Osnove računalniške arhitekture

Arhitekrura (Prvič se je pojavila leta 1964, razvil jo je IBM za rač. IBM s/360 in se do danes ni spremenila) računalnika je zgradba, ki jo vidi programer, ki programira strojnem jeziku. STROJNI jezik je jezik, ki ga sestavljajo ukazi, ki jih računalnik lahko direktno izvaja – vanj so vgrajeni.

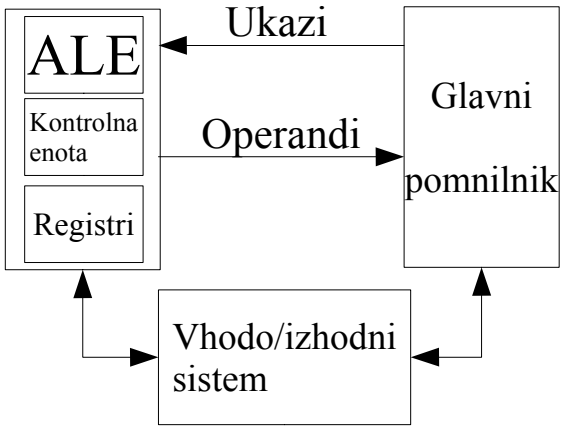
Kaj vidi programer v strojnem jeziku?

- zgradba in uporaba strojnih ukazov
- zgradba CPU – registri v CPU
- zgradba pomnilnika
- I/O sistem

Kaj računalnik dela

- iz pomnilika bere ukaze
- ukaze izvaja

Zgradba računalnika



Kontrolna enota vodi računalnik, shrani, prepoznava ukaze

V/I sistem pretvori ukaze iz strojne v primerno obliko za

Strojna oblika simbola 1 in 0
V eno pomnilniški celici je shranjen samo en bit (2 komb)
V dveh pomnilniških celicah štiri kombinacije

Štetje v dvojiškem sistemu

Osnova 2: RADIX
znaka: 0, 1

Šestnajstiški številski sistem

Osnova(RADIX) 16
Znaki, 0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F

Dvojiško v šestnajstiško

1 0011 1010 1011 0110 = $13AB6_{(H),\ (Hex)}$

Šestnajstiško v dvojiško

2A1F = 10101000011111

Zgodobina razvoja računalnikov

1. Obdobje mehanike 17 stol. C. Babbage (1792 – 1871)
   Njegov stroj (diferenčni stroj) je bil v groben podoben današnjim računalnikom. Prvič se je pojavila ideja o programu, ki narekuje stroju delovanje.
   Iz tega obdobja je izhajala prva programerka. Stroj ni bil nikoli dokončan, naredili so ga pa okrog 1940.
2. Obdobje elektromehanskih strojev
   Za pogon teh kalkutatorjev so se uporabljali električni motorji. Za njimi so se pojavili luknjaste kartice in relejev.
   Hollerith (1860 – 1929) je naredil stroj, ki je podatke sprejemal v obliki luknjastih kratic, jih tudi znal obdelat. Narejen je bil za štetje prebivalstva. Ustanovil je firmo Tabulating Machine Co., ki se je leta 1929 preimenovala v IBM.
   Harvard MARK I (H. Aikken 1900-1973) ZDA 1943
   Rač. Z3 (K. Zuse 1910 - 1969) NEMČIJA 1941
   Z23, leta 1962/63 – v Ljubljani

3. Obdobje elektronskih računalnikov
   Relejev preklopni čas je 1-10 ms
   Eletronka – preklopni čas $5*10^{-6}$s
   ENIAC (1945 ZDA) ~ 500×hitrejši kot MARK I dimenzije(30m × 1m × 3m) 18000 elektronk, 1500 relejev, poraba energije 140 kW
   EDVAC (1951) v tem računalniku je bil aprvič realizirana ideja, da računalnik dela po programu, ki je sestavljen v njegovem pomnilniku – Univerzalni računalnik - Pomnilnik 1024 × 16 bitov. Pri razvoju je sodeloval John von Neumann (1903 – 1957).
   1937 je angleški matematik Alan Turing objavil članek O izračunljivih številih ob uporabi pri odločitvenih problemih. TURINGOV stroj je tepretični model računalnika. Na njem se dal izračunati vse kar je izračunljivo.

Osnovni principi delovanja računalnika

VON NEUMANNOV računalniški model

- Teoretični del računanja
- Resničen stroj, ki deluje

VON NEUMANNOV stroj je vsak stroj, ki izpolnjuje naslednje pogoje:
1. Sestavljajo ga trije osnovi deli: CPU, glavni pomnilnik in I/O sistem
2. To je stroj, s shranjenim programom v glavnem pomnilniku in vodi delovanje stroja
3. Ukazi, ki sestavljajo program se izvajajo zaporedoma eden za drugim

Centralna procesna enota (CPE/CPU)

- Kontrolna enota
- Aritmetično-logična enota (ALE/ALU)
- Registri
  - Programsko dostopni (Vsebino lahko programsko preberemo, zapišemo, ...)
  - Programsko nedostopni (Te registre rabi CPU za svoje delovanje, programsko niso dostopni)

Glavni pomnilnik

Glasvni pomnilnik je sestavljen iz pomnilniških besed. Pomnilniška beseda ima svoj naslov in vsebino. Pomnilniško besedo sestvljajo posamezni BITI (enobitne pomnilniške celice).
Pomnilniška beseda – Tisto najmanjše število bitov, v pomnilniku, ki imajo svoj enoveljaven naslov

I/O sistem

Je namenjen prenosu informacij iz zunanjega sveta v računalnik in obratno. Pri tem pretvarja tudi med računalnikom in človekom razumljive podatke.

Delovanje VON NEUMANNOVEGA računalnika

Delovanje v popolnosti določajo ukazi, ki so shranjeni v glavnem pomnilniku. Računalnik ukaz prebere (fetch, prevzem), ga izvede (execute) in nadaljuje na naslednjem ukazu. Ukazi so shranjeni po naraščajočih pomnilniških naslovih. Računalnik mora ob RESET-u dobiti naslov prvega ukaza.

Osnovna naloga I/O naprav

Prenos informacij med pomnilnikom in I/O napravo. Dva načina prenosa:
1. Programski I/O (Programmed IO mode)
   Prenos poteka vedno preko CPU
   Prenos je vedno realiziran z nekim programom, ki ga izvaja CPU – od tod tudi ime
   Lahko uporablaj prekinitve ali ne (pooling)
   Slabosti: Počasen prenos, 100% zasedena procesna enota
   Prednosti: Cena, ker ne zahteva dodatnih naprav
   S prvim ukazom se podatek prenese iz I/O v CPU, z drugim ukazom pa v RAM.
2. Neposredni dostop do pomnilnika (DMA prenos)
   Prenos poteka brez sodelovanja CPU, vendar DMA krmilnik nadomesti delovanje CPE
   Slabosti: Višja cena (potrebna dodatna naprava)

Prednosti: Hiter prenos, CPU ni obremenjena, za prenos velikih količin podatkov
DMA naprava prebere iz naprave in direktno pošlje v glavni pomnilnik.

3. Prenos s pomočjo I/O procesorjev (CPU ne sodeluje pri prenosu)
   Prenos podoben kot DMA, samo, da lahko I/O CPU programiramo, DMA pa ne

Krmilniki I/O naprav

Vsaka naprava je preko krmilnika priključena na glavno vodilo. CPU-ju je vsak krmilnik viden kot določeno število podatkovnih registrov. V njih lahko CPU piše ali bere. Razlika glede na pomnilniške lokacije in registre je, da pisanje v registre sproži neko operacijo v napravi. Z branjem vsebine registra krmilnika ugotovimo stanje krmilnika oz. I/O naprave.
Naprave imajo najmanj 3 registre (lahko to tudi združeni v enega): podatkovni (za podatke), statusni (stanje naprave), kontrolni (krmiljenje naprave). Vsak od registrov mora imeti svoj naslov.

Naslovi registrov I/O naprav

1. Pomnilniško preslikan I/O
   Memory mapped I/O – Del naslovov iz pomnilniškega naslovnega prostora se uporabi za naslove I/O krmilnik
   En sam naslovni prostor, ki se ga delijo pomnilnik in I/O naprave – CPU ne vidi razlike med pomnilniško lokacijo in registrom I/O (CPU nima posebnih ukazov za dostop do I/O)
   Motorola uporablja tako rešitev za svoje procesorje.
2. Ločen I/O naslovni prostor
   Dve vrsti naslovov: Pomnilniški in I/O, dva ločena naslovna prostorta (Pomnilniški, I/O), pomnilniški naslovni prostor je neokrnjen
   Posebni (dodatni) ukazi za dostop I/O
   Intel uporablja tako rešitev za svoje procesorje.
3. Posredno preko I/O procesorjev
   Naslovi I/O registri so v posebnem naslovnem prostoru, ki ni viden iz CPU – ne more ga neposredno naslovit. Viden je samo iz I/O procesorjev.

Prenosne poti v računalniku

Prenosna pot: Povezava med CPU, glavnim pomnilnikom in I/O, po kateri se prenašajo informacije.
Želja je, da se prenese čim večje količine informacij v čim krajšem času.

1. Krajšamo čas enega prenosa – večanje frekvence urinega signala, s katerim je določen prenosni cikel
2. V enem prenosu prenesemo čim več informacij – večanje števila prenesenih bitov v enem prenosu – povečanje številaa povezovalnih linin.
3. Povečanje števila prenosnih poti
   Ena prenosna pot --> en prenos v določenem trenutku
   Več prenosnih poti --> več paralelno odvijajočih se prenosov hkrati

Dve vrsti povezovalnih poti

- Vodilo (BUS, OMNIBUS) – Ena sama prenosna pot, ki povezuje več naprav
- Povezava točka v točko (povezovalna pot, ki povezuje samo 2 napravi. Npr. CPU in glavni pomnilnik)

Temu rečemo povezovalna struktura. Boljša je, večja je hitrost.

Vrste signalov na vodilo

- Naslovni signali (določajo naslov lokacije, do katere hoče procesor dostopati)
- Podatkovni signali (prenšajo se podatki, število določa koliko bitol lahko istočasno prenesemo po vodilu)
- Kontrolni signali (določajo vrsto prenosa

Multipleksiranje

Prenos večih signalov po istih linijah
Deluje na principu Mater/slave. Master je naprava, ki vodi prenos, ga začne in tudi konča.
Arbitraža – Mehanizem, ki določa, kdo bo v danem trenutku MASTER vodila.

Zaporedje prenosov pri prenosu

1. Gospodar
   1. naslovni signali
   2. izbira naslovnega prostora
2. Gospodar
   1. Kontrolni signali (R/W)
   2. Širina prenosa
   3. Začetek in konec prenosa
3. Prenos podatkov

Glede na določanje začetka in konca prenosa ločimo:
- sinhronski prenos
- asinhrosnki prenos

Urin signal (Clock)

Periodičen električni signal, pravokotne oblike. Pozitivna, negativna fronta, prva in druga strmina.

Čas periode označijemo z T. Frekvenca je število period v eni sekundi. $f = \frac{1}{T}$

Pentium: $f = 1GHz \rightarrow T = 1ns$ Mc68HC11: $f = 8MHz \rightarrow T = \frac{1}{8 \cdot 10^6 \frac{1}{s}} = 125ns$

Sinhronski prenos

Prenos traja vedno celo število urinih period.
Čas prenosa je čas v naprej določen (na najpočasnejšo napravo) in vedno enak (npr: 3 cikle)
Začete in konec prenosa: Gospodar

Ahinhroni prenos

Čas prenosa ni v naprej določen. Prenos traja lahko poljubno dolgo. Gospodar začne prenos, suženj pa gospodarju sporoči, kdaj so podatki veljavni (potrditveni signal) in potem gospodar prenos konča. Če po določenem času ni potrditvenega signala, gospodar konča prenos in sporoči napako.

Časi pri prenosu

1. t_a Access time (dostopni čas)
   Čas, ki preteče od takrat, ko gospodar pošlje stabilne naslovne signale, do takrat ko naprava ta podatek sprejme.
2. t_s Setup time (vzpostavitveni čas)
   Čas, ki je potreben, da se podatek iz vodila prenese v registre ali pomnilnik.
3. t_h Hold time (držalni čas)
   Čas, koliko časa mora biti prisoten podatek še po koncu prenos. Pri večini naprav je ta čas enak 0.

Kapaciteta prenosne poti

Največje možno število prenesenih besed (bitov, bajtov) v eno sekundi

Oznaka za kapaciteto je B(andwidth). $B = \frac{količina\ informacije\ v\ enem\ prenosu}{čas\ enega\ prenosa}$

Primer: PCI vodilo
Širina vodila: 32 ali 64 bitov
Frekvenca vodila: 33MHz ali 66 MHz
En prenos traja eno urino periodo
B = ?

PCI 33bit / 66 MHz $f = 66MHz \Rightarrow T = 15.15\,ns$ $B = \frac{32\,bitov}{15{,}15\,ns} = 2{,}112 \cdot 10^9\,bitov/s = 264\,MB/s$

Čas potovanja elekričnega signala je 3,33 ns/m. Povprečni časi so 6 ns/m

Lokalnost pomnilniških dostopov

Prostorska lokalnost:

Zaporedni pomn ilniški naslovi, ki jih CPU pošilja v pomnilnik so blizu skupaj. - Način delovanja VON NEUMANNOVEGA računalnika, ker so ukazi na zaporednih naslovih

Časovna lokalnost:

V različnih časovnih obdobjih eni in isti naslovi večkrat ponovijo. - Zanke v programih

Glavni pomnilik – Pomnilniška hierarhija

Pomnilnik razdelimo na več samostojnih pomnilikov z različnimi lastnostmi.

CPE ⇔ Predpomnilnik ⇔ Glavni pomnilnik(RAM) ⇔ Navidezni pomnilnik(DISK)

Tronivojska pomnilniška hierarhija

	Dostopni čas	velikost
Predpomnilnik	ta~5ns	8Kb – 1Mb
Glavni pomnilnik	ta ~ 50ns	Gb
Navidezni pomnilik	ta ~ 10ms	nekaj 10 Gb

CPE vidi pomniliško hierarhijo kot glavni pomnilnik – z naslednjimi lastnostmi:

- Velikost navideznega pomnilika
- Hitrost blizu hitrosti predpomnilnika
- Cena: povprečje vseh cen

Amdahlov zakon

Case/Amdahlovi pravili (G. M. Amdahl IBM 360/370)

Če v računalniku za faktor N pohitrimo delovanje vseh operacij (enot), razen f operacij (enot), potem je celotno povečanje hitrosti delovanja računalnika

$$S(N) = \frac{1}{f + \frac{1-f}{N}} = \frac{1}{\frac{f*N+1-F}{N}} = \frac{N}{1+(N-1)*f}$$

1 – Vse operacije

(1-f) – Del operacij, ki jih pohitrimo N-krat

f – del operacij, ki ostanejo enako hitre

N=10 (10× pohitrimo) 90% enot računalnika

1 – f = 90% = 0.9

f = 10% = 0.1

$$S(N) = \frac{1}{0{,}1 + \frac{0{,}9}{10}} = \frac{1}{0{,}19} = 5{,}26$$

1. Case/Amdahlovo pravilo
   Velikost glavnega pomnilnika v bajtih mora biti najmanj enaka, številu (strojnih) ukazov, ki jih CPU izvede v eni sekundi.
2. Case/Amdahlovo pravilo
   Zmogljivost I/O sistema v bitih na sekundo mora biti najmanj enaka številu ukazov, ki jih CPU izvede v eno sekundi.

Računalnik, kot zaporedje navideznih računalnikov

Uporabniški pogled na zgradbo računalnikov

Nivo 5	Višji programski jeziki	Prevajanje ali interpretiranje
Nivo 4	Zbirni jezik	Prevajanje (Zbirnik)
Nivo 3	Operacijski sistem	Prevajanje (Zbirnik)
Nivo 2	Običajni strojni jezik	Delno interpretiranje
Nivo 1	Mikroproggramski jezik (Ni obvezen)	Interpretiranje
Nivo 0	Digitalna elektronika	Interpretiranje

Meja med fizičnim in programskim delom

Assembler: Zbirnik...

1. Zbirni jezik (Assembler)
2. Prevajalnik iz zbirnega v strojni jezik

Prevajanje:
Program v jeziku L2->program Prevajalnik->Prevedeni program v jeziku L1
Interpretiranje:
Program v jeziku L2->program Interpreter L2->L1

Predstavitev informacij v računalniku in aritmetika

Informacija se razdeli na ukaze in operande. Operandi so v osnovi lahko numerični (številčni) in nenumerični. Nenumerični operandi pa se delijo naprej na znake in logične spremenljivke. Numerični operandi se delijo na desetiška in dvojiška. Dvojiška števila se delijo na števila s fiksno vejico (ne-predznačena in predznačena) in števila s plavajočo vejico (enojna(32 bitov) in dvojna natančnost(64 bitov)).

Nenumerični operandi
Znaki (characters) ponavadi so predstavljeni v nizu znakov (string)
Znak (char) je predstavljen z neko abecedo
Abeceda: predpis (ali preslikava):
- do leta 1964 se je uporabljala BCD abeceda, ki je bila 6-bitna (2^6)
  26 črk angleške abecede (samo velike črke), 10 številk (0-9), 28 posebnih znakov
- 8-bitne abecede
  - EBCDIC (Extended Binary Coded Decimal Interchange Code); uporablja se v mainframe računalnikih
  - ASCII abeceda (American Standard Code for Information Interchange)
    V osnovi je 7-bitna (2^7=128->128 različnih znakov), razširjena ASCII abeceda je 8-bitna.
    b_7=0 osnovna ASCII abeceda
    b_7=1 2^8 – 256 kombinacij->nacionalna ASCII abeceda
- UNICODE 16-bitna abeceda
  2^{16} – 65.536 kombinacij BMP – Basic Multilinguall Plane (6.700 kombinacij neuporabljenih)
  Poznamo tri oblike:
  - UTF – 8 (ustreza ASCII abecedi)
  - UTF – 16
  - UTF – 32

Spodnji štirje biti v vsaki abecedi pri številih so enaki BCD zapisu. $3x = 0..9

Numerični operandi
Predstavitev števil v fiksni vejici (Fixed Point)
Desetiško število
$612,25=6*10^2+1*10^1+2*10^0+2*10^{-1}+5*10^{-2}$
osnova r=10
eksponent=položaj številke v številu

Zaporedje bitov:
$b_{n-1}\ b_{n-2}\ b_{n-3}$ ... $b_2\ b_1\ b_0\ b_{-1}\ b_{-2}\ b_{-3}\ b_{-m}$ = število, ki ga to zaporedje bitov predstavlja: $V(b)=\Sigma\ b_i*2^i$
b_i=0 ali 1 (n-1; i=-m)

$1101,01=1*2^3+1*2^2+0*2^1+1*2^0+0*2^{-1}+1*2^{-2}$
Če je vejica na skrajni desni konec->(desno od LSB(bita z najnižjo težo b_0)
Zapis s fiksno vejico Ξ na ta način so predstavljena cela števila Ξ integerji
Predznačena števila v fiksni vejici
- predstavitev s predznakom in velikostjo
- predstavitev z odmikom
- eniški komplement
- dvojiški komplement

Predstavitev s predznakom in velikostjo
MSB predstavlja predznak
b_{n-1}=0 pozitivno število
b_{n-1}=1 negativno število
Vrednost n-bitnega zaporedja:
$V(b)=(-1)^{b\ n-1}*\Sigma\ b_i*2^i$ (n-2; i=0)

$(-1)^0=1$
$(-1)^1=-1$

Primer n=3

0 1 1	+3	
0 1 0	+2	
0 0 1	+1	
0 0 0	+0	pozitivna števila
1 0 0	-0	negativna števila
1 0 1	-1	
1 1 0	-2	
1 1 1	-3	

Predstavitev z odmikom
K številu se prišteje konstanta, zato da je vrednost vedno pozitivna
Konstanta (BIAS) = odmik
n-bitno število->konstanta=2^{n-1} (lahko tudi 2^{n-1}-1)
n=3 (3-bitno zaporedje)
Odmik $2^{3-1}=4$

3(3+4)	1 1 1
2(2+4)	1 1 0
1(1+4)	1 0 1
0(0+4)	1 0 0
-1(-1+4)	0 1 1
-2(-2+4)	0 1 0
-3(-3+4)	0 0 1
-4(-4+4)	0 0 0

Eniški komplement
Najtežji bit Ξ predznak
Pozitivna števila – normalna predstavitev v dvojiški obliki
Negativna števila – predstavljena v komplementu ustreznega pozitivnega števila
COM Eniški komplement (vsak bit posebej komplementiramo)
Primer za n=3

0 1 1	+3		
0 1 0	+2		
0 0 1	+1		
0 0 0	+0	4 pozitivna števila	Dve ničli (Pozitivna in Negativna)
1 1 1	-0	4 pozitivna števila	
1 1 0	-1		
1 0 1	-2		
1 0 0	-3		

Dvojiški komplement
MSB – predznak (0+; 1-)
Pozitivna števila Ξ normalna dvojiška predstavitev
Negativna števila Ξ predstavljena v dvojiškem komplementu

Dvojiški komplement Ξ eniški+1
Primer za n=3

0 1 1	+3		
0 1 0	+2		
0 0 1	+1		
0 0 0	0	4 pozitivna števila	Ena sama ničla
1 1 1	-1	4 negativna števila	
1 1 0	-2		
1 0 1	-3		

V – overflow (previl) – Če prekoračimo obseg iz pozitivne na negativno stran

C – carry (prenos) – Če prekoračimo 8 bit

Predstavitev števil v plavajoči vejici

(floating point)
Za računalnik prirejena oblika, ki jo poznamo kot znanstvena notacija
$3{,}2 \times 10^6$ = Število × $\text{osnova}^{\text{eksponent}}$
$m \times r^e$ m – mantisa, r osnova (radix), e-eksponent

Dvojiški sistem: r=2
V računalniku hranimo samo mantiso in eksponent.
Mantisa predstavljena v obliki: Predznak in velikost (fiksna vejica)
Eksponent predstavljen v obliki: Predstavitev z odmikom (fiksna vejica)
Najprej je predznak, potem mantisa, in nato eksponent.

Če je mantisa večja in eksponent manjši je večja natančnost števil, ter manjši obseg, Če pa je mantisa manjša, pa eksponent večji pa ravno obratno.

Vejica je skrajno levo, tako da na levi strani mantice ni ničel... 0,1.... normalizirana mantisa. Prvi bit, ki je postavljen na 1 je normalni bit. Implicitna predstavitev normalnega bita je taka, da vejico postavimo za normalni bit, normalni bit pa si samo mislimo. Ničla pa je zapisana tako, da je mantisa 0, eksponent pa najbolj negativen.

Leta 1985 IEEE 754 je bil izdan standard za predstavitev števil v plavajoči vejici. Osnovne lastnosti so:
- osnova radix=2
- eksponent = predstavljen z odmikom
- mantisa = vrednost in predznak
- mantisa = normalizirana, normalni bit predstavljen implicitno
- dva formata: enojna (32 bitov) in dvojna (64 bitov) natančnost
  - enojna natančnost:
    Predznak (S,1 bit), eksponent (E, 8 bitov, odmik 127), mantisa (m, 23 bitov)
    Vrednost: $(-1)^s \times (1, m) \times 2^{(e-127)}$
  - dvojna natančnost
    Predznak (S,1 bit), eksponent (E, 11 bitov, odmik 1023), mantisa (m, 52 bitov)
    Vrednost: $(-1)^s \times (1, m) \times 2^{(e-1023)}$
- Standard definira tudi posebne vrednosti:
  +0 = Predznak: 0, eksponent: 00...00, mantisa: 00...00
  +∞ = Predznak: 0, eksponent: 11...11, mantisa: 00...00
  nan = Predznak: 0, eksponent: 11...11, mantisa: <> 0 Not a Number

5. Ukazi (strojni ukazi)

Strojni ukazi --> direktno izvajanje v kontrolni enoti CPU
Kontrolna enota je izdelana v fiksno ožičeni logiki. (danes RISC procesorji)
Strojni ukazi --> interpretiranje v enostavnejše mikroukaze
Kontrolna enota izvaja mikroukaze (enota je mikroprogramirana) (CISC procesorji)
Ukazi: Dve vrsti informacije:
- kaj narediti? - vrsta operacije, ki naj jo ukaz izvede.
- s čim? - operandi, nad katerimi naj se ukaz izvede.

Informacija o operaciji: OPERACIJSKA KODA
Informacija o operandih: INFORMACIJE O OPERANDIH:
- eksplicitno
- implicitno – info o operandih v operaijski kodi

Format ukaza: Ukaz je razdeljen na posamezna polja, ki vsebujejo informacije v ukazu. (Operacijska koda, ki vsebuje informacije o ukazu)
Ena operacija – več ukazov:
Operacija seštevanja:
- Ukaz za celoštevilčno seštevanje
- Ukaz za seštevanje v plavajoči vejici

- ukaz za seštevanje operandov različnih dolžin.

Operacijska koda:
Določeni število bitov uporabljenih za operacijsko kodo:
- Fiksna dolžina operacijske kode
  68HC11 Dolžina operacijske kode = 8 bitov
  Dolžina določa maksimalno število ukazov. => 2^8 = 256 ukazov
- Razširljiva operacijska koda
  Pogostejši ukazi imajo krajšo operacijsko kodo.

Osnovne lastnosti ukazov po katerih se medseboj razlikujejo procesorji:
1. Način shranjevanja operandov v CPU
   Programsko dostopni registri (majhen hiter pomnilnik v CPU
   +večja hitrost (Hitrejša tehnologija)
   +krajši ukazi (Krajši naslov)
   3 rešitve: akumulator, sklad, množica registrov
   Akumulator: En sam register v CPU:
   +V ukazu ni potreben naslov registra
   +preprosti ukazi, preprosti prevajalniki
   -veliko prenosov med CPE in glavnim pomnilnikom
   Sklad(stack): Pomnilniška struktura, ki deluje po pravilu last in, first out (Procesorjev, ki imajo sklad intergriranih vase ni veliko)
   Množica registrov v CPU: Vsak register ima enoveljaven naslov
   - Vsi registri so med seboj enakovredni
   - Registri so razdeljeni v skupine (podatkovni, naslovni)

   +Operande hranimo v registrih CPU dokler registerov ne zmanjka
   +Hitrejši dostop
   Primer: MC68HC11 – Procesor z Akumulatorjem
   P4 – Procesor z akumulatorjem + množica registrov (8 Splošnih 32bit, 6 segmentnih reg 32bit, 8 FPU registreo 80bit, 8 mmx registrov 64bit, 8 SSE in SSE2 registrov 128bit)
2. Število eksplicitnih operandov v ukazih
   Od števila operandov zavisi, kako kompliciran je računalnik.
   Danes: Največ trije eksplicitni operaandi v ukazu. Običajno so podani z naslovi, kjer so shranjeni
   Pri največ m eksplicitnih operandih v ukazu --> m-operandi ali m-naslovi Operandi so lahko shranjeni: V registrih v CPU, v glavem pomnilniku, v registrih I/O krmilnihov
   Delitev računalnikov na število eksplicitnih operandov ukazu:
   1. 3+1 Operandov rač (op koda, 3-je operandi, naslov naslednjega ukaza (ukazi niso bili na zaporednih lokacijah))
      OP3<--OP1+OP2
   2. 3-operandni (3-naslovni računalnikom)
      OP3<--Op1+OP2
   3. 2-operandi (1-naslovni računalniki
      OP2<--OP1 + OP2
   4. 1-operandni (enonaslovni računalnik)
      En sam register v CPE – akumulator
      AC<--AC+OP
   5. Brez-operandni računalniki
      Sklad(vrh)<--Sklad(vrh)+Sklad(vrh-1) (Najprej vzame dva podatka s sklada in na koncu enega porine na sklad)

3. Lokacija operandov in način njihovega naslavljanja

Kje so lahko shranjeni operandi?

1. V registrih v CPU (registerski operandi)
2. V glavnem pomnilniku (pomnilniški operandi)
3. V registrih I/O krmilnikov (Podobni so ostalima dvema)

Delitev računalnikov glede na lokacijo operandov, ki nastopajo v ukazih (Pri aritmetično-logičnih ukazih)

- Registerko – registerske (Vsi operandi so v registrih v CPE) (Load-store računalniki) Vsi ukazi, razen load in store, ki dostopata do pomnilnika, imajo operande v registrih (RISC računalniki).
  +Ukazi so kratni in enostavni, enostavna uporaba takih ukazov, krajši čas izvajanja in vedno enak
  -Za enako operacijo je potrebno več ukazov
- Registersko – pomnilniški (Nekaj operandov v pomnilniku (običajno eden), nekaj v registrih)
  +Ni potreben prenos operandov v register
  -Daljši ukazi, bolj zapleteni ukazi, čas izvajanja odvisen od dogajanja pri dostopu do pomnilnika
- Pomnilniško – pomnilniški (Vsak od operandov je lahko ali v pomnilniku ali v registru) CISC rač
  +Programiramo lahko brez uporabe registrov.
  -Ukazi so bolj zapleteni, čas izvajanja in dolžina ukazov je zelo različna

Informacija o tem, kje se operand nahaja – načini naslavljanja

3. osnovni načini naslavljanja:

1. Takojšnje naslavljanje(immediate addressing) (Operand je podan v ukazu)

V polju OPx je kar operand – takojšnji operand (konstanta)
LOAD R2, #27 R2 <-- 27
Uporaba: Vnos konstant v register, prištevanje konstant, ...
Število dostopov do pomnilnika, da pridemo do operanda: 1 dostopa

2. Neposredno naslavljanje(Direct addressing) (V ukazu neposreden ali posreden naslov ukaza)

V ukazu je neposredni ali registerski naslov: Ker se naslov ne da spreminajti včasih rečemo tudi absolutni naslov. Npr.: LOAD R1, $1010 (V register R1 naloži naslov podatek z naslova $1010)
Število dostopov do pomnilnika, da pridemo do operanda: 2 dostopa

3. Posredno naslavljanje(indirect addressing)

V ukazu je podan posredni naslov operanda.

- Pomnilniško posredno naslavljanje (Naslov pomnilniške lokacije, kjer je shranjen naslov operanda)
  LOAD R2, ($1234) ali @$1234 Na lokaciji $1234 se nahaja naslov kje je operand.
  Število dostopov do pomnilnika, da pridemo do operanda: 3 dostopa
- Registersko posredno naslavljanje
  V ukazu naslov nekega registra v CPU v katerem se nahaja začetni naslov pomnilniškega naslova
  V ukazu še odmik. Naslov pomnilniškega operanda = vsebina registra + odmik
  LOAD R2, 12, R0 (V register R2 se prenese se prenese vsebina iz pomnilnika na lokaciji R0 + odmik(12))
  Za dostop do različnih lokacij spremenimo samo register, odmika pa ne moremo spreminjati, ker je konstanta.
  1. Izvedbe registerskega posrednjega naslavljanja
  - Bazno naslavljanje Naslov OP = Rx + odmik (x= indeksni register (dolžina odmika v bitih = dolžina pomnilniškega naslova) / bazni register(dolžina odmika 8 ali 16 bit))
  - Indeksno naslavljanje Odmik pri baznem in indeksnim naslavljanjem je pozitivno število
  - Pre-dekrementno naslavljanje
  - Post-dekrementno naslavljanje
  - PC-relatnivno naslavljanje Odmik je število s predznakom v dvojiškem komplementu.

4. Vrste operacij, ki jih ukazi opravljajo

Razdelitev glede na vrsto operacije:

1. ALE operacije
   Operacije v fiksni vejici
   Dvooperandne operacije(+, -, *, /) in enooperandne operacije (+1, -1, ABS)
2. Prenosi podatkovnikontrolne operacije
   Prenašamo podatke iz izvora v ponor. (LOAD, STORE, PUSH, PULL, MOV)
3. Kontrolne operacije
   Spreminjajo vrstni red izvajanja ukazov (Brezpogojni in pogojni skoki, klic podprogramov) Te operacijo spreminjajo programski števec
4. Operacije v plavajoči vejici

Računajo se v posebni enoti.

5. Sistemske operacije
   To so priviligirane operacije (Omogočanje prekinitev, ...)
6. I/O operacije
   Operacije so podobne kot pri prenosu podatkov. Lahko so celo isti ukazi.

5. Vrsta in dolžina operandov
   Najpogostejše vrste operandov v strojnem jeziku
   1. BIT
   2. Znaki (character) vedno 8 bitov
   3. Cela števila (integer) fiksna vejica, dvojiški komplement (8, 16, 32 ali 64 bitov)
   4. Realna števila (real) plavajoča vejiva, enojna ali dvojna natančnost(32 ali 64 bitov IEEE 754)
   5. Desetiška števila (decimal) niz 8-bitnih znakov (ena števka predstavljena s 4 biti BCD) dolžina običajno 8 bitov (ena cifra), pakirana predstavitev 4 biti - ena cifra (P4 – 8 bit, pakirana predstavitev 2 cifri v 8 bit, 80 bitov - 20 pakiranih cifer)

   Sestavljeni pomnilniški operandi – sestavljeni so iz več pomnilniških besed. Zapisano so po dveh pravilih: Pravilo debelega in tankega konca. Poravnan pomnilniški operand: Kadar pomnilniški naslov na katerem se naslov nahaja (m) deljen s številom besed v operandu da ostanek 0. A(naslov) mod S(število besed)=0

   Pogostost izvajanja in trajanje posameznih vrst ukazov (operacij)
   Pogostost – Statična pogostost (Število posameznih vrst opercij v programu)
   * – Dinamična pogostost (Kolikokrat se posamezna vrsta operacije izvede)
   Tabela za dinamično pogostost izvajanja operacij.

i	***Vrsta operacij***	***Pi za program1***	***Pi za program2***	***CPI$_i$***	***Trajanje v ns***
1	LOAD / STORE	35%	20%	2	2,5
2	ALE	25%	30%	1	1,25
3	KONTROLNE	10%	10%	2	2,5
4	I/O operacije	30%	18%	3	3,75
5	Operacije v plavajoče vejice	0%	22%	5	6,25

Pi – procent časa izvajanja določene operacije (pogostost)

Čas izvajanja posameznih operacij

CPI – Clocks Per Instruction, Cycles Per Instruction (Trajanje izvajanja ukaza v urinih periodah)

Frekvenca urinega signala

f_{CPE} = 800 Mhz => t_{CPE}= $\frac{1}{800 * 10^6}[s] = 1.25[ns]$

Povprečno trajanje izvajanja enega ukaza v urinih periodah

$$CPI = \sum CPI_i * P_i$$

Primer za program 1 = CPI= 2*0.35 + 1*0.25 + 2*0.10 + 3*0.30

Koliko ukazov se v povprečju izvede v eni sekundi

Povprečno število ukazov v sekundi = $\frac{1}{CPI * t_{CPE}} = \frac{f_{CPE}}{CPI}$

Povprečno število ukazov se podaja v milionih na sekundo. = $\frac{F_{CPE}}{CPI * 10^6} = \frac{1}{CPI * t_{CPE} * 10^6}$ **MIPS**