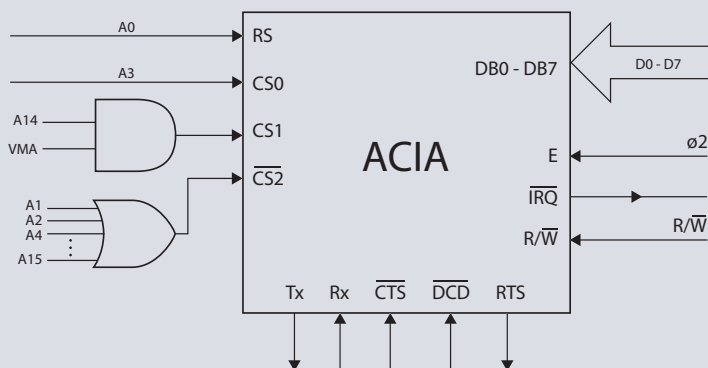
| | | | | | | | | | | | | | | | |
|-----|-----|-----|-----|-----|-----|----|----|----|----|----|----|----|----|----|----|
| A15 | A14 | A13 | A12 | A11 | A10 | A9 | A8 | A7 | A6 | A5 | A4 | A3 | A2 | A1 | A0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | X | X | X | X | X | X | X |

gdje X označava bilo 0 bilo 1 (X se tumači kao “nije važno”; engl. *don't care*).

Primjer 2.3.

Određimo adresni potprostor u kojem se pojavljuje ulazno-izlazna jedinica ACIA. Priključci čipa ACIA koji su u vezi s njezinim adresiranjem jesu (slika 2.10):

- RS (*Register Select*) – priključak za izbor registra u ACIA. Samo na temelju RS-a možemo zaključiti da se ACIA javlja procesoru kao sklop koji ima dva registra (jedan kad je RS = 0 i drugi kad je RS = 1),
- CS0 (*Chip Select*) – priključak za izbor čipa (ima istu funkciju kao priključak E u prethodnom primjeru),
- CS1 (*Chip Select*) – priključak za izbor čipa (ima istu funkciju kao priključak E u prethodnom primjeru),
- CS2 (*Chip Select*) – priključak za izbor čipa (ima istu funkciju kao priključak \bar{E} u prethodnom primjeru).



Sl. 2.10 Detalj priključenja ulazno-izlazne jedinice ACIA

Uvjet da bude čip ACIA izabran je sljedeći: *istodobno* moraju biti CS0 = 1 (na CS0 se dovodi adresna linija A3), CS1 = 1 (na CS1 se dovodi logičko I: A14 · VMA). Signal VMA (*Valid Memory Address*) upravljački je signal koji generira procesor i označava da je adresa koja se trenutno nalazi na adresnoj sabirnici *valjana*, i konačno, $\overline{CS2} = 0$ (na $\overline{CS2}$ se dovodi logičko II: A1 + A2 + A4 + A5 + A6 + A7 + A8 + A9 + A10 + A11 + A12 + A13 + A15).

Adresnom linijom A0 koja se dovodi na priključak RS izabire se jedan od dva registra u ACIA.

Na temelju navedenih uvjeta možemo odrediti adresni potprostor koji zauzima ACIA:

| A15 | A14 | A13 | A12 | A11 | A10 | A9 | A8 | A7 | A6 | A5 | A4 | A3 | A2 | A1 | A0 | Adresa |
|-----|-----|-----|-----|-----|-----|----|----|----|----|----|----|----|----|----|----|--------|
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 4008 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 4009. |

Dakle ACIA se javlja procesoru na dvjema sljednim adresama: 4008 i 4009 (heksadekadno), odnosno adresu ACIA možemo prikazati i kao:

| A15 | A14 | A13 | A12 | A11 | A10 | A9 | A8 | A7 | A6 | A5 | A4 | A3 | A2 | A1 | A0 |
|-----|-----|-----|-----|-----|-----|----|----|----|----|----|----|----|----|----|----|
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | X, |

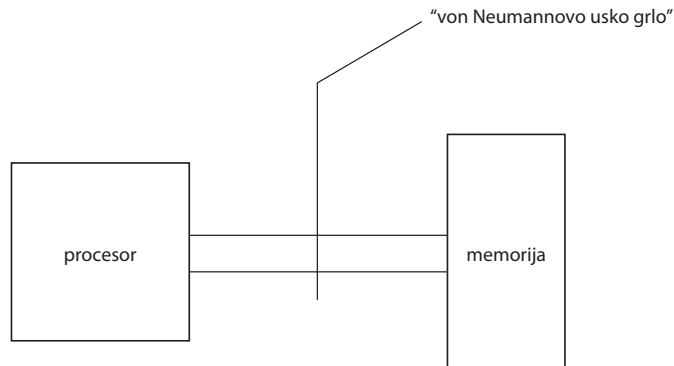
gdje X označava 0 ili 1.

2.4. VON NEUMANNOVO RAČUNALO – RAČUNALO SISD

Računala temeljena na von Neumannovom modelu razvrstavamo, u skladu s Flynnovom klasifikacijom, u SISD (*Single Instruction Stream Single Data Stream*) – arhitekturu računala s jednim instrukcijskim tokom i jednim tokom podataka. Zaista, ako pažljivo promotrimo sliku 2.2 vidimo da model ima samo jedan tok podataka i samo jedan instrukcijski tok. Pritom oba izvire iz memorijske jedinice, a “susreću” se u aritmetičko-logičkoj jedinici. Štoviše, fizička realizacija putova za prijenos podataka i instrukcija je takva da se isti put (sabirnica podataka) koristi za oba toka. To ima za posljednicu da *istodobno* na tom putu ne mogu postojati i jedan i drugi tok pa je obrada strogo slijedna (sekvencijalna). Nadalje, ako analiziramo aktivnosti tijekom instrukcijskog ciklusa (faze PRIBAVI i IZVRŠI), možemo utvrditi da je posebno intenzivan promet i instrukcija i podataka između procesora i memorijske jedinice: Tijekom faze PRIBAVI procesor obavezno jednom ili više puta uzastopce pristupa memorijskoj jedinici pribavljajući strojnu instrukciju. Tijekom faze IZVRŠI, ovisno o vrsti operacije koju treba izvesti, procesor također za većinu operacija pristupa memorijskoj jedinici da bi dohvatio operande ili pak pohranio rezultat.

Pojednostavljeno, model von Neumanovog računala mogao bi se predočiti slikom 2.11.

Vidimo da je na toj razini apstrakcije računalo prikazano samo procesorom, memorijskom jedinicom i spojnim putom između njih. Promet se podataka (u širem značenju te riječi – jer su i strojne instrukcije svedene na binarni kod) odvija spojnim putem, a brzina obrade očito ovisi o njegovoj propusnosti. Nažalost, propusnost spojnog puta takva je da se samo jedan podatak ili samo jedna strojna instrukcija mogu u vremenu prenijeti između procesora i memorijske jedinice. J. Backus, jedan od autora programskog jezika FORTRAN, u kritici von Neumannovog modela računala nazvao je taj spojni put “von Neumannovim uskim grlom”. Bacus tvrdi da mora postojati “... manje primitivan način...” ostvarivanja obrade podatka negoli je to “pumpanje” ogromne količine podataka naprijed-natrag kroz taj spojni put.



Sl. 2.11 Pojednostavnjeni model von Neumannovog računala

U knjizi ćemo se susresti s procesorima i računalima koji se temelje na von Neumannovim zamislama arhitekture, ali koji upotrebljavaju paralelizam na različitim razinama i predstavljaju otklon u odnosu na izvorni von Neumannov koncept.

Jedan od otklona je i harvardska arhitektura računala (nazvana po mjestu Harvard i zamisli koju je koristilo računalo Mark I) u kojoj se razlikuju memorijska jedinica za pohranu podataka i memorijska jedinica za pohranu instrukcija.

2.5. USPOREDBA TURINGOVOG STROJA I VON NEUMANNOVOG MODELA RAČUNALA

U prvom smo se poglavlju detaljnije upoznali s Turingovim strojem (TS), a u ovom smo poglavlju podrobnije opisali računalo temeljeno na von Neumannovom modelu. Usprkos tomu što je TS apstraktni izvršitelj i nije poslužio kao model u oblikovanju stvarnih računskih strojeva, zanimljivo ga je usporediti s von Neumannovim modelom računala.

Memorijska jedinica TS-a, nazvana *vanjska memorija*, s lijeva je i s desna neograničena vrpca podijeljena na polja u kojima se pohranjuje po jedan simbol iz vanjske abecede. Neograničenost vrpce tumačimo tako što po potrebi možemo vrpce dodavati proizvoljan broj polja s lijeve ili desne strane. Pristup se podacima pohranjenim na vrpici ostvaruje R/W glavom. Pristup podacima je sekvencijalan – vrijeme, odnosno broj taktova potreban za dohvat podatka ovisi o trenutnom položaju R/W glave i položaju polja u kojem se nalazi željeni podatak. R/W glava tijekom jednog takta može se pomaknuti samo za jedno polje (ulijevo ili udesno) u odnosu na njezin trenutni položaj. Poljima na vrpici nisu dodijeljene adrese, već se podacima pristupa na temelju položaja R/W glave. Stroj ima posebne naredbe za pomak glave (N, L, D).

Osim vanjske memorije, TS ima i dvije jednostavne *unutarnje memorije* za pohranu stanja stroja i naredbe za pomak R/W glave.

U vanjskoj memoriji pohranjuju se početni podaci, međurezultati i konačan rezultat. Program se *ne pohranjuje* u vanjskoj memoriji. S obzirom na to da je riječ o apstraktnom izvršitelju, nema ograničenja u obliku i načinu prikaza podataka.

Rad TS-a odvija se u *taktovima* kojima su definirani diskretni vremenski trenuci i u kojima TS mijenja svoje stanje. Rad stroja zamišljen je tako da se tijekom svakog takta dva pristupa vrpci: čita se simbol upisan u polju nad kojim se nalazi R/W glava, upisuje se (novi) simbol u promatrano polje.

Memorijska jedinica von Neumannovog modela računala je memorija ograničenog kapaciteta sastavljena od konačnog broja memorijskih lokacija. Svakoj je memorijskoj lokaciji *jednoznačno* pridružena adresa. Pristup podacima je izravan – temelji se na adresi memorijske lokacije pri čemu vrijeme pristupa ne ovisi o mjestu gdje je podatak pohranjen.

U memorijskoj se jedinici pohranjuju početni podaci, međurezultati, rezultati i programi u istom obliku – sljedova nula i jedinica.

Rad se računala odvija također u *taktovima* – diskretnim vremenskim trenucima koji su definirani signalom vremenskog vođenja. Tijekom instrukcijskog ciklusa procesor pristupa jednom ili više puta memorijskoj jedinici radi pribavljanja strojne instrukcije (faza PRIBAVI) te nijednom, jednom ili više puta tijekom faze IZVRŠI (broj pristupa memorijskoj jedinici ovisi o operaciji koja je određena operacijskim kodom strojne instrukcije).

Obrada informacije u TS-u odvija se u *logičkom bloku stroja L* u kojem je realizirana logička funkcija stroja δ : $S \times Q \rightarrow S \times P \times Q$.

Obrada podataka u von Neumannovom računalu odvija se u *aritmetičko-logičkoj jedinici*.

Program za TS, odnosno algoritam obrade predstavlja realizaciju logičke funkcije stroja δ koja je u obliku *funktionalne sheme Turingovog stroja* pohranjena u *logičkom bloku stroja L*.

Algoritam obrade u von Neumannovom računalu predložen je programom koji se sastoji od slijeda strojnih instrukcija. Program je pohranjen u memorijskoj jedinici u kojoj su pohranjeni i podaci.

Vanjska memorija TS-a ima ulogu i *ulazno-izlazne jedinice*, dok von Neumanovo računalo ima posebnu ulazno-izlaznu jedinicu. Postoji sličnost između vanjske memorije TS-a kao prikazne jedinice i memorijske jedinice IAS računala koja je realizirana elektronskim cijevima *Selectron*. U tom se slučaju memorijska jedinica računala može promatrati i kao izlazna grafička jedinica.