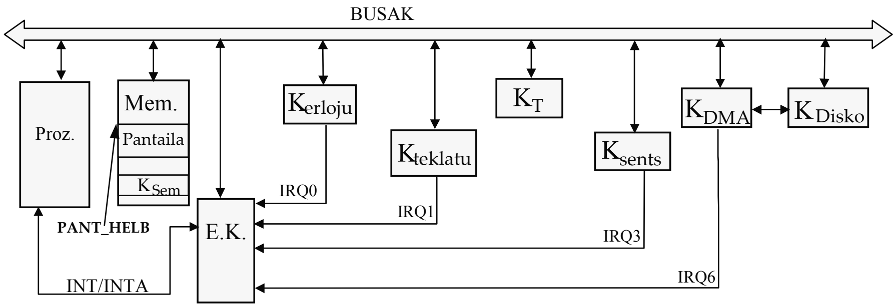
Konputagailuen Arkitektura I

Sarrera/irteerako azpisistema 3 (ebazpena): Teleordainketa-gunea

Autopista bateko bidesari finkoko zatietako irteeretan dauden ordainlekuetako kabina automatikoak (teleordainketa izenekoak) kontrolatzen dituen sistema bat diseinatu nahi dugu. Horretarako, irudiko eskeman adierazitako gailuak ditu sistemak, kabina bakoitzean.



Periferikoen kontroladoreen ezaugarriak honako hauek dira:

K_T: Teleordainketa kontroladorea da. Hiru erregistro ditu: kontrol-erregistroa (R_KON_KT), egoera-erregistroa (R_EGO_KT) eta datu-erregistroa (R_DAT_KT). Egoera-erregistroak 1 balioa hartuko du teleordainketa sistema duen auto bat detektatzen duenean. Datu-erregistroan —5 bytekoa—, detektatutako autoaren *IZ* identifikazio-zenbakia izango da. Kontrol-erregistroan, *strobe* sekuentzia bat egin behar da; horri esker, egoera-erregistroko balioa 0ra igaroko da eta kontroladorea prest geratuko da hurrengo autoa detektatzeko. Kontroladore honen sinkronizazioa **inkesta** bidez egin behar da.

K_Sem: Kabinaren goiko partean dagoen semaforoaren kontroladorea da. Memorian mapeatuta dagoen kontrol-erregistro bat du (SEM helbidean); bertan 1 idazten denean semaforoa gorri jarriko da, 0 idazten denean, berriz, berde.

K_sents: Kabinako irteerako langaren parean dagoen sentsore baten kontroladorea da. Eten bat eskatzen du bere aurretik auto bat erabat pasatu dela detektatzean.

K_DMA: DMA kontroladorea da. Memoriaren eta disko-unitate baten artean transferentziak egiteko erabiltzen da. Honako erregistro hauek ditu:

- **Kontrol-erregistroa** (R_KON_KDMA): erregistro honetan 1ekoa idaztean transferentzia hasiko da (automatikoki jartzen da 0an).
- **Helbide-erregistroa** (R_HEL_KDMA): transferituko den blokearen hasierako helbidea.
- **Luzera-erregistroa** (R_LUZ_KDMA): transferituko den blokearen luzera bytetan.
- **Egoera-erregistroa** (R_EGO_KDMA): transferentzia amaitzean, erregistro honek adierazten du transferentzia ondo joan den (1) edo erroreren bat gertatu den (0).

K_DISKO: Diskoaren kontroladorea da. Transferentzia bete ahal izateko kontroladore hau hasieratu behar da. Suposatuko dugu *KdiskoProgramatu()* izeneko errutina dugula hasieratze hori egiteko.

Beste kontroladore guztiak (erlojuarena, teklatuarena eta etenena) ikasgaian landutakoak dira. Sistema honetan **teklatua**rekiko sinkronizazioa **etenen** bidez egin behar da. Pantaila memorian mapeatuta dago, PANT_HELB helbidetik aurrera.

Sistemaren **funtzionamendua** honako hau izan beharko da. Kontrolatu beharreko kabinara teleordainketa-sistema duen auto bat iristen denean, kabinaren gaineko semaforoa berde badago, K_T kontroladoreak detektatuko du. Ordainketa automatikoki egingo denez gero (autoaren identifikazio-zenbakiari esker), sistemak kabinako irteeran dagoen langa altxatu behar du, ***altxatu_langa()*** errutinaren bitartez, autoa pasatzen uzteko.

Irteerako langa altxatuta mantenduko da autoa langa azpitik erabat pasatu arte eta 2 segundo gehiago, segurtasun arrazoiengatik, eta orduan jaitsiko da, ***jaitsi_langa()*** errutinaren bitartez. Sistema sinplifikatzeko, suposatuko dugu guztiz beharrezkoa dela langa jaitsi arte itxarotea hurrengo autoa detektatu ahal izateko.

Kobraketa automatikoak kudeatzeko, auto bat pasatzen den bakoitzean honako informazio hau gorde behar da memorian: *IZ* identifikazio-zenbakia —5 bytekoa— eta uneko ordua (*ORDUA* aldagai orokorrean dagoena) —5 bytekoa hori ere—. Informazio hori memorian idazteko, ***idatzmem (IZ, ORDUA, autokopurua)*** errutina daukagu eta bera arduratzen da zein memoria-helbidetan idatzi behar duen kontrolatzeaz: sistema hasieratzen denean, lehenengo autoari buruzko informazioa INF_HELB helbidetik aurrera idazten du; une horretatik aurrera, kabinatik pasatu diren autoen kopuruaren arabera kalkulatuko du helbidea. Orduari dagokionez, honako bi errutina hauek ditugu: ***HasieratuOrdua()***, *ORDUA* aldagaiari sistema eragileak esandako uneko balioa esleitzeko, eta ***EguneratuOrdua()***, *ORDUA* aldagai orokorra segundo batean gaurkotzen duena.

Kabinaren gaineko semaforoaren kolorea teklatuaren bidez kontrolatzen da:

- G tekla sakatzen denean, semaforoa gorri jarriko da eta ez da onartuko autorik pasatzea kabina horretatik (sistema sinplifikatzearren, suposatuko dugu tekla sakatu behar duen langileak sakatuko duela autorik ez dagoenean). Semaforoaren kolorea aldatzearekin batera, sistemak zera egin behar du: une horretara arte, semaforoa berde egon den bitartean, kabina horretatik pasatu diren autoei buruz memorian metatu duen informazio guztia diskora transferitu behar du, DMA bidez. DMA transferentzia amaitzean, errorerik gertatu baldin bada, beste 2 aldiz gehienez saiatuko da sistema transferentzia errepikatzen, baina 3 saioren ondoren transferentzia burutzea lortu ez bada, ***errorea_transferentzian()*** errutinari dei egingo zaio pantailan mezu bat azaltzeko, eta programari bukaera emango zaio. Errorerik gertatu ezean, funtzionamendu normalak jarraituko du (semaforo gorria). DMA kontroladorearen sinkronizazioa **etenen bidez** egin behar da.

- B tekla sakatzen denean, semaforoa gorri egonik eta DMA transferentzia amaituta, orduan semaforoa berde jarri eta funtzionamendu normalari ekingo dio sistemak (semaforo berdea). Semaforoa berde jarri ondoren pasatuko diren autoei buruzko informazioa berriro INF_HELB helbidetik aurrera gordeko da.

Hurrengoa eskatzen da: Idatzi lengoaia algoritmikoan beharrezkoak iruditzen zaizkizun zerbitzu-errutina guztiak eta programa nagusia. Komentatu ariketaren ebazpenerako egiten duzun edozein suposaketa. Kontuan hartuko da sistemaren portaera automata batez adieraztea.

Ebazpena

Lehenik, sistemaren funtzionamendua islatzen duen automata bilatzen saiatuko gara. Horretarako, biziki garrantzitsua da sistemaren funtzionamenduaren egoera edo etapa posible guztiak ondo bereiztea, eta baita ere egoera batetik beste batera pasatzeko egin beharreko urratsak identifikatzea ere. Oro har, egoeren arteko trantsizioak kanpoko gertaeren ondorioak izango dira, eta horiek periferikoen ekintzen bidez adieraziak izango dira, sinkronizazio-modua edozein izanik: inkesta bidezkoa ala etenen bidezkoa.

Ariketa honetan ez digute esaten zein diren sistemaren balizko egoerak, eta horregatik ongi irakurri behar dugu sistemaren funtzionamenduari buruz ematen diguten informazioa, egoerak ondorioztatzeko. Hala, azter ditzagun sistema honetan gerta daitezkeen gertaera posible guztiak eta ikus dezagun zein diren gertaera horien ondorioak, sistemaren egoeraren arabera. Horretarako, arreta handiz berrirakurriko dugu enuntziatua.

Hasteko, suposatuko dugu sistema martxan jartzen denean, kabinaren gaineko semaforoa berde egongo dela, eta sistema kotxeak detektatu eta pasatzen uzteko prest. Hasierako egoera horri *berdea* izena jarriko diogu.

1. *berdea* **egoera:**

- Egoeraren ezaugarriak:

  Egoera honetan, sistemak K_T sentsorearen inkesta egin behar du etengabe, ikusteko ea kotxeren bat kabinara hurbiltzen den. Ezezkoan, behin eta berriro irakurri behar da K_T sentsorearen egoera-erregistroa, kotxe bat detektatu arte; kotxea detektatzen den unean, langa altxatu beharko da, kotxeari pasatzen uzteko, eta kotxe horri dagokion informazioa gordetzeko.

  Beste aldetik, posible da, kotxerik ez dagoenean, langile batek G tekla sakatzea, eta orduan semaforoa gorri jarri behar da, eta une horretatik aurrera ez da kotxerik detektatu behar, berriro semaforoa berde egon arte (eta hori langileak B tekla sakatzen duenean gertatuko da, beste egoera batean, alegia).

  Hala, sistema egoera honetan dagoenean, bi gertaera posible daude, eta bakoitzak egoera-trantsizio desberdina eragiten du.

- Trantsizio posibleak:

a) K_T sentsoreak kotxe bat detektatzen duenean, langa altxatu behar da, kotxeari pasatzen uzteko. Horregatik, sistema beste egoera batera joango da: *kotxea_pasatzen* deituko diogu beste egoera horri.

K_T sentsorea inkestaz sinkronizatu nahi dugunez, haren gertaerak programa nagusian tratatuko dira, ez zerbitzu-errutina batean, etenen bidezko sinkronizazioan egiten den bezala. Hala, programa nagusiak nolabait detektatu beharko du kotxe bat hurbildu dela kabinara. K_T periferiko arrunta den neurrian, inkesta egiteko prozesadoreak haren egoera-erregistroa begiratzen du, nahi dugun gertaera detektatu arte. Hala, egoera-erregistroaren balioa 1 denean, kotxe bat detektatu du, eta dagozkion eginkizunak burutuko ditu. Balioa 0 denean, ordea, ez da ezer egin behar, egoera-erregistroa irakurtzen jarraitzea baizik.

↳ K_T sentsorean tratamenduaz arduratzen den programa-zatiaren eginkizunak:

Aipatu dugun egoera-trantsizioaz gain, K_T sentsorearen gertaeren tratamenduaz arduratzen den programa-zatiak beste eginkizun batzuk ere baditu (argi dago, inkestan kotxe bat hurbildu dela detektatu ondoren exekutatuko da zati hau; inkesta ez dugu hemen idatziko, orokorra delako, ez automataren mendekoa, horregatik, programa nagusia idaztean komentatuko dugu nola egin inkesta):

T.1) Langa altxatu behar da kotxeari pasatzen uzteko:

```
altxatu_langa();
```

T.2) K_T kontroladoreko datu-erregistroa (`R_DAT_KT`) irakurri behar da, kotxeari buruzko informazioa (identifikazio zenbakia, IZ) jasotzeko:

```
IZ = InPort(R_DAT_KT);
```

T.3) Kontroladoreko kontrol-erregistroan (`R_KON_KT`) "strobe" sekuentzia bat egin behar da. Honelaxe:

```
strobe(R_KON_KT);
```

T.4) Kotxeari buruzko informazioa memorian idatzi behar da:

```
idatzmem(IZ,ORDUA,auto_kopurua);
```

T.5) Beste kotxe bat pasatzen ari denez, aldagai global bat (`auto_kopurua`) erabiliko dugu kabinatik pasatzen diren kotxeen kopurua kontrolatzeko, eta une honetan eguneratu behar da:

```
auto_kopurua ++;
```

Gauza edo ekintza horiek automatan bertan argitzea komeni da, hori lagungarria izango baita programaren kodea idaztean. Horretarako, trantsizioaren geziaren azpian idatziko ditugu ekintza horiek guztiak, honelaxe:

b) Sistema *berdea* egoeran dagoen bitartean, langileak G tekla sakatzen badu, orduan sistema beste egoera batera pasatuko da, non semaforoa gorria egongo den eta ez den kotxerik detektatu behar. Horrez gain, kabina horretatik pasatu diren kotxe guztiei buruzko informazioa transferitu behar da memoriatik diskora, DMAren bitartez. Horregatik, DMA eta diskoa programatu behar ditugu, transferentzia egin dezaten. Egoera berriari *DMA_egiten* izena jarriko diogu.

Hauek dira trantsizio honetan egin beharreko eginkizunak:

↳ Teklatuaren zerbitzu-errutinaren eginkizunak:

Tek.1) Teklatuaren kontroladoreko datu-erregistroa (`R_DAT_Ktek`) irakurri beharko du, langileak zein tekla sakatu duen jakiteko:

```
tekla_kodea = InPort(R_DAT_Ktek);
```

Tek.2) Teklatuaren kasuan, kontroladoreko kontrol-erregistroaren gainean (`R_KON_Ktek`) "strobe" sekuentzia bat egin behar da. Honelaxe:

```
strobe(R_KON_Ktek);
```

Tek.3) Orain, tekla sakatu duen (MAKE) ala askatu duen (BREAK) begiratu behar dugu, eta gainera begiratu behar dugu ea G tekla den, eta hori guztia *berdea* egoeran soilik, beste edozein egoeratan ez baitzaio kasurik egin behar G teklari.

```
if ((MAKE(tekla_kodea))&&(ASCII_TAULA[tekla_kodea]== 'G')
&&(egoera_automata == berdea))
```

Tek.4) Semaforoa gorri jarri behar da. Semaforoaren kontroladoreko kontrol-erregistroa memorian mapeatuta dagoenez, dagokion memoriako posizioan (`SEM`) 1 balioa idatzi behar dugu, eta horretarako posizio hori aldagai global bat balitz bezala erabiliko dugu:

```
SEM = 1;
```

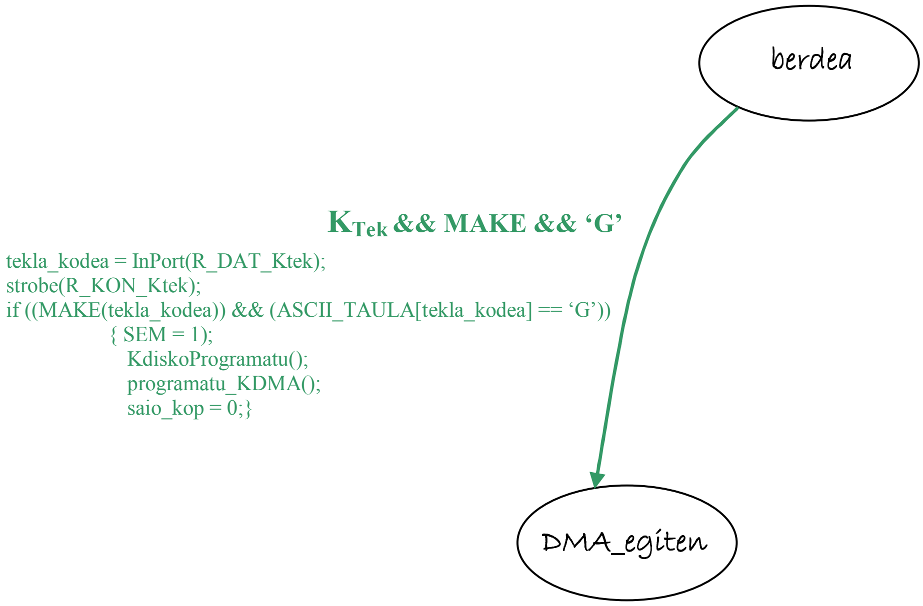
Tek.5) DMA transferentzia bat egin behar denez, diskoaren kontroladorea eta DMA kontroladorea programatu behar dira. Horrez gain, transferentzian errorerik gertatuz gero, gehienez hiru saio egin behar direnez, aldagai bat hasieratuko dugu egindako saioen kopurua kontrolatzeko:

```
KdiskoProgramatu();
Programatu_KDMA();
saio_kop = 0;
```

Hor erabili dugun `Programatu_KDMA()` funtzioa hauxe da:

```
OutPort(R_HEL_KDMA,INF_HELB);
OutPort(R_LUZ_KDMA,auto_kopurua × 10);
OutPort(R_KON_KDMA,1);
```

Helbidea INF_HELB da helbide horretatik aurrera metatu delako kotxeei buruzko informazioa, eta hortik aurrera dago diskora transferitu behar dugun informazioa; transferitu behar den blokearen luzera (`auto_kopurua ×` 10) da, kabinatik pasatzen den kotxe bakoitzeko 10 byteko informazioa metatu delako memorian, eta guztira `auto_kopurua` aldagaiak adierazitako kotxe-kopurua pasatu da kabinatik. Azkenik, kontrol-erregistroan 1 idaztean, transferentzia automatikoki hasiko da.



2. Kotxea_pasatzen **egoera:**

- Egoeraren ezaugarriak:
  Langa altxatuta dagoenez, kotxea pasatzen ari da.

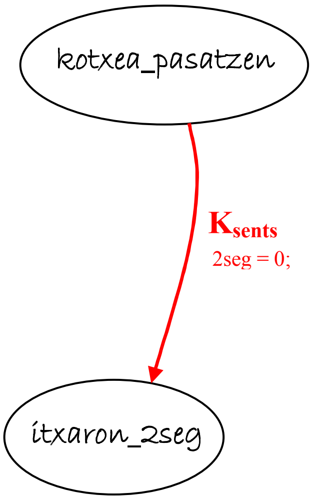
- Trantsizio posibleak:

a) Kabinaren irteeran sentsore bat dago, kotxea erabat pasatu dela detektatzen duena. Hortaz, egoera honetatik aterako da sistema Ksents sentsoreak etena eskatzen duenean, adierazteko kotxea jadanik pasatu dela. Segurtasun neurriengatik, langa berehala jaitsi beharrean, 2 segundoko tartea itxarongo du sistemak langa jaitsi aurretik (`2seg` aldagai globala beharko dugu). Hala, sentsoreak kotxea pasatu dela adierazten duenean, 2 segundoak kontatzeko "kronometroa" jarriko dugu martxan. Egoera berriari itxaron_2seg izena jarriko diogu.

↳ Sents sentsorearen zerbitzu-errutinaren eginkizunak:

Sents.1) `2seg` aldagaiaren hasieraketa:

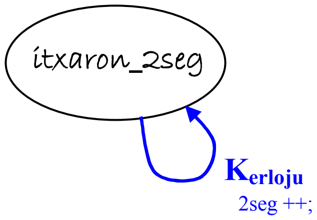
```
2seg = 0;
```



3. *itxaron_2seg* **egoera:**

- Egoeraren ezaugarriak:

  Egoera honetan sistema 2 segundoak pasatu zain dago, beraz, denbora pasa besterik ez da egiten, erlojuaren zerbitzu-errutinak `2seg` aldagaia eguneratu behar du:



↳ Erlojuaren zerbitzu-errutinaren eginkizunak:

Erloju.1) `2seg` aldagaiaren eguneraketa erlojuaren eten bakoitzean:

```
if (egoera_automata == itxaron_2seg)
   2seg++;
```

- Trantsizio posibleak:

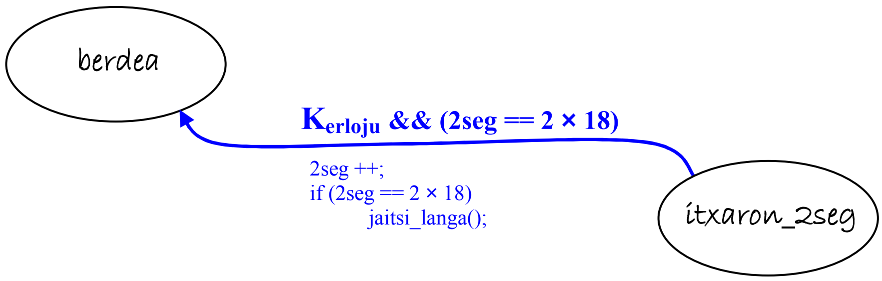
a) 2 segundoak pasatu ondoren, langa jaitsi behar da, eta sistema hasierako egoerara itzuliko da, hurrengo kotxea detektatzeko prest egotera.

Hala, honelaxe geratuko zaigu erlojuaren zerbitzu errutina egoera-trantsizio honi dagokionez:

Erloju.2) Egoera-trantsizioa, beharrezkoa baldin bada:

```
if (2seg == 2 × 18)
   {
      jaitsi_langa();
      egoera_automata = berdea;
   }
```

Eta automatan:



4. *DMA_egiten* **egoera:**

- Egoeraren ezaugarriak:

  Sistema iritsi da egoera honetara langile batek G tekla sakatu duenean, semaforoa berde zegoenean, eta horrekin batera DMA transferentziari hasiera eman zaio. Beraz, sistema egoera honetan dagoenean, DMA kontroladorea lanean ari da; transferentzia burutzean, DMA kontroladoreak etena sortuko du. Orduan, DMA kontroladorearen zerbitzu-errutina exekutatuko da, eta egoera-erregistroa irakurri beharko da ikusteko

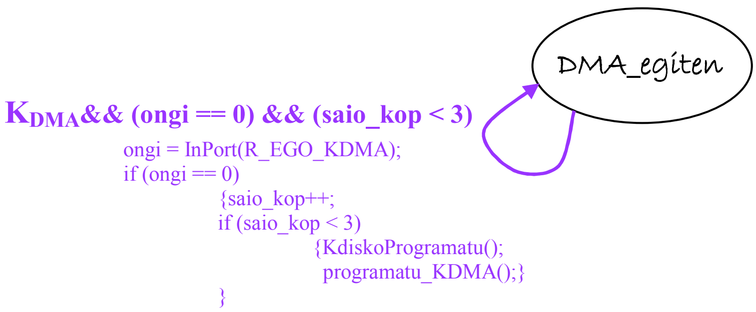
ea transferentzia ongi burutu den ala errorerik gertatu den. Horren arabera, ekintza desberdinak egin behar dira.
Hala, transferentzia ez bada ongi amaitzen, orduan saio kopurua hartu behar da kontuan: hiru aldiz baino gutxiago saiatu bada sistema transferentzia egiten, orduan berriro saiatu behar da, eta egoera berean mantendu. Bestela, hiru saio egin ondoren errorea ematen jarraitzen badu, orduan sistemaren egoera aldatuko da, programari amaiera emateko (trantsizioetan ikusiko dugu hau, hurrengo azpiatalean). Era berean, transferentzia ongi amaitzen bada hiru saio baino gutxiagotan, orduan sistema beste egoera batera joango da, non B tekla sakatu arte egongo den (hau ere trantsizioetan ikusiko dugu).
Hortaz, DMA kontroladorearen zerbitzu-errutinan bereizi beharko ditugu kasuak. Lehenik, errorea gertatzen bada baino hiru saio baino gehiago egin badira:

↳ DMA kontroladorearen zerbitzu-errutinaren eginkizunak:

DMA.1) Egoera-erregistroa irakurri; errorea gertatu bada, `saio_kop` aldagaia inkrementatu; hau 3 baino txikiagoa baldin bada, programatu berriro Kdisko eta KDMA beste transferentzia bat egiten saiatzeko:

```
ongi = InPort(R_EGO_KDMA);
if (egoera_automata == DMA_egiten)
   {
      if (ongi == 0)
         {
            saio_kop ++;
            if (saio_kop < 3)
               {
                  KdiskoProgramatu();
                  Programatu_KDMA();
               }
         }
   }
```

Automatan, honela islatzen da hori:



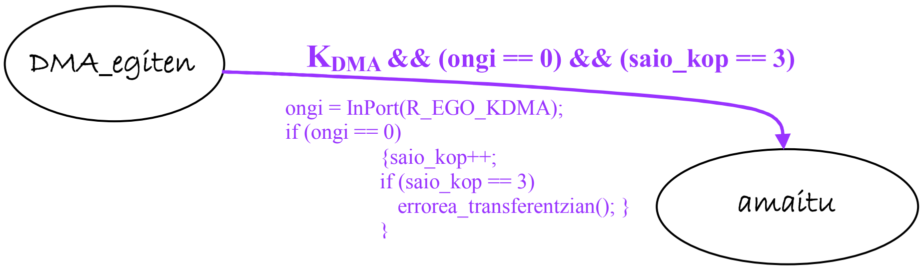
- Trantsizio posibleak:

a) DMA transferentzia hiru aldiz saiatu ondoren, oraindik errorea ematen jarraitzen badu, sistema *amaitu* egoerara pasatuko da, programari amaiera emateko.

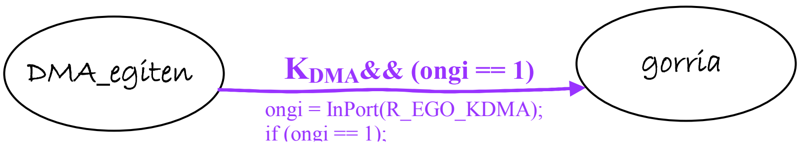
DMA kontroladorearen zerbitzu-errutinaren eginkizunak:

DMA.2) Egoera-erregistroa irakurri; errorea gertatu bada, `saio_kop` aldagaia inkrementatu; hau 3 denean, *amaitu* egoerara joan:

```
if (saio_kop == 3)
{
        errorea_transferentzian();
        egoera_automata = amaitu;
}
```



b) Transferentzia ongi amaitzen bada, orduan sistema *gorria* egoerara igaroko da, eta ez du beste ezer egin behar, langile batek B tekla sakatu arte, orduan hasierako egoerara itzuliko baita.



5. *gorria* **egoera:**

- Egoeraren ezaugarriak:
  Sistema egoera honetara iritsiko da DMA transferentzia ongi amaitzen bada, eta egoera honetan egongo da langile batek B tekla sakatu arte.

- Trantsizio posibleak:

a) Sistema *gorria* egoeran dagoenean, langile batek B tekla sakatzen badu, orduan sistema hasierako egoerara itzuliko da.

Teklatuaren zerbitzu-errutinaren eginkizunak:

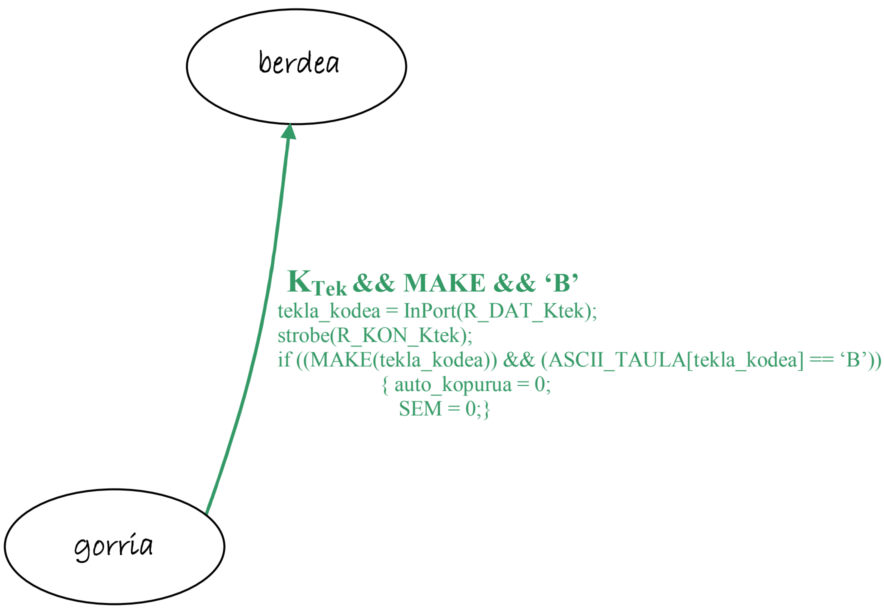
Tek.6) Semaforoa berde jarri behar da:

```
if ((MAKE(tekla_kodea))&&(ASCII_TAULA[tekla_kodea]== 'B')
&&(egoera_automata == gorria))
              SEM =0;
```

Tek.7) Sistemaren funtzionamendua berriro hasiko denez, auto_kopurua aldagaia hasieratu beharko dugu 0 balioarekin:

```
auto_kopurua = 0;
```

Honelaxe adieraziko dugu egoera-trantsizio hau automatan:



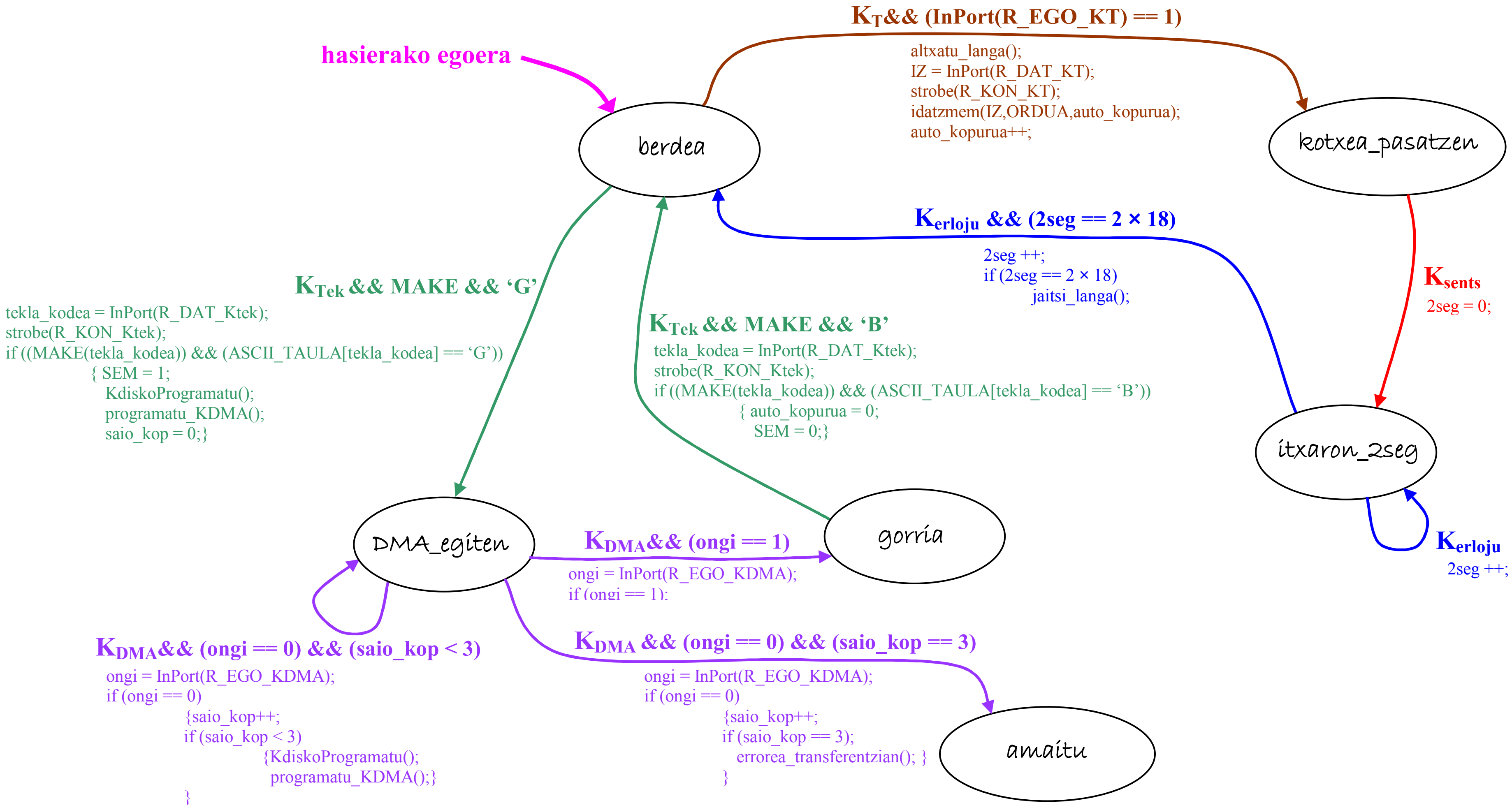
6. *amaitu* **egoera:**

- Egoeraren ezaugarriak:

Sistema egoera honetara iristen denean, programaren exekuzioa amaitu behar da. Horretarako, programa nagusian begizta bat jarriko dugu, sistema egoera honetan ez dagoen bitartean exekutatuko dena, eta egoera hau gertatzen denean, amaitzeko.

```
while (egoera_automata != amaitu)
```

Kasu posible guztiak denak batera bilduz, honelaxe geratuko da automata:



Eta, orain, kodea idaztea besterik ez da falta. Alde batetik, programa nagusia idatzi behar dugu, non automataren egoeraren menpeko begizta baten barruan KT periferikoaren inkesta egingo baita; begizta hori amaituko da automataren egoera *amaitu* denean. Horrez gain, periferikoen etenen zerbitzu-errutinak ere idatzi behar ditugu: K_{Sents}, K_{Tek}, K_{erloju} eta K_{DMA}.

Programa nagusia.

Programa nagusian, hainbat ekintza egin behar dira, hala nola: aldagaiak hasieratu (behar izanez gero), eten-bektorearen hainbat osagai aldatu (programaren hasieran) eta berreskuratu (bukaeran) –gure zerbitzu-errutinak exekutatuko direla ziurtatzeko, eta ez balizko sistema eragilearenak–, eta begizta bat ezarri gure programa behin eta berriro exekuta dadin, amaitu behar ez den bitartean. Begizta horren barruan, K_T periferikoaren inkesta egin behar da, detektatzeko noiz hurbiltzen den kotxe bat teleordainketa kabinara.

- Hasieratu beharreko aldagaiak:

  Hasteko, suposatuko dugu sistema martxan jartzen denean automata *berdea* egoeran egongo dela. Garbi izan behar dugu, beraz, sistemaren funtzionamendu egokia ziurtatzeko beharrezkoa dela uneoro jakitea zein egoeratan dagoen sistema, eta horretarako aldagai global bat behar da, `egoera_automata` izendatuko duguna, zeinaren hasierako balioa *berdea* izango baita. Horrekin batera, `auto_kopurua` aldagaia 0 balioarekin hasieratu beharko dugu, eta semaforoa berde jarri.

  Hala, honako hauek dira beharrezko hasieraketak:

  - ✓ `egoera_automata = berdea;`
  - ✓ `auto_kopurua = 0;`
  - ✓ `SEM = 0;    //semaforo = berdea`

  Horiez gain, `HasieratuOrdua()` funtzioa ere exekutatu behar da programaren hasieran, gure aplikazioaren denbora kontrolatu ahal izateko.

  Azpimarratu behar dugu ez dela beharrezkoa beste aldagaiak hasieratzea, erabili behar diren unean bertan balio egokiarekin hasieratzen direlako. Esate baterako, `2seg` aldagaia *kotxea_pasatzen* egoeratik *itxaron_2seg* egoerara pasatzean hasieratuko da 0 balioarekin, une horretan jarri behar delako martxan "kronometroa".

- Aldatu eta berreskuratu beharreko eten-bektorearen osagaiak:

  Sistema hau kontrolatzeko PC arrunt bat erabiltzen badugu, orduan guztiz beharrezkoa da eten-bektorearen hainbat osagai aldatzea, gure periferikoen etenei dagozkienak hain zuzen, kontrola guk nahi dugun moduan egin dezan PCak, hau da, gure periferikoek etena eskatzen dutenean, guk idatzitako zerbitzu-errutinak exekuta daitezen, eta ez sistema eragileak aurre programatuak dituenak .

  Hala, kasu honetan honako hauek aldatu behar ditugu: erlojuarena (eten-bektorearen 0x1C sarreran dagoena), K_Teklatu teklatuarena (0x09 sarrerakoa), K_sents sentsorearena (0x0B sarreran dagoena) eta DMA kontroladorearena (0x0E sarreran dagoena). Horretarako, suposatuko dugu jadanik idatzita dagoen funtzioa badugula, eta nahikoa dela aldatu beharreko sarrera parametro gisa pasatzea. Honelaxe:

✓ `Aldatu_EB(0x1C, 0x09, 0x0B, 0x0E);`

Era berean, gure programa amaitzen denean, orduan eten-bektorearen osagai horiek berreskuratu beharko ditugu, sistema eragileak PCaren kontrola har dezan berriz ere. Honelaxe:

✓ `Berreskuratu_EB(0x1C, 0x09, 0x0B, 0x0E);`

- Begiztaren kontrola:

  Ziurtatu behar dugu gure programa etengabe exekutatzen dela, amaitu behar ez den bitartean. Horretarako, automataren egoeraren menpekoa den baldintzapeko begizta bat jarriko dugu. Hala, begiztaren gorputza behin eta berriro exekutatuko da baldintza betetzen den bitartean. Kasu honetan, enuntziatuari erreparatuz, aurreikusita dago gure programaren exekuzioari amaiera emango zaiola DMA transferentzian hiru errore gertatzen badira. Kasu horretan, automata *amaitu* egoerara joango da. Hori dela eta honako baldintza hau erabil dezakegu:

  ✓ `while (egoera_automata != amaitu)`

- K_T periferikoaren inkesta:

  Esan bezala, K_T periferikoaren sinkronizazioa inkesta bidez egin behar da. Baina kotxeak detektatu behar dira semaforoa berde baldin badago soilik; semaforoa gorria dagoenean, ordea, ez dira kotxeak kontuan hartu behar. Hala, begizta nagusiaren barruan, programa nagusiak automataren egoera aztertuko du: *berdea* baldin bada, orduan K_T kontroladorearen egoera-erregistroa irakurriko du; haren balioa 0 baldin bada, berriro irakurri behar da egoera-erregistroa, egoera aldatzen ez den bitartean. Egoera-erregistroaren balioa 1 baldin bada, berriz, automataren egoera *berdea* izanik, orduan kotxe bat detektatu da teleordainketa gunean, eta pasoa eman behar zaio; horretarako, K_T periferikoari dagozkion eginkizunak egingo ditu programak:

```
while (egoera_automata != amaitu)
    {
       if (egoera_automata == berdea)
          {
             kotxe = InPort(R_EGO_KT);
             if (kotxe == 1)
                {
                   altxatu_langa();
                   IZ = InPort(R_DAT_KT);
                   strobe(R_KON_KT);
                   idatzmem(IZ,ORDUA,auto_kopurua);
                   auto_kopurua++;
                }
          }
    }
```

Laburbilduz, honelaxe geratuko da programa nagusiaren kodea:

```
void main()
{
            //aldagaien hasieraketa
 egoera_automata = berdea;
 auto_kopurua = 0;
 SEM = 0)  //semaforoa berde jarri
 HasieratuOrdua();

            //eten-bektoreko osagaien aldaketa
 Aldatu_EB(0x1C, 0x09, 0x0B, 0x0E);

            //begizta nagusia
 while (egoera_automata != amaitu);
   {
            //automataren egoera berdea baldin bada, detektatu kotxeak; bestela, ez.
     if (egoera_automata == berdea)
        {
         kotxe = InPort(R_EGO_KT);
            {
               if (kotxe == 1)
                      //kotxe bat detektatzen bada:
                 {
                   altxatu_langa();
                   IZ = InPort(R_DAT_KT);
                   strobe(R_KON_KT);
                   idatzmem(IZ,ORDUA,auto_kopurua);
                   auto_kopurua++;
                       //egoera-trantsizio egokia bideratu
                   egoera_automata = kotxea_pasatzen;
                 }
            }
        }
   }
            //eten-bektoreko osagaien berreskurapena programaren amaieran
 Berreskuratu_EB(0x1C, 0x09, 0x0B, 0x0E);
}
```

✍ Sents sentsorearen zerbitzu-errutina.

Ordainlekuko irteeran dagoen sentsore honek kotxea erabat pasatu dela detektatzean, “kronometroa” martxan jarri behar da, segurtasunagatik 2 segundotan zain egoteko langa jaitsi baino lehen. Hortaz, zerbitzu-errutina honetan `2seg` aldagaia 0 balioarekin hasieratu besterik ez da egin behar, eta automataren egoera aldatu, noski. Dena den, sentsorearen gainean egon daitezkeen interferentziak saihesteko, zerbitzu-errutinaren hasieran automataren egoera aztertuko dugu, eta zerbitzu-errutinaren gorputza bakarrik exekutatuko da sistema *kotxea_pasatzen* egoeran baldin badago, bestela ez.

```
void interrupt Ksents_ZE()
  {
   if (egoera_automata == kotxea_pasatzen)
     {
            //kronometroa martxan jarri 2 segundo kontatzeko
       2seg = 0;
            //egoera-trantsizio egokia bideratu
       egoera_automata = itxaron_2seg;
     }
   Eoi();
   IRET;
  }
```

K_{erloju} erlojuaren zerbitzu-errutina.

Kasu honetan, `tik` aldagaia beharko dugu, segundo bakoitzean `EguneratuOrdua()` funtzioari deitu behar baitzaio, ORDUA izeneko aldagai orokorra eguneratzeko. Baina gure aldagaiari dagokionez (`2seg`) independenteki eguneratuko dugu exekuzio bakoitzean, balizko arazoak saihesteko.

```
void interrupt Kerloju_ZE()
{
 static int tik = 0;
      //segundoak kontrolatu
 tik++;
 if (tik == 18)
      {
       tik = 0;
            //ORDUA aldagai globala segundoan behin eguneratu
      EguneratuOrdua();
      }

 if (egoera_automata == itxaron_2seg)
        {
           2seg++;
              //egoera-trantsizio egokiak bideratu
           if (2seg == 2 × 18)
              {
                  jaitsi_langa();
                  egoera_automata = berdea;
              }
         }
 IRET;
}
```

✍ K_{Tek} teklatuaren zerbitzu-errutina.

Teklak sakatzean/askatzean, automataren egoera kontuan hartuko dugu, edozein tekla edozein unetan sakatzen bada, ez egiteko kasurik.

```
void interrupt Ktek_ZE()
{
        //sakatutako teklari dagokion posizio-kodea irakurri
  tekla_kodea = InPort(R_DAT_Ktek);
        //"strobe" sekuentzia kontrol-erregistroaren gainean
  strobe(R_KON_Ktek);
        //egoera-trantsizio egokiak bideratu
 if ((MAKE(tekla_kodea))&&(ASCII_TAULA[tekla_kodea]=='G')
    &&(egoera_automata == berdea))
    {
              //semaforoa gorri jarri
       SEM = 1);
       KdiskoProgramatu();
       Programatu_KDMA();
       saio_kop = 0;
       egoera_automata = DMA_egiten;
    }
 else
 if ((MAKE(tekla_kodea))&&(ASCII_TAULA[tekla_kodea]=='B')
    &&(egoera_automata == gorria))
    {
              //semaforoa berde jarri
       SEM = 0;
       auto_kopurua = 0;
       egoera_automata = berdea;
    }
  Eoi();
  IRET;
 }
```

Kasu honetan, datu-erregistroa egoera guztietan irakurtzen da, eta `strobe` ere, egoera guztietan egiten da, baina sakatu dena G edo B tekla izanez gero, bakarrik egingo zaio kasu sistema *berdea* edo *gorria* egoeretan dagoenean, hurrenez hurren; bestela ez zaio kasurik egingo.

Aurreko zerbitzu-errutinan `Programatu_KDMA()` funtzioa erabili dugu. Hona hemen haren kodea:

```
void Programatu_KDMA()
{
        OutPort(R_HEL_KDMA,INF_HELB);
        OutPort(R_LUZ_KDMA,auto_kopurua × 10);
        OutPort(R_KON_KDMA,1);
}
```

✍ K_{DMA} gailuaren zerbitzu-errutina.

```
void interrupt KDMA_ZE()
{
 ongi = InPort(R_EGO_KDMA);
 if (egoera_automata == DMA_egiten)
    {
      if (ongi == 0)
      {
            saio_kop ++;
            if (saio_kop < 3)
            {
                 KdiskoProgramatu();
                 Programatu_KDMA();
            }
            else
            {
                 errorea_transferentzian();
                 egoera_automata = amaitu;
            }
      }
      else
       egoera_automata = gorria;
 Eoi();
 IRET;
}
```