OHARRA. S/Iko laborategiko ariketen ebazpenerako honako funtzio eta definizio hauek erabil ditzakezu:

`void DisableInts();` Etenak galarazteko: IF (*Interrupt Flag*) adierazleari 0 balioa ezartzen dio.

`void EnableInts();` Etenak baimentzeko (aldez aurretik galarazi baldin badira): IF adierazleari 1 balioa ezartzen dio.

`unsigned short Cs(name);` "name" izeneko zerbitzu-errutinari dagokion segmentuaren hasierako helbidea itzultzen du.

`unsigned short Ip(name);` "name" izeneko zerbitzu-errutinak dagokion segmentuaren barruan duen desplazamendua itzultzen du.

`unsigned char InPort(portua);` sarrera/irteerako espazioko "portua" helbideko erregistroaren edukia itzultzen du.

`void OutPort(portua, balioa);` sarrera/irteerako espazioko "portua" helbideko erregistroan parametro gisa pasatutako "balioa" idazten du.

`unsigned char IrakurByteFis(unsigned short Seg, unsigned short Desp);` "Seg" segmentuaren barruan "Desp" desplazamendua duen memoria posizioaren edukia (byte batekoa) irakurri eta itzultzen du.

`void IdatzByteFis (unsigned short Seg, unsigned short Desp, unsigned char kar);` "Seg" segmentuaren barruan "Desp" desplazamendua duen memoria posizioan parametro gisa pasatutako "kar" karakterea (byte batekoa) idazten du.

`unsigned char IrakurHitzFis(unsigned short Seg, unsigned short Desp);` "Seg" segmentuaren barruan "Desp" desplazamendua duen memoria posizioaren edukia (byte batekoa) irakurri eta itzultzen du.

`void IdatzHitzFis (unsigned short Seg, unsigned short Desp, unsigned char kar);` "Seg" segmentuaren barruan "Desp" desplazamendua duen memoria posizioan parametro gisa pasatutako "kar" karakterea (byte batekoa) idazten du.

`void Eoi();` *End Of Interruption*: Etenaren zerbitzu-errutinaren exekuzioa amaitu dela adierazteko: etenen kontroladoreak hori jasotzen duenean, ISR erregistroan 1 balioa duten bit guztien artean pisu handieneko bita desaktibatzen du, posible eginez lehentasun txikiagoko etenak zerbitzatzea.

`#define PANT_HEL 0xB000` Pantailaren hasierako helbidea.

`#define ATRIB_NORMALA 0x07` Atributu normala: karakterea beltzez pantaila zuriaren gainean.

`#define IRR_ERREG_EK8259 0x20` PCaren etenen kontroladoreko IRR erregistroaren helbidea sarrera/irteerako espazioan.

`#define IMR_ERREG_EK8259 0x21` PCaren etenen kontroladoreko ISR erregistroaren helbidea sarrera/irteerako espazioan.

`#define DATU_ERREG_TEKLATU 0x60` PCaren teklatuaren kontroladoreko datu-erregistroaren helbidea sarrera/irteerako espazioan.

`#define KONTROL_ERREG_TEKLATU 0x61` PCaren teklatuaren kontroladoreko kontrol-erregistroaren helbidea sarrera/irteerako espazioan.

`#define ASCII_RETURN 13` RETURN karakterearen ASCII kodea.

`#define BAL_EOI 0x20` EOI eragiketa bat burutzeko 0x20 helbidean idatzi behar den balioa.

Era berean, suposatuko dugu jadanik definituta dagoela bektore bat, ASCII_TAULA[], non teklen posizio-kodea indize gisa hartuta, tekla bakoitzari dagokion ASCII kodea lor daitekeen, baina MAKE kasuan lortutako posizio-kodea bakarrik erabil daiteke indize gisa, ez BREAK kasukoa. Hau da, taula horretan begiratu dezakegu zein den tekla bati dagokion ASCII kodea tekla sakatzen denean, inoiz ez tekla askatzean.

Ariketa hauen bidez PCaren periferiko batzuen funtzionamendua (pantaila, etenen kontroladorea, teklatua, erlojua, etab.) aztertu nahi dugu alde praktikoa kontuan izanda.

1. ariketa

Idatz ezazu pantaila '*' karaktereaz modu jarraian beteko duen aplikazioa. Hori egin ondoren, pantaila ezabatuko da eta berriz izarrak idazten jarriko da. Programa nagusia exekutatzen ari den bitartean teklatuaren etenak tratatuko dira. Tekla bat sakatzean, tekla hori goiko eskuineko izkinan azalduko da besteekiko desberdina den atributu batekin. Programa 'Q' tekla sakatzean amaituko da.

Ariketa hau egiteko, programa nagusiaz gain, funtzio hauek ere programatu egin behar dira:

```
void IdatzKar (int lerro, int zut, unsigned char kar, unsigned char atrib)
     //Kar karakterea pantailan idazten du, lerro eta zut
     //parametroek adierazitako posizioan
void PantEzabatu ()
     //Pantaila guztia ezabatzen du
void IrakurKarPant (int lerro, int zut, unsigned char *kar,
                    unsigned char *atrib)
     //lerro eta zut-ek adierazitako pantailako posizioko karakterea irakurtzen du
void AldatuEB (int sarrera, unsigned short IPBerria, unsigned short CSBerria,
               unsigned short *IPZah, unsigned short *CSZah)
     //Eten-bektorea aldatzen du gure zerbitzu-errutinaren helbidea jartzeko
void BerreskuratuEB (int sarrera, unsigned short IPZah, unsigned short CSZah)
     //Eten-bektoreko hasierako balioa berreskuratu
void Eoi ()
     //Eoi (End of Interruption)
unsigned char TeklatuDatErregIrakurri ()
     //Teklatuaren datu-erregistroa irakurri eta itzuli
void StrobeTeklatu ()
     //Teklatuari Strobe bidali kontrol-erregistroko 7 bitean
void interrupt ZerErrTek ()
     // Teklatuaren zerbitzu errutina
```

2. ariketa

Idatz ezazu karaktere-kate bat teklatutik inkesta bidez irakurtzen duen aplikazioa. Teklatutiko sarrera amaituko da 80 karaktere edo lerro bukaerako karakterea (RETURN tekla) irakurritakoan.

Ariketa hau egiteko 1. ariketan programatutako funtzioak erabil daitezke., Programa nagusia eta funtzio hauek programatu behar dira:

```
unsigned char IRR_Irakurri ()
     // IRR erregistroaren balioa itzultzen du
void TeketenGalarazi ()
     // Teklatuko etenak galarazi
     // Etenak galarazi eta baimendu
void TeketenBaimendu ()
     // Teklatuko etenak baimendu
     // Etenak galarazi eta baimendu
unsigned char TekInkestaIrakurri ()
     // Sakatutako teklaren ASCII kodea itzultzen du
          // Teklatuko inkesta
          // MAKE eta ASCII kodearen tratamendua
          // Teklatuaren Strobea
```

3. ariketa

Idatz ezazu 10 minutuz exekutatuko den aplikazio bat. Tarte horretan pantaila salbatzea emulatuko dugu modu erraz batean, pantailan testu bat idatzita dagoela suposatuaz. Ez bada teklarik sakatzen 30 segundotan zehar, aldagai lagungarri batean kopiatzen da pantailaren edukia eta pantaila ezabatuko da. Pantaila berreskuratuko da edozein tekla sakatzean.

Ariketa hau egiteko 1. ariketan programatutako funtzioak erabil daitezke., Programa nagusia eta funtzio hauek programatu egin behar dira:

```
void interrupt  ZerbErrErlojua ()
     // Erlojuaren zerbitzu errutina
void interrupt ZerErrTek ()
     // Teklatuaren zerbitzu errutina
```

4. ariketa

Alda ezazu 3. ariketa bi kasu hauetarako:

4.1.- Pantaila bakarrik berreskuratzen da ‘A’ tekla sakatzean.

4.2.- Pantaila edozein tekla askatzean berreskuratzen da, sakatua izan ondoren.

5. ariketa

Gure PC-an periferiko berri bat konektatu nahi dugu. Periferiko honek IRQ6 eten-lerroaren bidez eteten du eta bere erregistroak eta helbideak hauek dira:

Datu-erregistroa: `@ 0x279` (8 bitekoa) Kontrol-erregistroa: `@ 0x281` (8 bitekoa)

Gailuarekin egiten den edozein eragiketaren ondoren, `strobe` sekuentzia bat egin behar da kontrol-erregistroko 5 bitean (bitak 0tik 7ra zenbatuak daude).

Programa ezazu C lengoaian honako kodea: (a) `strobe` errutina, (b) eten-bektoreko sarrera bat aldatzen duen errutina, (c) eten-bektorea aldatzen duen errutina honi egiten zaion deia gailu honen etenen eskaerak zerbitzatzeko.

6. ariketa

PCaren arkitekturan oinarrituta joko bat programatu nahi dugu. Programak kalkulatu behar du zenbat aldiz sakatzen den pultsadore bat 10 minutuko denbora tartean. Pultsadore honek sakatzen den bakoitzean IRQ4 lerrotik eteten du eta sortzen duen eten bakoitzeko `strobe` sekuentzia bat egin behar da bere kontrol-erregistroko (`0x281` helbidea du) 3 bitean. Pultsadore honen sinkronizazioa inkesta bidez burutzea nahi dugu.

Programaren hasieran mezu bat aterako da pantailatik erabiltzaileari adieraziz kontaketa hasiko dela, erabiltzailea pultsadorea sakatzen has dadin. Programa bukatzean (10 minutu pasatzen direnean) beste mezu bat aterako da pantailatik pultsadorea guztira zenbat aldiz sakatu den adieraziz. Mezuak pantailaratzeko suposatu honako funtzioak dauzkagula inplementatuta: `IdatzTestu(Testu)` eta `IdatzZenbaki(zen)`.

Azaldutako funtzionamendua kontuan hartuta, idatzi C programazio lengoaian funtzio hauek:

a) Programa nagusia. Adierazi zein aldagai global diren beharrezkoak programaren funtzionamendurako.

b) Erlojuaren zerbitzu errutina: `void interrupt Erlojua()`

c) Beharrezkoa den `strobe` errutina: `void Strobe()`

d) Pultsadorearen etenak galarazten dituen errutina: `void PultsadoreaEtenakGalarazi()`

7. ariketa

PCaren arkitekturan oinarrituta, dardo-joko bat programatu nahi dugu. Dardo batek ituan (*dianan*) jotzen duenean, gailuak eten bat sortuko du IRQ3 lerrotik, eta datu-erregistroan (0x281 helbidea du) lortutako puntuazioa adieraziko du. Demagun dardo guztiek ituan joko dutela.

Jokoaren funtzionamendua oso erraza da: 6 jaurtiketatan zein jokalarik lortzen duen puntuazio altuena jakin nahi dugu, erabili duen denbora ere kontuan hartuta. Aurreneko dardoak ituan jotzen duenetik neurtuko dugu denbora. Programaren hasieran, mezu bat aterako da pantailatik, jokalariari jokoa hasiko dela adierazteko eta dardoak botatzen has dadin. Programa bukatzean, beste mezu bat aterako da pantailatik, lortutako puntuazioa eta behar izan duen denbora (segundoak) adieraziz. Mezuak pantailaratzeko, honako funtzio hauek erabiliko ditugu: `IdatzTestu(testu)` eta `IdatzZenbaki(zenb)`.

Azaldutako funtzionamendua kontuan hartuta, idatzi C lengoaian honako funtzio hauek:

a) Programa nagusia. Adierazi programaren funtzionamendurako beharrezkoak diren aldagai globalak.

b) Beharrezkotzat jotzen dituzun zerbitzu-errutina guztiak.

c) Periferiko baten eskaerak zerbitzatzeko, eten-bektorea aldatzen duen errutina.

8. ariketa

Gure sisteman IRQ5 eten-lerroa erabiltzen duen gailu bat konektatu dugu. Gailu honen erregistroak 8 bitekoak dira eta datu erregistro bat eta kontrol erregistro bat ditu. Hauek dira erregistroen helbideak:

```
Datu erregistroa: @ 0x279          Kontrol erregistroa: @ 0x281
```

Sistemak izan behar duen funtzionamendua hau da. Gailu honek eteten duen bakoitzean bere datu erregistroa irakurri behar da eta ondoren `strobe` sekuentzia bat egin behar da bere kontrol erregistroko 3 bitean. Datu erregistroko 4 bitak 1 balio badu pantailatik ‘1’ karakterea aterako da. Datu erregistroko 4 bitak 0 balio badu, berriz, pantailatik ‘0’ karakterea aterako da eta programaren exekuzioa bukatuko da. Gailu honen sinkronizazioa inkesta bidez egitea erabaki da.

Pantailatik idazteko IdatzKar funtzioa erabili behar da. Pantaila memorian mapeatuta dago, 60 lerro eta 132 zutabe ditu eta karaktereak atributurik gabe idazten dira. Karaktere bakoitza kodetzeko 2 byte erabiltzen dira.

Sistemaren funtzionamendua kontuan izanda, idatzi C programazio lengoaian funtzio hauek:

a) Inkesta egiten duen funtzioa: `unsigned char Inkesta()`. Funtzio honek gailuaren datu erregistroaren balioa itzuliko du.

b) Programa nagusia. Beste gauza batzuen artean, inkesta egiten duen funtzioari deituko dio gailuak datu erregistroan utzitako balioa jasotzeko. Balio horren tratamendua burutuko du programaren bukaera detektatzen ez duen bitartean (datu erregistroko 4 bitak 0 balio duenean).

c) Gailuaren etenak galarazten dituen funtzioa: `void KPer_Etenak_Galarazi()`

d) `Strobe` errutina: `void Strobe()`

e) Pantailan karaktere bat idazten duen errutina:

```
void IdatzKar (int lerro, int zut, unsigned short kar)
```

9. ariketa

Gure PC-an IRQ3 eten-lerroa erabiltzen duen gailu berri bat konektatu dugu. Gailu honen erregistroak 8 bitekoak dira eta datu erregistro bat eta kontrol erregistro bat ditu. Hauek dira erregistroen helbideak:

```
Datu erregistroa: @ 0x279    Kontrol erregistroa: @ 0x281
```

Gailu honek eteten duen bakoitzean 8 biteko datu erregistroan 4 biteko bi zenbaki uzten ditu, bat pisu txikieneko lau bitetan (Zen1) eta bestea pisu handieneko 4 bitetan (Zen2). Bi datu hauek irakurri eta Zen1 eta Zen2 aldagai orokorretan gorde ondoren, `strobe` sekuentzia bat egin behar da kontrol erregistroko 6 bitean (bitak 0tik 7ra zenbatuak daude). Bi balioak aldagaietan gordetzeaz gain, pantailatik atera behar dira, Zen1 3 lerroan eta 5 zutabean eta Zen2 4 lerroan eta 5 zutabean. Pantailaren ezaugarriak hauek dira: 30 lerro eta 60 zutabe ditu, karaktere bakoitzak byte bat okupatzen du eta atributurik gabe idazten da, eta PANHEL helbidetik aurrera mapeatuta dago.

Errutina hauek idaztea eskatzen da: a) azaldutako tratamendua burutzen duen zerbitzu-errutina: `void interrupt ZerPer()`, b) `strobe` errutina, `void Strobe(),` eta c) gailu honen etenak galarazten dituen errutina `void KPer_Etenak_Galarazi().`

10. ariketa

256 lerro eta 256 zutabe dituen eserleku multzo batean okupatuta dauden eserlekuak zeintzuk diren adierazten digun sistema bat kontrolatu nahi dugu. Sistema honetan Keser kontroladore bat daukagu, 16 biteko datu erregistro bat (`0x480` helbidean) eta 8 biteko kontrol erregistro bat (`0x482` helbidean) dituena. Eserlekuetako batean norbait eseri edo altxatzen denean eten bat sortzen du IRQ7 lerrotik, eta datu erregistroan zein eserlekutan gertatu den aldaketa adierazten du, erregistroaren 16 bitetatik pisu handieneko 8ek lerroa adierazten dute eta pisu txikieneko 8ek zutabea. Eserleku horretatik norbait altxatu bada, kontrol erregistroko 3 bita 0 izango da, eta norbait eseri bada berriz 1 (bitak 0tik 7ra zenbatzen dira). Sistema hau, norbait altxatu edo esertzen den bakoitzean, memorian `MEMHEL` helbidetik aurrera mapeatuta dagoen 256×256 osagaiko eserleku-matrizea eguneratzeaz arduratu behar da. Matrize honen osagai bakoitza byte batekoa da. Posizio bakoitzean, 1eko batekin posizio horri dagokion eserlekua beteta dagoela adierazten du eta 0ko batekin hutsik dagoela.

Keser kontroladorearen sinkronizazioa etenen bidez izan daitekeen arren, inkesta bidez burutzea erabaki da. Horretarako Keser gailuaren etenak galarazi eta baimentzeko gai izan behar dugu. Sistema honen funtzionamendua aurrera eramateko zenbait errutina egitea eskatzen da: (a) gailuaren etenak galarazten dituen funtzioa (`void Keser_Etenak_Galarazi()`) (b) inkesta egiten duen errutina (`void Inkesta_Keser()`*);* honek ESERLEKU eta EGOERA aldagai orokorretan eserlekuaren kodea eta eserlekuaren egoera (betea/hutsik) gorde behar ditu hurrenez hurren eta (c) programa nagusia, inkesta burutu eta memoriako matrizean eserlekuaren informazioa eguneratu beharko duena, besteak beste.

11. ariketa

Lantegi batean suteak automatikoki detektatzea nahi dute, horretarako PC bateragarri bat erabiliz (i8086 Intel mikroprozesadorea). Suteak detektatzeko sentsore berezi bat erabiltzen da, KSUA izanik bere kontroladorea. Sentsore honek sua dagoela detektatzean IRQ3 eten-lerrotik eten bat sortzen du, datu erregistroan (0x381 helbidea) suteari buruzko informazioa utziz (sutearen egoera adierazten duen karaktere bat gordetzen du). KSUA kontroladorearekin sinkronizazioa etenen bidez izan daitekeen arren, sinkronizazio hori inkesta bidez burutzea erabaki da (teklatuarekin egin izan dugun moduan). Horretarako beharrezkoa da kontroladore honen etenak galaraztea. Kontroladore honen etenak galarazten dituen errutina C lengoaian programatzea eskatzen da: `void KSUA_Etenak_Galarazi();`

Behin etenak galarazita inkesta bidezko sinkronizazioa erabiliko denez, inkesta hori burutzen duen errutina ere idatzi behar duzu C lengoaian: `unsigned char Inkesta_KSUA().` Errutina honek kontroladoreko datu erregistroan gordeta dagoen karakterea itzultzen du. `strobe` sekuentzia bat ere egin beharko da kontrol erregistroko (`0x382` helbidea) 2 bitean. `strobe`a egiten duen errutina kodetzea ere eskatzen da: `void strobe_KSUA().`

Aurreko errutinan irakurritako suteari buruzko informazioa pantailatik atera behar da. Pantaila memorian mapeatuta dago `C000h` helbidetik aurrera eta 48 lerro eta 160 zutabe dauzka. Idatzi, C lengoaia erabiliz, lerro eta zutabe parametroek adierazten duten posizioan karaktere bat idazten duen errutina:

```
void IdatzKar (int lerro, int zutabe, unsigned char kar,
               unsigned char atrib);
```

12. ariketa

Gure PC-an periferiko berri bat konektatu nahi dugu. Gailu honek IRQ5 eten-lerroaren bidez eteten du. Periferikoaren erregistroak, 2 datu-erregistro eta kontrol-erregistro bat, 8 bitekoak dira eta ez daude memorian mapeatuta. Erregistro hauen helbideak hauek dira:

Bi datu-erregistro, DAT1 eta DAT2: `@ 0x479` eta `@0x480`

Kontrol-erregistroa: `@ 0x481`

Gailuarekin egiten den edozein eragiketaren ondoren, `strobe` sekuentzia bat egin behar da kontrol erregistroko 2 bitean (bitak 0tik 7ra zenbatuak daude).

Programa ezazu C lengoaian kode hau: (a) `strobe` errutina (`void Strobe()`), (b) periferiko honen eskaerak zerbitzatzeko eten-bektorea aldatzen duen errutina, (c) eten-bektorea aldatzen duen errutina honi programa nagusitik egiten zaion deia (d) periferiko honen etenen tratamendurako zerbitzu-errutina (`void interrupt PerErr()`). Zerbitzu-errutina honek EMAITZA aldagaian 16 biteko zenbaki bat gordetzen du. 16 biteko zenbaki hau osatzeko, DAT1 erregistroan zenbakiaren pisu txikieneko 8 bitak daude eta DAT2 erregistroan pisu handienekoak.