A101, 10h

Razlika izmedu bezuvjetne, uvjetne i prekidne VJ, razlika izmedu RET, RETI i RETN, jos par pitanja u kodu tipa: zasto spremamo kontekst u potprogramima, gdje se nalazi prekidni potprogram...
P.S. Nisu nikog izbacili

EDIT: Lagan blic, treba nadopunit kod, zadatak je slican prvom zadatku(BVJ, UVJ i prekidna VJ). Treba napisati prekidni vektor zatim u prekidnom potprogramu prihvatiti prekid, ucitati podatak, poslati ga na drugu lokaciju i javiti kraj posluzivanja. U glavnom programu treba postaviti GIE u 1 (MOVE %B 10000, SR) i na kraju programa je trebalo poslati podatak na UVJ i obrisati njen bistabil stanja.

A101, 10h
Malo da objasnim one linije iz compasa iz dma, sta je koja adresa, zasto sam pop-o sr, gdje se prvo mijenja sr, pokazati memoriju u dma, i mislim da je sve pitao ret reti i retn, kaj je za kaj

Blic
Trebalo je staviti DW 300, onda u prekidnom programu LOAD R0, (ZADNJI), pa STORE R0, (DATA_P) i jos jedan STORE R0, (IEND_P), u glavnom je trebalo SR inic, MOVE %B 10000, SR, i na kraju jos neke labele minus i plus, dvije crte, prvu prepisemo iz minusa samo drugu vj stavimo, i drugu prepisemo samo umjesto data ili neceg ide STORE R2, (BRISI_U ili P)

Bio jako lagan zadatak sa prekidnom VJ, uvjetnom VJ i bezuvjetnom VJ. Trebalo je u prekidnom potprogramu sačuvati kontekst, prihvatit prekid i javit obrađeni prekid, na početku inicijalizirat stog, u glavnom programu ucitat podatak sa uvjetne vj i resetirat BS iste, i na kraju spremiti neki brojac natrag na labelu s koje je ucitan.

---

kako funkcionira ono mnozenje sa SHL R3,2,R4 da dobijem da se mnozi sa 5 recimo itd?
Blic inace prvi zadatak sa labosa treba spremiti kontekst, onda program sprema unutar nekih varijabli zatim treba poslati signal kraja i obnoviti kontekst. Na pocetku treba stack pointer inicijalizirati, te mnoziti R3 sa 5

---

blic danas u 18-20
jedan zadatak sa bezuvjetnom, uvjetnom i prekidnom VJ trebalo je nadopunit sto ide na pocetku prekidnog potprograma (push R0; store R0, (IACK_P) , na kraju prekidnog (pop R0, RETI buduci da je maskirajuci), provjera spremnosti uvjetne VJ (LOAD R0, (BS_U); CMP R0, 0; JR_EQ PETLJA) i sto ide na kraju petlje prije JR PETLJA, u ovom slucaju je trebalo stavit ADD R0, 4, R0

Blic grupa 16-18.
vj1 prenosi 600 podataka u memoriju od adrese 1000. Na kraju prijenosa mijenja se aktivan iz 1 u 2 te pocinje prebacivanje iz memorije na vj2. vj3 koja radi u prekidnom maskirajucem prima podatak koja vj je aktivna (1 ili 2).

prve dvije linije:
MOVE 10000,R7(ILI SP)
JP GLAVNI

potprogram pocinje na aresi 500:(3 linije, 1 neiskoristena)
LOAD R0,(AKTIVAN)
STORE R0,(P_IACK)nezz bas dobro labele
izlazak iz potprograma gdje se nesto jos radi
Pocetak glavnog (Inicijalizacija registara tj. 600 podataka , adresa 1000 itd.)
3 linije(ispitivanje spremnosti VJ1):
LOAD R0,(STATUS_1)(ili neki drugi registar osim onih inicijaliziranih)
CMP R0,0
JR_EQ PETLJA1
spremanje podataka
1 linija :(u komentaru pise "kraj petlje"
ADD R1,4,R1;povecanje adrese (tj.1000)
Ponovno inicijaliziranje na 600 i 1000
ispitivanje bistabila stanja VJ2
Spremanje podataka
2 linije su tu na prvu nista na drugu:
ADD R1,4,R1;(opeta povecanje adrese za citanje);
Ovo sa malim slovima komentiram da malo bolje shvatit program(nije potrebno pisat samo ovo sa linijama)

Blic: ponovno zadatak nadopuni naredbe, 15minuta, nije tezak zadatak. Treba samo pripaziti mozda na prekidni potprogram da se spremi status registar jer se rade aritmeticke radnje. To sam dodao zadnju minutu xD

Blic je bio valjda pet puta laksi nego onaj prvi, nije bilo trikova u opisu zadatka, potrebno je nadopuniti program koji radi sa bezuvjetnom, uvjetnom i prekidnom jedinicom.
Obratiti pozornost na incijalizaciju SRa, ORG prije prekidnog potprograma, petlju za ispitivanje spremnosti uvjetne jedinice, te dojavu prihvata i obrade prekida jedinica.

---

blic u 12

ORG 8
DW 400

(čitanje i pisanje) STORE(...), LOAD(...) ,STORE(...)
(provjeravanje spremnosti dviju VJ)....CMP ili fora s OR, i JR (uvjet ovisno o prethodnoj liniji)

Blic je bio dosta lagan, jedan program sa nadopunavanjem.
Trebalo je postavit prekidni vektor i adresu prekidnog vektora.
Učitati podatak sa lokacije,poslati ga prekidnoj vj i dojaviti da je prekid obraden.
Dvje petlje koje cekaju spremnost vanjskih jedinica-
za prvu CMP r?,1
JR_NE petlja vj2
i za drugu CMPr?,1
JR_NE petlja vj1 i to je to

U blicu je trebalo na početku glavnog programa dodati

MOVE 10000, SP
JP GLAVNI

U glavnom su bila 3 reda za provjeru spremnosti uvjetne jedinice, znači:

PETLJA LOAD RO, (STATUS_1)
AND R0,1,R0
JR_Z PETLJA

Na dva mjesta je trebalo povećati adresu memorije jer su se podatci čitali/pisali u nju u petlji.

ADD R1, 4, R1

I u prekidnom potprogramu je trebalo prihvatiti prekid, i uzeti broj sa memorijske lokacije, nešto ovako;
STORE R0, (IACK_1)
LOAD R0, (AKTIVNA)

---

u blicu dvije uvjetne i jedna prekidna jedinica, ne sjećam se točno ali VJ1 upisuje 600 podataka od početne adrese 1000 u memoriju, a kasnije VJ2 čita te podatke. u prekidnu jedinicu VJ3 se upisuje 1 ili 2, ovisno o tome koja uvjetna VJ trenutno radi. trebalo je nadopuniti na početku nakon `ORG 0
MOVE 10000, SR
JP GLAVNI

u prekidnom potprogramu javiti prihvat prekida i spremiti u DATA od vj3 koja uvjetna jedinica je trenutno aktivna:
LOAD R0, (AKTIVNA)
STORE R0, (VJ3_IACK)
(store r0, (VJ3_data) i ostatak je napisan)

u glavnom programu treba prvo provjeravati spremnost VJ1
PETLJA1 LOAD R2, (VJ1_BS)
CMP R2, 1
JR_NE PETLJA1
i malo dalje povećavati brojač memorijske lokacije,
ADD R1, 4, R1
u drugoj petlji, za VJ2 treba nadopisati samo ADD R1, 4, R1, ostatak je napisan

---

Blic:
MOVE 10000, SP; Spremiti kontekst, obraditi prekidnu vj i obnoviti kontekst (3. linije koda); Učitati neke podatke u registre i spremiti ih na drugu lokaciju (zavisno o parnosti/neparnosti)
Nije bilo nikavih postavljanja bitova, čekanja t mikro sek, ništa od CT-a, ništa od PIO-a.

Bio je jedan zadatak, u njemu bila bezuvjetna, uvjetna i prekidna jedinica..
Trebalo je ucitavat sa bezuvjetne 2'k brojeve i onda ako je pozitivan slat na uvjetnu, ako je negativan na bezuvjetnu, dok u međuvremenu radila prekidna, tako nešto...

I onda trebalo nadopunjavat neke stvari npr:
ORG 8
DW 400 ---- evo ovo trebalo napisat, jer se prekidni nalazio tu..

onda je u prekidnom potprogramu trebalo napisat dojavu prihvata prekida, pa ucitat neki podatak i poslat ga na prekidnu i dojavit kraj
a u glavnom na pocetku stavit MOVE %B 1100000, jer je bio na int2

Zadatak sasvim jednostavan, nadopunjavanje..

Po jedna vanjska jedinica Bezuvjetna/Uvjetna/Prekidna.. Uzimas neki podatak sa bezuvjetne, pozoves potprogram, on odredi jel paran ili neparan, ovisno o tome nesto promjeni u podatku te spremi podatak na Uvjetnu tj. Prekidnu jedinicu. Morale su se nadopuniti linije koje provjeravaju spremnost uvjetne, spremaju podatke u prekidnom potp (i RETI na kraju), a na poceku postaviti zastavice u SR.. Mislim da je to sve..

Bio jedan program sa 2 uvjetne vanjske jedinice i jednom prekidnom. Svaka prekidna sprema podatke na jednu od memorijskih adresa. Na labeli ZAMIJENI se nalazi -1. Podatci se sa VJ_1 salju u prekidni potprogram, a zadnji podatak poslan sa VJ_1 se sprema u zamijeni. Ako VJ_1 nije poslala podatak onda u ZAMIJENI treba biti -1. (to se inicijalno nalazi u programu):
Tako nešto je bilo, ovo se trebalo napisati. Znam da nema pretjerane koristi od samo tih naredbi (bez programa), ali možete vidjeti barem kakve stvari traže. **Pazite na dojavu o prihvatu prekida i kraju posluživanja!!! to je najčešća pogreška. I sretno svima ;)**
Za nadopuniti je bilo:

```
MOVE 10000, SP
JP GLAVNI
_____(PRAZNA CRTA)

; POTPROGRAM
STORE R0(IACK)
_____

STORE R0 (IEND)

; GLAVNI
MOVE %B 10100000
STORE R0, (BRISI_1)
STORE R0, (ZAMIJENI)
STORE R0, (BRISI_2)
```

BLIC kod Knezovića i Orsaga:

Bile su 3 vanjske jedinice: bezuvjetna, uvjetna i prekidna. Prekidna je spojena na INT 0.
Otprilike ovako su definirane:
DATA_B EQU 0FFFF1000
DATA_U EQU 0FFFF2000
STATUS_U EQU 0FFFF2004
DATA_P EQU 0FFFF3000
IACK_P EQU 0FFFF3004
IEND_P EQU 0FFFF3008

Uzima se podatak s bezuvjetne i šalje se na uvjetnu ili prekidnu (ovisno koja je spremna). Poziva se potprogram PAR_NEPAR (ovaj potprogram vam je napisan), on odredi jel podatak paran ili neparan, ako je paran prosljeđuje se, a ako nije onda se šalje -1.
Tako nekako je glasio zadatak. Bilo je 11 ili 12 praznih linija, a trebalo je nadopunit njih 8 ovim redoslijedom:

Najprije je trebalo u glavnom programu omogućit INT 0, bile su dvije prazne crte i mislim da nije bio komentar sa strane što treba napravit. Uglavnom, pišete:
MOVE %B 10010000, SR
druga crta ostaje prazna

Zatim je trebalo provjerit spremnost uvjetne jedinice:
PETLJA LOAD RO, (STATUS_U)
CMP R0, 1
JR_NE PETLJA
četvrta crta ostaje prazna

Nakon toga dolazi prekidni potprogram, na početku treba napisat:
ORG 1000
(Zbog toga jer na početku imate
ORG 08
DW 1000)

Zatim je trebalo spremit podatak na prekidnu jedinicu, pišete:
STORE R0, (DATA_P) ; ovdje imate čak i komentar
STORE R0, (IEND_P) ; dojava kraja prekida (OVA LINIJA NIJE ISKOMENTIRANA U TESTU)
treća crta ostaje prazna
I na kraju još trebate napisat:
RETI

Bio je blic drugaciji, no jako slican...
isto je bila prekidna, uvjetna i bezuvjetna
trebalo je napravit cijeli prekidni potprogram (bilo je vec napisano samo push,pop i reti), znaci trebalo je dojavit prihvat prekida, ucitat neki podatak sa fiksne lokacije, spremit ga na prekidnu jedinicu i dojavit kraj posluživanja.

u glavnom programu je trebalo omogucit prekide na INT2 (ako se dobro sjecam). uvjetna jedinica je bila skoro cijela napisana, trebalo je napravit JR_C NEPARAN jer su se ispitivali podaci sa uvjetne, ako je paran onda se nesto radi (sve je napisano tamo), a neparni se zanemaruju.

Blic... bio je u stilu prvog zadatka.
Trebalo je nadopisat linije koda u glavnom i potprogramu...
U glavnom je trebalo napisat liniju za LOAD podatka iz uvjetne jedinice.
Poanta je bila da se ulazni podatak množi s 5.
SHL za 2 polja je bio već upisan (shift za 2 polja je množ. s 4)
trebalo je samo napisat liniju s ADD za dodat još jednom početnu vrijednost kako bi ispalo mnozenje s 5...

Zadan je program koji ima 2 uvjetne jedinice na koje šalje podatke kad one budu spremne; u isto vrijeme postoji prekidna vj koja kad generira prekid, očitava sa mem. lokacije na koju su oni u programi spremali 1 ako je poslano na 1. vj, a 2 ako je poslano na 2. vj, što je poslano.
Cilj programa je da ako očita 1, šalje na ispis prekidne vj 1, ako 2 onda 2, a ako još ništa nema (nijedna uvjetna nije bila spremna još) onda šalje 0 (koja je tako i tako inicijalizirana od njihove strane). No to je nevažno, 90% programa je napisano i banalno, jednostavnije nego zvuči.

12 linija, treba ukupno nadopuniti 9 (to vam naravno ne kažu).

Mislim da su redom bile ove:
1) ORG 8
2) DW 400
3) LOAD R0, (ODAKLE)
4) STORE R0, (ISPIS_P)
5) STORE R0, (IEND_P)
6) OR R0, R0, R0
7) JR_Z ISPITAJ_2 << paziti jer se radi o prozivanju, mnogi su napisali ISPITAJ_1, to je krivo, ode 1 bod
8) OR R0, R0, R0
9) JR_Z ISPITAJ_1

---

blic: prenijeti 32-bitne NBC podatke s bezuvjetne vanjske jedinice na uvjetnu i prekidnu. i još neka fora s ispitivanjem parnosti al to vas ne zanima jer je taj potp napisan. ugl treba dodati sljedeće linije:
1.MOVE %B 10010000, SR

2.(ispitivanje spremnosti: LOAD R0, (ADRESA). CMP R0,1. JR_NE PETLJA).

3. ORG 1000(onaj broj koji ti oni zadaju ispod prekidnog vektora)

4.spremiti na prekidnu vj.: STORE R0, (ADRESA). STORE R0, (IEND).

5. RETI (jer se radi s INT0)

---

evo ovako ovo su rjesenja blica jer se bas ne sjecam najbolje ostatka koda...
ugl cini mi se da je ovako nekako zadatak isao, imamo vj1, vj2, vj3, imamo 300 brojeva koji se vrte, svaki broj mnozimo sa 5 (ili tak nesto) i sve spremamo u vj3 na adresi 4000 (?)

org0
move 10000, sp
.
.
org8
_____ rj: dw 500
.
.
.
e sad je tu isla provjera spremnosti i onda
_____rj: load r0, (citaj_)
_____rj:store r0, (brisi_)
.
.
.
_____rj: add r4,r3,r4

push r1
push r0
_____
_____
_____
_____ rj: tu je islo store, move sr, r0 i push
.
.
.
isto tako 4 linije kao i gore...takoder kao sto je kolega rekao nije ih trebalo sve popunit
ret