Arhitektura osnovnog računara

Arhitektura osnovnog računara

- Osnovni računar PDP 8, slika 9.1
- Registri: PC, OPR, MAR, MBR, AC
- RAM 4096 x 16

Registri osnovnog računara

- Akumulator AC ima 16 bita
- Za komuniciranje sa memorijom postoje dva registra MBR od 16 bita i MAR od 12 bita
- Flip flop E je pridružen AC
- Flip flop I, bit režima za neposredno i posredno adresiranje
- OPR sadrži kod naredbe u užem smislu
- Kontrolni registar čine I, OPR, MAR
- PC programski brojač, 12 flip flopova

Memorija osnovnog računara

- RAM memorija se sastoji od 4096 riječi, svaka riječ ima 16 bita
- Memorija je podijeljena na dva dijela
  - Prvi dio je program
  - Drugi dio su podaci koji se obrađuju
- Učitavanje programa u memoriju - loadovanje

Oblik naredbe osnovnog računara

- Naredba se sastoji od ukupno 4 + 12 bitova
- Tri vrste naredbi, slika 9.2:
  - *Naredbe koje se odnose na memoriju*, bitovi 1-4 nijesu 111, prvi bit može biti 0 ili 1 što označava posredno i neposredno adresiranje, prestali bitovi sadrže memorijsku adresu, ukupno 7 naredbi
  - *Naredbe koje se odnose na registre*, počinju sa 0111, ostali bitovi su svi 0 osim jednog, ukupno 12 naredbi
  - *Ulazno – izlazne naredbe*, počinju sa 1111, ukupno 6 naredbi

Kontrolna jedinica

- Procesor se sastoji od kontrolne ili upravljačke i aritmetičko-logičke jedinice
- Kontrolna jedinica, slika 9.4, glavni zadatak je generisanje upravljačkih signala koji definišu mikro-operacije, pet vrsta ulaza, $t_0 – t_3$, $c_0 – c_3$, I, $q_0 – q_7$, $B_5 – B_{16}$
- Registar SC ima dva bita, generator taktova, ulaz u vremenski dekoder 2/4, izlazi su $t_0 – t_3$ i mijenjaju se po istom šablonu 1000, 0100, 0010, 0001

Kontrolna jedinica (2)

- Flip flop S ima ulogu dozvole za vremenski dekoder pa se vremenski signali generišu samo ako S = 1, ako je S = 0 izlaz iz dekodera je 0000, računar se zaustavlja
- Kada se jednom aktivira startni prekidač, računar postupa po istom šablonu: iz memorije se pročita naredba sa adrese koja je upisana u PC, naredba se upisuje u MBR, mode bit prepiše se u I, a kod naredbe u OPR, kod se dekodira u upravljačkoj jedinici, gleda se I za memorijske naredbe

Kontrolna jedinica (3)

- Riječ u MBR je jednog od sljedeća tri tipa: naredbe, adresa argumenta ili argument
- Kada se iz memorije čita naredba onda je računar u *fetch ciklusu*
- *Indirect* ciklus je čitanje adrese argumenta
- *Execute* ciklus obuhvata i čitanje argumenta
- *Interrupt* ciklus
- Zadatak kontrolne jedinice je smjenjivanje ciklusa

Ciklusi

Dva flip flopa za razlikovanje ciklusa F i R, dovode se na ulaz ciklusnog dekodera 2/4, izlazi dekodera su c_0 – c_3, jedan ciklus traje četiri takta

- F=0, R=0 → c_0 = 1, fetch ciklus
- F=0, R=1 → c_1 = 1, indirect ciklus
- F=1, R=0 → c_2 = 1, execute ciklus
- F=1, R=1 → c_3 = 1, interrupt ciklus

Smjenjivanje ciklusa, prekidni ciklus samo za ulazno-izlazne operacije, jedna naredba je najviše 4 ciklusa ti. 16 taktova

Ciklusi (2)

- Obično je redosljed ciklusa određen sa $c_0 = 1$, $c_1 = 1$, $c_2 = 1$, $c_3 = 1$
- Ako se ne koristi mogućnost prekida biće preskočeno $c_3 = 1$, odnosno poslije $c_2 = 1$ biće $c_0 = 1$
- Ako je adresa zadata neposredno (I = 0) biće preskočeno $c_1 = 1$, odnosno poslije $c_0 = 1$ biće $c_2 = 1$

Ciklusi (3)

- Prekidni ciklus koristi se za efikasno programiranje ulazno-izlazne komunikacije
- Primjer, radi se sa neposrednim adresama, bez mogućnosti prekida, tada
  - $c_0 = 1, c_1 = 0, c_2 = 0, c_3 = 0$, tokom 4 takta
  - $c_0 = 0, c_1 = 0, c_2 = 1, c_3 = 0$, tokom 4 takta
  - Dalje se periodično ponavlja
- Izvršavanje svake naredbe traje 4 ciklusa ili manje, odnosno 16 taktova ili manje

Kontrolni signali

- Filpflop I sprovodi se direktno na upravljačku kombinacionu mrežu, posredno ili neposredno adresiranje
- Registar OPR uvodi se na dekoder 3/8 (dekoder stanja) sa izlazima q_0,..., q_7, koji se dovode na upravljačku kombinacionu mrežu
  - Ako je q_0=1, ili …, q_6=1 radi se o naredbi I vrste
  - Ako je I=0 i q_7=1 radi se o naredbi II vrste
  - Ako je I=1 i q_7=1 radi se o naredbi III vrste

Kontrolni signali (2)

- U kombinacionu upravljačku mrežu uvode se signali $B_5,...,B_{16}$, pri čemu je B_i = MBR(i), koriste se za naredbe druge i treće vrste, te naredbe ne traže argument iz memorije pa se adresno polje koristi da se zada vrsta naredbe
- Imamo 12 naredbi druge vrste, samo jedan B_i je 1
- Slično je za naredbe treće vrste samo je njihov broj manji, pa nijesu iskorišćene sve mogućnosti

Prijemni ciklus

- Ciklus se sastoji od sljedećih mikro-operacija

  c_0t_0: MAR ← PC (prenošenje adrese naredbe)

  c_0t_1: MBR ← M, PC++ (naredba se donosi)

  c_0t_2: I ← MBR(1), OPR ← MBR(2-4)

  c_0t_3: if q_7' && I = 1 then R ← 1 else F ← 1

- Ciklus traje četiri takta, poznaje se po c_0, način izvršavanja je isti za sve naredbe, zadatak je da se iz memorije donese naredba i pripremi teren za naredni ciklus

Prijemni ciklus (2)

- Dok je $t_0 = 1$, PC se upisuje u MAR jer sadrži adresu naredbe koja je na redu za izvršavanje
- Dok je $t_1 = 1$, u MBR se donosi sadržaj lokacije M, čitanje memorije, istovremeno se PC uveća za 1 čime je pripremljen za izvršavanje sljedeće naredbe
- Dok je $t_2 = 1$, prvi bit se prenosi u I, bitovi 2-4 upisuju se u OPR, adresni dio 5-16 ostao je u MBR

Prijemni ciklus (3)

- Dok je $t_3 = 1$, definišu se radnje koje treba da uslijede, ako je $B_1B_2B_3B_4 = 0111$ (naredba druge vrste) ili $B_1B_2B_3B_4 = 1111$ (naredba treće vrste), onda treba preći u izvršni ciklus, ako je $B_2B_3B_4 <> 111$ i $B_1 = 0$ (naredba prve vrste sa direktnom adresom) opet treba preći u izvršni ciklus, ako je $B_2B_3B_4 <> 0111$ i $B_1 = 1$ (naredba prve vrste sa indirektnom adresom) treba preći u indirektni ciklus

Prijemni ciklus (4)

- Veličina B_1 vidi se u I, veličine $B_2B_3B_4$ vide se posredstvom OPR po q_0, …, q_7
- Prilikom ulaska u prijemni ciklus bilo je F=0, R=0, ako na završetku ciklusa bude u F=0, R=1 prelazi se u indirektni ciklus, ako bude F=1, R=0 prelazi se u izvršni ciklus

Idirektni ciklus

- Ciklus se sastoji od sljedećih mikro-operacija

  c_1t_0: MAR ← MBR(5-16)

  c_1t_1: MBR ← M[MAR]

  c_1t_2:

  c_1t_3: F ← 1, R ← 0

- Ciklus se poznaje po c_1, tokom ciklusa vrši se čitanje memorije sa lokacije upisane u MAR
- Uvijek se prelazi u izvršni ciklus

Izvršni ciklus

- Četiri mikro-operacije tokom ovog ciklusa zavise od vrste mašinske naredbe
- Primjer, naredba ADD, $B_2B_3B_4 = 001$

  $q_1c_2t_0$: MAR ← MBR(5-16)

  $q_1c_2t_1$: MBR ← M

  $q_1c_2t_2$: EAC ← AC + MBR

  $q_1c_2t_3$: F ← 0

- Ciklus se poznaje po c_2=1, ADD se poznaje po q_1=1, flipflop E za prenos

Izvršni ciklus (2)

- Može da bude I=0 i I=1, što nema značaja za odvijanje izvršnog ciklusa, jer je u oba slučaja obezbijeđeno da u trenutku kada $q_1 c_2 t_0$=1 dio 5-16 registra MBR sadrži adresu drugog sabirka
- Prelazak u novi prijemni ciklus ili prekidni ciklus

Prekidni ciklus

- Sistem prekida stavlja na raspolaganje mogućnost da program P, čije je izvršavanje u toku, može da prepusti upravljanje računarom drugom programu Q
- Pomoćni program Q služi da obavi neku kratku obradu (ulaz-izlaz), poslije ovoga obnavlja se rad programa P kao da nije bio ni prekidan
- Prekidi su asinhroni događaji

Naredbe koje se odnose na memoriju

- Naredbe prve vrste, tabele 9.1 i 9.2
- Kod operacije opisan je dijelom 2-4, memorijska riječ sa kojom se radi opisana je dijelom 5-16, prvi bit naredbe smješta se u I
- Naredbe: AND – 000, ADD – 001, LDA – 010, STA – 011, BUN – 100, BSA – 101, ISZ - 110

Naredbe koje se odnose na memoriju (2)

- Ako je I=0, onda se efektivna adresa m nalazi u MBR(5-16), M je sadržaj odgovarajuće memorijske lokacije
- Ako je I=1, onda je izvršnom prethodio indirektni ciklus, sada je m sadržaj memorijske lokacije koju definiše dio 5-16 same naredbe
- Primjer, AC=2000, M[63]=3000, M[3000]=14500
  - Naredba = 0001 63 (ADD), važi m=63, M=3000
  - Naredba = 1001 63 (ADD), m=3000, M=14500

Naredbe koje se odnose na memoriju (3)

LDA, STA za komunikaciju između AC i memorije, fizičku adresu promjenljive određuje prevodilac

- BUN, u registar PC dovodi se MBR(5-16)
- BSA, poziv potprograma, bezuslovni skok, zatečena vrijednost PS je povratna adresa
- ISZ, odgovara IF naredbi u višim jezicima

Naredbe koje se odnose na registre

- Tabele 9.3 i 9.4
- Naredbe se izvršavaju tokom t_3, za jedan takt
- Cijeli brojevi zapisuju se u potpunom komplementu, znak zavisi od prvog bita slijeva
- Naredbe: CLA, CLE, CMA, CME, CIR, CIL, INC, SPA, SNA, SZA, SZE, HLT

Ulazno-izlazna jedinica

- Osnovni računar ima jednu jedinicu koja je istovremeno ulazno-izlazni uređaj, teleprinter sa tastaturom i štampačem, serijski prijem i slanje podataka, ulazni registar INPR, izlazni registar OUTR, komunikacija sa akumulatorom je paralelna, slika 9.5
- INPR sastoji se od 8 bita, FGI je flipflop koji je setovan u trenutku kada je nova informacija raspoloživa, resetuje se kada računar prihvati informaciju, usklađivanje brzina ulaznog uređaja i računara

Ulazno-izlazna jedinica (2)

- Na početku rada je FGI=0, kada se pritisne taster u INPR upisuje se kod dužine 8 bita i FGI=1, dok je FGI=1 pritisak ma kojeg tastera ne može da promijeni registar INPR, računar prenosi paraleno sadržaj INPR u AC i postavlja FGI=0
- Izlazni registar OUTR funkcioniše na sličan način samo je smjer preokrenut
- Programski upravljan način ulazno-izlazne komunikacije

Ulazno-izlazna jedinica (2)

- Na početku rada je FGI=0, kada se pritisne taster u INPR upisuje se kod dužine 8 bita i FGI=1, dok je FGI=1 pritisak ma kojeg tastera ne može da promijeni registar INPR, računar prenosi paraleno sadržaj INPR u AC i postavlja FGI=0
- Izlazni registar OUTR funkcioniše na sličan način samo je smjer preokrenut
- Programski upravljan način ulazno-izlazne komunikacije

Programski upravljan ulaz-izlaz

- Program upravlja prenosom i prenos se izvodi samo u slučaju pripravnosti spoljašnjeg uređaja
- Realizacija pomoću petlje:
  - Ispitivanje FGI
  - Započinje prenos ako je FGI=1, inače skok na prethodnu naredbu
- Neefikasno, za štampanje jednog karaktera potreban veliki broj čitanja i provjere vrijednosti FGO

Programski upravljan ulaz-izlaz (2)

- Alternativa je moguća ako postoji način da procesor naznači da treba odštampati neki karakter koji je smješten na određenom mjestu, a da nakon toga uređaj sam štampa, kada se završi štampa uređaj obavještava procesor, koji je u međuvremenu izvršavao neke druge naredbe
- Potreba je
  - hardverska dogradnja – uvođenje prekidnog ciklusa i flipflopa IEN
  - Softverska dogradnja – uvođenje prekidne rutine

Programski upravljan ulaz-izlaz (3)

- Flipflop IEN je flag koji označava da li je dozvoljen prekid (interrupt enable)
  - Računar postavlja IEN=0 kada “ne želi” da bude prekidan
  - Kada je IEN=1, i FGI ili FGO setovan, doći će do prekida, izvršavanje programa se prekida i započinje izvršavanje prekidne rutine, poslije prekidne rutine nastavlja se izvršavanje prekinutog programa od mjesta na kom je bio prekinut
  - Adrese 0 i 1 imaju specijalnu namjenu, 0 za čuvanje povratne adrese, 1 za skok na prvu naredbu prekidne rutine, slika 9.6

Ulazno-izlazne naredbe

- Tabela 9.5
- INP i OUT odnose se na nižih osam bitova AC, dvije uzastopne ulazne informacije čine jednu riječ
- SKI provjerava vrijednost FGI
  - ako je FGI=1 onda se jedna naredba preskoči i, po pravilu, izvrši naredba INP
  - ako je FGI=0 onda se izvršava sljedeća naredba (preskočena u prethodnom slučaju), a to je po pravilu, skok na ponovno provjeravanje FGI
  - Slično za SKO

Prekidni ciklus

- U prekidni ciklus ulazi se na kraju izvršavanja neke naredbe, tj. kada je c_2=1 i t_3=1, u izvršnom ciklusu odlučuje se koji je sljedeći

  c_2t_3: if IEN&&(FGI || FGO) then R ← 1 else F ← 0

- Prekidni ciklus

  c_3t_0: MBR(5-16) ← PC, PC ← 0

  c_3t_1: MAR ← PC, PC ← PC + 1

  c_3t_2: M ← MBR, IEN ← 0

  c_3t_3: F ← 0, R ← 0

Prekidni ciklus (2)

- Opis
  - t_0 sadržaj PC prenosi u MBR, tj. taj sadržaj će biti poslat u memoriju, PC se anulira
  - t_1 anulira se MAR, PC se postavlja na 1
  - t_2 piše se u memoriju, onemogućavaju se prekidi
  - t_3 izaziva se prelazak u prijemni ciklus
  - Rezime: zatečena vrijednost PC upisana je na adresu 0, PC je postavljen na 1, prethodno se na 1 smješta komanda za bezuslovni skok na prekidnu rutinu

Sistem prekida osnovnog računara

- Program koji koristi ulaz/izlaz može se pisati na nekoliko načina:
  - ne koristi mogućnost prekida
  - djelimično koristi prekide
  - kompletan ulaz/izlaz sa prekidima naročito ako je komunikacija obimna, velika iskorišćenost ulazno/izlaznih uređaja

Prekidna rutina

- Tipične faze prekidne rutine
  - zatečene vrijednosti AC i E sačuvati negdje u memoriju
  - izvršava se prekidna rutina u užem smislu
  - sačuvane vrijednosti AC i E vratiti u registre
  - uključiti prekide
  - izvršiti bezuslovni skok na naredbu sa adrese 0

Prekidna rutina (2)

- Precizno definisana namjena adresa 0 i 1
- Precizno definisati djelove memorije za korisničke programe i za prekidne rutine
- Hardverski prekidi generisani u hardveru
  - sa perifernog uređaja, pokušaj izvršavanja nedozvoljene radnje
- Softverske prekide generiše program koji se izvršava
- Prioriteti prekida, kontroler prekida (hardver koji upućuje procesoru broj prekida koji je potrebno obraditi)

Osnovni računar naspram PDP-8

- Upravljačka tabla računara
  - dugmad CLEAR, START, STOP, lampa RUN, zvonce BELL
- Podrška za ručno loadovanje, SWITCH registar sa tri dugmeta LOAD PC, LOAD MAR, DEPOSIT
- Ulazno-izlazna jedinica sa prekidačima FGI=0, FGO=0
- Izmjene u hardveru
- Tehnički parametri