Vježbe 9-12: Mikroprocesor MC68000

Mikroprocesor MC68000 nalazi se uvijek u jednom od tri stanja obrade: normalnom stanju, stanju iznimke ili stanju "halt". U normalnom stanju procesor pribavlja naredbe iz memorije, izvodi ih i sprema rezultate u memoriju ili u registre. Poseban slučaj je stanje 'stop', u koje se dospijeva pozivom naredbe STOP, što ima za posljedicu prestanak pribavljanja i izvršavanja naredbi. Iz stanja 'stop' izlazi se vanjskim reset-om ili prekidom.

Stanje iznimke je način na koji MC68000 odgovara na devijacije od normalnog izvršavanja naredbi. Te se devijacije, odnosno iznimke, mogu podijeliti na one uzrokovane izvana i one uzrokovane unutarnjim stanjem procesora.

izvana izazvane iznimke
- reset
- sabirnička pogreška
- prekid

iznimke izazvane stanjem procesora
- ilegalna adresa
- praćenje (engl. trace)
- ilegalna naredba
- neugrađena naredba
- privilegirana naredba
- naredba TRAP, TRAPV, CHK
- dijeljenje s nulom

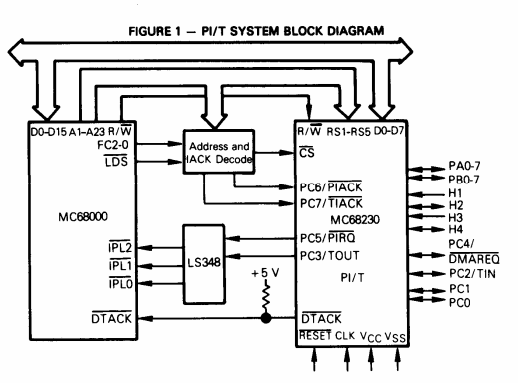
U stanje "halt" procesor ulazi ako se dogodi katastrofalna sklopovska pogreška (npr. dvije uzastopne sabirničke pogreške). Iz stanja "halt" izlazi se samo vanjskim reset-om.

Procesor MC68000 može raditi u korisničkom načinu (*engl. user state*) ili nadglednom načinu (*engl. supervisor state*). Za razliku od nadglednog, u korisničkom načinu nije dozvoljena upotreba cijelog skupa naredbi (nisu dozvoljene naredbe koje mogu mijenjati sistemski dio označnog registra, naredba STOP itd.). Zastavica S u označnom registru određuje način rada: S=1 označava nadgledni, a S=0 korisnički način.

Detaljniji opis procesora MC68000 može se pronaći u S. Ribarić: "Arhitektura mikroprocesora", Tehnička knjiga Zagreb, četvrto dopunjeno izdanje. Obradu iznimki za MC68000 potrebno je proučiti iz S. Ribarić: "Naprednije arhitekture mikroprocesora", Školska knjiga Zagreb, str.149-160.

MC68230 Parallel Interface/Timer kao vremenski sklop

Programirljivi sklop MC68230 sastoji se od dvije logički neovisne cjeline: paralelnog komunikacijskog međusklopa i vremenskog sklopa. Vremenski sklop sadrži 24-bitovno brojilo koje može koristiti vanjsko generirani ili sistemski signal vremenskog vođenja. Može se koristiti za generiranje periodičnih prekida, pravokutnih impulsa, prekida nakon programiranog vremenskog perioda itd. U/I vrata mogu se konfigurirati kao jednosmjerna ili dvosmjerna, 8 ili 16 bitova širine. Mogu koristiti vektorske ili autovektorske prekide, kao i obavljati DMA prijenos. Na slici 1 prikazan je načelan način spajanja MC68230 na MC68000. U okviru vježbi sklop će biti korišten kao vremenski sklop, pa je ovdje opisan upravo taj način rada.



Slika 1. Povezivanje sklopa MC68230 s procesorom MC68000.

U sljedećoj tablici opisane su uloge signalnih linija sklopa MC68230:

D0 – D7	Dvosmjerna sabirnica podataka. Služi za prijenos podataka od/prema procesoru MC 68000 ili nekog drugog vodećeg modula.
RS1-RS5	(Register select) Ovim se ulazima bira registar kome vodeći modul (MC 68000) želi pristupiti.
R/W*	(Read/Write Input) Određuje smjer prijenosa podataka. Signal postavlja vodeći modul (MC 68000) – visoka razina određuje ciklus pisanja, a niska ciklus čitanja
CS*	(Chip Select Input) Omogućava se pristup registrima sklopa. Signal generira odgovarajući adresni dekoder.
DTACK*	(Data Transfer Acknolwedge Output) Izlaz koji signalizira završetak sabirničkog ciklusa. DTACK* postavlja MC68230 nakon što podatke postavi ili pročita s podatkovne sabirnicu. Kompatibilan je sa MC68000, kao i sa drugim vodećim modulima Motoroline serije (npr. MC68450 DMA upravljački sklop. Linija se mora pritegnuti preme napajanju da bi se osigurala visoka razina signala između sabirničkih ciklusa.
RESET*	Ulaz koji se koristi za inicijalizaciju PI/T funkcija. Svi upravljački i podatkovni registri se brišu i većina internih operacija je onemogućena.
CLK	Ulaz signala vremenskog vođenja (može biti neovisan o taktu MC68000).
PA0 – PA7 PB0 – PB7	Port A i port B. Koriste se u komunikacijskom radu sklopa.
H1 - H4	(Handshake Pins) Pinovi za rukovanje (koriste se u komunikacijskom radu sklopa)
Port C PC0-PC7/ linije posebne namjene: TIACK*, PIACK*, PIRQ*, DMAREQ*, TOUT, TIN*,	Višenamjenske linije. Mogu se koristiti kao 8 ulazno/izlaznih linija opće namjene ili kao bilo koja kombinacija od 6 linija posebne namjene i 2 linije opće namjene (PC0-PC1). Linije TIN, TOUT i TIACK* su vezane za rad vremenskog sklopa. TIN može biti korišten kao ulaz vanjskog takta koji okida na rastući brid ili kao upravljački ulaz za pokretanje/zaustavljanje vremenskog sklopa (vremenski sklop radi ako je ovaj pin u visokoj razini) TOUT može poslužiti kao (aktivan nisko) izlaz preko kojega se generira zahtjev za prekid ili kao izlaz na kojem se generira pravokutni signal. TIACK* je ulaz koji se koristi kao potvrda zahtjeva za prekid.

Registarski model

U sljedećoj tablici prikazani su registri sklopa MC 68230 koji se koriste kada sklop radi kao vremenski sklop. U lijevom stupcu tablice navedena je kombinacija na adresnim ulazima RS5-RS1 kojom se bira pojedini registar. Prilikom određivanja stvarne adrese na kojoj se javljaju pojedini registri obratiti pažnju na shemu spajanja !

Bitovi koji se ne koriste označeni su kao "*". Prilikom čitanja podatka te bitovne pozicije poprimaju vrijednost "0". Pokušaj pisanja na te bitovne pozicije ne mijenja njihovo stanje. Svi registri adresiraju se kao 8-bitne lokacije, pa ime se može pristupiti MOVE.B naredbom. Međutim, uređenje bajtova je takvo da se višebajtnim registrima (npr. CPR) može pristupiti i odjednom, korištenjem naredbe MOVEP.

Bitovi za izbor registra RS5-RS1	Značenja bitova u registrima 7	6	5	4	3	2	1	0	Registar
0 0 0 0 0									ne koriste se za vremenski sklop
. . .									
0 1 1 1 1									
1 0 0 0 0									Timer Control Register (TCR)
1 0 0 0 1									Timer Interrupt Vector Register (TIVR)
1 0 0 1 0	*	*	*	*	*	*	*	*	(null)
1 0 0 1 1									Counter Preload Register (CPR) – High (najznačajniji bajt)
1 0 1 0 0									CPR - Mid
1 0 1 0 1									CPR – Low (najmanje značajan bajt)
1 0 1 1 0	*	*	*	*	*	*	*	*	(null)
1 0 1 1 1									Count Register (CR) - High
1 1 0 0 0									CR – Mid
1 1 0 0 1									CR – Low
1 1 0 1 0	*	*	*	*	*	*	*	ZDS	Timer Status Register (TSR)
1 1 0 1 1	*	*	*	*	*	*	*	*	(null)
1 1 1 0 0	*	*	*	*	*	*	*	*	(null)
1 1 1 0 1	*	*	*	*	*	*	*	*	(null)
1 1 1 1 0	*	*	*	*	*	*	*	*	(null)
1 1 1 1 1	*	*	*	*	*	*	*	*	(null)

Opis rada vremenskog sklopa

MC68230 kao vremenski sklop može biti upotrijebljen u različite svrhe:

- periodičko generiranje zahtjeva za prekid;
- generiranje pravokutnog signala;
- generiranje zahtjeva za prekid nakon definiranog vremenskog perioda;
- mjerenje proteklog vremena.

PI/T vremenski sklop ima 24-bitno sinkrono brojilo prema dolje koje se puni preko tri 8-bitna registra za punjenje brojila (*Counter Preload Registers* – CPR). To brojilo može biti upravljano preko izlaza 5-bitnog (dijeli sa 32) djelila frekvencije (*prescaler*), pri čemu se kao ulazni takt koristi sistemski takt (ulaz CLK) ili vanjski takt doveden na ulaz TIN**.** Okidanje brojila može biti i izravno (bez dijeljenja frekvencije), vanjskim taktom dovedenim na ulaz TIN.

Osnovno djelovanje vremenskog temelji se na signaliziranju stanja 0 u brojilu. Kada brojilo poprimi stanje 0, postavlja se zastavica detekcije nule (*Zero Detect Status* – ZDS) u status-registru vremenskog sklopa (*Timer Status Register* – TSR). To se stanje može programski provjeriti (od strane mikroprocesora), ili se vremenski sklop može konfigurirati da generira prekid.

Ponašanje vremenskog sklopa može se oblikovati upisom podatka u 8-bitni upravljački registar (*Timer Control Register* – TCR). U njemu se određuje:

- uloga višenamjenskih linija (C-port ili linije TIN, TOUT i TIACK*);
- da li se brojilo nakon detekcije stanja 0 ponovo inicijalizira iz registra za punjenje brojila (CPR) ili nastavlja brojanje s vrijednošću $FFFFFF_{16}$;
- koji se ulaz koristi za okidanje brojila;
- da li se koristi dijelilo frekvencije (*prescaler*);
- da li je vremenski sklop omogućen (pokreće se ili zaustavlja brojilo).

Ponašanje vremenskog sklopa razlikuje se u ovisnosti da li je pokrenut (u stanju *run*) ili zaustavljen (stanje *halt*). Kada je brojilo zaustavljeno, vrijedi:

- Prethodni sadržaj brojila nije promijenjen i pouzdano se može očitati iz brojačkih registara (Counter Registers);
- Djelilo frekvencije se postavlja na $1F neovisno da li se koristi;
- ZDS statusni bit se briše, neovisno o stanju brojila.

U stanju *run* vrijedi:

- Brojilo je okidano na način koji određuje sadržaj upravljačkog registra (TCR);
- Stanje brojila nije pouzdano čitljivo;
- Djelilo frekvencije se svakim taktom umanjuje za 1 ako se koristi;
- ZDS status bit se postavlja kada 24-bitno brojilo prelazi iz 000001_{16} u 000000_{16}.

Prilikom programiranja vremenskog sklopa potrebno je voditi računa o sljedećim pravilima:

- Pravila vezana za stanja *run* i *halt* (vidjeti iznad).
- Kada je se sklop resetira (aktivan signal RESET), svi bitovi upravljačkog registra (TCR) se brišu i time konfiguriraju dualnu funkciju višenamjenskih linija kao ulaze C-porta.
- Sadržaj registra za punjenje brojila (CPR) i samog brojila ne mijenja se resetiranjem sklopa.
- Registri brojila osiguravaju direktno čitanje podataka sa svakog dijela 24-bitnog brojila, ali podaci koji se pokušaju zapisati na njihovu adresu biti će ignorirani. Ovi registri se mogu čitati u bilo kojem trenutku, ali njihovo očitanje u stanju *run* nije pouzdano.
- CPR se može čitati i upisivati u svakom trenutku, neovisno o načinu rada sklopa.
- Ulazna frekvencija 24-bitnog brojila dobivena izravno preko linije TIN ili iz izlaza djelila (*prescaler*-a) mora biti između 0 i ulazne frekvencije na liniji CLK podijeljene sa 32, bez obzira na izabranu konfiguraciju.

- Za konfiguracije u kojima se koristi djelilo (kao ulazni takt može biti korišten signal s linije CLK ili linije TIN) sadržaj CPR se prenosi u brojilo prvi put kad djelilo prelazi iz $00 u $1F nakon što se prijeđe u stanje *run*. Kasnije se svakim prolaskom djelila kroz stanje 0 brojilo umanjuje za 1 (ili se ponovno puni sadržajem CPR).
- Ako se djelilo frekvencije ne koristi, sadržaj CPR-a se prenosi u brojilo na prvi brid ulaznog signala TIN nakon postavljanja u stanje *run*. Na svaki sljedeći brid brojilo se dekrementira ili se puni iz CPR-a.
- Najniža vrijednost dopuštena u CPR-u za korištenje sa brojilom je $000001.

Upravljački registar vremenskog sklopa (TCR)

7	6	5	4	3	2	1	0
TOUT/TIACK* Control			0-Detect Control	*	Clock Control		Timer Enable

Ovaj registar određuje način rada vremenskog sklopa.

- Bitovima 7-5 bira se funkcija linija PC3/TOUT i PC7/TIACK*.
- Bit 4 određuje da li brojilo, kada dođe do 0, ponovno inicijalizira iz CPR-a ili nastavlja brojati s $FFFFFF_{16}$.
- Bit 3 se ne koristi
- Bitovi 2-1 određuju putanju od CLK i TIN ulaza do kontrolera brojila.
- Bit 0 omogućava vremenski sklop

Ovaj registar može se čitati ili upisivati u svakom trenutku. Resetiranjem sklopa (signalom RESET*) svi bitovi ovog registra se postavljaju u 0.

TCR 7 6 5	FUNKCIJA: Upravljanje prekidnim linijama
0 0 X	PC3 i PC7 imaju ulogu C-porta
0 X 1	PC3/TOUT koristi se kao TOUT. Na tom se izlazu prilikom rada vremenskog sklopa generira pravokutni signal, s time da se stanje na izlazu mijenja svakim prolaskom brojila kroz stanje 0. Kada je brojilo zaustavljeno, ovaj izlaz je u visokom stanju. Linija PC7/TIACK* koristi se kao PC7.
1 0 0	PC3/TOUT koristi se kao TOUT, i to kao izlaz za zahtijevanje prekida. Međutim, u ovom načinu rada *zahtijevanje prekida je onemogućeno*, pa je ova linija uvijek u stanju visoke impedancije. Linija PC7/TIACK* koristi se kao TIACK*, tj. za primanje potvrde prekida. Međutim, kako je zahtijevanje prekida onemogućeno, sklop ne odgovara u slučaju aktiviranja ove linije.
1 0 1	PC3/TOUT koristi se kao TOUT, i to kao izlaz za zahtijevanje prekida. Zahtijevanje prekida je *omogućeno*. Ova linija prelazi u nisku razinu (postaje aktivna) kada bit ZDS u statusnom registru (TSR) poprimi vrijednost 1, tj. brojilo je u stanju 0. Linija PC7/TIACK* koristi se kao TIACK*, tj. za primanje potvrde prekida. U ovom načinu rada podržan je *vektorski prekid*, tj. u ciklusu potvrde prekida vremenski sklop na sabirnicu podataka postavlja *vektorski broj*.
1 1 0	PC3/TOUT koristi se kao TOUT, i to kao izlaz za zahtijevanje prekida. Međutim, u ovom načinu rada *zahtijevanje prekida je onemogućeno*, pa je ova linija uvijek u stanju visoke impedancije. Linija PC7/TIACK* koristi se kao PC7.
1 1 1	PC3/TOUT koristi se kao TOUT, i to kao izlaz za zahtijevanje prekida. Zahtijevanje prekida je *omogućeno*. Ova linija prelazi u nisku razinu (postaje aktivna) kada bit ZDS u statusnom registru (TSR) poprimi vrijednost 1, tj. brojilo je u stanju 0. Linija PC7/TIACK* koristi se kao PC7. U ovom načinu rada podržan je *autovektorski prekid*.

TCR 4	FUNKCIJA: upravljanje detekcijom nule
0	Kada brojilo stigne do 0, puni se iz CPR-a s prvim sljedećim taktom, te nastavlja brojati.
1	Nakon detekcije nule, brojilo nastavlja kontinuirano brojati.

TCR bit 3 se ne koristi i uvijek je 0.

TCR 2 1	FUNKCIJA: upravljanje taktom
0 0	PC2/TIN ulazna linija obavlja C-port funkciju. Za okidanje brojila koristi se signal s ulaza CLK i djelilo frekvencije (*prescaler*). Djelilo frekvencije se umanjuje za 1 svakim padajućim bridom signala CLK. 24-bitno brojilo se dekrementira ili se puni iz CPR-a kada djelilo prelazi iz stanja $00 u stanje $1F. Bit 0 (*Timer Enable*) određuje da li se sklop nalazi u stanju *halt* ili *run*.
0 1	PC2/TIN služi kao ulaz TIN, a za okidanje brojila koriste se CLK i djelilo frekvencije. Djelilo se dekrementira na padajući brid CLK. 24-bitno brojilo se dekrementira ili se puni iz CPR-a kada djelilo prelazi sa $00 na $1F. Sklop je u stanju *run* kada je bit 0 (*Timer Enable*) u 1, a ulaz TIN je u visokoj razini. U suprotnom, sklop je u stanju *halt*.
1 0	PC2/TIN služi kao ulaz TIN kojim se okida brojilo, pri čemu se koristi djelilo frekvencije (*prescaler*). Djelilo se dekrementira na rastući brid TIN-a, nakon sinkroniziranja sa internim taktom. 24-bitno brojilo se dekrementira ili puni iz CPR-a kada djelilo prelazi sa $00 na $1F. Bit 0 (*Timer Enable*) određuje da li se sklop nalazi u stanju *halt* ili *run*.
1 1	PC2/TIN služi kao ulaz TIN kojim se okida brojilo, a djelilo frekvencije (*prescaler*) se ne koristi. 24-bitno brojilo se dekrementira ili puni iz CPR-a na rastući brid TIN-a, nakon sinkroniziranja sa internim taktom.. Bit 0 (*Timer Enable*) određuje da li se sklop nalazi u stanju *halt* ili *run*.

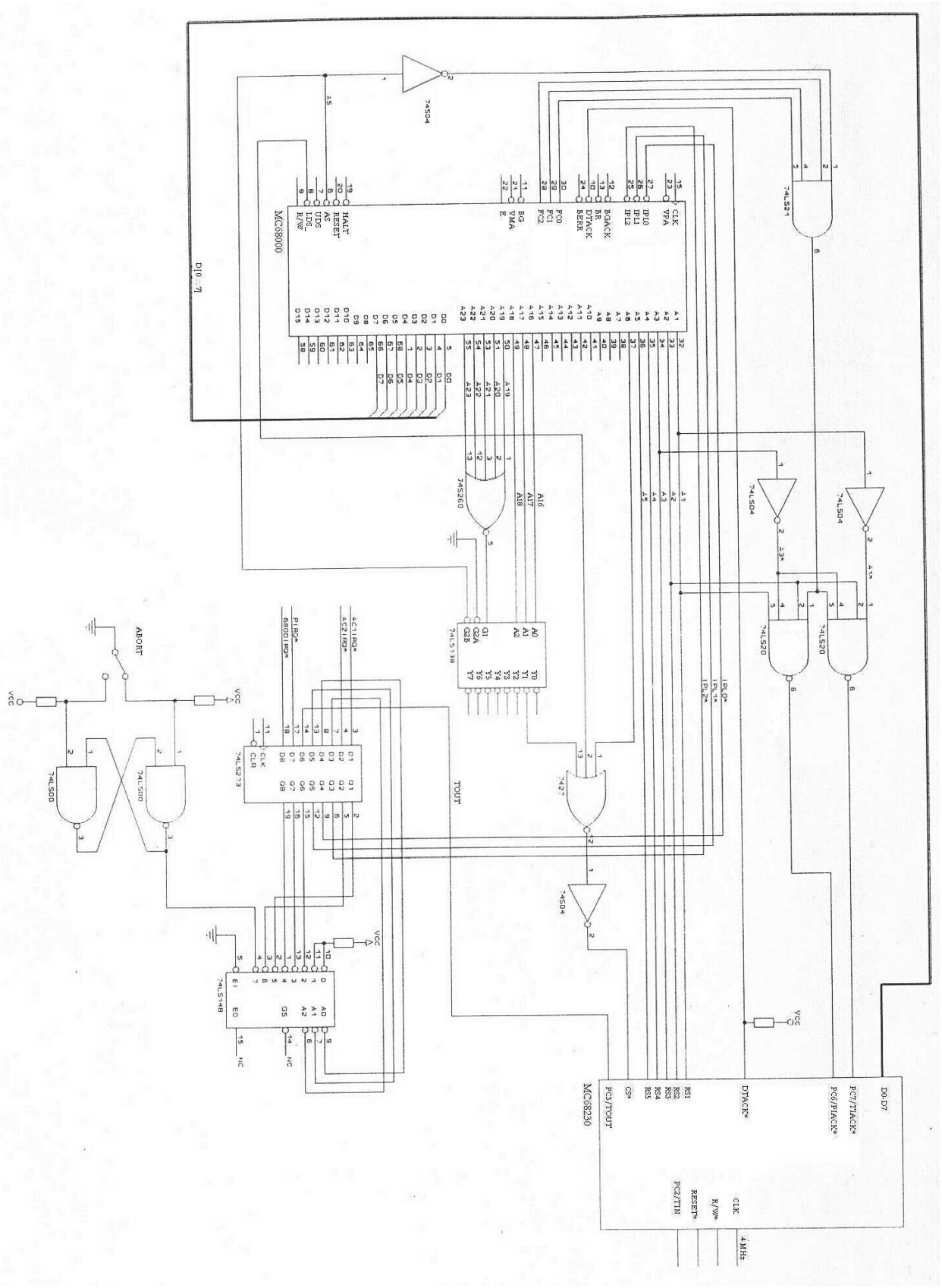
TCR 0	FUNKCIJA: omogućavanje vremenskog sklopa
0	Vremenski sklop je onemogućen (zaustavljen).
1	Vremenski sklop je omogućen (pokrenut).

Registar prekidnog vektora (**TIVR** – Timer Interrupt Vector) sadrži 8-bitni vektorski broj koji se preko podatkovne sabirnice šalje procesoru u ciklusu potvrde prekida (ako je prekid prihvaćen i aktivirana je linija potvrde prekida TIACK). Ovaj se registar može čitati i pisati u svakom trenutku. Prilikom resetiranja sklopa (signal RESET*), TIVR se puni vrijednošću $0F_{16}$.

Registar za punjenje brojila (**CPR** – Counter Preload Register H, M, L) se sastoji od tri 8-bitna registra koji služe za pohranjivanje 24-bitne inicijalne vrijednosti koja se upisuje u brojilo prilikom pokretanja i nakon prolaska kroz stanje 0 (ovisno o načinu rada). Svakom dijelu registra (H, M, L) može se pristupiti zasebno instrukcijom MOVE.B, a može se i pristupiti istovremeno grupi 8-bitnih registara korištenjem naredbe MOVEP.W ili MOVEP.L. Ako se koristi naredba MOVEP.L, potrebno je adresirati nul-registar koji neposredno prethodi CPR-H, jer se na taj način formira 32-bitovni registar čiji se najznačajniji bajt (nul-registar) zanemaruje. Registri se mogu čitati i pisati u svakom trenutku. Da bi se osigurao ispravan rad sklopa PI/T, vrijednost $000000 ne može se pohraniti u CPR. Resetiranje sklopa ne utječe na sadržaj registra CPR.

Registar brojila (Counter Register H, M, L) je skup od tri 8-bitne adrese s kojih se može pročitati stanje brojila. Čitanje stanja brojila tijekom njegovog rada (stanje *run*) nije pouzdano. Operacija pisanja na ove adrese rezultira normalnim sabirničkim ciklusom, ali se podaci ignoriraju. Pristupanje ovom registru se, kao i kod registra CPR, može obaviti uporabom naredbi MOVE.B ili MOVEP.L (uz adresiranje nul-registra).

Statusni registar vremenskog sklopa (**TSR** – Timer Status Register) sadrži jedan bit (ZDS) iz kojeg se može očitati stanje nule. To bridom okidani bistabil koji se postavlja u 1 kada 24-bitno brojilo prelazi iz 000001_{16} u 000000_{16}. ZDS bit se postavlja u 0 izravnim upisom podatka čiji najmanje značajni bit ima vrijednost 1 ! ZDS bit se briše i prilikom zaustavljanja brojila, kao i resetiranjem sklopa. Registar se može čitati u svakom trenutku, pri čemu se neiskorišteni bitovi čitaju kao 0.



Slika 2. Shema sustava 68000 IR.

VJEŽBA 9. Mjerenje vremenskog perioda pomoću MC68230

Odrediti na kojim adresnim lokacijama se javlja MC68230 u sustavu 68000 IR (vidi sl.2). Napisati program koji će izmjeriti vrijeme potrebno za obavljanje zadane operacije. Program treba imati slijedeću strukturu:

```
početak
        napuni CPR;
        inicijaliziraj TCR i omogući TIMER;
        zadana operacija;
        zaustavi TIMER;
        očitaj CNTR;
kraj;
```

"Zadanu operaciju" najbolje je izvesti kao programsku petlju, čije se trajanje (u ciklusima) može procijeniti na temelju tablice instrukcija MC68000. Usporediti tako procijenjeno trajanje operacije s vremenom izmjerenim vremenskim sklopom. Izmjeriti trajanje operacije za dvostruko veći broj ponavljanja i usporediti rezultat s prethodnim.

VJEŽBA 10. Generiranje prekida nakon određenog vremenskog intervala

Programirati TIMER za generiranje jednog zahtjeva za prekid nakon zadanog vremenskog perioda. Koristiti liniju TOUT za zahtjev za prekid i TIACK* za potvrdu zahtjeva. Čemu služi i kako radi naredba STOP?

```
/* GLAVNI PROGRAM*/
početak
        napuni TIVR;
        napuni CPR;
        inic. TCR i omogući TIMER;
        STOP #n;
        očitaj CNTR;
kraj;
```

```
/*PREKIDNI PROGRAM*/
        početak
                zaustavi TIMER;
                ispiši "A";
        povratak;
```

VJEŽBA 11. Gniježđenje iznimki

Napisati glavni program koji će izazvati iznimku TRAP, a rutina za obradu te iznimke inicijalizirat će TIMER kao u vježbi 7.

```
/*GLAVNI PROGRAM*/
početak
        ispiši "A";
        TRAP #x;
        ispiši "G";
kraj;

/*RUTINA TIMER-a */
pocetak
        zaustavi TIMER;
        ispiši "C";
povratak;
```

```
/*RUTINA TRAP #x */
početak
        ispiši "B";
        napuni TIVR;
        napuni CPR;
        inic. TCR i omogući TIMER;
        STOP #n;
        ispiši "F";
povratak;
```

VJEŽBA 12. Višestruki prekidi

Koristiti tipku ABORT (prekid razine 7) za prekidanje procesora dok ispisuje slova "C" na ekran. Pri tome je potrebno sačuvati autovektor razine 7 (nalazi se na adresi $7C_{16}$), a umjesto njega upisati adresu vlastite rutine.

```
/* GLAVNI PROGRAM */
početak
        ispiši "A";
        TRAP #x;
        ispiši "G";
kraj;


/* RUTINA TRAP #x */
isto kao u vj.7


/* PREKID RAZINE 7 */
početak
        ispiši "D";
povratak;
```

```
/* RUTINA TIMER-a */
početak
        zaustavi TIMER;
        (7CH) --> -(SSP);
        adr.prekida 7 --> $7C;
        i:=0;
                ponavljaj
                        i:=i+1;
                        ispiši "C";
                dok je i<500;
        (SSP)+ --> $7C;
        ispiši "E";
povratak;
```

Dodatak I: Skup instrukcija mikroprocesora MC68000

INSTRUKCIJA	OPIS INSTRUKCIJE	**Oblik**	VELIČINA	ZASTAVICE **X N Z V C**
ABCD	BCD_Broj1 + BCD_Broj2 + X → BCD_Broj2, X(ako je došlo do preljeva). Zbraja 2 BCD broja i sadržaj X zastavice te rezultat stavlja u 2. BCD broj	Dx,Dy -(Ax),-(Ay)	B--	* U * U *
ADD	Bin_Broj1 + Bin_Broj2 → Bin_Broj2 Zbraja 2 broja i rezultat stavlja u 2. broj	Dn,<ea> <ea>,Dn	BWL	* * * * *
ADDA	Broj + An → An Zbraja Broj sa sadržajem Adresnog registra	<ea>,An	-WL	- - - - -
ADDI	#Broj + (ea) → (ea) Zbraja neposredni podatak (#Broj) sa sadržajem (ea)	#x,<ea>	BWL	* * * * *
ADDQ	Brzo zbrajanje s brojevima od 1 do 8	#<1-8>,<ea>	BWL	* * * * *
ADDX	Bin_Broj1 + Bin_Broj2 + X→ Bin_Broj2 Zbraja 2 broja i sadržaj X zastavice te rezultat stavlja u 2. broj	Dy,Dx -(Ay),-(Ax)	BWL	* * * * *
AND	Binarno 'I' između dva broja i rezultat u drugi broj	<ea>,Dn Dn,<ea>	BWL	- * * 0 0
ANDI	Binarno 'I' između neposrednog podatka (#Broj) i broja te rezultat u drugi broj	#<data>,<ea>	BWL	- * * 0 0
ASL	Aritmetički posmak lijevo. Najviši bit ostaje nepromijenjen i kopira se u C i X zastavice, u najniži bit (LSB) se zapisuje 0.	#<1-8>,Dy Dx,Dy <ea>	BWL	* * * * *
ASR	Aritmetički posmak desno. Najviši bit ostaje nepromijenjen. Najniži bit kopira se u C i X zastavice.	#<1-8>,Dy Dx,Dy <ea>	BWL	* * * * *
Bcc	Skok ako je ispunjen uvjet 'cc'	Bcc.S <label> Bcc.W <label>	BW-	- - - - -
BCHG	Testira bit i mijenja njegovo stanje. Z = 1 ako je bit bio 0	Dn,<ea> #<data>,<ea>	B-L	- - * - -
BCLR	Testira bit i postavlja ga na 0. Z = 1 ako je bit bio 0	Dn,<ea> #<data>,<ea>	B-L	- - * - -
BSET	Testira bit i postavlja ga na 1. Z = 1 ako je bit bio 0	Dn,<ea> #<data>,<ea>	B-L	- - * - -
BSR	Skok na podprogam. Povratak iz potprograma instrukcijom RTS	BSR.S <label> BSR.W <label>	BW-	- - - - -
BTST	Testira bit. Z = 1 ako je bit bio 0	Dn,<ea> #<data>,<ea>	B-L	- - * - -
CHK	Testira da li se sadržaj (ea) nalazi između 0 i vrijednosti Dn. Ako se nalazi dešava se izuzetak (TRAP)	<ea>,Dn	-W-	- * U U U
CLR	Sadržaj (ea) postavlja na 0	<ea>	BWL	- 0 1 0 0
CMP	Oduzima (ea) od Dn i postavlja zastavice. Sadržaj (ea) i Dn se ne mijenja. Koristi se za uspoređivanje dvaju brojeva.	<ea>,Dn	BWL	- * * * *
CMPA	Oduzima (ea) od An i postavlja zastavice. Sadržaj (ea) i An se ne mijenja. Koristi se za uspoređivanje dvaju adresa.	<ea>,An	-WL	- * * * *
CMPI	Isto kao i CMP, ali sa neposrednim podatkom.	#<data>,<ea>	BWL	- * * * *
CMPM	Uspoređuje dva broja čije su memorijske lokacije u Ax i Ay	(Ay)+,(Ax)+	BWL	- * * * *
DBcc	Ako uvjet 'cc' nije ispunjen smanjuje Dn za 1, i skače na <label>	DBcc Dn,<label>	-W-	- - - - -
DIVS	Djeli cjelobrojno predznačno Bin_Broj2 sa Bin_Broj1. Ostatak.w i Rezultat.w sprema u Bin_Broj2.l	<ea>,Dn	-W-	- * * * 0
DIVU	Djeli cjelobrojno nepredznačno Bin_Broj2 sa Bin_Broj1. Ostatak.w i Rezultat.w sprema u Bin_Broj2.l	<ea>,Dn	-W-	- * * * 0
EOR	Binarno 'isključivo ILI' između dva broja i rezultat u drugi broj	Dn,<ea>	BWL	- * * 0 0
EORI	Binarno 'isključivo ILI' između neposrednog podatka (#Broj) i broja te rezultat u drugi broj	#<data>,<ea>	BWL	- * * 0 0
EXG	Mijenja sadržaje dvaju registara	Rx,Ry	--L	- - - - -
EXT	Proširuje (Dn.b na Dn.w) ili (Dn.w na Dn.l) tako da kopira najviši bit (Dn.b ili Dn.w) preko dijela registra na koji se vrijednost registra proširuje	Dn	-WL	- * * 0 0
ILLEGAL	Instrukcija izaziva iznimku (Exception)	ILLEGAL		- - - - -
JMP	Bezuvjetni skok na (ea)	<ea>		- - - - -
JSR	Bezuvjetni skok na potprogram (ea)	<ea>		- - - - -
LEA	Stavlja (ea) u An	<ea>,An	--L	- - - - -
LINK	Dodaje prostor na stog tako da stari pokazivač stoga spremi na stari stog i zatim napuni pokazivač stoga sa sadržajem An + #pomak	An,#<displac>		- - - - -
LSL	Posmak ulijevo. Najniži bit postaje 0, najviši bit se kopira u C,X.	Dx,Dy #<1-8>,Dy <ea>	BWL	* * * 0 *
LSR	Posmak udesno. Najviši bit postaje 0, najniži bit se kopira u C,X.	Dx,Dy #<1-8>,Dy <ea>	BWL	* * * 0 *
MOVE	Kopira vrijednost iz (ea1) u (ea2)	<ea>,<ea>	BWL	- * * 0 0

MOVE	Kopira vrijednost (ea).b u CCR	<ea>,CCR	–W–	I I I I I
MOVE	Kopira vrijednost (ea).w u SR	<ea>,SR	–W–	I I I I I
MOVE	Kopira vrijednost SR u (ea).w	SR,<ea>	–W–	- - - - -
MOVE	Kopira vrijednost USP u An registar, ili vrijednost An u USP	USP,An An,USP	––L	- - - - -
MOVEA	Kopira vrijednost (ea) u An	<ea>,An	–WL	- - - - -
MOVEM	Kopira specificirane registre na (ea) ili sa (ea) u specificirane registre	<rglst>,<ea> <ea>,<rglst>	–WL	- - - - -
MOVEP	Stavlja pojedinačne byte-ove iz Dn registra na na gornje byte-ove riječi na koje pokazuje x(An) ili obrnuto (pri čitanju iz memorije)	Dn,x(An) x(An),Dn	–WL	- - - - -
MOVEQ	Stavlja 8-bitnu predznačenu vrijednost u Dn kao dugu riječ	#<-128, +127>,Dn	––L	- * * 0 0
MULS	Množi cjelobrojno predznačno Bin_Broj1i Bin_Broj2.	<ea>,Dn	–W–	- * * 0 0
MULU	Množi cjelobrojno nepredznačno Bin_Broj1i Bin_Broj2.	<ea>,Dn	–W–	- * * 0 0
NBCD	Negira BCD brojeve i zbraja ih sa X zastavicom. (0 - BCD_broj - X→ BCD_broj, X)	<ea>	B––	* U * U *
NEG	Mijenja predznak broju u (ea) (0 - broj → broj)	<ea>	BWL	* * * * *
NEGX	Mijenja predznak broju (ea)+X (0 - broj -X→ broj, X)	<ea>	BWL	* * * * *
NOP	Ne radi baš ništa	NOP		- - - - -
NOT	Komplementira broj. Binarni 'NOT' broja	<ea>	BWL	- * * 0 0
OR	Binarni 'OR' broja	<ea>,Dn Dn,<ea>	BWL	- * * 0 0
ORI	Binarni 'OR' broja sa neposrednim podatkom	#<data>,<ea>	BWL	- * * 0 0
PEA	Stavlja (ea) na stog	<ea>	––L	- - - - -
RESET	Resetira sve vanjske uređaje	RESET		- - - - -
ROL	Rotacija u lijevo bitova unutar registra. Najviši bit se kopira u C zastavicu, svi ostali bitovi se posmiču u lijevo, C zastavica se kopira u najniži bit.	#<1-8>,Dy Dx,Dy <ea>	BWL	- * * 0 *
ROR	Rotacija u desno bitova unutar registra. Najniži bit se kopira u C zastavicu, svi ostali bitovi se posmiču u desno, C zastavica se kopira u najviši bit.	#<1-8>,Dy Dx,Dy <ea>	BWL	- * * 0 *
ROXL	Rotacija u lijevo bitova unutar registra preko X zastavice. Najviši bit se kopira u C i X zastavicu, svi ostali bitovi se posmiču u lijevo, X zastavica se kopira u najniži bit.	#<1-8>,Dy Dx,Dy <ea>	BWL	* * * 0 *
ROXR	Rotacija u desno bitova unutar registra preko X zastavice. Najniži bit se kopira u C i X zastavicu, svi ostali bitovi se posmiču u desno, X zastavica se kopira u najviši bit.	#<1-8>,Dy Dx,Dy <ea>	BWL	* * * 0 *
RTE	Povratak iz obrade iznimke	RTE		I I I I I
RTR	Povratak iz potprograma sa povratkom CCR registra	RTR		I I I I I
RTS	Povratak iz potprograma	RTS		- - - - -
SBCD	BCD_Broj2 - BCD_Broj1 - X → BCD_Broj2, X (ako je došlo do podljeva). Oduzima 2 BCD broja i sadržaj X zastavice te rezultat stavlja u 2. BCD broj	Dx,Dy -(Ax),-(Ay)	B––	* U * U *
Scc	U byte na (ea) upisuje $FF (-1) ako je cc istina inače upisuje 0	<ea>	B––	- - - - -
STOP	Stavlja neposredni podatak u SR i zaustavlja zahvaćanje izvršavanja naredbi dok se ne dogodi TRACE prekid ili RESET iznimka	#<data>		I I I I I
SUB	Oduzima dva broja. (BIN_broj2 - BIN_broj1 → BIN_broj2)	Dn,<ea> <ea>,Dn	BWL	* * * * *
SUBA	Oduzima broj od An. (An - BIN_broj1 → An)	<ea>,An	–WL	- - - - -
SUBI	Oduzima neposredni podatak od broja. (BIN_broj2 - #broj → BIN_broj2)	#x,<ea>	BWL	* * * * *
SUBQ	Brzo oduzimanje neposrednih podataka iz intervala (1, 8) od broja.	#<data>,<ea>	BWL	* * * * *
SUBX	Oduzima dva broja sa X zastavicom. (BIN_broj2 - BIN_broj1 - X → BIN_broj2, X)	Dy,Dx -(Ay),-(Ax)	BWL	* * * * *
SWAP	Mijenja položaj riječi unutar duge riječi Dn.	Dn	–W–	- * * 0 0
TAS	Testira sadržaj (ea), setira najviši bit u (ea) i setira N ili Z flag	<ea>	B––	- * * 0 0
TRAP	Starta izvršavanje TRAP iznimke.	#<vector>		- - - - -
TRAPV	Starta izvršavanje TRAP iznimke ako je V zastavica postavljena	TRAPV		- - - - -
TST	Testira sadržaj (ea) i setira N ili Z flag	<ea>	BWL	- * * 0 0
UNLK	Oslobađa stog koji je rezervirala LINK instrukcija. An registar se ne smije promjeniti od izvođenja LINK instrukcije.	An		- - - - -

OPIS SIMBOLA ZA ZASTAVICE

*****	Postavlja se u ovisnosti o rezultatu operacije
–	Ne mijenja se
0	Postavljen na 0
1	Postavljen na 0
U	Stanje nakon operacije nije definirano
I	Postavljeno s neposrednim podatkom

OPIS SIMBOLA ZA OBLIKE INSTRUKCIJA

<ea>	Efektivna adresa
<data>	Neposredni podatak
<label>	Labela
<vector>	Vektor TRAP iznimke
<rg.lst>	Lista registara koje prebacuje MOVEM instrukcija
<displ.>	Veličina novog stoga kojeg kreira LINK instrukcija

Adresni načini

Data Register Direct	**Dn**
Address Register Direct	**An**
Address Register Indirect	**(An)**
Address Register Indirect with Post-Increment	**(An)+**
Address Register Indirect with Pre-Decrement	**-(An)**
Address Register Indirect with Displacement	**w(An)**
Address Register Indirect with Index	**b(An,Rx)**
Absolute Short	**w**
Absolute Long	**l**
Program Counter with Displacement	**w(PC)**
Program Counter with Index	**b(PC,Rx)**
Immediate	**#x**
Status Register	**SR**
Condition Code Register	**CCR**

Uvjeti koji se koriste za Bcc , DBcc i Scc instrukcije

Uvjeti koji se postave nakon izvođenja CMP D0,D1 instrukcije.

Odnos	Nepredznačni	Predznačni
D1 < D0	CS – Carry Bit Set	LT – Less Than
D1 <= D0	LS – Lower or Same	LE – Less than or Equal
D1 = D0	EQ – Equal (Z-bit Set)	EQ – Equal (Z-bit Set)
D1 != D0	NE – Not Equal (Z-bit Clear)	NE – Not Equal (Z-bit Clear)
D1 > D0	HI – HIgher than	GT – Greater Than
D1 >= D0	CC – Carry Bit Clear	GE – Greater than or Equal
	PL – PLus (N-bit Clear)	MI – Minus (N-bit Set)
	VC – V-bit Clear (No Overflow)	VS – V-bit Set (Overflow)
	RA – BRanch Always	

SAMO DBcc – F – Never Terminate (DBRA is an alternate to DBF)
T – Always Terminate

SAMO Scc – SF – Never Set
ST – Always Set

Dodatak II: Nadgledni program TUTOR

Vježbe se obavljaju na školskom sustavu MC 68000 pod upravljanjem nadglednog programa TUTOR. On omogućava pregled i izmjenu sadržaja registara, pregled i izmjenu sadržaja memorijskih lokacija, pokretanje izvođenja programa, postavljanje prekidnih točaka i slično. Pregled naredbi dan je u tablici, a njihov detaljniji opis može se pronaći u uputama na radnom mjestu u laboratoriju. Valja voditi računa o memorijskom prostoru koji je predviđen za korisničke programe, a on se nalazi u memorijskom području **$0900** do **$7FFF**.

Naredba	Opis
MD <adresa> [<broj>][;<opcije>]	Prikaz sadržaja navedenog broja memorijskih lokacija (bajtova) počevši od zadane adrese. Opcijama se može zadati format podatka (B - bajt, W - riječ, L – duga riječ) ili ispis disasemblirane liste naredbi (opcija DI).
MM <adresa> [;<opcije>]	Prikaz i omogućavanje izmjene sadržaja navedene memorijske lokacije. Opcije su jednake kao kod naredbe **MD.**
MS <adresa> <podatak>	Blok memorije počevši od zadane adrese puni se zadanim podacima. Npr. **MS 2000, ABCD** postavlja riječ $ABCD na memorijsku adresu $2000. **MS 2000, 'ABCD'** upisuje zadani niz znakova od adrese $2000.
.A0 do **.A7** **.D0** do **D7** **.PC, .SR** **.SS, .US**	Prikaz sadržaja navedenog registra (adresnog, podatkovnog, statusnog, programskog brojila, te nadglednog ili korisničkog kazala stoga). Ukoliko se iza registra navede novi sadržaj, on se upisuje u registar. Npr. **.A3 12AB** će u registar A3 upisati broj $000012AB.
DF	Prikazuje sadržaje svih registara procesora MC 68000.
BF <poč_adr><završ_adr><podatak>	Punjenje bloka memorije zadanog početnom i završnom adresom zadanim podatkom (zadaje se riječ). Npr. **BF 2000 2006 FFFF** upisat će riječ $FFFF na memorijske lokacije ad adrese $2000 do $2006.
BM <poč_adr><završ_adr><odred_adr>	Premještanje bloka memorije zadanog početnom i završnom adresom na zadanu odredišnu adresu.
BT <poč_adr><završ_adr>	Ispitivanje zadanog bloka memorije upisivanjem i čitanjem podataka (*destruktivno* ispitivanje).
BS <poč_adr><završ_adr>'niz' **BS <poč_adr><završ_adr><podatak> [<maska>][;<opcije>]**	Pretraživanje zadanog bloka memorije. Npr. znakovni niz bismo tražili naredbom **BS 1000 2000 'ABCD'**. Drugi oblik naredbe služi za pronalaženje numeričkog podatka, koji se može maskirati (logičkom funkcijom I), a kao opcija navodi se format podatka (B, W ili L).
DC <izraz>	Pretvaranje izraza u heksadekadski i dekadski oblik. Podrazumijevani oblik podataka je heksadekadski (eksplicitno se označava znakom $, dok se dekadski oblik specificira znakom &). Primjeri: **DC &120, DC $100, DC $15 + &12 - $A.**
BR <adresa>[;<broj_ponavljanja>]	Postavlja prekidnu točku na zadanu adresu. Ako se navede broj ponavljanja, zaustavljanje izvođenja programa događa se tek nakon zadanog broja prolazaka kroz prekidnu točku.
NOBR [<adresa1><adresa2>...]	Uklanjaju se prekidne točke s navedenih adresa. Ako se ne navede lista adresa, brišu se sve prekidne točke.
GO [<adresa>]	Pokreće se program od zadane adrese. Ako se adresa ne navede, izvođenje započinje od adrese u programskom brojilu. Izvođenje se zaustavlja ako se naiđe na prekidnu točku ili ako dođe do iznimke.
GT <završna_adresa>	Pokreće se program od adrese u programskom brojilu. Izvođenje se zaustavlja na zadanoj završnoj adresi (ili ako se naiđe na prekidnu točku ili ako dođe do iznimke). Završna adresa je zapravo privremeno postavljena prekidna točka.
GD [<adresa>]	"Izravno" se pokreće program od zadane adrese. Ako se adresa ne navede, izvođenje započinje od adrese u programskom brojilu. U ovom načinu izvođenja ne provjeravaju se prekidne točke.

TR [<broj>]	Započinje se izvođenje jedne po jedne instrukcije programa od adrese u programskom brojilu. Nakon izvršenja instrukcije, prikazuju se registri procesora i disasemblira sljedeća instrukcija. U trace-modu rada TUTOR-a pritisak na tipku <CR> ("Enter") izaziva izvršavanje sljedeće instrukcije. Za izlaz iz tog načina rada potrebno je unijeti neku naredbu i <CR>. Može se navesti broj instrukcija koje će se izvesti u slijedu.
TT <završna_adresa>	Započinje se izvođenje jedne po jedne instrukcije programa od adrese u programskom brojilu (kao TR). Instrukcije se izvode u slijedu sve do navedene završne adrese. Nakon toga se sljedeća instrukcija može izvesti sa <CR>
HE	Prikaz liste raspoloživih naredbi TUTOR-a.

Funkcije TRAP #14

U sustavu s nadglednim programom TUTOR predviđen je niz funkcija koje omogućavaju interakciju vaših asemblerskih programa s korisnikom. Te su funkcije u ovom sustavu pridijeljene naredbi programske zamke TRAP #14. Funkcije se odabiru na temelju *funkcijskog broja*, koji se prije poziva TRAP #14 upisuje *u registar **D7***. Neke funkcije zahtijevaju i neke parametre, koji se upisuju u određene podatkovne i/ili adresne registre, te rezultat izvršavanja vraćaju vrijednosti u nekim registrima. *Napomena: funkcije ne jamče da neće promijeniti vrijednosti ostalih registara, pa je međurezultate potrebno pohraniti prije poziva neke od funkcija !*

Raspoložive funkcije ukratko su opisane u tablici. Funkcije se mogu podijeliti u 3 grupe:

1. programi za ulaz/izlaz jednog znaka ili niza znakova sa u/i pristupa;
2. programi za pretvorbu:
3. programi za prijenos upravljanja na TUTOR.

Funkc. broj	Simbolički naziv	Opis
247	INCHE	Unos jednog znaka s tipkovnice. Znak se pohranjuje u D0 (najmanje značajan oktet). Funkcija koristi i uništava sadržaj registara D1 i A0 !
241	PORTIN1	Unos znakovnog niza s tipkovnice. Pri pozivu funkcije A5 pokazuje na blok memorije gdje se pohranjuje niz, A6 pokazuje sljedeću slobodnu lokaciju. Nakon povratka iz funkcije, A6 pokazuje prvi znak **iza** unesenog niza. Unos se završava nailaskom na <CR>.
224	PORTIN1N	Kao PORTIN1, ali se ne prelazi automatski u novi red.
248	OUTCH	Ispis jednog ASCII kodiranog znaka iz registra D0 na ekran. Funkcija koristi i uništava sadržaj registara D0, D1 i A0 !
243	OUTPUT	Ispis znakovnog niza. A5 pokazuje na blok memorije gdje je niz pohranjen, a A6 pokazuje prvi znak iza niza. Poslije izvršavanja A5 također pokazuje prvi znak iza niza.
227	OUT1CR	Ispis znakovnog niza i prelazak u novi red. A5 pokazuje na blok memorije gdje je niz pohranjen, a A6 pokazuje prvi znak iza niza. Poslije izvršavanja A5 također pokazuje prvi znak iza niza. Funkcija koristi i uništava sadržaj registara D0, D1 i A0 !
236	HEX2DEC	Pretvara (heksadekadski) broj smješten u D0 u ASCII kodiran dekadski broj. A6 pokazuje na početak memorijskog bloka u koji će se znakovi pohraniti. Završna vrijednost A6 je uvećana za broj dekadskih znamenaka. Funkcija uništava sadržaj registra D0 !
235	GETHEX	Pretvara ASCII predstavljenu heksa znamenku (zapisanu u najmanje značajnom bajtu D0) u odgovarajući broj (pohranjuje se u najmanje značajni bajt D0).
234	PUTHEX	Pretvara 1 heksa znamenku iz registra D0 u ASCII oblik. A6 pokazuje na početak memorijskog bloka u koji će se znak pohraniti. Završna vrijednost A6 je uvećana za 1. Funkcija uništava sadržaj registra D0 !
233	PUT2HX	Pretvara 2 heksa znamenke iz registra D0 u ASCII znakove. A6 pokazuje na početak memorijskog bloka u koji će se znakovi pohraniti. Završna vrijednost A6 je uvećana za 2. Funkcija uništava sadržaj registara D0 i D2 !

232	PUT4HX	Pretvara 4 heksa znamenke iz registra D0 u ASCII znakove. A6 pokazuje na početak memorijskog bloka u koji će se znakovi pohraniti. Završna vrijednost A6 je uvećana za 4. Funkcija uništava sadržaj registara D0, D1 i D2 !
231	PUT6HX	Pretvara 6 heksa znamenke iz registra D0 u ASCII znakove. A6 pokazuje na početak memorijskog bloka u koji će se znakovi pohraniti. Završna vrijednost A6 je uvećana za 6. Funkcija uništava sadržaj registara D0, D1 i D2 !
230	PUT8HX	Pretvara 8 heksa znamenke iz registra D0 u ASCII znakove. A6 pokazuje na početak memorijskog bloka u koji će se znakovi pohraniti. Završna vrijednost A6 je uvećana za 8. Funkcija uništava sadržaj registara D0, D1 i D2 !
226	GETNUMA	Pretvara ASCII kodirani heksadekadski broj (niz znakova na čiji početak pokazuje A5) u (heksadekadski) broj koji se zapisuje u D0. A6 mora pokazivati znak iza zadnje znamenke ASCII kodiranog broja.
225	GETNUMD	Pretvara ASCII kodirani dekadski broj (niz znakova na čiji početak pokazuje A5) u (heksadekadski) broj koji se zapisuje u D0. A6 mora pokazivati znak iza zadnje znamenke ASCII kodiranog broja.
229	START	Prijenos upravljanja s korisničkog programa na TUTOR. Pritom se obavlja reinicijalizacija sustava. Kazalo stoga (A7) inicijalizira se na 'SYSSTACK'. Vektori iznimki, smješteni u nižem dijelu sistemskog RAM-a također se inicijaliziraju. Gornji dio RAM-a puni se nulama. Registar statusa postavlja se u $2700 – prekidna maska razine 7 i nadgledni način rada.
228	TUTOR	Prijenos upravljanja s korisničkog programa na TUTOR, ali bez reinicijalizacije sustava. A7 i SR se inicijaliziraju kao i kod funkcije START.