

GRAFIKA

Az AMIGA eltérően pl. a C64 - estől mindig valamilyen grafikus üzemmódot használ. Ezáltal a grafika szabadon keverhető a betűkkel. A hardware segítségével négy alapvető grafikus üzemmódot valósíthatunk meg: Az egyszerű, az Extra Halfbright, a Hold And Modify, és a Dual Playfield. Négy felbontás használható:

320x256 Kis felbontású - nem interlace
320x512 Kis felbontású - interlace
640x256 Nagy felbontású - nem interlace
640x512 Nagy felbontású - interlace

BITMEZŐK

Képzeljünk el egy képernyőt, amelyen 320 x 256 pontot lehet elhelyezni. Minden pontnak két féle állapota lehet: Bekapcsolt vagy bekapcsolatlan. Mivel csak ez a két állapot lehetséges, hozzá kell rendelni a képernyő pontjait a tárbeli bitekhez. Minden pont megfelel egy bitnek, és minden bit megfelel egy pontnak. Az első pont az 1. byte utolsó bitje, a második az 1. byte utolsó előtti bitje, a kilencedik pont a 2. byte utolsó bitje, stb. A teljes képernyő 320 x 256 pontjának megfelelő bitek együttese alkotja a bitmezőt. Ebből kiderül az is, hogy egy 320 x 256 pixeles bitmező tárigénye 81920 bit (10 Kbyte). Az amiga a színeket a piros(red), a zöld(green), és a kék(blue) alapszínek keverésével hozza létre. Mindhárom színösszetevőnek 16 - 16 intenzitása lehet, ezáltal 16x16x16, azaz 4096 színárnyalat érhető el. A színárnyalatok mindegyike meghatározható egy 12 bites számmal, melynek első 4 bitje piros összetevő, a második 4 összetevő a zöld, a harmadik 4 bitje pedig a kék összetevő mennyiségét határozza meg. A színek kiválasztása 32 darab 12 bites színregiszter segítségével történik. Ezekbe beírhatunk egyet - egyet a 4096 lehetséges szín közül, így a továbbiakban csak az adott színregiszterre kell hivatkozni.

GRAFIKUS ÜZEMMODOK

Egyszerű Grafikus üzemmód:

Ez a leggyakrabban használt mód. Egy bitmezővel egy háttér és egy tintaszín használható. Ha a képpontnak megfelelő bit értéke 1, akkor tintaszínű a pont, ha 0, akkor papírszínű. Több szín kezelését a következőképpen oldották meg a gép tervezői: Egy képet, amely 32 színt használhat, öt bitmező képvisel. Így minden képpontnak öt bit felel meg (minden bitmezőből egy - egy), amelyek egy öt bites számot alkotnak. Ez a szám meghatároz egy színregisztert, és a képpont a regiszterben található színt veszi fel. (A 0. regiszter mindig a háttérszínt tartalmazza). Ilyenkor 50 Kbyte-ra van szükség a képernyő adatainak tárolásához. Ha csak 16,8 vagy 4 színt használunk, akkor elegendő 4,3,2 bitmező is.

Extra Halfbright üzemmód:

Mivel a színeket csak a 32 színregiszteren keresztül tudjuk elérni, a regiszterek száma egyben korlát is. 32 színnél többet csak különleges trükkökkel lehet elérni. Az egyik ilyen trükk ez az üzemmód, amely magyarul különleges félfényerőt jelent. A név elég sokat elárul: Ugy használhatunk 64 színt, hogy a 32 színregiszter színeinek félfényerejü változata lesz a második 32 szín. Az üzemmód 6 bitmezőt igényel. Az alsó öt bitmező a már megismert módon a 32 alapszín valamelyikét határozza meg. Ha a hatodik bit egy, akkor az alsó öt bitmező által meghatározott szín félfényerővel lesz látható, a képernyőn. Ezt egyszerűen oldja meg a gép, a 4 bites színösszetevők mindegyikét elosztja kettővel.

Hold And Modify üzemmód:

Ez a legbonyolultabb grafikus üzemmód. Itt lehetőség van mind a 4096 szín egyidejű használatára, amihez ugyancsak 6 bitmező szükséges. Az üzemmód azon a felismerésen alapul, hogy egy képen belül ritkán van szükség hirtelen színváltásra, sokkal inkább folyamatos átmenetekre a világosból a sötétbe és fordítva. A HAM működése a következő: Az 1-4 bitmezők tartalmának 4 féle jelentése lehet. Hogy ezek közül mikor melyik érvényes, azt az 5 - 6 bitmezők tartalma határozza meg. Ha az 5 - 6 bitmezők tartalma nem 0, akkor az alsó 4 bitmező csak a három színösszetevő egyikét határozza meg, a képpont másik két összetevője ugyanaz lesz, mint a tőle balra eső pontnak. Ebből kiderül a HAM hátránya is: Ahhoz, hogy egy színtől eljussunk egy tetszőleges másikhoz, 3 képpontra van szükség: Először az egyik, aztán a másik, végül az utolsó színösszetevőt módosítva tudjuk csak elérni a kívánt színt, így éles kontrasztok nem hozható létre.

Dual Playfield üzemmód:

Lehetőség van arra is, hogy két grafikus képernyőt használjunk egyszerre, egymástól függetlenül. A képernyők két különálló képernyőnek tekinthetők, melyek közül az egyiknek átlátszó a háttere, így azon keresztül ráláthatunk a másik képre. Ez különösen játékoknál hasznosítható jól. Egyszerre maximum 2 képernyő kezelhető, maximum 3 - 3 bitmezővel (8 - 8 színnel). A képernyőn soronként épül fel a kép, a sugár 262 sort rajzol a képernyő tetejétől az aljáig. Ebből alapüzemmódban 200 használható. Egy teljes képernyőt 1/60-ad másodperc alatt rajzol fel a gép.

EGY ALAP KÉPERNYŐ DEFINIÁLÁSA

Először is eldöntendők a következők:

- A képernyő szélessége és magassága
- A színek maximális száma (2-32 vagy 4096)
- A vízszintes felbontás (320 vagy 640)
- A függőleges felbontás (200 vagy 400)
- DATA FETCH és MODULO

Ezután a megfelelő méretű memóriaterületet le kell foglalni (ALLOCATE), be kell állítani a megfelelő címeken elhelyezkedő HW mutatókat, meg kell írni a COPPER listát, hogy a grafikus rendszer fel tudja építeni az új képernyőt.

SZÍNEK SZÁMA

- színek számának eldöntése
- színek betöltése a színregiszterekbe
- kellő számú bitmező lefoglalása

A színek száma	Bittérképek száma
1-2	1
3-4	2
5-8	3
9-16	4
17-32	5
4096	6

A 32 színregiszter egymás után helyezkedik el, a \$DFF180 címtől kezdődően.

COLOR00	Háttérszín, keretszín HW által fixen
COLOR01	1. számú szín, 2 színű a képernyőn a tintaszín
COLOR02	2. számú szín
.	.
COLOR31	31. számú szín

Ha (opcionális) GENLOCK kártyát használasz, melynek bemenő jele videokamera, videomagnó, lézerlemez, akkor a háttérszín állandóan a külső jelnek megfelelően változik, a képernyőn nem állandó.

A színregiszterek felépítése a következő:

15-12 bitek	nem használtak
11-08 bitek	a vörös színösszetevő mértéke
07-04 bitek	a zöld színösszetevő mértéke
03-00 bitek	a kék színösszetevő mértéke

A színregiszterek feltöltése pl. a következőképpen történhet:

```
LEA    COLOR00,A0 ; színregiszterek báziscíme A0-ba
MOVE.W #$FFF,(A0) ; a háttérszín fehér
MOVE.W #$6FE,2(A0) ; az első szín világoskék
```

A színregiszterek csak írhatóak, olvasásuk legtöbbször a rendszer összeomlásához vezet, bár SUPERVISOR állapotban a háttérszín regiszter kiolvasható. Ha a program futása során szükség van a színregiszterek tartalmára, akkor a RAM -ban kell róla másolatot tartani. Ha már eldöntöttük, hogy hány bitmezőt kívánunk használni, akkor ezt tudatnunk kell a géppel is. A kívánt számot a BPLCON0 regiszter megfelelő bitjeibe kell beírni. Az e célra szolgáló bitek a 14 - 12 bitek. A hatodik bitmező kizárólag DUAL PLAYFIELD és a HAM üzemmódokban használatos. A regiszter bitjei önnállóan nem állíthatóak, az összeset újra kell írni egy esetleges változtatáskor.

2 kisfelbontású bitmező bekapcsolása:

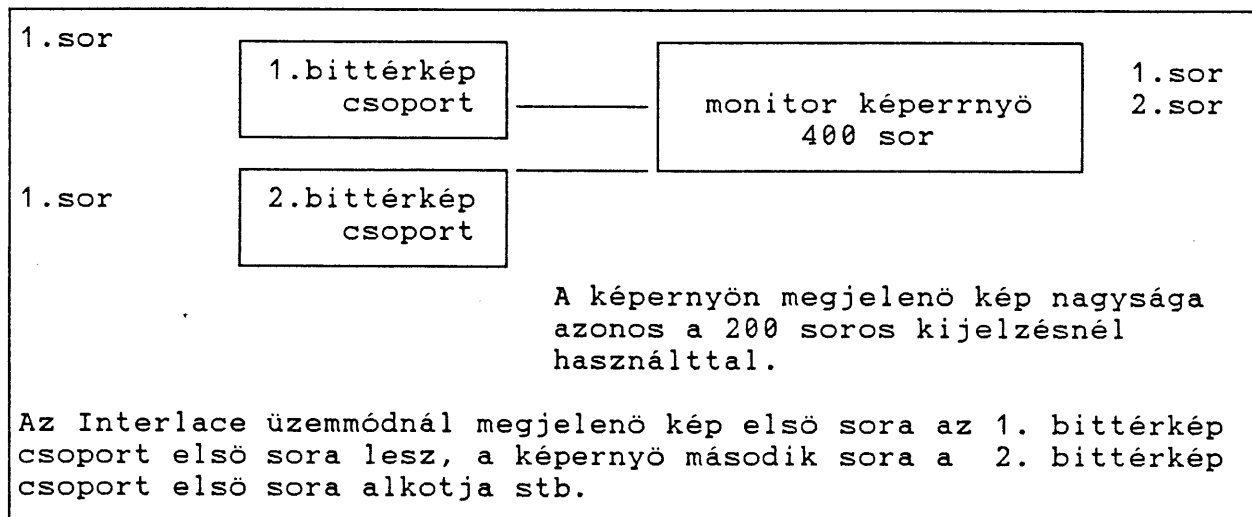
```
BPLCON0 EQU $DFF100 ; BPLCON0 címe
MOVE.W #$2200,BPLCON0
```

A BPLCON0 regiszter további bitjei a következő célokra foglaltak:

- HAM üzemmód
- DUAL PLAYFIELD üzemmód
- Composit video színek
- GENLOCK audio
- Fényceruza
- Interlace üzemmód
- Külső szinkronizálás

VISSZINTES ÉS FÜGGŐLEGES FELBONTÁS BEÁLLÍTÁSA

Egy televízió készülék csak a kisfelbontású képernyő megjelenítésére alkalmas minőségi megkötesek nélkül. Nagyfelbontású monochrome vagy színes RGB monitron alkalmazható a nagyfelbontású üzemmód is. Ha egy kép nagyfelbontásban lett megtervezve, a kisfelbontásban csak fél méretű lesz. A vízszintes felbontás a BPLCON0 15. bitje segítségével állítható be. Nagyfelbontásnál beállítandó, egyébként törölni kell. Nagyfelbontású üzemmódban maximum 4 bittérkép, ezáltal maximum 16 szín használható. Interlace módban 2 külön bittérkép csoportnak kell lennie. Az elsőben a páros sorok, a másodikban a páratlan sorok szerepelnek. A COPPER e két helyről felváltva szedi elő a sorok adatait. Az üzemmód a BPLCON0 2. bitjével (LACE) állítható. Ha a bit magas, akkor Interlace üzemmód, egyébként normál üzemmódban van.



Példa 2 nagyfelbontású, Interlace bitmező bekapcsolására:

```
BPLCON0 EQU $DFF100
MOVE.W #$A204,BPLCON0
```

MEMÓRIAFOGLALÁS A BITTÉRKEPEK SZÁMÁRA

Célszerű a bittérképeket a memóriában egymás után elhelyezni. Az egyes bittérképek kezdőcíme mindig a bittérkép bal felső sarkának címe.

Memóriaigények:

320x200 Kisfelbontású, nem interlace	8000Byte
320x400 Kisfelbontású, interlace	16000Byte
640x200 Nagyfelbontású, nem interlace	16000Byte
640x400 Nagyfelbontású, interlace	32000Byte

Az egyes bitmezők kezdőcímeit 2 - 2 16 bites regiszterrel lehet beállítani (BPLxPTH,BPLxPTL). Az "x" a kívánt bitmező sorszáma. A két regiszter összesen 19 Byte-on tárolja a címet.

Példa 2 bitmező kezdőcímének beállítására:

```
BPL1PTH EQU $DFF0E0 ; Az 1. bittérkép címének felső 3 bitje
BPL1PTL EQU $DFF0E2 ; Az 1. bittérkép címének alsó 16 bitje
BPL2PTH EQU $DFF0E4 ; A 2. bittérkép címének felső 3 bitje
BPL2PTL EQU $DFF0E6 ; A 2. bittérkép címének felső 16 bitje

MOVE.W $21000,BPL1PTH ;Az 1. bittérkép $21000-tól
MOVE.W $25000,BPL2PTH ;A 2. bittérkép $25000-tól
```

A BITMEZŐK KEZELÉSE A KÍVÁNT SZÍNEK BEÁLLÍTÁSÁHOZ

Egy bitmező esetében ha a bitmezőt nullázod, a képernyő COLOR00 (háttérszínű lesz).

```
LEA $21000,A0 ; Mutató a bittérkép kezdetére
MOVE.W #2000,D0 ; 2000 LongWord=8000Byte
LOOP: MOVE.L #0,(A0)+ ; Bitmező nullázása
SUBQ.W #1,D0 ; Számláló növelése
BNE.S LOOP ; Vissza,ha még nem kész
```

Ha egyes biteket bekapcsolunk, akkor azok COLOR01 színűek lesznek. Kettő vagy több bitmező esetén az egyes pixelek színét az összetartozó bitek kombinációja adja.

DISPLAY ABLAK MÉRETÉNEK BEÁLLÍTÁSA

A megtervezett képernyőt még el kell helyezni a kijelzőn (a TV képernyőn) is. A TV képernyőn bárhol elhelyezhető a kívánt méretű grafikus képernyő. A MODULO és a FETCH megfelelő beállításával elérhető az is, hogy teljes képernyős grafika legyen. Ekkora felbontás elérheti a 700 x 550 pixelt is. Meg kell határozni, hogy az egyes sorok hol kezdődjenek és hol végződjenek a kijelzőn. Alapesetben a képernyő 0,0 pozíciója valójában a kijelző \$81,\$2c koordinátájú pontjaival esik egybe. E beállítás a kisfelbontású nem interlace képernyőt a kijelző közepére állítja.

DISPLAY ABLAK KEZDŐ POZÍCIÓJÁNAK BEÁLLÍTÁSA

A kezdőpozíciót a DIWSTRT (Display Window Start) elnevezésű regiszter vezérli. E regiszter tartalmazza a függőleges és vízszintes pozíciókat is (VSTART,HSTART). A normál DIWSTRT \$2c81.

Példa HSTART=\$81, WSTART=\$2c

```
DIWSTRT      EQU $DFF08E
              MOVE.W #$2C81, DIWSTRT
```

A DISPLAY ABLAK VÉGPOZÍCIÓJÁNAK BEÁLLÍTÁSA

A jobb alsó sarkot deiniáljuk ezzel, a DIWSTOP regiszter segítségével. E regiszter szintén tartalmazza a kívánt függőleges és vízszintes komponenst is (VSTOP,HSTOP). A HSTOP értéke a kívánt érték-256(-\$100) legyen. A normál HSTOP \$1C1 (\$C1), a VSTOP \$F4, DIWSTOP=\$F4C1

```
DIWSTOP      EQU $DFF090
              MOVE.W #$F4C1, DIWSTOP
```

AZ ADATOK KIÍRÁSA A KÉPERNYŐRE

Miután meghatároztuk a képernyő ablak méretét meg kell adni az adatok képernyőre vitelét. Ehhez minden sorkezdet és sorvég vízszintes helyzetét kell beírni az adatkezelő regiszterekbe. Az adatkezelő regiszterek 4 pontnyi lépésközzel rendelkeznek, minden egyes pozíció négy pontnyira van az előzőtől (pl. a 0. pont a 0. pozíció, a 4. pont az 1. pozíció stb.). Az adatelérés kezdete és a kijelző ablak kezdőpontja egymással összefügg. Ajánlott, hogy az adatelérés kezdő értéke legfeljebb 16 pontnyi felbontásra korlátozódjon.(8 órajel alacsony felbontásban, 4 órajel magas felbontás esetén.) A hardwarenek szüksége van egy kis időre az első adat elérés után mielőtt ténylegesen kijelezhetné az adatot. Ennek eredményeként különbség van a képernyőabalak kezdőértéke és az adatelérés kezdete között. Alacsony felbontásnál 8.5, magasnál 4.5 órajel. A normál kisfelbontású adatelérés kezdőértéke DDFSTRT=\$0038, nagyfelbontásúé DDFSTRT=\$003c.

Az értékeket a következő módon kaphatjuk meg:

```
DDFSTRT=DDFSTOP-(8*(szavak száma 1 sorban -1)) ; kis felbontás
DDFSTRT=($81/2-4.5)=$3C
```

A normál kis felbontású DDFSTOP=\$00D0, a nagyfelbontású DDFSTOP=\$00D4.

Példa DDFSTRT=\$0038, DDFSTOP=\$00D0

```
DDFSTRT      EQU $DFF092
DDFSTOP      EQU $DDF094
              MOVE.W #$0038, DDFSTRT
              MOVE.W #$00D0, DDFSTOP
```

Azt is meg kell határozni a rendszernek, hogy a memória mely byte-jai tartoznak a kijelző vízszintes vonalához, ehhez a MODULO értékét kell meg adni. A MODULO a memória azon bytejainak a számára utal, amely a vízszintes vonal utolsó szava, és a következő sor első szava között van. Így a MODULO képessé teszi a rendszert arra, hogy lineáris formában tárolt bittérkép adatokat téglalap formában értelmezze. Az alap kijelzési módnál ahol a képernyő nagysága megegyezik a memóriáéval, a MODULO értéke 0. A bittérkép címmutatókat (BPLxPTH és BPLxPTL) a képernyőre kerülő adatok elérésére használja. Ezek dinamikus használatúak, miután az adat-elérés (megjelenítés) megkezdődött, a mutatók automatikusan növekednek, hogy a következő elérendő szóra mutassanak (2 byteonként éri el az adatokat). Amikor a sor végére jutott (ezt a DDFSTOP határozza meg) a MODULO értéke hozzáadódik a bittérkép mutatókhoz, ezzel a következő sor elejére pozicionált a rendszer. (Jegyezzük meg: a mutatók mindig páros számot tartalmaznak mert az adatokat szavanként kezeli!) Két MODULO regiszter van, a BPL1MOD a páratlan, a BPL2MOD a páros bit- mezőkhöz.

Példa (BPL1MOD = 0) :

```

BPL1MOD      EQU $DFF108
              MOVE.W #0, BPL1MOD

```

Nagyfelbontású üzemmódban a FETCH értékét meg kell növelni, mivel 40 byte helyett 80 byte-ot kell egy sorban beolvasni. Interlace üzemmódban a MODULO-n szintén változtatni kell, mivel először a páratlan sorok, majd a páros sorok elővétele történik.

```

1.sor START
2.sor START+40
3.sor START+80
4.sor START+120

```

Látható, hogy a MODULO értékét 40-re kell állítani, hogy a nem - kívánatos sorokat átugorjuk. Ha minden megfelelően be van állítva, engedélyezhető a bitmező DMA, a DMACON regiszter BPLEN (BitPlane Enable) bitje segítségével. Composit video kimeneten a színek engedélyezése nélkül, csak fekete - fehér képet kapunk. A színeket a BPLCON0 regiszter 9.bitjének beállításával engedélyezhetjük. RGB kimenetet használva nem érdekes a bit állapota.

ÖSSZEGZÉS

Egy képernyő bekapcsolása a következő folyamatból áll:

A KÉPERNYŐ ADATAINAK MEGHATÁROZÁSA

- a. Pixelek száma egy sorban
 - maximum 320 (kisfelbontású üzemmód)
 - Törölni kell a 15. bitet a BPLCON0 regiszterben
 - maximum 640 (nagyfelbontású üzemmód)
 - Be kell állítani a 15.bitet a BPLCON0 regiszterben (HIRES)

- b. Színek száma
 - A kívánt színek betöltése a színregiszterekbe
 - Bittérképek számának meghatározása
 - Bittérkép regiszterek beállítása
 - 12-14 bitek a BPLCON0 regiszterben (bittérképek száma)
 - BPLxPTH regiszterek beállítása (bittérképek kezdőcímei)
- c. Interlace ?
 - Interlace (400 sor)
 - 2.bitet beállítani a BPLCON0 regiszterben (LACE)
 - Normál (200 sor)
 - 2.bitet törölni a BPLCON0 regiszterben

2. MEMÓRIA LEFOGLALÁSA

- (byte/sor) * sorok száma * a bittérképek száma

3. DISPLAY ABLAK DEFINIÁLÁSA

- A start pozíció beírása a DIWSTRT regiszterbe
 - Vízszintes pozíció a 0-7 bitekbe
 - Függőleges pozíció a 8-15 bitekbe
- A stop pozíció beírása a DIWSTOP regiszterbe
 - Vízszintes pozíció a 0-7 bitekbe
 - Függőleges pozíció a 8-15 bitekbe

4. DATA FETCH BEÁLLÍTÁSA

- DDFSTRT regiszterbe a start pozíció
- DDFSTOP regiszterbe a stop pozíció

5. MODULO MEGHATÁROZÁSA

- Interlace üzemmódban 40
- Normál üzemmódban 0

6. COPPER UTASÍTÁS LISTA MEGÍRÁSA, LISTA ELINDÍTÁSA

7. SZINES ÜZEMMÓD ENGEDÉLYEZÉSE

A 9.bit beállítása a BPLCON0 regiszterben
(Composite monitor esetén)

Két komplett példaprogram az eddig elhangzottak bemutatására:

Az első létrehoz egy 320 * 200 pixel felbontású, egy bitmezőt használó képernyőt a \$21000 címtől.

A COPPER lista kezdőcíme \$20000

```

CUSTOM      EQU      $DFF000
BPLCON0     EQU      $100
BPLCON1     EQU      $102
BPLCON2     EQU      $104
BPL1MOD     EQU      $108
DDFSTRT     EQU      $092
DDFSTOP     EQU      $094
DIWSTRT     EQU      $08E
DIWSTOP     EQU      $090
VPOSR       EQU      $004
COLOR00     EQU      $180
COROR01     EQU      $182
COLOR02     EQU      $184
COLOR03     EQU      $186
DMACON      EQU      $096
COP1LCH     EQU      $080

; A COPPER lista kezdőcíme $20000
LEA          CUSTOM,A0          ; A0-ba a báziscímet
MOVE.W       #$1200,BPLCON0(A0) ; 1.bitmező színes
MOVE.W       #0,BPLCON1(A0)    ; vízszintes scroll
                                   értéke
MOVE.W       #0,BPL1MOD(A0)    ; MODULO=0
MOVE.W       #$0038,DDFSTRT(A0) ; FETCH start
MOVE.W       #$00D8,DDFSTOP(A0) ; FETCH stop
MOVE.W       #$2C81,DIWSTRT(A0)
MOVE.W       #$F4C1,DIWSTOP(A0)
MOVE.W       #$0F00,COLOR00    ; háttérszín vörös
MOVE.W       #$0FF0,COLOR01    ; 1.szín sárga

; Bitmező feltöltése #$ff00ff00 mintával (csíkozás)
MOVE.L       #$21000,A1        ; mutató a bitmező
                                   kezdetére
MOVE.L       #$FF00FF00,D0     ; mintázat
MOVE.L       #2000,D1          ; 2000
                                   LongWord=8000Byte
LOOP:        MOVE.L       D0,(A1)+ ; írás
SUBQ.W       #1,D1             ; a számláló növelése
BNE.S        LOOP              ; vissza amíg nincs
                                   kész

; A COPPER lista kreálása $20000-re
MOVE.L       #$20000,A1        ; mutató a kezdetre
LEA          COPPERL,A2        ; mutató a lista
                                   adataira
CLOOP:       MOVE.L       (A2),(A1)+ ; egy LongWord
                                   áthelyezése
CMPI.L       $FFFFFFFE,(A2)+    ; az utolsó LongWord
BNE.S        CLOOP

; COPPER lista kezdőcímének átadása a COPPER-nek
MOVE.L       #$20000,COP1LCH(A0)
MOVE.W       COPJMP1(A0),D0     ; COPPER start

; DMA start
MOVE.W       #$8380,DMACON(A0) ; bitmező és COPPER
                                   engedélyezése
BRA          .....            ; további rutinok

; COPPER lista adatok
COPPERL:     DC.W         $00E0,$0002 ; BPL1PTH
DC.W         $00E2,$1000 ; BPL1PTL
DC.W         $FFFF,$FFFE ; lista vége

```

A második példa egy nagyfelbontású, Interlace üzemmódú, 2 színű képernyőt kapcsol be:

```

LEA        CUSTOM,A0
MOVE.W    #$9204,BPLCON0(A0) ; 1 bitmező, színes
MOVE.W    #0,BPLCON1(A0)    ; vízszintes scroll
                                   értéke
MOVE.W    #80,BPL1MOD(A0)    ; MODULO=0
MOVE.W    #$003C,DDFSTRT(A0) ; FETCH start
MOVE.W    #$00D4,DDFSTOP(A0) ; FETCH stop
MOVE.W    #$2C81,DIWSTRT(A0)
MOVE.W    #$F4C1,DIWSTOP(A0)
MOVE.W    #$000F,COLOR00(A0) ; Háttérszín kék
MOVE.W    #$0FFF,COLOR01(A0) ; COLOR01 fehér

; Bitmező $20000-re
LEA        $20000,A1          ; Mutató a bitmezőre
LEA        CHARLIST,A2        ; Mutató a mintára
MOVE.W    #400,D1              ; 400 sor
MOVE.W    #20,D0               ; 20 LongWord/sor
L1:        MOVE.L    (A2),(A1)+
SUBQ.W    #1,D0
BNE.S     L1
MOVE.W    #20,D0
ADDQ.L    #4,A2
CMPI.L    #$FFFFFFFE,(A2)
BNE.S     L2
LEA        CHARLIST,A2

L2:        SUBQ.W    #1,D1          ; Sorszámláló
                                   növelése
BNE.S     L1                    ; Ha van még sor, vissza

; DMA start
MOVE.W    #$8300,DMACON(A0) ; DMA engedélyezése
Eddig a példa nem tartalmaz COPPER listát. A következő részben
lévő LOOP (hurok) a függőleges visszafutás pillanatára vár. Amikor
ez a pillanat eljött, ellenőrzi a LOF bitet a VPOSR regiszterben.
Ha a LOF=0, akkor a bitmező mutatót $20050-re állítja. Ha a LOF=1,
akkor pedig $20000-re.

VLOOP:     MOVE.W    INTREQR(A0),D0
AND.W     #$8020,D0
BEQ.S     VLOOP
MOVE.W    #$0020,INTREQ(A0)
MOVE.W    VPOSR(A0),D0
BPL.S     VL1
MOVE.W    #$20000,BPL1PTH(A0)
BRA.S     VLOOP
VL1:       MOVE.L    #$20050,BPL1PTH(A0)
BRA.S     VLOOP

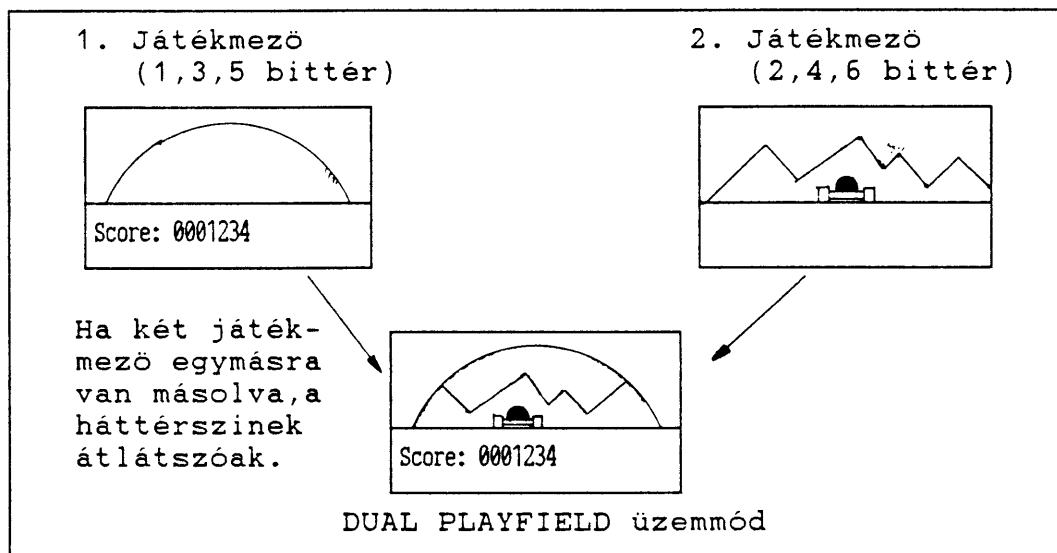
; Karakter lista
DC.L      $18FC3DF0,$3C6666D8,$3C66C0CC,$667CC0CC
DC.L      $7E66C0CC,$C36666D8,$C3FC3DF0,$00000000
DC.L      $FFFFFFFF

```

A DUAL PLAYFIELD KÉPERNYŐ KEZELÉSE

Az egyes képernyőkön bármit megváltoztathatsz, a változás nincs befolyással a másik képre. Az üzemmód a következő korlátozások kivételével megegyezik a normál üzemmóddal:

- a képernyők egyenként maximum 3 bitmezőt használhatnak
- a színek száma maximum 7, plusz az átlátszó háttérszín
- a két képernyő külön színregisztereket használ
- be kell kapcsolni a DUAL bitet a BPLCON0 regiszterben



Az átlátszó színt az első képernyőnél a COLOR00 színregiszter, a többi színt a COLOR01-COLOR07 regiszterek, a második képernyőnél a háttérszínt a COLOR08 színregiszter tartalmazza, a többit pedig a COLOR09- OLOR15 színreiszterek.

BITMEZŐ HOZZÁRENDELÉS DUAL PLAYFIELD ÜZEMMÓDBAN

A 3 páratlan sorszámú bitmező (1,3,5) alkotja az első képernyőt a párosak (2,4,6) pedig a másodikat. Ha nagyfelbontású Üzemmodot akarsz használni, akkor csak 2-2 bitmező használható (4-4 szín) ! Az első képernyőnél az 5. bitmező bitje a legnagyobb helyiértékű.

A színek hozzárendelése:

BIT KOMBINÁCIÓ	SZÍN
000	átlátszó
001	COLOR01
010	COLOR02
011	COLOR03
100	COLOR04
101	COLOR05
110	COLOR06
111	COLOR07

A második képernyőnél a 6. bitmező bitje a legnagyobb helyiértékű.

A színek hozzárendelése:

000	átlátszó
001	COLOR09
010	COLOR10
011	COLOR11
100	COLOR12
101	COLOR13
110	COLOR14
111	COLOR15

Nagyfelbontású üzemmódban az első képernyőnél:

00	átlátszó
01	COLOR01
10	COLOR02
11	COLOR03

Nagyfelbontású üzemmódban a második képernyőnél:

00	átlátszó
01	COLOR09
10	COLOR10
11	COLOR11

DUAL PLAYFIELD PRIORITÁSOK ÉS VEZÉRLÉSEK

A két képernyő prioritása felcserélhető. Ha a BPLCON2 regiszter 6. bitje (PF2PRI) magas, akkor a 2. képernyő lesz elől, ha 0 akkor az első.

A két képernyő vezérléséhez a következőket kell még tudnod:

- a két képernyő lehet egymástól eltérő méretű, és felbontású is
- a két képernyő egymásól függetlenül is scrollozható

FIGYELEM! Különös gondot kell fordítani ha csak az egyik képernyőt scrollozzuk és a másik álló helyzetben marad. Amikor alacsony felbontású képernyőt scrollozunk egy szóval többet kell elérni (nagy felbontás esetén kettővel), azonban csak egy adatelérés kezdet és egy adatelérés vég regiszter van, ezért ezt mindkét játékmező használja. Be kell állítani a kezdő és vég regisztereket a scrollozott értékre, majd a MODULO -t és a bittérkép mutatókat arra a játékmezőre, amelyet nem akarunk mozgatni. Ezenkívül alacsony felbontás esetén a mozditandó képernyő MODULO-t 2-vel, magas felbontás esetén 4-gyel kell csökkenteni.

A DUAL PLAYFIELD ÜZEMMÓD BEKAPCSOLÁSA

Ha a BPLCON0 regiszter 10. bitje (DBLPE) magas, akkor az üzemmód aktív.

ÖSSZEGZÉS

A kezdeti lépések megegyeznek a normál képernyő bekapcsolásával, plusz a következők:

- A színek betöltése a színregiszterekbe
0-7 regiszterek az 1. képernyő színei
9-15 regiszterek a 2. képernyő színei
- Bitmező szükséglet
1,3,5 csak az 1. képernyőhöz
2,4,6 csak a 2. képernyőhöz
- MODULO beállítás
BPL1MOD az 1. képernyőhöz
BPL2MOD a 2. képernyőhöz
- Prioritás definiálása
Ha a 2. képernyő prioritása nagyobb, akkor a
BPLCON2 6. bitje legyen 1
- DUAL PLAYFIELD üzemmód bekapcsolása
BPLCON0 regiszter 10. bitje legyen 1

BITMEZŐK ÉS DISPLAY ABLAKOK MAXIMÁLIS MÉRETBEN

Az előbbieken ismertetett szabványos méretű bitmezők mellett használható egyéb, azoktól eltérő is. A következőkben ismertetjük hogyan definiálható és használható egy ilyen nagy képernyő (BIG PICTURE), és hogyan mozgatható az ablak a BIG PICTURE előtt.

A MODULO MEGHATÁROZÁSA

Ha a képernyő szélesebb, mint a display ablak, akkor a MODULO-nak más értéket kell adni, hogy a FETCH kívánt mennyiségű adatot olvasson egy sorhoz. Ha a sor 80 byte-nál szélesebb, BIG PICTURE-ről beszélünk.

A MODULO FELADATA

Ha a kép a memóriában nagyobb mint a megjelenő képernyő, létre kell hozni egy MODULO értéket, hogy a megfelelő szavakat érje el. Pl.: egy 640 képpontnyi rajznak csak egy szabványos 320-as részét szeretnénk kijelezni, ebben az esetben minden sor 40 byte hosszúságú, a memóriában viszont 80. Az első sor végének elérése után a MODULO értéke hozzáadódik (40 byte) a sormutatókhoz, így valóban a második sor elejére jutunk.

A MEMÓRIA LEFOGLALÁSA

Nagyobb memóriát foglaló képekhez nagyobb helyre van szükség. Az alábbi formula segítségével kiszámítható a képernyő memóriaigénye:
byte soronként * sorok száma * bitmezők száma
Pl.: 2 bitmező, 120 byte soronként: $120 * 200 * 2 = 48000$ byte memória

A KÉPERNYŐ ABLAK KEZDŐHELYZETÉNEK KIVÁLASZTÁSA

A képernyő ablak kezdő pozícióját a bal felső sarok vízszintes és függőleges koordinátáinak meghatározása jelenti. Egy regiszter (DIWSTRT) tartalmazza mind a vízszintes, mind a függőleges kezdőértéket (a vízszintes - HSTART, a függőleges - VSTART). A HSTART nyolc bitje adja az első 256 helyzetet a bal oldaltól számítva, a VSTART nyolc bitje az első 256 helyzetet adja a képernyő tetejétől számítva. A képernyőablakot úgy állítjuk kezdőpozícióba, hogy a DIWSTART 0 - 7 bitjeibe a HSTART értékeit, a 8 - 15 bitjeibe pedig a VSTART értékeket töltsük.

A KÉPERNYŐABLAK VÉGHELYZETÉNEK KIVÁLASZTÁSA

A véghelyzet a jobb alsó sarok vízszintes és függőleges koordinátáinak beírásával határozhatjuk meg, ezeket a HSTOP és VSTOP állítja a DIWSTOP regiszterben (a DIWSTOP 0 - 7 bitjei a VSTOP-ot, a 8 - 15 bitjei a HSTOP-ot tartalmazzák.)

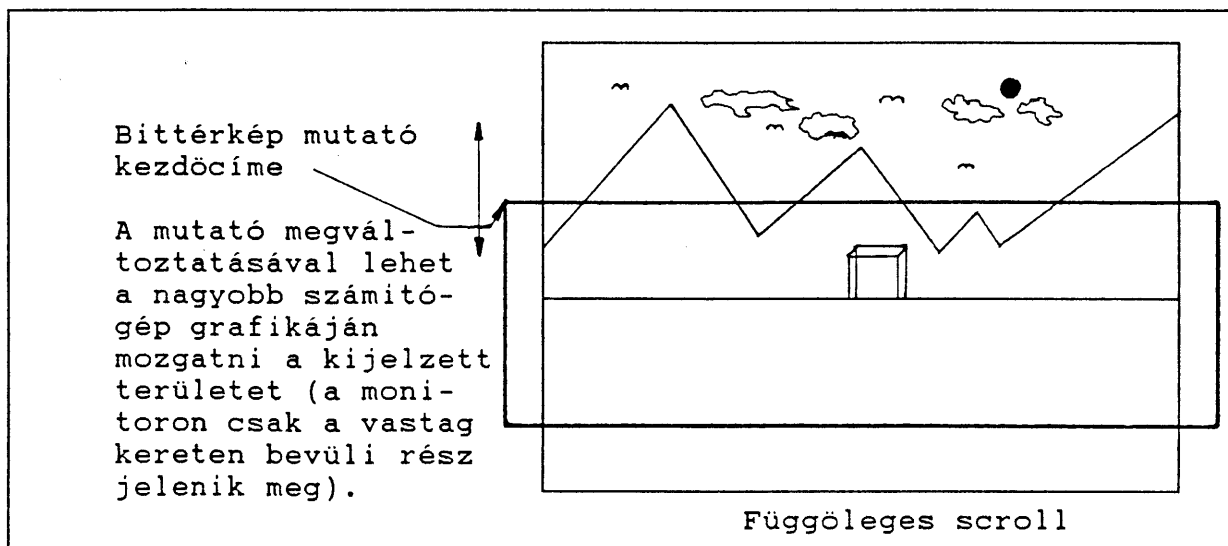
A MAXIMÁLIS DISPLAY ABLAK MÉRET

A maximális méretet a sorok és oszlopok maximális száma határozza meg. Függőlegesen korlátot képez az, hogy ada nem jeleníthető meg a függőleges sugárviasszafutás ideje alatt. Ez az időintervallum a 0 - 19 rastersorok alatt van. E 20 sor miatt csak 242 sor jeleníthető meg függőlegesen (484 sor Interlace üzemben). Vízszintesen hasonló a helyzet. A hardver által kötött a bal szélső pozíció min. \$18 (DFFSTRT), a jobb szélső pozíció max. \$d8 (DFFSTOP). Kis felbontásnál maximum 25 word (50 byte) lehet a FETCH, nagyfelbontásnál maximum 49 word (98 byte). E korlátok miatt kis felbontásban 376 pixel (23.5 word), nagyfelbontásban 752 pixel a max. vízszintes felbontás.

A KÉPERNYŐK SCROLLLOZÁSA

Ha a képernyőt scrollozni kívánod nagyobbra kell megtervezni, mint a display ablakot. DUAL PLAYFIELD üzemmódban a két képernyő külön-külön is scrollozható.

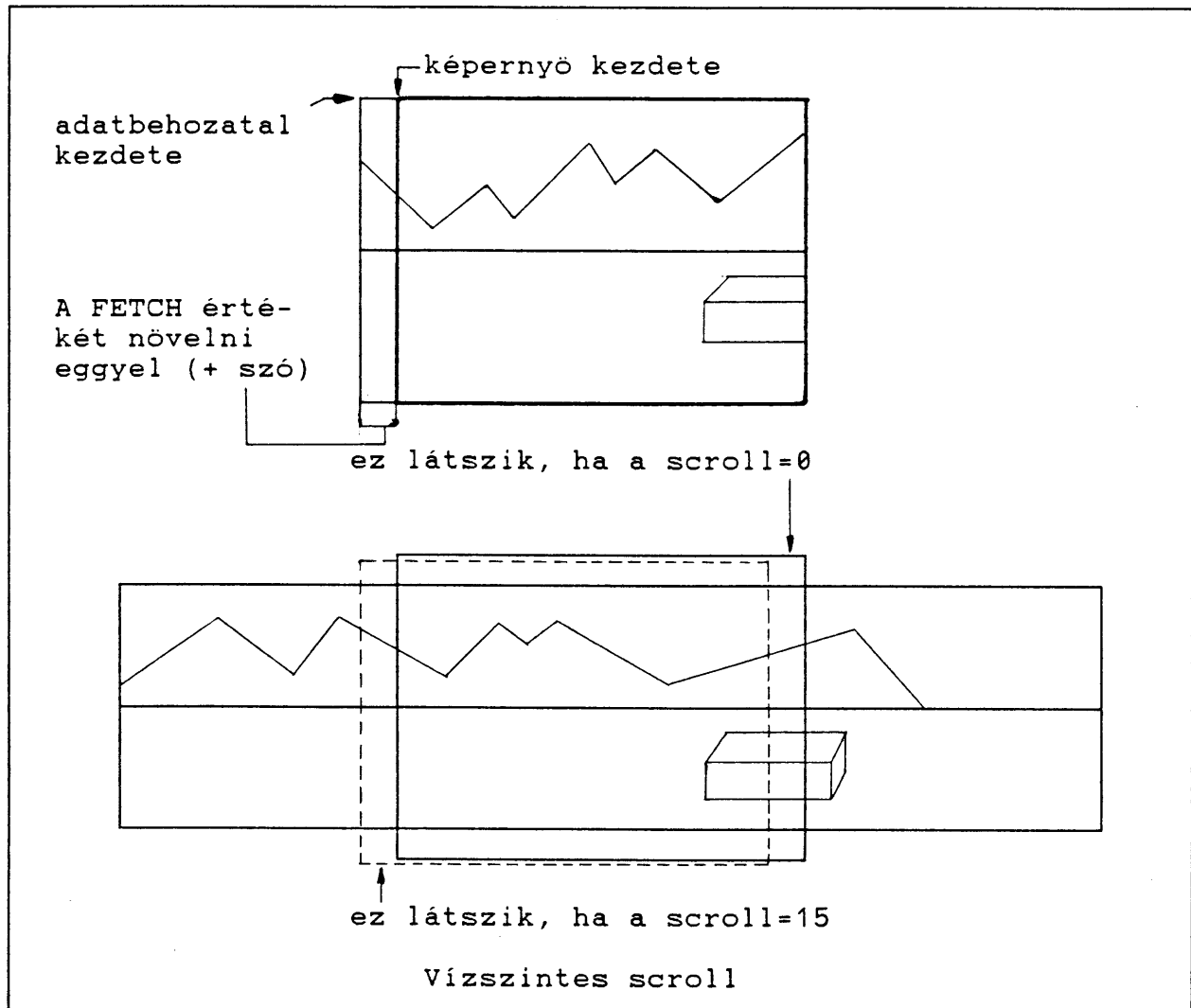
FÜGGŐLEGES SCROLL



A függőleges scroll elég egyszerűen hajtható végre. Csak a bitmezők mutatóit kell megváltoztatni úgy, hogy a kép egy sorral elmozduljon. Ekkor természetesen a grafika függőleges méretének nagyobbnak kell lennie, mint az ablak. (BIG PICTURE)

Egy scroll lépés minimum a következő lehet: 200 soros felbontásnál a kép 1/200-ad része, 400 soros felbontásnál a kép 1/400-ad része. Természetesen nagyobb lépések is előidézhethők, ha csak egy sorral növeljük (csökkentjük) a mutatókat. Ha a kezdőcímek nőnek, a kép felfelé mozdul.

VÍZSZINTES SCROLL



A vízszintes scroll más módon idézhető elő, mint a függőleges. A FETCH értékét növelni kell egyvel (így egy sorban 1 word-el, 16 bittel többet olvasunk be), valamint módosítani kell a display időzítését. A pluszként beolvasott word bitjei lépkednek be a display ablak látható területére. Maximum 16 bit lehet a scroll mértéke (csak ennyi bit van a plusz word-ben). Nagyfelbontású üzemmódban a scroll 2 pixelenként léptethető. A FETCH időtartama alatt a sprite-ok tiltva vannak !

A scrollhoz a következők szükségesek:

- a bitmezőket megfelelő szélességűre kell állítani
- a DATA FETCH regiszter tartalmát növelni kell eggyel
- az időzítő biteket be kell állítani
- a MODULO átállítása
- a COPPER lista módosítása a függőleges visszafutás ideje alatt

DATA FETCH BEÁLLÍTÁSA A VÍZSZINTES SCROLLHOZ

A normál FETCH értéke nem scrollozó képhez, \$38. Ha vízszintesen akarsz, a FETCH értékét állítsd egy word-el korábbra. (DDFSTRT=\$30) Ekkor a 7. sprite tiltott! A DDFSTOP értéke maradjon a normális értéken (\$00d0). A FETCH minden képernyőre kihat!

MODULO BEÁLLÍTÁSA A VÍZSZINTES SCROLLHOZ

Az adatelérés kezdete scrollozás nélkül általában \$38. Ha vízszintes scrollozást alkalmazunk, akkor az adatelérés egy szóval korábban kezdődjön (DDFSTRT=\$30, a DDFSTOP változatlan marad). A MODULO értéke két szóval kevesebb, mint az utolsó elérendő szó és a következő sor első szava közötti különbség.

Pl.: 40 byte-os képernyőt használunk egy 80 byte-os memóriaképből, akkor 42 byte hosszúságú adatok tartoznak egy sorhoz a scrollozás esetén. Ilyenkor a MODULO értéke 36 byte. Mindkét bittérkép csoport (DUAL PLAYFIELD) MODULO értékét a BLTxMOD regiszterben állíthatjuk be. Vízszintes scrollozásnál a késleltetés értékét a BPLCON1 regiszter 0 - 7 bitjei tartalmazzák, mindkét játéklemező késleltetése külön állítható. A 0 - 3 bitek az 1, 3, 5 bittérképeket, a 4 - 7 bitek a 2, 4, 6 bittérképeket állítják. Mindkét játéklemező késleltetését be kell írni annak ellenére, hogy esetleg az egyiket nem használjuk (ebben az esetben a 0 - 3 és 3 - 7 bitek értékei azonosak legyenek)!

A következő példa a vízszintes scroll időzítést 7-re állítja minden képernyőn:

```
BPLCON1      EQU      $DDF102      ; Vízszintes scroll regiszter
               MOVE.W   #$77,BPLCON1
```

ÖSSZEFOGLALÁS

Egy scrollozandó játéklemező beállításának lépései azonosak egy egyszerű játéklemezőével, kivéve a következőket:

- Meg kell határozni az adatelérést, ez vízszintes soronként egy plusz szót jelent, így 16 ponttal korábban kell kezdeni.
- Meg kell határozni a MODULO értékét. A MODULO két szóval kisebb mint hagyományos esetben.
- Függőleges scrollhoz a növekmény nagyságával kell változtatni a bittérkép mutatókat (ezt a függőleges késleltetési idő alatt kell megtenni).

A HOLD AND MODIFY ÜZEMMÓD

A HAM használja mind a 6 bitmezőt. Az 5, 6 bitmező írja elő a módosítást. Az üzemmód a 0-15 színregisztereket használhatja. A 4 alsó bitmező ezeket választhatja ki, ha az 5, 6 bitmezők tartalma 00. Ha csak 5. bitmezőt használunk, a 6. bitmező hiányzó tartalmát a gép 0-nak veszi.

- Ha az 5, 6 bitmezők tartalma 01, akkor a kék és zöld színösszetevők megegyeznek az előző pixelével a vörös színösszetevők, pedig az 1 - 4 bitmezők bitjei határozzák meg.
- Ha az 5, 6 bitmezők tartalma 10, akkor a zöld és vörös színösszetevők megegyeznek az előző pixelével, a kék színösszetevőt pedig az 1 - 4 bitmezők bitjei határozzák meg.
- Ha az 5, 6 bitmezők tartalma 11, akkor a kék és vörös színösszetevők megegyeznek az előző pixelével, a zöld színösszetevőt pedig az 1 - 4 bitmezők bitjei határozzák meg.

A BPLCON0 regiszter 11. bitje (HOMOD) kapcsolja be az üzemmódot.

Az üzemmód bekapcsolása a következőképpen történik:

- HOMOD bit (11.) magasra állítása
- DBLPE bit (10.) törlése (DUAL PLAYFIELD nem engedélyezett)
- HIRES bit (15.) törlése (HIRES nem engedélyezett)
- BPU2, BPU1, BPU0 bitek (14-13-12) beállítása
101 - 5 bitmező
111 - 6 bitmező

A következő példa generál egy 6 bitmezős képernyőt. Az összes színregisztert feketére állítjuk, a színeket a HAM üzemmód segítségével generáljuk.

; Először a vezérlő regisztereket állítjuk be
;

```
LEA      CUSTOM,A0
MOVE.W   #$6A00,BPLCON0(A0)
MOVE.W   #0,BPLCON1(A0)
MOVE.W   #0,BPL1MOD(A0)
MOVE.W   #0,BPL2MOD(A0)
MOVE.W   #$0038,DDFSTRT(A0)
MOVE.W   #$00D0,DDFSTOP(A0)
MOVE.W   #$2C81,DIWSTRT(A0)
MOVE.W   #$F4C1,DIWSTOP(A0)
```

```

;
; Az összes színregiszter feketére állítása
;
                MOVE.W    #32,D0
                LEA        CUSTOM+COLOR00,A1
CREGLOOP       MOVE.W    #0000,(A1)+
                SUBQ.W    #1,D0
                BNE.S     CREGLOOP
;
; A 6 bitmező FILL-ezése
;
                MOVE.W    #2000,D0
                MOVE.W    #21000,A1
                MOVE.W    #23000,A2
                MOVE.W    #25000,A3
                MOVE.W    #27000,A4
                MOVE.W    #29000,A5
                MOVE.W    #2B000,A6
FPLLOOP        MOVE.L    #55555555,(A1)+
                MOVE.L    #33333333,(A2)+
                MOVE.L    #0F0F0F0F,(A3)+
                MOVE.L    #00FF00FF,(A4)+
                MOVE.L    #FFFFFFF,(A5)+
                MOVE.L    #00000000,(A6)+
                SUBQ.W    #1,D0
                BNE.S     FPLLOOP
;
; COPPER lista $20000-re
;
                MOVE.L    #20000,A1
                LEA        COPPERL,A2
CLOOP:         MOVE.L    (A2),(A1)+
                CMPI.L    #FFFFFFFE,(A2)+
                BNE.S     CLOOP
;
; COPPER lista címének átadása a COPPER-nek
;
                MOVE.L    #20000,COP1LCH(A0)
                MOVE.W    COPJMP1(A0),D0
;
; DMA Start
;
                MOVE.W    #8380,DMACON(A0)
                BRA.S     .....
;
; COPPER lista 6 bimezőhöz. Első bitmező $21000-re, 2.bitmező
; $23000-re, 3.bitmező $25000-re, 4.bitmező $27000-re, 5.bimező
; $29000-re, 6.bimező $2b000-ra
;
COPPERL:       DC.W      $00E0,$0002
                DC.W      $00E2,$1000
                DC.W      $00E4,$0002
                DC.W      $00E6,$3000
                DC.W      $00E8,$0002
                DC.W      $00EA,$5000
                DC.W      $00EC,$0002
                DC.W      $00EE,$7000
                DC.W      $00F0,$0002
                DC.W      $00F2,$9000
                DC.W      $00F4,$0002
                DC.W      $00F6,$B000
                DC.W      $FFFF,$FFFE

```

KÜLSŐ VIDEO FORRÁS JELEINEK HASZNÁLATA

A GENLOCK elnevezésű külön beszerezhető áramkör segítségével az AMIGA képe rákeverhető egy külső forrás (kamera, video magnó, stb.) jelére. A GENLOCK a háttérszint változtatja a külső jelnek megfelelően, így a háttérszínből áll elő a külső kép.

KÉPERNYŐ REGISZTEREK ÖSSZEFOGLALÁSA

BPLCON0	bit 0	Nem használt
	bit 1	ERSY (External Synchronisation Enable) külső videó szinkronizáció engedélyezése 1 - engedélyezett, 0 - tiltott
	bit 2	LACE (Interlace Enable) interlace üzemmód engedélyezése 1 - be, 0 - ki
	bit 3	LPEN (LightPen Enable) fényceruza engedélyezése 1 - be, 0 - ki
	bit 4-7	Nem használt
	bit 8	GAUD (Genlock Audio Enable) Genlock audio engedélyezése 1 - be, 0 - ki
	bit 9	COLOR_ON (Color Enable) Composite színes üzemmód engedélyezése 1 - be, 0 - ki
	bit 10	DBPLF (Dual Playfield Enable) Dual playfield üzemmód engedélyezése 1 - be, 0 - ki
	bit 11	HOMOD , HAM üzemmód engedélyezése 1 - be, 0 - ki
	bit 12	BPU0
	bit 13	BPU1
	bit 14	BPU2
		000 - csak háttérszín
		001 - 1 bitmező, PLANE 1
		010 - 2 bitmező, PLANE 1-2
BPLCON1		011 - 3 bitmező, PLANE 1-3
		100 - 4 bitmező, PLANE 1-4
		101 - 5 bitmező, PLANE 1-5
		110 - 6 bitmező, PLANE 1-6
		111 - Nem használt
	bit 15	HIRES 1 - be(640 pixel), 0 - ki(320 pixel)
	bit 3-0	PF1H 1.képernyő időzítés vízszintes scrollhoz
	bit 7-4	PF2H 2.képernyő időzítés vízszintes scrollhoz
BPLCON2	bit 15-8	Nem használt
	bit 0-5	Képernyő Sprite prioritás
	bit 6	PF2PRI képernyő prioritás Dual-Playfield üzemmódban 1 - a 2.képernyő prioritás a nagyobb 0 - az 1.képernyő prioritása a nagyobb
	bit 7-15	Nem használt

DDFSTR	Data Fetch Start bit 0-2 Nem használt bit 3-7 Pixel pozíció (H4-H8) bit 8-15 Nem használt
DDFSTOP	Data Fetch Stop bit 0-2 Nem használt bit 3-7 Pixel pozíció (H4-H8) bit 8-15 Nem használt
BPLxPTH	Bitmező mutatók felső 3 bitje (x=az aktuális bitmező sorszáma)
BPLxPTL	Bitmező mutatók alsó 16 bitje
DIWSTR	Display ablak start bit 15-8 VSTART (V7-V0) bit 7-0 HSTART (H7-H0)
DIWSTOP	Display ablak stop bit 15-8 VSTOP (V7-V0) bit 7-0 HSTOP (H7-H0)
BPL1MOD	1. bitmező MODULO (a páratlan sorszámu bitmezökhöz) 2. bitmező MODULO (a páros sorszámu bitmezökhöz)

A SZÍNEK ÖSSZEGZÉSE

Minden színregiszter csak írható ! A színregiszterek felépítése:

bit 15-12 Nem használt
bit 11-08 Vörös színösszetevő
bit 07-04 Zöld színösszetevő
bit 03-00 Kék színösszetevő

Néhány színvariáció:

\$FFF Fehér	\$1FB Világoszöld
\$F00 Vörös	\$6CE Világoskék
\$F90 Narancs	\$FB0 Arany narancs
\$91F Bíbor	\$FF0 Ciromsárga
\$2C0 Sötétzöld	\$C88 Barna
\$DB9 Sárgásbarna	\$CCC Világosszürke
\$0DB Aqua	\$999 Középszürke
\$D00 Téglavörös	\$6FE Égszínkék
\$F80 Vöröses narancs	\$0FF Kék
\$06D Sötétkék	\$FD0 Kadmium sárga
\$C1F Ibolya	\$F1F Bíborvörös
\$FAC Rózsaszín	\$0F0 Zöld
\$0BB Kékeszöld	

SZÍNKIVÁLASZTÁS KISFELBONTÁSÚ ÜZEMMÓDBAN

One Playfield		Dual Playfield	
Normál üzemmód	HAM üzemmód	Színregiszter	
5,4,3,2,1 bitmezők	4,3,2,1 bitmezők	sorszama	
		Playfield 1	
		5,3,1 bitmezők	
00000	0000	000	0*
00001	0001	001	1
00010	0010	010	2
00011	0011	011	3
00100	0100	100	4
00101	0101	101	5
00110	0110	110	6
00111	0111	111	7
		Playfield 2	
		6,4,2 bitmezők	
01000	1000	000**	8
01001	1001	001	9
01010	1010	010	10
01011	1011	011	11
01100	1100	100	12
01101	1101	101	13
01110	1110	110	14
01111	1111	111	15
10000	.	.	16
10001	.	.	17
10010	.	.	18
10011	.	.	19
10100	Nem	Nem	20
10101	használt	használt	21
10110	ebben	ebben	22
10111	az	az	23
11000	üzemmódban	üzemmódban	24
11001	.	.	25
11010	.	.	26
11011	.	.	27
11100	.	.	28
11101	.	.	29
11110	.	.	30
11111	.	.	31

* A 0.színregiszte a hátérszint definiálja

** Az álatszó szín a 8.színregiszter

SZÍNKIVÁLASZÁS HAM ÜZEMMÓDBAN

6. bitmező	5. bitmező	Színregiszterek normális használata	
0	0 Normál üzemmód		
0	1 Zöld és Vörös tartás	B=a 4-1 bitmezők tartalma	
1	0 Zöld és Kék tartás	R=a 4-1 bitmezők tartalma	
1	1 Kék és Vörös tartás	G=a 4-1 bitmezők tartalma	

SZÍNKIVÁLASZTÁS HIRES ÜZEMMÓDBAN

One Playfield 4,3,2,1 bitmezők	Dual Playfield	Színregiszter sorszáma
Playfield 1		
3,1 bimezök		
0000	00*	0**
0001	01	1
0010	10	2
0011	11	3
0100	.	4
0101	Nem	5
0110	használt	6
0111	.	7
Playfield 2		
4,2 bitmezők		
1000	00*	8
1001	01	9
1010	10	10
1011	11	11
1100	.	12
1101	Nem	13
1110	használt	14
1111	.	15

* Atlátszó szín

** A 0.színregiszter a háttérszint definiálja

A SPRITE

A SPRITE olyan hardware objektum, amelyet a képernyőtől teljesen függetlenül mozgathatunk a képernyő előtt, és melynek alakja és színei szabadon, a saját igényeknek megfelelően definiálható. 8 sprite használható egy - egy DMA csatorna segítségével. Egy normál sprite 16 pixel széles lehet, magassága nincs korlátozva. A színek száma 4, ebből egy az átlátszó.

A SPRITE ADATAI

Az első lépés a sprite(ok) megtervezése, majd az őket leíró adatstruktúrák elhelyezése a memóriában. A tervezés folyamán a következőket kell figyelembe venni:

- A szélesség maximum 16 pixel
- A függőleges méret korlátlan
- Maximum 3 szín + az átlátszó
- A képernyő pozíció tetszőleges

POZÍCIONÁLÁS

Egy sprite képernyőn elfoglalt pozícióját X, Y koordinátákkal adhatjuk meg. A 0, 0 pozíció a képernyő bal felső sarka. A koordináták a sprite bal felső pixelének koordinátáját írják le. A pozíciók függetlenek a közben alkalmazott egyéb képernyő üzemmódoktól és felbontásoktól. A sprite terület it is (mint C64-nél) látható és nem látható területekre osztódik, de itt a nem látható terület megszüntethető a display ablak teljes képernyőre való kiterjesztésével.

VÍZSZINTES POZÍCIONÁLÁS

A vízszintes pozíció (X) a 0-447 értékeket veheti fel. Alapállapotú display ablak mellett a látható terület 64 - 383-ig terjed. Egy sprite X pozíciója a legfelső sor bal szélső pixelének pozícióját írja.

FÜGGŐLEGES POZÍCIONÁLÁS

A függőleges pozíció (Y) 0 - 262-ig terjed. Alapállapotú display ablak mellett a látható terület 44 - 243 -ig terjed. Egy sprite Y pozíciója a legfelső sor pozícióját írja le.

EGY SPRITE MEGTERVEZÉSE

Egy sprite pixeleinek színe a képernyő színeinek előállításához hasonlóan, itt is több (2) bitmező használatával történik. Egy sprite sort két word (2 * 16 bit) ír le. Mint már a második fejezetből kiderült, e két word-dal 4 színregiszter írható elő. A 8 sprite párokat alkot. Egy - egy pár azonos 4 - 4 színregisztert használ. A 00 bitpárok speciális jelentésűek, mivel ez a mintázat írja elő az átlátszó színt. A sprite -ok között fix prioritás van kiosztva, ez nem változtatható meg. A sprite-ok és a képernyők- (Dual Playfield) között viszont felcserélhető a prioritás. A sprite -ot úgy tervezzük meg, hogy először a kívánt színeket számokkal jelöljük 0-3-ig, majd következő lépésként a számokat átkonvertálva bináris számokká, megállapítható az adott sor leíró két 16 bites word. A sorokat leíró word-öket kell a későbbiekben kell eltárolni.

AZ ADATSTRUKÚRA FELÉPÍTÉSE

A struktúra 16 bites word-ök sorozata a memóriában. A word-ök némelyike pozícionálási, vezérlési információkat, a többi az alak és szín információkat hordozza. A struktúra csak az alsó 512 KByte -ban helyezhető el, mivel a DMA itt fér hozzá.

A struktúra felépítése:

1. vezérlő word SPRxPOS

bit 15-8 függőleges pozíció (VSTART) legkisebb 8 bitje
bit 7-0 vízszintes pozíció (HSTART) legnagyobb 8 bitje

2. vezérlő word SPRxCTL

bit 15-8 függőleges stop pozíció (VSTOP) legkisebb 8 bitje
bit 7 sprite-ok összefüzésekor használt
bit 6-3 nem használt (legyen mindig 0)
bit 2 függőleges pozíció (VSTART) legmagasabb bitje
bit 1 függőleges stop pozíció (VSTOP) legmagasabb bitje
bit 0 vízszintes pozíció (HSTART) legkisebb bitje

$(VSTOP - VSTART + 1) = A$ sprite sorainak száma

További word-ök:

A sprite színeit leíró wordpárok sorozata.

Utolsó word:

Az adatstruktúra végét jelző word pár. (\$00000000)

A SZÍNEKET LEÍRÓ WORD-ÖK

Magas, alacsony word sorrendben követik egymást. Sprite-ok használata esetén az utolsó 16 színregisztert használjuk a sprite színek megadására. Például a 0. sprite a 17 - 19 színregisztereket használhatja:

Bitminta	Szín
-----	-----
00	átlátszó
01	17
10	18
11	19

A STRUKTÚRA VÉGÉT JELÖLŐ WORD

Ha a függőleges raszterszámláló eléri a VSTOP értékét, a továbbiakban a sprite nem jelenik meg a képernyőn.

A következő struktúra egy SPACE SHIP-et ír le:

SPRITE: DC.W \$6D60,\$7200 ; VSTART, HSTART, VSTOP
DC.W \$0990,\$07E0
DC.W \$13C8,\$0FF0
DC.W \$23C4,\$1FF8
DC.W \$13C8,\$0FF0
DC.W \$0990,\$07E0
DC.W \$0000,\$0000 ; A struktúra vége

A SPRITE-OK BEKAPCSOLÁSA

A sprite-ok használata és a bekapcsolás a következőkből áll:

- HW Sprite mutatók beállítása
- Sprite DMA engedélyezése, ha még nincs bekapcsolva
- A függőleges visszafutás ideje alatt, a pozíciók módosítása

MEGJEGYZÉS

Amennyiben a sprite DMA-t akkor kapcsoljuk ki amikor egy sprite kirajzolása folyamatban van (VSTART után, VSTOP előtt), a rendszer folytatja a sprite megjelenítését a legutoljára beolvasott sor adataival. Ekkor a képernyőn egy függőleges sáv látható. Emiatt a sprite-ok kikapcsolását csak a függőleges visszafutás ideje alatt végezzük el.

SPRITE MUTATÓK BEÁLLÍTÁSA

Mindegyik sprite DMA csatorna egy-egy mutató segítségével olvassa a sprite adatokat. A mutatók elnevezése SPR x PTH (felső 3 bit) és SPR x PTL (alsó 16 bit). A következő példa a 0. sprite pointert beállítja \$20000-re. Ezt a feladatot egyébként a COPPER-rel szokás elvégeztetni:

```
SPR0PTH      EQU      $DFF120
SPR0PTL      EQU      $DFF122
              MOVE.L   #$20000, SPR0PTH
```

E mutató tartalma dinamikusan változik, mindig az éppen beolvasott sprite adatra mutat.

EGY SPRITE MEGJELENÍTÉSI PÉLDA

A már előzőleg is említett SPACE SHIP jelenik meg a 65,128 sprite pozícióban:

```
; Először egy kétszínű képernyőt kapcsolunk be
;
```

```
LEA          CUTSOM, A0          ; HW chipek báziscíme
MOVE.W       #$1200, BPLCON0(A0) ; 1 bittérkép, színes
MOVE.W       #$0000, BPL1MOD(A0) ; MODULO=0
MOVE.W       #$0000, BPLCON1(A0) ; Vízszintes scroll=0
MOVE.W       #$0024, BPLCON2(A0) ; Sprite prioritás
              ; nagyobb a képernyőnél
MOVE.W       #$0038, DDFTSRT(A0) ; FETCH start
MOVE.W       #$00D0, DDFSTOP(A0) ; FETCH stop
MOVE.W       #$2C81, DIWSTRT(A0) ; Display ablak start
MOVE.W       #$F4C1, DIWSTOP(A0) ; Display ablak stop
```

```

;
; Színregiszterek beállítása
;
        MOVE.W  #$,COLOR00(A0)      ; Háttérszín sötétkék
        MOVE.W  #$,COLOR01(A0)      ; Előtérszín fekete
        MOVE.W  #$,COLOR17(A0)      ; COLOR17 sárga
        MOVE.W  #$,COLOR18(A0)      ; COLOR18 lila
        MOVE.W  #$,COLOR19(A0)      ; COLOR19 bíborvörös
;
; COPPER lista $20000-re
;
        MOVE.L  #$20000,A1           ; Mutató a COPPER lis-
                                     ; ta kezdetére
        MOVE.L  COPPERL,A2           ; Mutató a forrásra
CLOOP:   MOVE.L  (A2),(A1)+
        CMP.L   #$FFFFFFFE,(A2)+
        BNE.S   CLOOP
;
; A spriteadatok átmásolása $25000-re
;
        MOVE.L  #$25000,A1           ; Mutató a struktúra
                                     ; kezdetére
        LEA     SPRITE,A2            ; Mutató a forráslis-
                                     ; tára
SPRLOOP: MOVE.L  (A2),(A1)+
        CMP.L   #$00000000,(A2)+    ; Lista vége
        BNE.S   SPRLOOP
;
; $30000-en egy ál sprite sruktúrát hozunk létre. Mivel minden
; sprite-ot be kell kapcsolni, az 1-7 sprite-okat ide irányítjuk.
;
        MOVE.L  #00000000,$30000
;
; COPPER lista címének átirása
;
        MOVE.L  #$20000,CUSTOM+COP1LC
;
; A bittérkép feltöltése $FF-ekkel
;
        MOVE.L  #$21000,A1
        MOVE.W  #$2000,D0
FLOOP:   MOVE.L  #$FFFFFFFF,(A1)+
        SUBQ.W  #1,D0
        BNE.S   FLOOP
;
; DMA start
;
        MOVE.W  CUSTOM+COPJMP1,D0
        MOVE.W  #$83A0,(CUSTOM+DMACON)
        BRA     ...
;
; COPPER lista. 1.bitmező $21000-től, 0.sprite $25000-től,
; többi sprite $30000-től
;

```

```

COPPERL:      DC.W      $00E0,$0002
               DC.W      $00E2,$1000
               DC.W      $0120,$0002
               DC.W      $0122,$5000
               DC.W      $0124,$0003
               DC.W      $0126,$0000
               DC.W      $0128,$0003
               DC.W      $012A,$0000
               DC.W      $012C,$0003
               DC.W      $012E,$0000
               DC.W      $0130,$0003
               DC.W      $0132,$0000
               DC.W      $0134,$0003
               DC.W      $0136,$0000
               DC.W      $0138,$0003
               DC.W      $013A,$0000
               DC.W      $013C,$0003
               DC.W      $013E,$0000
               DC.W      $FFFF,$FFFE

```

```

;
; Sprite adatok
;

```

```

SPRITE:      DC.W      $6D60,$7200
               DC.W      $0990,$07E0
               DC.W      $13C8,$0FF0
               DC.W      $23C4,$1FF0
               DC.W      $13C8,$0FF0
               DC.W      $0990,$07E0
               DC.W      $0000,$0000

```

SPRITE-OK MOZGATÁSA

A sprite-ok mozgatása úgy történhet, hogy a koordinátákat előíró word-öket a függőleges visszafutás ideje alatt az új (következő) koordinátára módosítjuk. A sprite-ok ütközése figyelhető egymás között vagy a két képernyő (DUAL PLAYFIELD) között is. Az ütközéseket a hardware figyeli. Az előző példánál maradva, a VSTART és HSTART pozíciók \$25000-től, a HSTOP pozíció \$25002-től található. A sprite mozgatásához ezeket a mutatókat kell módosítani. Változtassuk a VSTART-t 1-gyel, a HSTART-ot 2-vel.

```

MOVE.B #151,D0 ; Alap vízszintes számláló
MOVE.B #194,D1 ; Alap függőleges számláló
MOVE.B #64,D2  ; Alap vízszintes pozíció
MOVE.B #44,D3  ; Alap függőleges pozíció
MOVE.B #1,D4   ; Vízszintes lépésköz
MOVE.B #2,D5   ; Függőleges lépésköz

```

```

;
; Várakozás a függőleges visszafutásra
;

```

```

VLOOP:      MOVE.W  CUSTOM+INTEREQ,D6
               AND.W  #$0020,D6
               BEQ.S  VLOOP
               MOVE.W  #$0020,CUSTOM+INTEREQ

```

```

;

```

```

      ADD.B    D4,D2      ; Vízszintes pozíció módosítása
      SUBQ.W   D4,D0      ; Vízszintes számláló csökkentése
      BNE.S    L1
      MOVE.B   #151,D0    ; Vízszintes számláló újratöltése
      EOR.B    #$FE,D4    ; A módosító érték negálása

L1:    MOVE.B   D2,$25001; Az új HSTART érték beírása
      ADD.B    D5,D3      ; Függőleges pozíció módosítása
      SUBQ.W   #1,D1      ; Függőleges számláló csökkentése
      BNE.S    L2
      MOVE.B   #194,D1    ; Függőleges számláló újratöltése
      EOR.B    #$FE,D5    ; A módosító érték negálása

L2:    MOVE.B   D3,$25000; Az új VSTART érték beírása
      MOVE.B   D3,D6      ; Az új VSTOP kiszámítása
      ADD.B    #6,D6      ; VSTOP=VSTART+6
      MOVE.B   D6,$25002; Az új VSTOP érték beírása
      BRA.S    VLOOP

```

Mivel nincs minden egyes sprite-nak ki- és be kapcsolási lehetősége, kizárólag egyszerre kapcsolható sprite, így a nem használt sprite-okat a példában is látható módon egy üres sprite adatstruktúrára kell irányítani.

SPRITE SORSZÁMA	SZÍNREGISZTEREK
0 és 1	17-19
2 és 3	21-23
4 és 5	25-27
6 és 7	29-31

EGY SPRITE DMA CSATORNA TÖBBSZÖRÖS FELHASZNÁSA

Egy sprite DMA csatorna többszörös megjelenítésre is alkalmas a következő eltételek betartása mellett.

- A következő sprite képernyőn elfoglalt Y pozíciója csak nagyobb lehet, mint az öt megelőző
- Az egyes rész sprite-ok X pozíciója tetszőleges, ebben nincs korlátozás

A normális sprite használatkor egy sprite struktúra végét két 0 word jelzi. Ha ezzel találkozik a rendszer, akkor ennek a csatornának a használatára nincs szükség a további rasztersorokban. A csatorna többszörös elhasználása abból áll, hogy a két word előtt pluszként (az előbbieken leírt feltételek betartásával) sprite struktúrákat szúrunk be. A korlátozás amiatt van, hogy egy rasztersoron belül csak két DMA ciklus van a sprite adatok olvasására csatornánként.

A következő példaprogram egy SPACE SHIP-et és egy másik objektumot jelenít meg:

```

        LEA      CUSTOM,A0
        MOVE.W   #$0F00,COLOR17(A0) ; COLOR17=Vörös
        MOVE.W   #$0FF0,COLOR18(A0) ; COLOR18=Sárga
        MOVE.W   #$0FFF,COLOR19(A0) ; COLOR19=Fehér

SPRITE:  DC.W     $6D60,$7200
        DC.W     $0990,$07E0
        DC.W     $13C8,$0FF0
        DC.W     $23C4,$1FF8
        DC.W     $13C8,$0FF0
        DC.W     $0990,$07E0
        DC.W     $8000,$8D00          ; VSTART, HSTART, VSTOP
                                         ; az új sprite-hoz

        DC.W     $1818,$0000
        DC.W     $7E7E,$0000
        DC.W     $7FFE,$0000
        DC.W     $FFFF,$2000
        DC.W     $FFFF,$2000
        DC.W     $FFFF,$3000
        DC.W     $FFFF,$3000
        DC.W     $7FFE,$1800
        DC.W     $7FFE,$0C00
        DC.W     $3FFC,$0000
        DC.W     $0FF0,$0000
        DC.W     $03C0,$0000
        DC.W     $0180,$0000
        DC.W     $0000,$0000          ; Sprite adatok vége

```

ÁTLAPOLT SPRITE-OK

Amennyiben 16 pixel-nél szélesebb, összetettebb, többszínű sprite-okat használunk, akkor a következő fogással megoldható ez a probléma is. A sprite prioritásokat kihasználva pl. 2 egymást átfedő sprite segítségével többszínű sprite, vagy két összeérő sprite egyidejű mozgatásával szélesebb sprite-ok alkothatók.

ÖSSZEÉRŐ SPRITE-OK

Ezzel a módszerrel 16 színű (+ az átlátszó) spriteokat definiálhatunk, a következő feltételek mellett:

- Két csatornát kell használni sprite-onként, két sprite struktúrával amelyek azonos méretűek, és azonos pozicionálással rendelkeznek
- A második sprite 2. kontroll word-jében az ATTACH bitet be kell állítani

A 16 szín lefoglalja az összes színregisztert 17 - 31-ig. A plusz színek azáltal érhetők el, hogy a normál sprite-okkal szemben 4 bit írja elő egy sprite pixel színét. Az összekapcsolható sprite-ok az egy párba tartozóak.

Sprite 1-0
 Sprite 3-2
 Sprite 5-4
 Sprite 7-6

A négy bit által előállított 0 - 15 bináris számok a következő színregisztereket írják elő.

BITKOMBINÁCIÓ	BINÁRIS SZÁM	SZÍNREGISZTER
0	0000	16*
1	0001	17
2	0010	18
3	0011	19
4	0100	20
5	0101	21
6	0110	22
7	0111	23
8	1000	24
9	1001	25
10	1010	26
11	1011	27
12	1100	28
13	1101	29
14	1110	30
15	1111	31

* Nem használt. Ez átlátszó pixel

A magasabb sorszámú sprite tartalmazza a magasabb helyiértékű biteket:

1.bitsor - 1.sprite első bitsora
 2.bitsor - 1.sprite második bitsora
 3.bitsor - 2.sprite első bitsora
 4.bitsor - 2.sprite második bitsora

Az üzemmód az ATTACH bit segítségével (páratlan sorszámú sprite 2. kontroll word-jének 7.bitje) kapcsolható be. A következő példa a SPACE SHIP-et 6 színüként definiálja:

```

SPRITE0:    DC.W    $6D60,$7200    ; VSTART=65,HSTART=128
            DC.W    $0C30,$0000    ; Az 1.színt meghatározó word
            DC.W    $1818,$0420
            DC.W    $342C,$0E70
            DC.W    $1818,$0420
            DC.W    $03C0,$0000
            DC.W    $0000,$0000    ; 0.sprite vége
SPRITE1:    DC.W    $6D60,$7200    ; Pozíciók megegyeznek a 0.
                                   ; sprite-éval, de az ATTACH
                                   ; bit magas !
  
```

DC.W \$07E0,\$0000
DC.W \$0FF0,\$0000
DC.W \$1FF8,\$0000
DC.W \$0FF0,\$0000
DC.W \$07E0,\$0000
DC.W \$0000,\$0000 ; Az 1.sprite vége

SPRITE REGISZTEREK ÖSSZEGZÉSE

A sprite regiszterek a 8 sprite-nak megfelelően 8 regiszter készletből állnak. Egy készlet 5 regiszterből áll.

SPRxPTH és SPRxPTL

E két regiszter tartalmazza az x.sprite adatainak címét.

SPRxPOS

Ez a regiszter a sprite adatstruktúrában található, a sprite pozícióját írja elő:

	Bit 15-8	Függőleges start pozíció (V7-V0)
	Bit 7-0	Vízszintes start pozíció (H8-H1)
SPRxCTL		
	Bit 15-8	Függőleges stop pozíció (V7-V0)
	Bit 7	ATTACH bit. Ez a bit csak páratlan sorszámú sprite-oknál értelmezett
	Bit 6-3	Foglalt, későbbi alkalmazásokra. Értékük legyen 0
	Bit 2	A függőleges startpozíció legfelső bitje. (V0)
	Bit 1	Függőleges stop pozíció legfelső bitje. (V8)
	Bit 0	Vízszintes startpozíció legalsó bitje (H0)

SPRxDATA, SPRxDATB

E regisztereket normál esetben csak a display processzor használja munkaterületként, átmeneti adattárolóként.

SPRITE SZÍNEK ÖSSZEFOGALÁSA

NORMÁL SPRITE

SPRITE	BITMINTA	SZÍN

0 és 1	00	átlátszó
	01	17
	10	18
	11	19
2 és 3	00	átlátszó
	01	21
	10	22
	11	23
4 és 5	00	átlátszó
	01	25
	10	26
	11	27
6 és 7	00	átlátszó
	01	29
	10	30
	11	31

ÖSSZEKAPCSOLT SPRITE

BITMINTA	SZÍNREGISZTER
0000	átlátszó
0001	17
0010	18
0011	19
0100	20
0101	21
0110	22
0111	23
1000	24
1001	25
1010	26
1011	27
1100	28
1101	29
1110	30
1111	31

COPPER

A COPPER az AMIGA speciális célú segédprocesszorainak egyike. Direkt memóriáhozáféréssel dolgozik. A FAT AGNUS elnevezésű chipben található a BLITTER-rel együtt. Elsődleges feladata a grafiku vezérlőregiszterek gyors felfrissítése. Ezt a folyamatot az alatt az idő alatt végzi el, mialatt a képernyőt felrajzoló sugár függőlegesen visszafut. Szabályozza csaknem az egész grafikus rendszert, függetlenül a 68000-es főprocesszortól, amely így felszabadul a képernyőszerkesztés egyébként időigényes feladata alól. Közvetlenül képes befolyásolni a többi chip regisztereit. Ez úton vezérelhető a színskála, a BLITTER, a sprite-ok, és a hang is. A COPPER 3 utasítást ismer, ezek a MOVE, a WAIT, és a SKIP.

A MOVE segítségével tetszőleges értéket írhatunk a HW regiszterek bármelyikébe. A WAIT arra utasítja a COPPER-t, hogy amíg a video sugár el nem ér egy előírt pontot a képernyőn, addig várakozzon. A várakozás alatt a rendszerbusz teljesen szabad a 68000-es, és a többi segédchip számára, használható bármelyik DMA csatorna. Amikor a feltétel teljesül, a COPPER lefoglalja az összes memóriaciklust, amíg elvégzi az előírt műveletet. Segítségével az időzítések pontosan kezelhetők. Várakozás alatt a rendszerbusz teljesen szabad, a 68000-es használhatja bármelyik DMA csatornát. A SKIP utáni parancsot nem hajtja végre a COPPER, ha az aktuális videó sugárpozíció nagyobb, vagy egyenlő, mint a paraméterként megadott érték. A COPPER csak az alsó 512 KByte-ban tud dolgozni, ugyanis címregisztereit csak 19 bites címeket tudnak kezelni. A COPPER a páratlan órajelciklusokban dolgozik. Ezáltal elkerülhető lett az összeütközés a hang, a diszk, a memóriafrissítés, a sprite és a kifelvbontású display hozzáférésekkel, mivel ezek a páros ciklusokban dolgoznak. A COPPER a harmadik legnagyobb prioritású eszköz a 68000-es, és a BLITTER után.

Minden COPPER utasítás két 16 bites szóból áll, melyek a memóriában folytonosan követik egymást. Amikor a COPPER egy utasítást beolvas, mindig 1 szót olvas be. A MOVE és a SKIP utasítások végrehajtása 2, a WAIT-é 3 memóriaciklusig (6 CPU ciklusig) tart. Amellett hogy a COPPER direkt módon csak a HW regisztereket tudja írni, kezelheti a memóriát is egy BLITTER művelet elindításával.

A MOVE UTASÍTÁS

Egy megadott értéket ír a szintén megadott célregiszterbe. Az első szó a célregiszter címe, a második a beírandó adat.

1.szó	(IR1)	Bit 0	- Mindig nulla!
		Bit 1-8	- A célregiszter címe (DA1-8)
		Bit 9-15	- Nem használt, értéke legyen 0
2.szó	(IR2)	Bit 0-15	- A 16 bites adat

A COPPER a következő regiszterekbe tud írni:

- Amelyeknek (relatív) címe \$020 vagy annál nagyobb
- A \$010 és \$020 címek közé eső regiszterbe, ha a COPPER DANGER Bit magas. E bit a COPCON regiszterben található.
- A \$010 relatív címnél alacsonyabb című regisztereket nem képes írni, mivel itt a legtöbb regiszter csak olvasható.

MEGJEGYZÉS

A COPPER által kezelt címek mind ún. HW regiszter címek. Ezek báziscíme \$DFF000, a HW címek e báziscímhez viszonyított offszetek.

A következő példa a MOVE utasítás segítségével az 1.bitmező mutatóját \$21000, a 2.bitmező mutatóját \$25000-re állítja:

```
DC.W $00E0,$0002 ; $0002 a $0E0 regiszterbe (BPL1PTH)
DC.W $00E2,$1000 ; $1000 a $0E2 regiszterbe (BPL1PTL)
DC.W $00E4,$0002 ; $0002 a $0E4 regiszterbe (BPL2PTH)
DC.W $00E6,$5000 ; $5000 a $0E6 regiszterbe (BPL2PTL)
```

A WAIT UTASÍTÁS

A WAIT arra utasítja a COPPER-t, hogy várjon, ha a video sugár számlálója kisebb mint az utasításban előírt pozíció. Amíg az utasítás aktív, addig a COPPER lekapcsolódik a rendszerbuszról, nem használ fel egyetlen memóriaciklust sem. Az első utasításszó a kívánt vízszintes és függőleges sugárpozíciókat, a második szó engedélyező biteket tartalmaz az alábbiak szerint:

1.szó	(IR1)	Bit 0	- értéke mindig 1
		Bit 1-7	- vízszintes sugárpozíció (HP)
		Bit 8-15	- függőleges sugárpozíció (VP)
2.szó	(IR2)	Bit 0	- értéke mindig 0
		Bit 1-7	- vízszintes összehasonlítást engedélyező bitek (HE)
		Bit 8-14	- függőleges összehasonlítást engedélyező bitek
		Bit 15	- Blitter Finished Disable bit, értéke alapesetben 0

A következő példa a 150. sorra (\$96) várakoztatja a COPPER-t:

```
DC.W $9601,$FF00 ; Vízszintes összehasonlítás nincs
```

A következő példa a 255. sor 254. pozíciójára vár. Ez a pozíció sohasem következik be. Ennél az utasításnál a COPPER az utasításlista feldolgozását előlről kezdi. A WAIT 255,254 így a COPPER lista végét jelzi.

DC.W \$FFFF,\$FFFE

A következőkben leírtak a WAIT és a SKIP utasításra is vonatkoznak.

VÍZSZINTES SUGÁRPOZÍCIÓ

A vízszintes raszter értéke \$00 és \$e2 között változik. A legkisebb helyiértékű bit nem kezelt, emiatt 113 vízszintes pozíció érhető el. Ez kislebontású üzemben 4 pixelnek, nagy felbontású üzemben 8 pixelnek felel meg. A \$0F és \$35 közötti értékek általában nem használtak. Az alap (320 pixeles) képernyőn a \$04 és \$47 közötti terület háttérszínű.

FÜGGŐLEGES SUGÁRPOZÍCIÓ

A függőleges raszter értéke \$00 és \$FF között változik. Mivel valójában 312 függőleges pozíció van, komplikációk lehetnek az utolsó 256. sortól. A következő módon azonban megoldhatóak a problémák:

Várakozás a 260. sor 0. pozíciójára:

WAIT (255,0) ; E ponton kerül 0 a függőleges raszter számlálóba
WAIT (4,0) ; Ezáltal 255+4=260. sorban állsz

A COPPER REGISZTEREINEK HASZNÁLATA

COPPER LISTA CÍM MEGHATÁROZÓ REGISZTEREK

COP1LCH Az 1. COPPER lista címének felső 3 bitje
COP1LCL Az 1. COPPER lista címének alsó 16 bitje
COP2LCH A 2. COPPER lista címének felső 3 bitje
COP2LCL A 2. COPPER lista címének alsó 16 bitje

Látható, hogy összesen 19 bit áll rendelkezésre a cím tárolására. Emiatt a COPPER-t érintő dolgok csak az alsó 512K-ban helyezkedhetnek el. A fenti 2 regiszterben egy-egy indirekt COPPER ugrócím van elhelyezve. Amikor a COPPER elindul, akkor e címek valamelyikétől kezdve olvassa a memóriatartalmat, mint COPPER listát. (Ez a COPPER programszámlálójának kezdőértéke.) Ha a regisztereket írjuk, a COPPER az új címre ugrik a következő utasítás elővételi ciklusban.

UGRÁSI CÍMET TARTALMAZÓ REGISZTEREK

Ha ezekbe a regiszterekbe írunk, a COPPER betölti programszámlálójába a megfelelő COPxLCH regiszter tartalmát. A COPPER képes írni a COPxLCH regisztereket, miáltal programozható ugrások is létrehozhatóak. Ha a MOVE-val beírunk egy címet, pl. a COP2LC-be és szintén a MOVE-val írunk a COPJMP2-be, akkor a COPPER egy indirekt ugrást hajt végre az új címre. COPJMP1 COPPER újraindítása a COP1LC-ben tárolt címtől COPJMP2 COPPER újraindítása a COP2LC-ben tárolt címtől.

VEZÉRLŐ REGISZTEREK

A COPPER szinte valamennyi HW regisztert tudja írni. Néhány regisztert azonban csak abban az esetben, ha egy speciális (DANGER Bit) magasra van állítva. A \$0010 HW cím alatti regiszterek nem írhatóak (a legtöbb regiszter ezen a területen csak olvasható), a \$0010 és \$0020 között csak akkor ha a DANGER Bit magas, míg \$0020 és \$00FF között mindig írhatóak. A COPPER CONTROLL regiszter mindig írható. Neve COPCON, egyetlen hasznos bitet tartalmaz, az elsőt. A bit elnevezése CDANG. E bit védi a \$0010 és \$001F közötti regisztereket, többek között a BLITTER controll regisztert is. Ha a CDANG 0, akkor a regiszterek nem írhatóak.

Megjegyzés: A CDANG bitet egy reset törli!

COPPER LISTA ÖSSZEFOGLALÁSA

A COPPER lista tartalmazza mindazon tevékenységet, amelyeket a COPPER-nek a videósugár függőleges visszafutásakor vagy a kép felrajzolása alatt el kell végeznie. Nemcsak a grafika vezérelhető, módosíthatóak az összes HW regisztereken keresztül elérhető funkciók. A következő példában a bitmező mutatót és a sprite mutatót állítjuk, mialatt a sugár a függőleges visszafutást végzi:

```
WAIT   az első display sorra
MOVE   cím1 az 1.bitmező mutatóba
MOVE   cím2 a 2.bitmező mutatóba
MOVE   cím3 az 1.sprite mutatóba
stb...
```

A képernyő bal felső sarka a 0, 0 a jobb alsó sarok a 226,262 raszterpozíció. Az első koordináta az X. A WAIT addig vár, amíg a raszter számláló egyenlő vagy nagyobb nem lesz a paraméterben megadottnál.

EGY KOMPLETT COPPER LISTA

A lista bekapcsol két bittérképet \$21000-re és \$25000-re. A képernyő tetején a színregiszterek a következő értéket kapják:

```
COLOR00  fehér
COLOR01  vörös
COLOR02  zöld
COLOR03  kék
```

A 150. rasztersorban a színregiszterek újratöltődnek:

```
COLOR00 fekete
COLOR01 sárga
COLOR02 cián
COLOR03 lila
```

A lista:

```
DC.W $00E0,$0002 ; $0002 a BPL1PTH-ba
DC.W $00E2,$1000 ; $1000 a BPL1PTL-be
DC.W $00E4,$0002 ; $0002 a BPL2PTH-ba
DC.W $00E6,$5000 ; $5000 a BPL2PTL-be

;
; A színregiszterek feltöltése
;
DC.W $0180,$0FFF ; fehér ($0FFF) a COLOR00-ba
DC.W $0182,$0F00 ; vörös ($0F00) a COLOR01-be
DC.W $0184,$00F0 ; zöld ($00F0) a COLOR02-be
DC.W $0186,$000F ; kék ($000F) a COLOR03-ba

;
; Várakozás a 150.rasztersorra
;
DC.W $9601,$FF00 ; A vízszintes pozíciót nem figyeljük
;
; Színregiszterek újratöltése
;
DC.W $0180,$0000 ; fekete ($0000) COLOR00-ba
DC.W $0182,$0FF0 ; sárga ($0FF0) COLOR01-be
DC.W $0184,$00FF ; cián ($00FF) COLOR02-be
DC.W $0186,$0F0F ; lila ($0F0F) COLOR03-ba

;
; A COPPER lista vége (várakozás egy nem létező pozícióra)
;
DC.W $FFFF,$FFFE ; 255.sor 254.pozíció
```

COPPER START ÉS STOP

COPPER START RESET UTÁN

Bekapcsolás vagy reset után inicializálni kell a COPPER címző regisztereit, írni kell a COPJMP-be, majd a COPPER DMA-t engedélyezni kell:

```
MOVE.L COPPERLISTA,A0 ; A COPPER lista címe A0-ba
MOVE.L A0,COP1LCH
MOVE.W COPJMP1,D0 ; A COPPER programszámlálójának
; feltöltése az új értékkel
MOVE.W $SETBIT+COPPERDMA,D0
MOVE.W D0,DMACONW ; COPPER DMA engedélyezése
```

COPPER STOP

A COPPER megállítására nincs külön utasítás. A lista végét egy nem érvényes WAIT utasítás jelzi. Ha ezzel találkozunk a COPPER, akkor programszámlálóját újratölti (előlről kezdi a listát végrehajtani). Megállapodás szerint ez az érvénytelen utasítás a WAIT 255,254. Ha a COPPER-t végérvényesen ki akarod kapcsolni, akkor ezt a DMACON regiszter segítségével teheted meg. Ez a regiszter vezérli az összes DMA csatornát. A 7. bit (COPEN) a COPPER engedélyező bit, ha magas akkor a COPPER fut.

A SKIP UTASÍTÁS

A SKIP utasítás hatására a COPPER átlépi a következő utasítást, ha a rasterszámláló értéke nagyobb vagy egyenlő mint a paraméterként megadott érték. A WAIT és a SKIP utasítást az különbözteti meg egymástól, hogy a WAIT második szavának 0. bitje 0, a SKIP-é viszont 1.

1.szó	(IR1)	Bit 0	- Mindig 1
		Bit 15-8	- Függőleges pozíció (VP)
		Bit 7-1	- Vízszintes pozíció (HP)
2.szó	(IR2)	Bit 0	- Mindig 1
		Bit 15	- BLITTER Finished Disable bit
		Bit 14-8	- Függőleges összehasonlítás engedélyező bitek. (VE)
		Bit 7-1	- Vízszintes összehasonlítás engedélyező bitek. (HE)

NA következő példában a COPPER átlépi a következő utasítást, ha a függőleges rasterszámláló nagyobb vagy egyenlő mint 100 (\$64):

```
DC.W    $6401,$FF01
```

Amennyiben magában a listában módosítjuk a COP1LCH tartalmát, akkor akár felételes ugrások is kezelhetők. A WAIT és SKIP utasítások második szavainak 1 - 14 bitjei azt írják elő, hogy az összehasonlítás során a rasterszámláló mely bitjei legyenek figyelembe véve. Ha pl. a függőleges pozíciónak csak az utolsó 4 bitjét kívánjuk figyelni, akkor csak az utolsó 4 bitet kell beállítania 2. szóban (11 - 8 bitek). így 16 rastersoronként kapunk egyezést.

```
DC.W    $5001,$FFFE    ; WAIT amíg VP=$50, HP=0
```

```
DC.W    $0F01,$0F00    ; WAIT amíg VP=xxxx1111
```

```
;
```

```
; A következő utasítás a $DFF09C címre (IRQ regiszter) $0010-et ír.  
; Ez beállítja a COPPER IRQ bitet, miáltal a COPPER IRQ jelet tud  
; küldeni a 68000-esnek
```

```
;
```

```
DC.W    $009C,$8010    ; MOVE.L $8010,$DFF09C
```

```
DC.W    $A001,$FF01    ; Ha VP>=160, akkor átugorja a következőt
```

```
;
```

```
; A következő utasítás egy ál-MOVE. Egy kieroszakolt COPPER RESART  
;-ot vált ki a COP2LC-ben tárolt címtől. Ha VP>=160, akkor az  
; utasítást átugorja.
```

```
;
```

```
DC.W    $008A,$0000    ; MOVE.L #0,COPJMP2
```

COPPER HASZNÁLATA INTERLACE ÜZEMMÓDBAN

Az interlace képernyő függőleges felbontása 400 sor, az alapállapotú felbontás kétszerese. Az üzemmódban a video sugár kétszer futja át a képernyőt, míg a kép előáll. Elsőre a páratlan, másodjára páros sorokat rajzolja fel. A következő táblázat az interlace kép tárolását magyarázza:

ADAT A KÉPERNYŐN	ADAT A MEMÓRIÁBAN
Páros mező 1.sora	1.sor
Páratlan mező 1.sora	2.sor
Páros mező 2.sora	3.sor
Páratlan mező 2.sora	4.sor
.	.
.	.
.	.
Páros mező utolsó sora	399.sor
Páratlan mező utolsó sora	400.sor

A COPPER-nek ebben az üzemmódban két listára van szüksége. A két lista közötti átváltásnak a függőleges visszafutás alatt kell megtörténnie, a COP x LC regiszter megfelelő átírásával.

A COPPER HASZNÁLATA A BLITTERREL

Ha elindítunk egy BLITTER műveletet, akkor amíg a BLITTER FINISHED visszajelzést meg nem kapjuk, nem adható ki újabb BLITTER művelet. A BLITTER regiszterek a jelzés megkapása előtt is módosíthatóak. A WAIT utasítás tartalmaz egy plusz bitet (BFD Blitter Finished Disable). Alapesetben ez a bit magas. Ekkor csak a videosugár pozíciója van kiha- tással a WAIT utasításra. Azonban ha a bitet töröljük, a WAIT a pozíció egyezés mellett a BLITTER művelet befejezésére is várni fog. A BLITTER kész, ha a bit magasra vált.

A COPPER ÉS A 68000-ES

Ha a 68000-es munkáját meg akarjuk szakítani (INTERRUPT), a COPPER MOVE utasításának segítségével írjunk 1- et az INTREQ regiszter COPEM bit- jébe:

Bit 15	SET/CLR	Set/Clear vezérlő bit. Ha e bit magas, akkor a regiszterbe történő írás a biteket beállítja, egyébként törli.
Bit 4	COPEM	Coprocesszor Interrupt a 68000-es felé.

A COPPER UTASÍTÁSAINAK ÖSSZEGLALÁSA

A következő táblázat összesítve tartalmazza az egyes bitpozíciók értelmezését a három Copper utasítás esetében:

Parancs	Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MOVE	W1	0	0	0	0	0	0	0	RA8	RA7	RA6	RA5	RA4	RA3	RA2	RA1	0
	W2	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
WAIT	W1	VP7	VP6	VP5	VP4	VP3	VP2	VP1	VP0	HP8	HP7	HP6	HP5	HP4	HP3	HP2	1
	W2	BFD	VM6	VM5	VM4	VM3	VM2	VM1	VM0	HM8	HM7	HM6	HM5	HM4	HM3	HM2	0
SKIP	W1	VP7	VP6	VP5	VP4	VP3	VP2	VP1	VP0	HP8	HP7	HP6	HP5	HP4	HP3	HP2	1
	W2	BFD	VM6	VM5	VM4	VM3	VM2	VM1	VM0	HM8	HM7	HM6	HM5	HM4	HM3	HM2	0

W1 - első utasításszó
 W2 - második utasításszó
 RA - célregiszter címe
 D - adat, amelyet a célregiszterbe kívánunk írni
 VP - függőleges raszterpozíció értéke
 HP - vízszintes raszterpozíció értéke
 VM - függőleges összehasonlítás engedélyező bitek (MASZK)
 HM - vízszintes összehasonlítás engedélyező bitek (MASZK)
 BFD - Blitter Finished Disable, Blitter funkció vége

BLITTER

BEVEZETÉS

A memória négyszögletű területeinek mozgatására alkalmas eszköz (ez általában grafikus térkép), de vonalak rajzolására is alkalmas. Négy DMA csatornát használ: az A, B, C adatforrás csatornákat és a D adatcél csatornát. Ezeket a BLTCON0 regiszter 11, 10, 9, 8 bitjei kapcsolják be (USEA, USEB, USEC, USED). A forráscsatornákból az adatok egy tárolóregiszterbe kerülnek, itt logikai műveleteket végezhetünk velük, majd az adatcél csatornán át a BLITTER-be továbbítódnak.

BLITTER MŰVELETEK

ADATMÁSOLÁS

Ez a BLITTER elsődleges feladata: nagy adatblokkokat mozgat. Ehhez az szükséges, hogy megadjuk a forrás címét, a cél címét és a blokk méretét (szélesség, magasság). Ezután az adatbusz (egyébként üres állapotában) automatikusan végrehajtja byte-onként a másolást. A másolás végét egy jelző FLAG és egy megszakítás fogja követni.

SZÁMLÁLÓK ÉS MÓDOSÍTÓK

A számláló arra a címre mutat, ahol a forrás vagy a cél következő byte-ja van. Ezzel a memória alsó 512K-ja címezhető. A feladatra 16 bites regisztereket használ.

BLTAPTL - BLTAPTH
 BLTBPTL - BLTBPTH
 BLTCPTL - BLTCPTH
 BLTDPTL - BLTDPTH

A módosítók feladata az, hogy egy nagyobb képen belüli kisebb képet változtathassunk meg. A módosító tulajdonképpen a nagy és kis kép szélessége közötti különbség. Négy ilyen módosító van:

BLTAMOD, BLTBMOD, BLTCMOD, BLTDMOD

Ezzel lehetővé válik, hogy különböző nagyságú bittérképekre ill. bittérképekről másoljunk egy kisebb darabot. Ez úgy valósul meg, hogy a beállított kezdőértéktől növekszik a számláló regiszter értéke, majd elérve a kép szélességet a módosító értékével ugrik. Vigyázni kell arra, hogy a számlálót és a módosítót a processzor byte- szóként értelmezi de bele az adatokat byte-ként tölti (a byte-szónál a 0. bit mindig 0, tehát a byte szóban levő érték a kétszerese lesz a valódi módosítónak vagy számlálónak. Pl. ha 4-et akarunk módosítónak, akkor 8-at kell byte szóban tartalmazni).

CSÖKKENŐ ÉS NÖVEKVŐ CÍMZÉS

Fontos, hogy az átmásolásnál semmi esetre se írassuk azt a területet amelyben a forrás vagy a cél található, erre szolgál ez a funkció. Ha alacsonyabb területről magasabbra másolunk akkor csökkenő üzemmódba kell másolni, ha pedig magasabbról alacsonyabbra akkor növekvő üzemmódot használjunk (a csökkenő üzemmód a BLTCON1 regiszter 1. bitjének magas állása, míg a növekvőt az alacsonyabb állása jelzi).

NÉGYSZÖG ÉS VONALAS CÍMKERESÉS

A négyszög területek magasságát és szélességét a BLTSIZE regiszterbe írhatjuk. A számláló regiszter megadja a helyet ahol az átvinni kívánt adatok találhatóak. A BLITTER hozzáad (növekvő címezéssel) ill. levon (csökkenő címezéssel) 2-t minden 16 bites byte-szó átvitelekor, egész addig amíg el nem éri a blokk szélességét, ezután hozzáadja a módosító regiszter tartalmát. Így a módosító regiszter értéke az a szám, amelyet a számláló regiszterhez kell adni egy sor végén, hogy a következő sor kezdőcímét megkapjuk. A BLITTER-t vonalas üzemmódba is állíthatjuk úgy, hogy vagy a szélesség, vagy a magasság egy lesz.

A BLITTER LOGIKAI MŰVELETEI

A logikai műveletekre az A,B,C forráscsatorna áll rendelkezésre. Ezek általában egy bittérkép különböző részelemeit tartalmazzák. Mivel mindhárom egyforma négyszög, ezért bitenként különböző logikai műveleteket hajthatunk végre közöttük, ezért az LF kontroll byte felelős a BLTCON0 regiszterben. Ez a byte generálja a három csatornával létrehozható 256 féle logikai műveletet. Mivel logikai műveleteket lehet a BLITTER-rel végrehajtatni, ezért kiválóan alkalmas animációs hatások létrehozására. Két féle módon számíthatjuk ki a kívánt logikai értéket: logikai egyenletekkel ill. diagram segítségével.

LOGIKAI MŰVELETEK

A három csatorna nyolc különböző kombinációja jöhet létre:

ABC	ABC	ABC	ABC	ABC	ABC	ABC	ABC	
7.	6.	5.	4.	3.	2.	1.	0.	bit az LF byte-ban

(Az aláhúzott csatorna hamis értékű, a betű önmagában igaz értéket jelöl. A csatornák között logikai ÉS művelet hajtódik végre, tehát csak akkor egy értékű, ha mindhárom állítás igaz.)

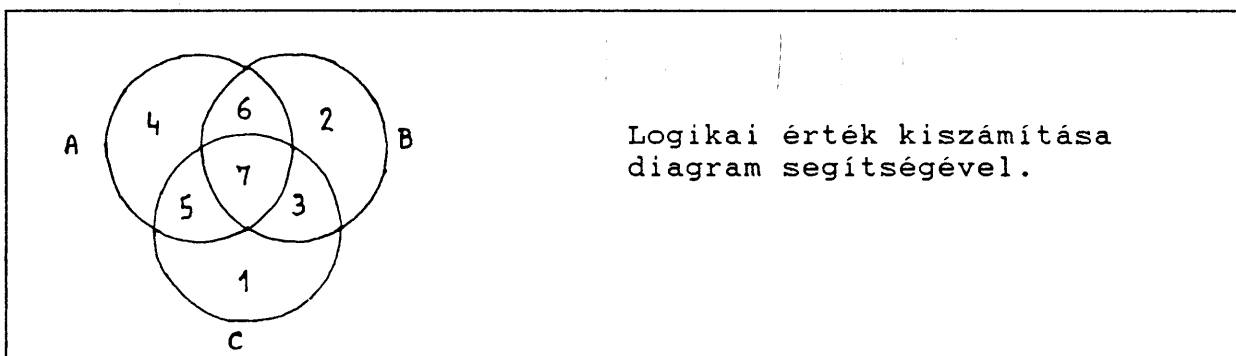
Példa egy egyenletre: $D = ABC + \underline{ABC}$
 $D = AB (C + \underline{C})$
 $D = AB$

Az általában használt egyenletek és ezek LF értékei a táblázatban megtalálhatók.

EGYENLET	LF	EGYENLET	LF
D=A	F0	D=AB	C0
D=A	0F	D=AB	30
D=B	CC	D=AB	0C
D=B	33	D=AB	03
D=C	AA	D=BC	88
D=C	55	D=BC	44
D=AC	A0	D=BC	22
D=AC	50	D=AC	11
D=AC	0A	D=A+B	F3
D=AC	05	D=A+B	3F
D=A+B	FC	D=A+C	F5
D=A+B	CF	D=A+C	5F
D=A+C	FA	D=B+C	DD
D=A+C	AF	D=B+C	77
D=B+C	EE	D=AB+AC	CA
D=B+C	BB		

DIAGRAM SEGÍTSÉGÉVEL

Ez egy sokkal érthetőbb és könnyebb módszer a helyes LF érték kiszámításához.



1 Ha pl. a D=A-t választjuk, akkor csak azok a számok kellenek, amik egy-egy betű halmazába tartoznak (itt ezek 7,6,5,4). Ezután már csak a számoknak megfelelő biteket kell bekapcsolni a helyes LF értékhez:

ABC	ABC	ABC	ABC	ABC	ABC	ABC	ABC	
7.	6.	5.	4.	3.	2.	1.	0.	bit az LF byte-ban
1	1	1	1	0	0	0	0	tehát a helyes LF érték = F0
F				0				

2 Ha két bitből álló kombinációt választunk (pl.D=AB), akkor azokat a számokat kell venni, amelyek a két betű halmazainak metszetében vannak (ez a példában a 7 és a 6, ennek megfelelően LF=11000000, azaz C0)

3 Ha a kombináció hamis állításra kérdez (pl D=A), akkor mindazokat a számokat kell venni, amelyek nem tartoznak az A halmazába (a metszetekkel együtt)

pl D=AB akkor a szám 2,3 azaz 00000110 LF=06
 pl D=A akkor a szám 1,3,2 azaz 00000111 LF=07

4 Ha két betűkombinációt alkalmazunk (összeadunk)
 pl. D=AB + BC , akkor logikai VAGY-ot használunk:

AB	6,7	11000000
BC	3,7	10001000

		11001000 azaz LF=C8

ELMOZDÍTÁS

A BLITTER tartalmaz egy elmozgató áramkört is, amely az A és B forráscsatornán használható. Ezeket az adathalmazokat 0 - 15 bittel tudja eltolni (scroll-művelet). Két mozgató irányító regiszter van. A BLTCON 15 - 12. bitjei (az A forrás elmozdítási értékét adják), és a BLTCON1 15-12. bitjei (a B forrás elmozdítási értékét adják). Vonalas rajzolás közben ezek a bitek más feladatokat látnak el.

MASZKOLÁS

Ha egy tárgy szélessége az egy byte-szó (16 bit) nem egész számú többszöröse, a BLITTER lemaszkolhatja mind a bal, mind a jobb szélén lévő biteket azért, hogy csak azzal az aktuális bithatárral dolgozzunk, amely éppen bekeretezi a tárgyat. A lehetőség különösen akkor hasznos, ha betűket kell sűrítetten tárolni (nincs a betűk között szünet). Az első szó maszk azt jelenti, hogy a forrás minden vízszintes sorában az első byte-szót maszkolhatjuk, az A forrásnál ezt a BLTAFMW regiszter kezeli. A maszkolást azzal hajtjuk végre, hogy az eredetileg csupán 1-et tartalmazó regisztert 0-kal töltjük fel. Csak azok a bitek jelennek meg a forrás első byte-szavából, amelyekhez az első szó maszk regiszterben 1-et rendeltünk. Az utolsó szó regiszter szintén hasonló módon működik (BLTALMW). Ha csak egy byte hosszúságú a rajzunk, akkor az első szó maszk és az utolsó szó maszk egy byte-szóra vonatkozik, és kettő értékei között logikai ÉS hajtódik végre: az egy byte-szó így csak ott nem lesz maszkolható, ahol ezt mindkét maszk megengedi. (Alapesetben mindkét regiszter csupa 1-et tartalmaz, értéke tehát FFFF).

NULLA KERESÉS

Az ún. blitter nulla egy jelzőbit feladata, hogy jelezze: a logikai mű- velet hamis vagy igaz eredménnyel zárult-e. A nulla jezőbit (BZERO) a DMACONR regiszterben magas lesz, ha a logikai művelet eredménye hamis. Ezt a funkciót általában ütközések vizsgálatára használhatjuk. Átfedi-e egymást a két kép? A logikai művelet ekkor $D=AB$, ha ez igaz lesz (logi- kai és A,B között), akkor nem jelez a bit. Ha csak az átfedést (nulla keresést) vizsgáljuk, és a D (adatcél) csatornára nincsen szükségünk, (nem akarunk képeket létrehozni), akkor ezt a D csatornát ki is lehet kapcsolni (BLTCON regiszter USED bitje).

TERÜLET FELTÖLTÉS

Az adatmásolás mellett a BLITTER egyidőben területfeltöltést is tud végezni. Ennek csak egy feltétele van: a feltöltendő terület bal és jobb oldalát egybefüggő vonalnak kell határolni.

Normál feltöltés:

- a módosítót be kell állítani a megfelelő értékre
- a forrás és a cél számlálóit ugyanarra az értékre kell állítani (ha csak feltöltést végez a BLITTER). Ez az érték a legmagasabb cím kell hogy legyen.
- be kell állítani a csökkenő üzemmódot (a feltöltési műveletek csak így működnek).
- Az ún. FCI kontroll-bitet használva beállíthatjuk a feltöltés módját. Ez a BLTCON1 regiszter 2.bitje. Ha ez 0, akkor a határok között, ha 1, akkor a határokon kívül tölt fel. Az FCI úgy működik, hogy minden 1 értékű bit után megváltoztatja a töltési módot.
- A BLTSIZE regiszterbe meg kell határozni a feltöltési utasítások alá tartozó négyszög méreteit (ez azért fontos, mert a vonal nem csak függőleges lehet a feltöltendő terület határán).

A kizáró feltöltés:

A BLTCON1 regiszteren belül kétféle feltöltési mód választható: Az IFE (normál), ill. az EFE (kizáró) feltöltés. A kizáró feltöltés azt jelen- ti, hogy töltés közben a bal oldali sávhatárt a BLITTER automatikusan eltávolítja. Ez az éles sávhatárok kialakításánál használható jól. A feltöltés műveletei hasonlóak az előzőkhöz.

VONAL RAJZOLÁSA

A fent leírt funkciók mellett a BLITTER vonalrajzoló üzemmóddal is rendelkezik. Ezt a BLTCON1 0.bitjének magasra állításával választhatjuk. A BLITTER ekkor a következő műveleteket tudja:

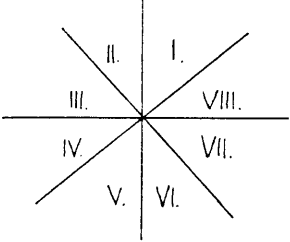
- 1024 pontnyi vonal rajzolása
- Inverz vagy normál színű vonal húzása
- Egybefüggő vagy mintázott vonal rajzolása

Ebben az üzemmódban a BLITTER regisztereinek egy része más feladatot végez el mint egyébként.

SÍKNYOLCADOK HASZNÁLATA

A síknyolcadok használata egyszerűsíti a vonal megrajzolását az X,-X,Y,-Y tengelyek közötti szimmetria kihasználásával.

Az AMIGA síknyolcad felosztását a BLTCON1 regiszter 4,3,2 bitjeiben állíthatjuk be a következő módon:

	4	3	2		4	3	2	bitek
	I	1	1	0	V	1	0	1
	II	0	0	1	VI	1	0	0
	III	0	1	1	VII	0	0	0
	IV	1	1	1	VIII	1	0	0
Síknyolcadok								

A BLITTER MŰVELETEK ÉS A RENDSZER DMA

A BLITTER műveletei a rendszer többi részeinek működésére is hatással van.

Blitter DMA elsőbbsége:

A Blitter különböző adatelérési, módosítási és tárolási műveleteit a DMA kérelmeken keresztül hajtja végre, és ez a memóriaelérést megosztja a rendszer többi eszközeivel. A memóriát elérő minden eszköznek van egy elsőbbségi szintje rendelve. A lemezegység DMA, hanggenerátor DMA, bit-térkép DMA és sprite DMA mindegyike rendelkezik a legmagasabb elsőbbségi szinttel. Bizonyos körülmények között a bittérkép DMA elsőbbséget élvez a sprite DMA-val szemben. A video sugár minden egyes vízszintes pásztázása közben ezen négy eszköz mindegyikéhez az időzítési rések egy csoportja van hozzárendelve. Ha egy eszköz nem kéri a hozzárendelt időzítési adagját, azokat más eszközök használják el. Ezeknek az eszközöknek van adva az elsőbbség, mert hibás DMA ciklusok esetén adatvesztést okoznak (Pl. kimaradó hang vagy rásztersor stb.) A COPPER közvetlenül ezen elsőbbségi szint után következik, a legalsó csőnyabb szint a BLITTER-nek, majd (ez alá rendelve) a 68000-es processzornak van. A BLITTER elsőbbségénél fogva sokkal gyorsabban végzi az adatmozgatást (Kb 70MHz órajelnek megfelelően).

DMA IDŐZÍTÉSI RÉSZ HOZZÁRENDELÉS

Egy vízszintes pásztázási sor alatt (63 Mikrosec) 227.5 szinusz órajel elérési ciklus van. Egy memóriaciklus kb.280 mikrosec időtartamú. Az időzítési rész hozzárendelése vonalanként a következő:

- 4 ciklus a memória frissítésére
- 3 ciklus a lemezegység DMA (2 byte csatornánként)
- 16 ciklus sprite DMA számára (2 byte csatornánként)
- 80 ciklus a bittérkép DMA számára

BITTÉRKÉP / PROCESSZOR BUSZ MEGOSZLÁS

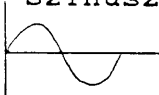
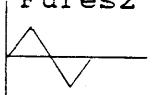
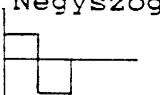
A 68000-es csak a páros számú memóriaelérési ciklusokat használja. A 68000-es a teljes processzor utasítási időnek kb.felét a belső műveletek futtatására, másik felét a memória elérésére fordítja. Néhány 68000-es utasítás nem illeszkedik tökéletesen a páros számú ciklusok hozzárendeléséhez és ciklusok elvesztését okozhatja. Ilyenkor a 68000-esnek várni kell a következő memória résig. Mindazonáltal a legtöbb utasítás nem okoz ciklusvesztést, így a processzor teljes sebességgel fut az idő legnagyobb részében ha a blitter DMA nem akadályozza. Ha a képernyőn négy vagy kevesebb alacsony felbontású (320x...) bittérképet alkalmazunk, a 68000-es választható memóriaciklusokat kaphat, ha ennél több bittérképet használunk a képfrissítés kezdő ellopni a szabad részeket. (Pl. 6 bittérképnél az időzítési rész közel 50%-át használja fel a bittérképre)

A BLITTER MŰVELETI HATÁSAI

Amin már említettem általában a blitter magasabb prioritással rendelkezik mint a processzor a DMA ciklusok tekintetében, azonban vannak bizonyos esetek, amikor a két eszköz megosztja a ciklusidőt. Ha van lehetősége a blitter ellop minden szabad ciklust, ez akkor aktív, ha a BLTPRI (Blitter elsőbbség bit) kapcsolja a DMACON regiszterben. Ha a BLTPRI 1, akkor a blitter minden elérhető ciklust megkaparint (ez potenciálisan minden ciklus is lehet). Ha ez a bit 0, akkor a DMA vezérlő megvizsgálja a processzor cikluskérését, ha három ciklust a blitter működése miatt nem kap, akkor a negyediket odaadja.

BEVEZETÉS AZ AKUSZTIKÁBA

A hang a levegő rezgéseinek segítségével kerül fülünkbe, amely végül a hangot számunkra hallhatóvá teszi. Egy hang ábrázolható grafikusan is, a légnyomás, és az idő függvényében. Minden hang leírható egy rá jellemző grafikonnal. A hanghullámok egy ismétlődő szakaszát nevezzük a hang periódusának. A zörejszerű hangok hullámformája nem ismétlődő. A periódusok ismétlődő gyakorisága a frekvencia, ez határozza meg egy hang magasságát. A frekvencia mértékegysége a Hertz (Hz). $1 \text{ Hz} = 1 \text{ periódus/másodperc}$. Ha magasabb a frekvencia, akkor magasabb a hang is. Az ember $20\text{Hz}-20\text{KHz}$ -ig terjedő tartományt hallja. Egy denevér kb. 200 KHz -ig hall. A hullámforma amplitudója (nagysága) a hangerőt, maga a hullámforma a hangszínt, a hang minőségét határozza meg. A mellékelt ábrán egy szabályos szinuszhullámot láthatunk, az eddig szóba hozott alapfogalmak feltüntetésével. Egy hang elektronikus tárolásához a hang jellemzőit kell rögzítenünk. Minél több információt tárolunk, a hangminőség annál jobb lesz (HIFI). A számítógép nem képes analóg hangok tárolására, emiatt a hangot valamilyen módszerrel át kell alakítani (digitális) numerikus információvá. Ezt a feladatot analóg-digitális átalakítóval lehet megoldani.

	127	Színusz	Fűrész	Négyszög
	-127			
mintavételi	0	0	0	127
szeletek	1	39	20	127
	2	75	40	127
	3	103	60	127
	4	121	80	127
	5	127	100	127
	6	121	00	-127
	7	103	60	-127
	8	75	40	-127
	9	39	20	-127
	10	0	0	-127

A digitalizált mintavétel elve

AZ AMIGA HANGKÉPZÉSE

Az AMIGA négy hangcsatornával rendelkezik. Bármelyik csatorna a program futásától függetlenül képes komplex hangok megszólaltatására. A csatornák mindegyike képes a többi modulálására. A csatornák párosával, jobb és bal oldalként vannak kivezetve (sztereó). A hangcsatornák mindegyike egy-egy 8 bites digitál-analóg átalakítót tartalmaz, melyek közvetlen memóriáhozáféréssel dolgoznak (DMA). A DMA egy rastersoron belül két hangmintát közvetít az átalakítóknak. Ha a hang állandó, a DMA automatikusan biztosítja a folyamatos hangminta vételt. Az AMIGA-n két módja van a hang létrehozásának. Az automatikus (DMA), és a direkt (manuális) mód.

HANG MEGSZÓLALTATÁSA

Egy hang megszólaltatása a következő lépésekből áll:

- csatornák számának meghatározása
- a kívánt hullámformák (a hangminták) rögzítése a memóriában
- a hardware címregiszterek beállítása
- a hangerő beállítása
- a periódus beállítása
- a kívánt csatornák engedélyezése, hang DMA start

CSATORNÁK SZÁMA

A 0-3 csatornák a bal oldali kimenetre, a 1-2 csatornák a jobb oldali kimenetre van kötve. A csatornák számának kijelölésénél vegyük figyelembe ezt is.

HULLÁMFORMÁK DEFINIÁLÁSA ÉS RÖGZÍTÉSE

E fejezetben egy tiszta hangot adó szinuszos hullámot fogunk létrehozni. A hangmintákat egy-egy nyolc bites adat írja le, ezáltal egy adat-word két mintát tartalmaz. A minták értéke 128-tól 127-ig terjed. Az első hangmintát leíró byte-nak páros címnek kell kezdődnie! A hangminta címe beírandó a megfelelő AUDxLC regiszterbe. Mivel ez a regiszter is csak 19 bites címet tud kezelni, emiatt a hangminták is csak az alsó 512K-ban helyezkedhetnek el (a DMA csak az alsó fél MByte-ban tud dolgozni).

```
AUD0LC:          EQU      AUD0LCH
WHERE0DATA:       LEA      AUDI0DATA,A0
                  MOVE.L   A0,AUD0LC
```

A hangminta hossza word-okban az AUDxLEN regiszterbe írandó. A mintapéldában a hossz 16 word.

```
SETAUD0LENGHT:   MOVE.W   #16,AUD0LEN
```

HANGERŐ BEÁLLÍTÁSA

A hangerő csatornánként külön-külön állítható. A hangerő beállítására 6 bit áll rendelkezésre, amelyek segítségével 0-64 hangerő érték állítható be. A hangerő értéke az AUDxVOL regiszterbe írandó. Ha a hangerő a maximálisra van állítva (64), és a hangminta amplitúdója maximális (-128,+127), akkor a kimeneten +- 8.4V kimeneti jelet (800mV p-p) kapunk.

```
SETAUD0VOLUME:   MOVE.W   #48,AUD0VOL
```

PERIÓDUS BEÁLLÍTÁSA

A minták visszajátszásakor létrejövő hang magassága az egyes lejátszási sebességével (perioduseval) változtatható. Nagyobb magasságú hang gyorsabb lejátszási sebességgel oldható meg. E célra csatornánként egy-egy 16 bites regiszter szolgál. A regiszter egy egyszerű számlálót szimulál. A regiszterbe beírt értéktől 0-ig számol el a gép (minden egyes hang DMA ciklusban egyel csökkentve a regiszterben lévő értéket), mielőtt a következő mintát elküldi a digitál-analóg átalakítóknak. Emiatt nagyobb beírt érték kisebb hangmagasságot eredményez. A minimálisan beírható érték 124, maximális 65535.

A MINTVÉTELI PERIÓDUS ÉRTÉKÉNEK KORLÁTOZÁSA

A mintavételi időzítő érték minimális korlátja amiatt van, hogy a DMA egy rasztorsoron belül csak meghatározott ideig bírja kiszolgálni a hangot. A hangcsatorna egy soron belül két hangmintát (16 bitet) kap a DMA-tól. A maximális mintavételi frekvencia a következő képlet alapján kiszámítható:

$2 \text{ hangminta/rasztorsor} * 262.5 \text{ rasztorsor/kép} * 59.94 \text{ kép/sec} = 31469$
A képlet eredménye elméleti maximum, a hardware miatt az érték lecsökken 28867-re. A rendszer órajele 279.365 nanosec-onként ismétlődő impulzusok sorozata (7.19MHz). A maximális mintavételi sebesség: 28867 hangminta/sec - 34642 mikrosec/hangminta A minimális időzítő érték. (34642 mikrosec/minta) / (0.279365 mikrosec/ciklus) = 124 időzítő érték/hangminta A fentiek miatt, amennyiben 124-nél kisebb értéket írunk be időzítés-ként, azt 0-nak tekintí a gép, és nem történik semm várákozás két hang- minta-csomag elküldése között.

AZ IDŐZÍTŐ ÉRTÉK MEGHATÁROZÁSA

Időzítő érték = a kívánt periódus idő / a rendszer órajelének periódusideje

Ha szinuszos hangot kívánunk megszólaltatni, akkor a következő hang- minta táblázatot építsük fel a memóriában:

0,90,127,90,0,-90,-127,-90

1 kHz-es esetében: egy ciklus 1/1000-ed másodpercig tart. Mivel egy ciklus 8 darab 16 bites mintából áll össze, a hangmintákat 1/8000-ed másodpercenként kell megszólaltatni. Ehhez a beállítandó időzítő érték: $(1/8000)/(0.279365) = 125$ mikrosec/minta $(0.279) = 447$ Az időzítő regiszterbe (AUDxPER) 1 KHz-es, 8 hangmintából álló hangnál 447-et kell írni:

SETAUD0PERIOD: MOVE.W #447,AUD0PER

Jó minőségű hangminták úgy hozhatók létre, ha az egyes hangminta érékek között a lehető legkisebb idő telik el.

A HANG MEGSZÓLALATÁSA

Ha az eddig említett értékeket beállítottuk, akkor a hang megszólal- tatható a az audio DMA engedélyezésével. A kívánt hangcsatorna a DMACON regiszter megfelelő AUDxEN bitjének magasra állításával kapcsolható be. A hang DMA indul, amennyiben DMAEN bit magas a DMACON regiszterben:

Bit 15	SETCLR	Ha ez a bit magas, a regiszter bitjei beállítódnak, egyébként törlődnek
Bit 9	DMAEN	Audio DMA engedélyezése. 1-be, 0-ki
Bit 3	AUD3EN	A harmadik hangcsatorna engedélyezése
Bit 2	AUD2EN	A második hangcsatorna engedélyezése
Bit 1	AUD1EN	Az első hangcsatorna engedélyezése
Bit 0	AUD0EN	A nulladik hangcsatorna engedélyezése

A következő példában bekapcsoljuk a 0. csatornát, és engedélyezzük a hang DMA-t:

```
SET:          EQU      $08000
AUD0EN:       EQU      $01
DMAEN:        EQU      $0200
BEGINCHAM0:   MOVE.W   #(SET+AUD0EN+DMAEN),DMACON1
```

HANG DMA KIKAPCSOLÁSA:

A megfelelő AUDxEN bit törlésekor az adott csatorna elhallgat. Hasonlóan ha a bitet beállítjuk, a csatorna megszólal:

```
CLEAR:        EQU      0
STOPAUDCHAN0: MOVE.W   #(CLEAR+AUD0EN),DMACON1
```

Következő mintaprogram megszólaltat a 0. hangcsatornán egy 1KHz-es szinuszos hangot:

```
AUD0LC:       EQU      AUD0LCH
SET:          EQU      $08000
CLEAR:        EQU      0
AUD0EN:       EQU      $01
DMAEN:        EQU      $0200
              DC.W      0
SINEDATA:     DC.B      0,90,127,90,0,-90,-127,-90
MAIN:         LEA       SINEDATA,A0 ; Mutató a hangminták kezdetére
WHERE0DATA:   MOVE.L    A0,AUD0LC   ; A hangminták címének beírása
SETAUD0LENGHT: MOVE.W   #4,AUD0LEN  ; Hangminta hossza word-ökben
SETAUD0VOLUME: MOVE.W   #64,AUD0VOL ; Maximális hangerő a 0.csatornára
SETAUD0PERIOD: MOVE.W   #447,AUD0PER ; Időzítés 1KHz-hez
BEGINCHAM0:   MOVE.W   #(SET+DMAEN+AUD0EN),DMACON1
```

KOMPLEX HANGOK MEGSZÓLALTATÁSA

Ha a hang megszólalása közben módosítjuk a hangregisztereke, akkor ér- dekes hatásokat érhetünk el. A következő példa szinusz és háromszög formájú hangokat felváltva szólaltat meg.

Ha (hullám=háromszög), akkor AUD0LCL-be a szinuszos adatok címe.

Ha (hullám=szinusz), akkor AUD0LCL-be a háromszög adatok címe.

HANGMODULÁCIÓK

AMPLITUDÓ MODULÁCIÓ

A hangerőt kell változtatgatni a hang megszólalásának ideje alatt. Az AUDxVOL regiszter felépítése:

Bit	15-7	Nem használt	
Bit	6	Maximális hangerő (64)	(V6)
Bit	5-0	További hangerő bitek (1-63)	(V5-V0)

FREKVENCIA MODULÁCIÓ

Az időzítő értéket kell változtatgatni a hang megszólalásának ideje alatt. Az AUDxPER regiszter felépítése:

Bit	15-0	16 bites időzítő érték (124-65535)	(P15-P0)
-----	------	------------------------------------	----------

Ha egy csatorna kimenő jelével kívánunk egy másikat modulálni, akkor használjuk az ADKCONR regiszter bitjeit:

Bit	15	SET/CLR	Ha ez a bit magas, a regiszter bitjei beállítódnak, egyébként törlődnek.
Bit	7	ATPER3	A harmadik csatorna frekvencia moduláció céljára használva, kimenő jele nincs.
Bit	6	ATPER2	A 2.csatornát frekvenciamodulálja a 3.csatorna
Bit	5	ATPER1	Az 1.csatornát frekvenciamodulálja a 2.csatorna
Bit	4	ATPER0	A 0.csatornát frekvenciamodulálja az 1.csatorna
Bit	3	ATVOL3	A 3.csatorna amplitudó moduláció céljára használva, kimenő jele nincs.
Bit	2	ATVOL2	A 2.csatornát amplitudó modulálja a 3.csatorna
Bit	1	ATVOL1	Az 1.csatornát amplitudó modulálja a 2.csatorna
Bit	0	ATVOL0	A 0.csatornát amplitudó modulálja az 1.csatorna

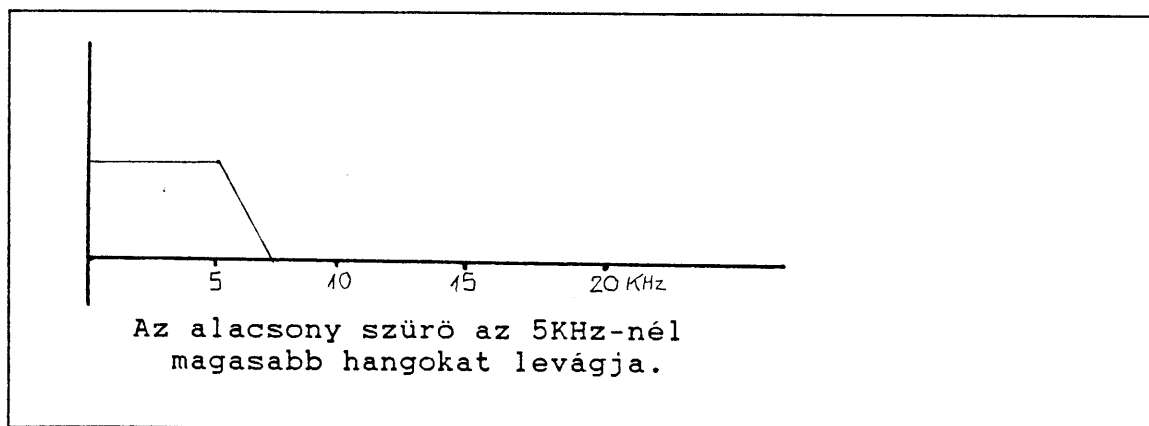
Amennyiben egy csatornára mindkét modulációs bitet bekapcsoljuk, a csatorna elnémul.

HIFI HANGOK LÉTREHOZÁSA

A következő szempontok betartásával növelhető a hanghűség, optimális esetben közel CD minőség is elérhető.

- A hullám átmenetek zökkenőmentesek legyenek
- Figyeljünk arra, hogy lehetőleg azon az értéken fejeződjön be amelyen elkezdődött. Így elkerülhetőek a kapcsolási hangok, a zörejek.
- Egy hang minél több hangmintából álljon. Lehetőleg minél több egymástól különböző mintát vegyünk a hangból.
- Két hangminta között a lehető legkisebb legyen a szünet. Minimális időzítő érték legyen.
- A zajokat, torzításokat próbáljuk kiköszöbölni, csökkenteni.
- A minimális mintavételi frekvencia a legnagyobb frekvenciájú összetevőnél is 7KHz-cel nagyobb legyen.
- A hardware magas hangszűrőt kapcsoljuk ki

Minimális időzítő érték	Mintavételi sebesség	Max.kimeneti frekvencia
124	29KHz	7KHz
256	14KHz	7KHz
320	11KHz	4KHz



A szűrő a 7KHz feletti hangokat teljesen levágja, emiatt minimum 256, max. 124 időzítő értékek használatakor a szűrő minden további nélkül kikapcsolható.

DIREKT (NEM DMA) HANGGENERÁLÁS

Létrehozható hang más módon is, igaz ez kissé több munkát követel a programozótól. A hang kimenet puffer regisztereibe direkt MOVE utasításokkal (IRQ rutin segítségével) is írhatunk értékeket, melyeknek megfelelő sebességgel való ismétlése létrehozza a kívánt hangot. Mivel ez a módszer igen nagy mértékben lekötí a processzort, nem tárgyaljuk részletesen.

AZ EGYENLŐEN TEMPERÁLT ZENEI SKÁLA

Időzítés	Hang	Ideális frekvencia (AUDxLEN=0)	Valós frekvencia (AUDxLEN=0)
500	A	440.0	440.4
400	A#	466.2	466.1
453	B	493.9	493.9
428	C	523.3	522.7
404	C#	554.4	553.8
381	D	587.3	587.2
360	D#	622.3	621.4
339	E	659.3	659.9
320	F	690.5	699.1
302	F#	740.0	740.8
285	G	784.0	785.0
269	G#	830.6	831.7
254	A	880.0	880.8
240	A#	932.3	932.2
226	B	987.8	989.9
214	C	1046.5	1045.4
202	C#	1108.7	1107.5
190	D	1174.7	1177.5
180	D#	1244.5	1242.9
170	E	1318.5	1316.0
160	F	1396.9	1398.3
151	F#	1400.0	1481.6
143	G	1560.0	1564.5
135	G#	1661.2	1657.2

DECIBEL ÉRTÉKEK A HANGERŐSSÉG EGYES SZINTJEIHEZ

Hangerő	Decibel	Hangerő érték	Decibel érték

64	0.0	32	-6.0
63	-0.1	31	-6.3
62	-0.3	30	-6.6
61	-0.4	29	-6.9
60	-0.6	28	-7.2
59	-0.7	27	-7.5
58	-0.9	26	-7.8
57	-1.0	25	-8.2
56	-1.2	24	-8.5
55	-1.3	23	-8.9
54	-1.5	22	-9.3
53	-1.6	21	-9.7
52	-1.8	20	-10.1
51	-2.0	19	-10.5
50	-2.1	18	-11.0
49	-2.3	17	-11.5
48	-2.5	16	-12.0
47	-2.7	15	-12.6
46	-2.9	14	-13.2
45	-3.1	13	-13.8
44	-3.3	12	-14.5
43	-3.5	11	-15.3
42	-3.7	10	-16.1
41	-3.9	9	-17.0
40	-4.1	8	-18.1
39	-4.3	7	-19.2
38	-4.5	6	-20.6
37	-4.8	5	-22.1
36	-5.0	4	-24.1
35	-5.2	3	-26.6
34	-5.5	2	-30.1
33	-5.8	1	-36.1
		0	-végtelen

REGISZTEREK

Ezeket az értékeket \$DFF000 báziscímhez hozzá kell adni.
A *-gal jelzett bitek nem használtak.

BLTxPTH	x=A	050	Az x BLITTER csatorna számláló regisz-
	x=B	04C	párjai, a BLITTER DMA forrás (A,B,C) és
	x=C	048	iránycímét (D) tartalmazza. Fel kell töl-
	x=D	054	teni a BLITTER által mozgatott adatok
BLTxPTL	x=A	052	kezdőcímével. Miután a BLITTER végrehaj-
	x=B	04E	totta a műveleteket, az utolsó adat
	x=C	04A	címet tartalmazza (+MODULO+növekmény).
	x=D	056	A BLTxPTH a felső három, a BLTxPzL az
			alsó 15 bitet tartalmazza. Vonalrajz-
			olásnál. A BLTAPL akkumlátor és a (2Y-X)
			kezőértékkel kell feltölteni, ahol Y/X a
			vonallal meredeksége. A BLTCPT és BLTDPT-
			nek a vonal kezdőcímét kell tartalmaznia.

BLTxMOD	x=A	064	Ez az x BLITTER modulo. A modulo egy
	x=B	062	olyan szám, amelyet egy sor végén
	x=C	060	hozzáad a HW, hogy a következő sor
	x=D	066	elejét kapja meg. Mind a 4 területnek
			saját moduloja van, így különböző
			méretűek lehetnek a másolandó és a
			célterületek.

Vonalhúzásnál: A BLTAMOD és BLTBMOD meredekség tároló regiszter, és eredetileg a (4Y-4X), ill. 4Y értékekkel kell feltölteni. Y/X a meredekség A BLTCMOD és BLTDMOD regisztereket a szélesség értékével kell feltölteni. Ez azon területet jelenti, ahova a vonalat húzni akarjuk (ez általában a képernyő szélesség).

BLTAFHW	044	Maszk regiszter első szó.
BLTALHW	046	Maszk regiszter utolsó szó.
		A regiszterekben lévő minta és a másolandó terület között AND művelet hajtódik végre, tehát a másolandó rész elejére és végére bit szélességű adat is kapcsolható.
		Vonalhúzásnál és feltöltéskor minden bitjének 1-nek kell lennie.

BLTxDAT	x=A	074	Az x BLITTER forrás adatregiszter. Ez tartalmazza az x(A,B,C) forrás adatait, ezt általában a BLITTER DMA írja, de használhatja a processzor is.
---------	-----	-----	--

Vonalhúzásnál: A BLTADAT indexregiszterként szerepel, és 8000-val kell feltölteni. A BLTBDAT a mintázatot tartalmazza, \$FF a tartalma, ha a vonal folyamatos.

BLTDDAT	080	A BLITTER irányregiszter. A forrásregiszterekből a logikai műveletek végrehajtása után az adatok ide kerülnek (átmeneti tárolás)
---------	-----	--

BLTCON0	040	A BLITTER vezérlő regisztere. Két fő állapotuk van, területmozgatás és vonalhúzás, ezt a BLTCON1 0.bitjébe beállíthatjuk.
BLTCON1	042	

BLITTER TERÜLETFELTÖLTÉSÉNél

Bit	BLTCON0	BLTCON1		
15	ASH3	BSH3	ASH0-3	A forrás léptetési érték
14	ASH2	BSH2	BSH0-3	B forrás léptetési érték
13	ASH1	BSH1	USEA-D	forrás vezérlő bitek
12	ASH0	BSH0	LF0-7	logikai funkciók bitjei
11	USEA	*	EFE	kizáró feltöltés beállít.
10	USEB	*	IFE	beleértő feltöltés beáll.
09	USEC	*	FCI	feltöltés átvitel bit
08	USED	*	DESC	csökkenő címzés bit
07	LF7	*	LINE	vonalhúzás bekapcsolás most 0
06	.	*		
05	.	*		
04	.	EFE		
03	.	IFE		
02	.	FCI		
01	.	DESC		
00	LF0	LINE(=0)		

BLITTER VONALHÚZÁSNAI

15	START3	TEXTURE3	START0-3 sor kezdőpontja \$0-15
14	START2	TEXTURE2	TEXTURE0-3 vonalhúzási minták
13	START1	TEXTURE1	LF0-7 4A-val kell feltölteni
12	START0	TEXTURE0	SIGN előjel értéke
11	1	0	SING 1bit vízszintes vonalanként
10	0	0	SUD Néha fel vagy le
09	1	0	SUL Néha fel vagy balra
08	1	0	AUL Mindig fel és balra
07	LF7	0	
06	.	SIGN	
05	.	0	
04	.	SUD	
03	.	SUL	
02	.	AUL	
01	.	SING	
00	LF0	LINE(=1)	

MEGJEGYZÉS: Ha a SUL, SUD, AUL bitek a vonalhúzásnál sik-
nyolcadokat jelzik.

	SUD	SUL	AUL
0	1	1	0
1	0	0	1
2	0	1	1
3	1	1	1
4	1	0	1
5	0	1	0
6	0	0	0
7	1	0	0

BLTSIZE 058 BLITTER kezdet és méret (szélesség és magasság).
Vonalhúzásnál: szélesség=2, magasság=a vonal hossza. Ennek
a regiszternek az írása indítja meg a BLITTER műveletet,
tehát ezt töltjük fel utoljára.

BIT 15 14 13 12 11 10 09 08 07 06 05 04 03 02 01 00
h9 h8 h7 h6 h5 h4 h3 h2 h1 h0 w5 w4 w3 w2 w1 w0
h0-9 függőleges lépésköz (1024 pixel max, 1 bit=1 pixel)
w0-5 vízszintes lépésköz (1024 pixel max, 1 bit=16 pixel)

BPLxPTH x=1 0E0 y=1 0E2 A bittérképek címei (1-6)
BPLyPTL x=2 0E4 y=2 0E6 A BPLxPTH a felső 3 bitet,
x=3 0E8 y=3 0EA a BPLyPTL az alsó 15 bitet
x=4 0EC y=4 0EE tartalmazza
x=6 0F0 y=5 0F2
x=6 0F4 y=6 0F6

BPLxDAT x=1 110 Bittérkép adatok (párhuzamos, soros átala-
x=2 112 kítás). Ezek a regiszterek fogadják RAM-ból
x=3 114 érkező adatokat. A COPPER használja, de a
x=4 116 processzor is írhatja, gyakorlatilag puffer
x=5 118 funkciókat lát el.
x=6 11A

BPL1MOD 108 Bittérkép modulo páratlan (1,3,5)
BPL2MOD 10A Bittérkép modulo páros (2,4,6) a bittér-
képekhez. A modulo az az érték amit a sor
végén a HW hozzáad a számláló értékéhez,
hogy a következő sor elejére mutasson.

BPLCON0 100 Kontroll regiszterek - kontrollbitek
BPLCON1 102 - vízszintes scroll
BPLCON2 104 - képernyő prioritás

BIT	BPLCON0	BPLCON1	BPLCON2	
15	HIRES	*	*	HIRES 640-es felbontás
14	BPU2	*	*	BPL0-2 bittérképek
13	BPU1	*	*	bekapcsolása
12	BPU0	*	*	HOMOD kitartás és mó-
11	HOMOD	*	*	dosítás mód
10	DBLPF	*	*	DPLPF Dual Playfield
09	COLOR	*	*	COLOR összetevő videó
08	GAUD	*	*	szin keverés
07	*	PF2H3	*	GAUD GENLOCK hang be-
06	*	PF2H2	PF2PRI	állítás (BKGND-re)
05	*	PF2H1	PF2P2	LPEN fényceruza be
04	*	PF2H0	PF2P1	LACE interlace be
03	LPEN	PF1H3	PF2P0	ERSY külső szinkron az
02	LACE	PF1H2	PF1P2	input (HSYNC, VSYNC
01	ERSY	PF1H1	PF1P1	pad-ek)
00	*	PF1H0	PF1P0	PF2PRI 2.játékmező az
				1. felett

PF2P0-2 2.játékmező prioritása a sprite-okkal szemben

PF1P0-2 1.játékmező prioritása a sprite-okkal szemben

PF2H0-3 2.játékmező vízszintes scroll

PF1H0-3 1.játékmező vízszintes scroll

CLXCON 098 Ütközés vezérlés. Ez a regiszter ellenőrzi, hogy mely bittérképekre vonatkozik és ha az ütközés bekövetkezik, mi legyen az állapotuk. Ezen kívül vezérli a páratlan sprite-ok bekapcsolását egyenként az ütközés vizsgálatába azáltal, hogy logikai or-t hajt végre a páros számú sprite-okkal.

BIT

15	ENSP7	bekapcsolja a 7.sprite-ot (OR a 6-kal)
14	ENSP5	bekapcsolja az 5.sprite-ot (OR a 4-el)
13	ENSP3	bekapcsolja a 3.sprite-ot (OR a 2-kel)
12	ENSP1	bekapcsolja az 1.sprite-ot (OR a 0-kal)
11	ENBP6	bekapcs. a 6.bittérkép ütközését
10	ENBP5	bekapcs. az 5.bittérkép ütközését
09	ENBP4	bekapcs. a 4.bittérkép ütközését
08	ENBP3	bekapcs. a 3.bittérkép ütközését
07	ENBP2	bekapcs. a 2.bittérkép ütközését
06	ENBP1	bekapcs. az 1.bittérkép ütközését
05	MVBP6	illesztési érték 6.bittérkép ütközéséhez
04	MVBP5	illesztési érték 5.bittérkép ütközéséhez
03	MVBP4	illesztési érték 4.bittérkép ütközéséhez
02	MVBP3	illesztési érték 3.bittérkép ütközéséhez
01	MVBP2	illesztési érték 2.bittérkép ütközéséhez
00	MVBP1	illesztési érték 1.bittérkép ütközéséhez

Az ütközések vizsgálatához illesztési érték is szükséges. A bekapcsolt bittérképek mindig ütköznek, ezért ha minden bittérkép ki van kapcsolva, akkor állandóan ütközést indukál az illesztési értéktől függetlenül.

CLXDAT 00E ütközési adatregiszter.

BIT

15 Nem használt
14 4.(vagy 5.) sprite ütközése a 6.(vagy 7.)-kel szemben
13 2.(vagy 3.) sprite ütközése a 6.(vagy 7.)-kel szemben
12 2.(vagy 3.) sprite ütközése a 4.(vagy 5.)-kel szemben
11 0.(vagy 1.) sprite ütközése a 6.(vagy 7.)-kel szemben
10 0.(vagy 1.) sprite ütközése a 4.(vagy 5.)-kel szemben
09 0.(vagy 1.) sprite ütközése a 2.(vagy 3.)-kel szemben
08 2.játékmező ütközése a 6.(vagy 7.) sprite-tal
07 2.játékmező ütközése a 4.(vagy 5.) sprite-tal
06 2.játékmező ütközése a 2.(vagy 3.) sprite-tal
05 2.játékmező ütközése a 0.(vagy 1.) sprite-tal
04 1.játékmező ütközése a 6.(vagy 7.) sprite-tal
03 1.játékmező ütközése a 4.(vagy 5.) sprite-tal
02 1.játékmező ütközése a 2.(vagy 3.) sprite-tal
01 1.játékmező ütközése a 0.(vagy 1.) sprite-tal
00 1.játékmező ütközése a 2.játékmezőével

COLORxx 00=180 32 darab színregiszter van (0-31), és ezeket
 01=182 együtt színpalettának nevezik. 12 bites kódokat
 . tartalmaznak, melyek 4-4-4 bittel a zöld, a vörös
 . és a kék színeket tartalmazzák (RGB)
 10=194 BIT 15 14 13 12 11 10 09 08 07 06 05 04 03 02 01 00
 . * * * * R3 R2 R1 R0 G3 G2 G1 G0 B3 B2 B1 B0
 . R0-3 A vörös színösszetevő
 30=1BC G0-3 A zöld színösszetevő
 31=1BE B0-3 A kék színösszetevő

COPCON 02E COPPER vezérlő regiszter. Ez a regiszter csak egy
 hasznos bitet tartalmaz (CDANG, 01.bit).
 Amennyiben ennek értéke 1, megengedi a COPPER a
 BLITTER hardware elérését (ez a reset hatására
 törlődik).

COPJMP1 088 COPPER újraindítási regiszterek.
COPJMP2 08A A címeket a rendszer kis időintervallumonként
 vizsgálja. Ha írunk beléjük, a COPPER indirekt
 ugrást hajt végre az 1.vagy 2.COPPER első helyzet
 regiszterekben lévő címmel. A COPPER maga is
 tudja ezeket írni.

COP1LCH 080 COPPER 1. első helyzetregiszter felső 3 bitje
COP1LCL 082 alsó 15 bitje
COP2LCH 084 COPPER 2. első helyzetregiszter felső 3 bitje
COP2LCL 086 alsó 15 bitje
 Ezek a regiszterek tartalmazzák a fent leírt ugrócí-
 meket.

COPINS 08C COPPER utasítás-elérés azonosító.
A címen lévő adatot a copper akkor generálja,
amikor utasításokat tölt a saját utasítás
regiszterébe. Ez lényegében minden copper
ciklusban megtörténik, kivéve a MOVE utasítás
2.ciklusát (IR2)

BIT	MOVE		WAIT		SKIP
	IR1	IR2	IR1	IR2	
15	*	RD15	VP7	BFD	Az IR1 és IR2
14	*	RD14	VP6	VE6	regisztereinek
13	*	.	.	VE5	elnevezése azo-
12	*	.	.	.	nos a WAIT utas.
11	*	.	.	.	IR1, IR2 tartal-
10	*	RD10	.	.	mával.
09	*	.	VP1	VE1	
08	DA8	.	VP0	VE0	
07	DA7	.	HP8	HE8	
06	.	.	HP7	HE7	
05	.	.	.	HE6	
04	.	.	.	.	
03	.	.	.	.	
02	.	.	.	.	
01	DA1	RD01	HP2	HE2	
00	0	RD00	1	0	
IR1	Első utasítás regiszter				
IR2	Második utasítás regiszter				
DA1-8	Irány cím a MOVE számára. Az IR1 elérési ideje alatt íródik be (RGA haszn.)				
RD0-15	RAM adatok amelyeket a MOVE mozgat az IR2 elérési idő alatt (DA megadott címre)				
VP0-7	Függőleges nyaláb helyzet összehasonlító bitek				
VE0-6	Maszk bitek (összehasonlítás engedélyezve)				
HP2-8	Vízszintes nyaláb összehasonlító bitek				
HE2-8	Maszk bitek (összehasonlítások engedélyezése)				
BFD	BLITTER készen bekapcsoló				

Ha a bit igaz, a blitter készen jelzés nem lesz hatással a copperre. Ha a bit nulla, akkor a blitter készen jelzésnek igaznak kell lennie (a többi bit összehasonlítás mellett) mielőtt a copper beléphetne a várakozó helyzetből, vagy egy új utasításra ugorhatna. Nem szabad elfelejteni, hogy a VP7 összehasonlítás bitet nem lehet maszkolni. A copper alapvetően egy két ciklusos szerkezet amely a buszt csak a páratlan órajel ciklusokban használja (4 memória ciklus lesz új utasításonként). Ezzel megelőzhető a kérelmek ütközése a kijelzővel, hanggal, diszkkal, memória frissítéssel ill. sprite-okkal, amelyek a páros számú ciklusokban dolgoznak. Ezért a copper-nek csak a blitter és a processzor fölött kell elsőbbséget élveznie. Két indirekt ugrási regiszter van (COP1LC és COP2LC), ezek 19 bites mutatóregiszterek, melyek értéke módosítja a program-számlálót ugrások inicializálásához. Az adatokat átviszi a rendszer programszámlálóba, amikor a COPJMP1 ill. COPJMP2 címet írjuk. Emellett a COP1LC-t használja még a rendszer a minden egyes függőleges késleltetési idő kezdetén. Fontos, hogy az ugró regiszterek egyike inicializálódjon és az ugrásvezérlő COPJMP regiszter értéket kapjon bekapcsolás után közvetlenül, mielőtt majd a copper DMA inicializálódik. Ez egymeghatározott kezdőcímet és állapotot tud így biztosítani.

DIWSTRT 08E Kijelző ablak kezdete (bal felső sarok)
 DIWSTOP 09E Kijelző ablak vége (jobb alsó sarok)

A képernyő ablak a kijelző hasznos információt tartalmazó területe.

BIT 15 14 13 12 11 10 09 08 07 06 05 04 03 02 01 00
 V7 V6 V5 V4 V3 V2 V1 V0 H7 H6 H5 H4 H3 H2 H1 H0

A DIWSTRT függőlegesen képernyő felső 2/3 részébe, vízszintesen a bal alsó 3/4 részbe kerülhet.

A DIWSTOP függőlegesen az alsó 1/2 részbe (V0=/=V7) és vízszintesen a jobb oldal 1/4 részébe.

DDFSTRT 092 Kijelző adatelérés kezdet
 DDFSTOP 094 Véghelyzet

Ezek a regiszterek irányítják a bittérkép DMA kijelző adatelérés kezdőcímei ill. végcímei időzítését. A függőleges kijelző időzítés adódik a fent leírt képernyőablak kezdet és vég értékéből. A bittérkép modulok függenek a bittérkép vízszintes méretéből és az adatelérési ablak nagyságából.

BIT 15 14 13 12 11 10 09 08 07 06 05 04 03 02 01 00
 * * * * * * * * H8 H7 H6 H5 H4 H3 * *

A * biteket nullára kell állítani.

A következő táblázat a különböző regisztertartalmakhoz mutatják a kezdő és vég időzítést.

DDFSTRT	H8	H7	H6	H5	H4	DDFSTOP	H8	H7	H6	H5	H4
Extra széles	0	0	1	0	1						
Széles	0	0	1	1	0		1	1	0	1	1
Normál	0	0	1	1	1		1	1	0	1	0
Szűk	0	1	0	0	0		1	1	0	0	1

DMACON 096 DMA vezérlés írás (törlés+beállítás)
 DMACONR 002 olvasás (a blitter státuszé is)
 Ez vezérli a DMA összes csatornáját, és tartalmazza a blitter DMA státuszt is.

BIT	FUNKCIÓ	MAGYARÁZAT
15	SET/CLR	Az 1 bitek törlődnek, a 0 nem változik
14	BBUSY	BLITTER foglaltsági állapot bit
13	BZERO	BLITTER logikai 0 állapot bit
12	*	
11	*	
10	BLTPRI	BLITTER DMA elsőbbség a proc. felett
09	DMAEN	Bekapcsol minden DMA-t
08	BPLEN	Bittérkép DMA be
07	COPEN	Copper DMA be
06	BLTEN	Blitter DMA be
05	SPREN	Sprite DMA be
04	DSKEN	Diszk DMA be
03	AUD3EN	A 3. hangcsatorna DMA be
02	AUD2EN	A 2. hangcsatorna DMA be
01	AUD1EN	Az 1. hangcsatorna DMA be
00	AUD0EN	A 0. hangcsatorna DMA be

DSKPTH 020 Diszk mutató felső 3 bit
DSKPTL 022 alsó 15 bit

Ez a regiszterpár tartalmazza a diszk DMA adatok 18 bites címét. Ezeket a címregisztereket a processzornak vagy a COPPER-nek inicializálnia kell mielőtt a diszk DMA-t engedélyezzük.

DSKLEN 024 Diszk hosszúság regiszter

A diszk DMA adatok hosszát (sorszámát) tartalmazza. Két kontroll bitet is tartalmaz.

BIT	FUNKCIÓ	MEGJEGYZÉS
15	DMAEN	Diszk DMA bekapcsolás
14	WRITE	Diszk írás (RAM-ból) ha 1
0-13	LENGHT	Diszk DMA adathossz

DSKDAT 026 Diszk DMA adat írás

DSKDATR 008 olvasás

Ez a DMA adatpuffer. Két byte-nyi adatot tartalmaz, amely vagy a diszkre íródik, vagy onnan olvasódik. Az írást a DSKLEN 14.bitje (ld. fenn) határozza meg. A DMA vezérlő automatikusan ide szállítja az adatokat, amikor az adatok elfogytak, (a hossz 0) akkor megszakítást generál.

DSKBYTR 01A Diszk adat byte és állapot jelző olvasás.

Ez a diszk processzor adatpuffere. Olvasási üzemmódban a diszkről az adatok ebbe a regiszterbe kerülnek.

BIT	FUNKCIÓ	MEGJEGYZÉS
15	DSKBYT	Diszk byte kész
14	DMAON	Végrehajt egy AND-et
13	DISKWRITE	A DSKLEN 14.bitjének másolata
12	WORDEQUAL	Akkor 1, ha a DSKSYNC regiszter éppen a
11-08	*	diszkről vett adatot egyenlíti ki
07-00	DATA	Diszk byte adatok

DSKSYNC 07E Diszk szinkronizáló regiszter. A diszk olvasás szinkronjához tartalmazza az illesztési kódot.

INTEREQ	09C	Megszakítás kérelem írás (törlés+beállítás)
INTEREQR	01E	olvasás

A processzor lekérdezi a biteket, és ha be vannak kapcsolva az processzor megszakítást eredményezhet. Mind a SET, mind a CLEAR (beállítás,törlés) művelet szükséges ahhoz, hogy a adatokat betölthessük a regiszterekbe. Ezek az állapotbitek nem állítódnak alaphelyzetbe automatikusan egy-egy művelet után. A bitkiosztás megegyezik a megszakítás kapcsolóbitek bitkiosztásával (ld.lent).

INTENA	09A	Megszakítás kapcsoló bitek írás (törlés+beállítás)
INTENAR	01C	olvasás

A regiszter a megszakításo kapcsolóbitjeit mutatja.

BIT	FUNKCIÓ	MAGYARÁZAT
15	SET/CLR	1 a bitek törlődnek, 0 nem változik
14	INTEN	Master megszakítás (csak bekapcsolni kell)
13	EXTER	Külső egység megszakítás
12	DSKSIN	A diszk adatot kódolja (illeszti)
11	RBF	A soros port fogadó puffer tele
10	AUD3	A 3. hangcsatorna blokk vége jel
09	AUD2	A 2. hangcsatorna blokk vége jel
08	AUD1	A 1. hangcsatorna blokk vége jel
07	AUD0	A 0. hangcsatorna blokk vége jel
06	BLIT	Blitter készen jel
05	VERTB	Függőleges késleltetés kezdete
04	COPER	Copper megszakítás
03	PORTS	I/O portok vagy I/O időzítők megsz.kérélmek
02	SOFT	Software által megszakításra fenntartva
01	DSKBLK	Diszk blokk vége jel
00	TBE	A soros port átviteli puffer üres

JOY0DAT	00A	Botkormány, egér adatok 0. (bal oldal)
JOY1DAT	00C	1. (jobb oldal) Mindkét cím egy pár 8 bites egérszámlálót olvas.

Mind a 4 számlálót a bemenő csatornákról érkező jelek hajtják meg. Minden számláló 1-es és 0-ás bitje használható arra, hogy a hozzájuk tartozó bemenet állapotát meghatározzuk, így ezek jabemenő jelek a botkormány fogadására is alkalmasak lesznek.
(1,3 bemenő láb=Y 2,4 láb=X)

BIT	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
0DAT	y7	y6	y5	y4	y3	y2	y1	y0	x7	x6	x5	x4	x3	x2	x1	x0
1DAT	y7	y6	y5	y4	y3	y2	y1	y0	x7	x6	x5	x4	x3	x2	x1	x0

A következő táblázat az egér-botkormány kapcsolólábázat használatát mutatja. A lábázat jelei beolvasódnak a DENISE chip-be, az órajel megfelelő ütemében. Nem szabad elfeledkezni, hogy a botkormány funkciói aktiv-alacsony szinten vannak a kapcsolólábazon.

			DENISE MINTAVÉTEL		
LÁB	JOY	MOUSE	LÁB	NÉV	ÓRAJEL
L1	előre	Y	38	M0V	CCK
L2	balra	YQ	38	M0V	CCK*
L3	hátra	Y	9	M0H	CCK
L4	jobbra	YQ	9	M0H	CCK*
R1	előre	Y	39	M1V	CCK
R2	balra	YQ	39	M1V	CCK*
R3	hátra	Y	8	M1H	CCK
R4	jobbra	YQ	8	M1H	CCK*

Miután a mintavétel megtörtént, a csatlakozó lábjeleit az egérszámláló meghajtására használjuk. A bal és jobb botkormány funkció létrehozása érdekében azonban szükséges, hogy logikailag kombináljunk (kizáró vagy) a számláló alsó két bitjét. Ez a következő táblázatban látható:

ÉRZÉKELNI

előre	Y1 xor Y0 (09 bit xor 08 bit)
balra	Y1
hátra	X1 xor X0 (01 bit xor 00 bit)
jobbra	X1

JOYTEST 036 Azonnal írja mindegyik egér botkormány számlálóba az adatokat.

Egér számláló tesztadatok:

BIT	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
0DAT	y7	y6	y5	y4	y3	y2	xx	xx	x7	x6	x5	x4	x3	x2	xx	xx
1DAT	y7	y6	y5	y4	y3	y2	xx	xx	x7	x6	x5	x4	x3	x2	xx	xx

POT0DAT 012 Potméter párok adatai: BAL
POT1DAT 014 JOBB

Ezek a címek egy 8 bites számlálót olvasnak (összesen 4 ilyen van). A számlálókat két irányító csatlakozóról érkező jelek állítják be, melyekhez egyenként két láb tartozik.

BIT	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
JOBB	y7	y6	y5	y4	y3	y2	y1	y0	x7	x6	x5	x4	x3	x2	x1	x0
BAL	y7	y6	y5	y4	y3	y2	y1	y0	x7	x6	x5	x4	x3	x2	x1	x0

CSATLAKOZÓ	IRÁNY	LÁB	*	LÁB	ELNEVEZÉS(a PAULA chipben)
jobb	Y	RY	9	36	POT1Y
jobb	X	RX	5	35	POT1X
bal	Y	LY	9	33	POT0Y
bal	X	LX	5	32	POT0X

POTGO 034 Potméter kapu adatírás és kezdet.

POTGOR 016 Potméter kapu adatolvasás.

Ez a regiszter egy 4 bites kétirányú I/O kaput vezérel, amely ugyanazt a négy lábat használja, mint a fentebb leírt potméter számláló.

BIT	FUNKCIÓ	LEÍRÁS
15	OUTRY	I/O engedélyezve a PAULA 36.lába számára
14	DATRY	I/O adata a PAULA 36. lába számára
13	OUTRX	I/O adata a PAULA 35. lába számára
12	DATRX	I/O adata a PAULA 35. lába számára
11	OUTLY	I/O adata a PAULA 33. lába számára
10	DATLY	I/O adata a PAULA 33. lába számára
09	OUTLX	I/O adata a PAULA 32. lába számára
08	DATLX	I/O adata a PAULA 32. lába számára
07 -01	0	A chip ID kódjának fenntartva(jelenleg0)
00	START	Indítja a potmétereket

REFPTR 028 Frissítés számláló.
Ezt a regisztert a dinamikus RAM címgenerátoraként
használják. Csak tesztelés céljából írható, a processzor
soha nem írhatja.

SERDAT 030 Soros port adat és állj bitek írása.
Ez a cím adatokat ír egy átviteli adatpufferbe. Az adat
innen egy soros shift regiszterbe kerül amikor ez üres
lesz, hogy átvitelre kerüljön. Ezt egy TBE (átviteli
puffer üres) megszakítás beállítást éri el. Az adat-szó
részeként feltétlenül kell egy állj bitnek is szerepelnie
Az adat-szó hosszát a stop bit helye állítja be.
BIT 15 14 13 12 11 10 09 08 07 06 05 04 03 02 01 00
 0 0 0 0 0 0 S D8 D7 D6 D5 D4 D3 D2 D1 D0
S=állj bit D=adatbitek

SERDATR 018 Soros port adat és állapotjelző olvasás.
Ez a cím egy fogadó adatpuffert olvas. Az adatok ide egy
fogadó shift regiszterből kerülnek, amikor az megtelt.
Soros megszakítás kérelem biteket is olvas a rendszer
ezen a címen az adatblokkal együtt:
BIT
15 OVRUN soros port fogadó túlfutott
14 RBF soros port fogadó puffer tele
13 TBE soros port átviteli puffer üres
12 TSRE soros port átviteli shift regiszter üres
11 RXD az RXD láb VART soros adatot fogad,
 amelyet a processzornak kell tesztelnie
10 x
09 STP állj bit
08 STP-DB8 adat bit ha LONG, állj bit ha nem
07 DB7 adat bitek
06 DB6
05 DB5
04 DB4
03 DB3
02 DB2
01 DB1
00 DB0

SERPER 032 Soros port periodus és vezérlő.
Ez a regiszter tartalmazza a LONG vezérlő bitet, valamint
egy 15 bites számot, amely a soros port átviteli sebessé-
gét határozza meg (BAUD RATE). Ha a szám N akkor az átvit-
teli sebesség 1 bit minden (N+1) * 0.2794 Mirkrosec-ben.
BIT
15 LONG A soros fogadást 9 bites szám állítja
14-0 RATE Az átviteli sebességet határozza meg

SPRxPTH x=0 120 y=0 122 Sprite mutató felső 3 bit
SPRyPTL x=1 124 y=1 126 alsó 15 bit
 x=2 128 y=2 12A Ez a regiszterpár tartalmazza a sprite
 x=3 12C y=3 12E DMA adatok 18 bites címét. A regisztert
 x=4 130 y=4 132 minden függőleges késleltetés alatt a
 x=5 134 y=5 136 processzornak vagy a COPPER-nek inicia-
 x=6 138 y=6 13A lizálni kell.
 x=7 13C y=7 13E

```

-----
SPRxPOS   x=0   140   y=0   142   Sprite víz.- függ. kezdőérték adatok
SPRyCTL   x=1   148   y=1   14A   Sprite függőleges állj és vezérlőbitek
          x=2   150   y=2   152   Ez a két regiszter a helyzet, méret és
          x=3   158   y=3   15A   jellemzők vonatkozásában együtt dolgo-
          x=4   160   y=4   162   zik. Általában a SPRITE DMA csatorna
          x=5   168   y=5   16A   tölti fel őket a vízszintes késleltetés
          x=6   170   y=6   172   alatt, azonban a processzor is bármikor
          x=7   178   y=7   17A   írhatja őket.
SPRxPOS   REGISZTER
BIT
15-08   SV7-SV0   Függ.kezdőérték(legmagasabb 8.bit SPRxCTL-ben)
07-00   SH8-SH1   Vízszintes kezdőérték, a legalacsonyabb bit a
          (SH0) az SPRxCTL regiszterben
SPRyCTL   REGISZTER
BIT
15-08   EV7-EV0   Függőleges végérték alsó 8 bitje
          07   AIT   Sprite kijelölés vezérlő bit(páratlan sprite)
06-04   *
          02   SV8   Függőleges kezdőérték 8.bit
          01   LV8   Függőleges végérték 8.bit
          00   SH0   Vízszintes kezdőérték 0.bit
Ha ebbe a regiszterbe írunk, az bekapcsolja a SPRITE-ok
vízszintes kiegyenlítő áramkörét.

```

```

-----
SPRxDATA   x=0   144   y=0   146   Ezek a regiszterek pufferelek a sprite-
          x=1   14C   y=1   14E   ok megjelenítési adatait. Ezeket álta-
          x=2   154   y=2   156   lában a sprite DMA csatorna tölti fel,
          x=3   15C   y=3   15E   de a processzor is írhatja. Amikor a
          x=4   164   y=4   166   vízszintes összehasonlítás történik, a
          x=5   16C   y=5   16E   pufferek értéke shift regiszterekbe
          x=6   174   y=6   176   töltődik és soros formában a kijelzőre
          x=7   17C   y=7   17E   kerül, a legmagasabb helyiértékű elő-
          szőr és bal oldalon. Az A regiszterbe
          való írás bekapcsolja a sprite-ot, ha
          be van kapcsolva, az A és B pufferekben
          lévő adatok a képernyőre kerülnek.

```

```

-----
STREQU     038   Pillanattároló a vízszintes szinkron számára a VB és
          EQU-val együtt.

```

```

-----
STRVBL     03A   Pillanattároló a vízszintes szinkron számára a VB-vel
          együtt.

```

```

-----
STRHOR     03C   Pillanattároló a vízszintes tároló számára

```

```

-----
STRLONG    03E   Pillanattároló hosszú vízszintes vonal számára
Az első három pillanattároló címek egyike a cél cím buszra
kerül az első frissítési ciklus alatt. A negyedik pillanat-
tároló cím a második időzítési ciklus alatt használatos
minden második vonalon, hogy a hosszú számmal rendelkező
vonalak azonosítsa. Négy frissítési ciklus van, és az
amelyet nem használ a rendszer pillanattárolásra, egy
0-ás (0FF) címet hagy a cél cím buszon.

```

```

-----
VPOSR      004   Olvasás: Függ. legmagasabb helyiértékű bit (és keret
                váltás)
-----
VPOSW      02A   írás: Függ. legmagasabb helyiértékű bit (és keret
                váltás)
                BIT 15 14 13 12 11 10 09 08 07 06 05 04 03 02 01 00
                LOF *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  * V8
                LOF = Hosszú keret
-----
VHPOSR     006   Olvasás: Vízszintes és függőleges pozíciója a nyaláb-
                nak vagy fényceruzának.
-----
VHPOSW     02C   írás: Vízszintes és függőleges pozíciója a nyalábnak
                vagy fényceruzának.
                BIT 15 14 13 12 11 10 09 08 07 06 05 04 03 02 01 00
                V7 V6 V5 V4 V3 V2 V1 V0 H8 H7 H6 H5 H4 H3 H2 H1
                Felbontás: 1/160-ad része a képernyő szélességének
                (280 ns)
-----

```

PORTOK

VEZÉRLŐ PORTOK

A gép hátulján jobb oldalon található két 9 pólusú cannon csatlakozó aljzat a vezérlő hardware-ek csatlakoztatására. E csatlakozókra lehet csatlakoztatni a mouse-okat, botkormányokat, fényceruzákat, stb.

VEZÉRLŐ PORTOK OLVASÁSA

Mivel a csatlakozókra több, egymástól különböző eszköz is csatlakoztatható, nekünk kell megoldani a különbségek áthidalását.

MOUSE, TRACKBALL

Ezeknél az eszközöknél a joystick vonalait a következőképpen használjuk:

- A joystick jobb és hátra vonalai a vízszintes elmozdulás jelzésére
- A joystick bal és előre vonalai a függ. elmozdulás jelzésére használtak.

Ha elmozdulás történik, akkor impulzusok jelennek meg a megfelelő vonalakon.

A SZÁMLÁLÓK OLVASÁSA

A mouse, trackball számlálók a JOY0DAT (bal oldali) és a JOY1DAT (jobb oldali csatlakozó) regiszterek.

```

        Bit 15-8   Mouse/trackball függőleges számláló
        Bit  7-0   Mouse/trackball vízszintes számláló

```

A SZÁMLÁLÓK HATÁRÉRTÉKEI

E számlálók folyamatosan számlálnak 2 határérték között, növekvő vagy csökkenő irányban. A számláló leolvasása a függ. visszafutás ideje alatt történjek, mentsük el a kiolvasott értéket, és hasonlítsuk össze az előzőleg leolvasott számláló tartalommal. A két érték összehasonlításakor megállapítható az elmozdulás iránya és sebessége. A mouse 200 impulzust ad inch-enként (24.5 mm). A függ. visszafutás 1/60-ad másodpercenként aktuális. Ha a két leolvasott érték különbsége kisebb, mint 127 akkor a különbség azonnal kiolvasható. Ha az elmozdulás sebessége nagyobb mint 72 inch/sec, akkor a különbség nagyobb mint 127. Ekkor a különbséghez hozzá kell adni 255-öt a valódi különbség meghatározásához.

Előző érték	Aktuális érték	Elmozdulás
200	100	fel(balra)
100	200	le(jobbra)
200	45	le *
45	200	fel **

* Mivel $200-45=155$, és ez nagyobb mint 127, a valódi érték $255-(200-45)=100$ Az elmozdulás lefelé történt

** Mivel $45-200=-155$, ennek abszolút értéke nagyobb 127-nél, a valódi érték $255+(-155)=100$ Az elmozdulás felfelé történt

Az Amiga mouse-okon 2 tűzgomb van. A vezérlő port 3 tűzgomb funkciót képes kezelni.

- 1.tűzgomb (bal oldali mouse gomb) a 6. érintkező a csatlakozón. A bal oldali tűzgombok állapota a páros című 8520 periféria kezelő port regiszterein keresztül olvasható le. Ha a megfelelő bit magas, a gomb lenyomott állapotú.
- 2. tűzgomb (jobb oldali mouse gomb) a 9. érintkező a csatlakozókon. A jobb oldali gombok állapota a potenciométer porton keresztül olvasható le. Kis ellenállás lenyomott gombot jelez.
- 3. tűzgomb az 5. érintkező a csatlakozókon. Ez a csatlakozó proportionális kontroller esetében használt.

JOYSTICK

A botkormány a 4 különböző iránynak megfelelően 4 kapcsolót tartalmaz, plusz egyet a tűzgombhoz. Az irányok kapcsolói az 1,2,3,4 (előre,hátra, balra,jobbra) csatlakozókra vannak kötve. A tűzgomb a 6. csatlakozóra van kötve. Normál esetben minden kapcsoló nyitott, a vonalak magas szintűek. Ha egy kapcsoló záródik, akkor a vonal lezáródik a testre, nulla szintű lesz. A joystick kapcsolóinak állapota szintén a JOY0DAT és JOY1DAT regiszterekből olvasható ki. Ha a kapcsoló zárt, a bit 0.

Bit 1 jobb kapcsoló
Bit 9 bal kapcsoló
Bit 1 (XOR) 0 hátra kapcsoló
Bit 9 (XOR) 0 előre kapcsoló

LINEÁRIS ESZKÖZÖK – PADDLE

Egy vezérlő port két potenciométert tud egyszerre kezelni. A paddle- ben (Lineáris Joystick) potméterek úgy vannak elhelyezve, hogy az egyik az X, a másik az Y irányú mozgást érzékeli. Mind két potméterhez tartozik egy-egy tűzgomb.

A különböző vezérlő eszközök bekötése az alábbi:

MOUSE TRACKBALL DRIVING			Analóg		
Csatlakozó Joystick	kontroller	paddle	joystick	lightpen	
1	előre*	V-pulzus	(NH)	(NH)	
2	hátra*	H-pulzus	(NH)	(NH)	
3	balra*	VQ-pulzus	bal gomb	1.gomb	
4	jobbra*	HQ-pulzus	jobb gomb	2.gomb	
5	(NH)	3.gomb	jobb poti	POT X	
6	1.gomb	1.gomb	(NH)	(NH)	tűzgomb
7	+5V	+5V	+5V	+5V	pos.
8	GND	GND	GND	GND	+5V
9	2.gomb	2.gomb	bal poti	POT Y	GND

* alacsony szinten aktív

A LINEÁRIS ESZKÖZÖK HASZNÁLATA

Használatuk bonyolultabb mint az eddig tárgyalt eszközöké. A függőleges visszafutás idelye alatt egy meghatározott értéket kell írunk a POTGO regiszterbe (\$DFF034). Ez az érték általában \$0001. A potenciométer helyzete megváltoztatja ezt az értéket, amit a következő függőleges visszafutáskor leolvashatunk. A potméter leolvasás a 7 - 8 rastersorban történik. Ha az összehasonlítás kisebb értéket mutat ki mint az előző, akkor növeli a POTxDAT regiszterben lévő számláló tartalmát. Ha a beolvasott érték nagyobb, mint az előző akkor a számláló értéke marad az eddigi. Mivel a számlálók csak 255-ig számolnak, nem tudjuk a teljes képernyő magasságot átfogni.

LINEÁRIS ESZKÖZÖK REGISZTEREI

POT0DAT - Port 0 Data
POT1DAT - Port 1 Data
Bit 15-8 POTxY Függőleges érték
Bit 7-0 POTxX Vízszintes érték

A számlálók értékei törölődnek amikor a POTGO regiszterbe írunk.

POTENCIOMÉTER SPECIFIKÁCIÓ

A potenciométerek csak lineárisak legyenek. Maximális ellenállásuk 528Kohm lehet. A belső kapacitás 0.047 mikroFarad $\pm 10\%$, a maximális időzítés 16.6 milisec (1 képernyő létrehozásának idelye)

FÉNYCERUZA, LIGHTPEN

A fényceruza csak a bal oldali portba csatlakoztatható.

A bekötés:

Csatlakozó	Használat
7	+5V
8	GND
5	TÜZGOMB
6	CAPTURE BEAM POSITION

FÉNYCERUZA REGISZTEREK HASZNÁLATA

VPOSR	Bit 15	LOF	interlace üzemmódban használt
	Bit 14-1		nem használt
	Bit 0	V8	függ.pozíció legfelső bitje
VHPOSR	Bit 15-8	V7-V0	függ.pozíció
	Bit 7-0	H7-H1	vízszintes pozíció

A raszter pozíciók felbontása a következő:

függőlegesen	1 sor nem interlace módban
	2 sor interlace módban
vízszintesen	2 kisfelbontású pixel

A fényceruza engedélyezéséhez magasra kell állítani a BLPCON0 regiszter harmadik bitjét. Amikor a fényceruza lenyomásra kerül, a VPOSR regiszter tartalmát el kell menteni. Az elmentés a függőleges visszafutás végén történjen (20. rasztorsor). A regiszter használata:

- Várakozás a fényceruza gomb lenyomására
- A függőleges visszafutás ideje alatt (0-19 sorok) olvassuk ki a VPOSR regisztert. (LONG)
- A 15 bitet hasonlítsuk össze \$10500-zal (V=261)
- Ha a VPOSR regiszter értéke nagyobb mint \$10500, a fényceruza nincs a képernyőn. Ha kisebb, akkor a VPOSR regiszter a fényceruza valós helyzetét tartalmazza.

SPECIÁLIS VEZÉRLŐK ADAPTÁLÁSA

A vezérlőportok vonalai átdefiniálhatóak a POTGO regiszter(csak írható) bitjei segítségével. A POTGOR regiszterből (\$DFF016 csak olvasható) a POTGO regiszter tartalma. A POT kontroll regiszter egy 4 bites kétirányú I/O portot vezérel.

A POTGO regiszter:

Bit 15	OUTRY	Output engedélyezés a jobb port 9. csatlakozójára
Bit 14	DATRY	I/O data a jobb port 9. csatlakozójáról
Bit 13	OUTRX	Output engedélyezés a jobb port 5. csatlakozójára
Bit 12	DATRX	I/O data a jobb port 5. csatlakozójáról
Bit 11	OUTLY	Output engedélyezés a bal port 9. csatlakozójára
Bit 10	DATLY	I/O data a bal port 9. csatlakozójáról
Bit 9	OUTLX	Output engedélyezés a bal port 5. csatlakozójára
Bit 8	DATLX	I/O data a bal port 9. csatlakozójáról
Bit 7-1	X	Foglalat chip azonosításra
Bit 0	START	Start pots (kondenzátor kisütése, számláló start)

A DISK KONTROLLER

A diszk kontroller 4 db kétoldalas, 3.5-es vagy 5.25-es inches meghajtót képes kezelni. Egy 3.5 inches meghajtó a gépbe beépítve található. A többi meghajtó külsőleg csatlakoztatható. A diszk funkciókat több re-giszteren keresztül vezérelhető:

- Kiválasztás, motor vezérlés, lemezérzékelés
- Disk DMA csatorna vezérlése, engedélyezése
- Disk írás/olvasás
- Disk formátum kezelés
- IRQ generálás

MEGHAJTÓ KIVÁLASZTÁS, VEZÉRLÉS ÉS LEMEZÉRZÉKELES

A disk rendszer két 8520-as portot használ, plusz egy FLAG IRQ portot. CIA A (\$BFE001), port A, 4 bitje és a CIA B (\$BFD000), port B, teljesen foglalt a disk céljára.

CIA A	PA5	DSKRDY*	Meghajtó rendben (alacsony-aktív)
CIA A	PA4	DSKTRACK0*	A meghajtó feje a 00-ás track-en áll
CIA A	PA3	DSKPROT*	A lemez írásvédett
CIA A	PA2	DSKCHANGE*	A lemez ki lett véve a meghajtóból. A következő fejmozgásig megmarad a jelzés.
CIA B	PB7	DSKMOTOR*	Meghajtó motor vezérlés. Ez a jel nem szabványos az AMIGA rendszerben. Mindazon drive-ok motorja elindul, amelyek SEL* vezetéke aktív. Minden drive motorja kikapcsolódik reset hatására.
CIA B	PB6	DSKSEL3*	3.meghajtó kiválasztása (alacsony aktív)
CIA B	PB5	DSKSEL2*	2.meghajtó kiválasztása
CIA B	PB4	DSKSEL1*	1.meghajtó kiválasztása
CIA B	PB3	DSKSEL0*	0.meghajtó kiválasztása(belső drive)
CIA B	PB2	DSKSIDE*	Fej kiválasztás (0 ha az alsó fej)
CIA B	PB1	DSKDIREC	Fej léptetés iránya (0 - befelé)
CIA B	PB0	DSKSTEP*	Fej léptetés
CIA B	FLAG	DSKINDEX*	Disk INDEX impulzusok(\$BFD000 4.bit) 6.szintű megszakítást generál

DISK DMA CSATORNA VEZÉRLÉSE

Az adatáramlás a disk és a memória között DMA segítségével történik.

A disk DMA vezérlése 4 részből áll:

- A mozgatandó terület címmutatójának tárolása
- Az adathossz átadása
- Az adatáramlás irányának átadása (olvasás/írás)
- DMA engedélyezés

CÍMMUTATÓ

A címet egy 19 bites cím határozza meg. A legalsó bit mindig nulla (A start csak páros cím lehet). Ez a cím beírható a DSKPTH (felső 3 bit) és DSKPTL (alsó 16 bit) regiszterekbe.

HOSSZ, IRÁNY, DMA ENGEDÉLYEZÉS

Ezek az információk a DSKLEN regiszterbe tárolandók:

Bit 15	DMAEN	Disk DMA engedélyezés
Bit 14	WRITE	Disk írás (RAW -ha 1-)
Bit 13-0	LENGTH	Mozgatandó word-ök száma

A hardware egy speciális folyamatot kíván a disk DMA előtt. Ez megakadályozza az íráshibát. Röviden, a DMAEN bitet a DSKLEN regiszterben kétszer kell beállítani. A második beállítás hatására indul el a disk DMA.

Példa:

- írjunk \$4000-et a DSKLEN regiszterbe, ez engedélyezi a disk DMA-t
- A többi adat beírása a regiszterbe
- Az előbbi adat ismételt beírása, ez indítja el a DMA-t
- Amikor a DMA befejezi a műveletet, a DSKLEN regiszterben ismét \$4000 jelenik meg

Amikor egy adat word mozgatásra került, a LENGTH értéke csökken, a DSKPTH, DSKPTL értéke nő. Amikor a LENGTH értéke eléri a nullát, az adatátviteli művelet befejeződik. A disk-ről való olvasásra egy módszer:

Olvassunk be egy teljes sávot (a pufferbe), majd keressük meg benne a kívánt szektor(okat). Amennyiben nincs elég szabad memória egy teljes track méretű pufferhez, a disk hardware támogatja a szektor keresést.

TOVÁBBI DISK REGISZTEREK

DSKBYTR - Disk Data Byte and Status Read

Ez a regiszter a disk mikroprocesszor adatpuffere. Olvasáskor az adat ami a lemezről kiolvasásra került, bekerül a regiszterbe, és a BYTEREADY bit TRUE-ra vált. A BYTEREADY törlődik, ha a regisztert olvassuk. E regisztert normál esetben a rendszerszoftver a disk forgásának szinkronizálására használja a processzorhoz, az írás vagy olvasás alatt.

Bit 15	BYTEREADY	Jelzi, hogy a regiszter egy byte-ot tartalmaz
Bit 14	DMAON	
Bit 13	DISKWRITE	A DSKLEN regiszter write bitjének engedélyezése
Bit 12	WORDEQUAL	Jelzi, hogy a DISKSYNC regiszter tartalma megegyezik a disk input stream-mel.
Bit 11-8		Nem használt
Bit 7-0	DATA	Disk Byte adat

ADKCON és ADKCONR - Audio és Disk vezérlő regiszterek

Az ADKCON regiszter csak írható, az ADKCONR csak olvasható. Az alsó 8 bit a hang céljára használt:

Bit 15	CLR/SET	Ha e bit magas, a regiszterbe való írás a biteket beállítja, egyébként törli
Bit 14	PRECOMP1	A PRECOMP felső bitje
Bit 13	PRECOMP0	A PRECOMP alsó bitje
		00 nincs
		01 140 Ns
		10 280 Ns
		11 560 Ns
Bit 12	MFMPREC	0 - GCR PRECOMP
		1 - MFM PRECOMP
Bit 11	UARTBRK	1 - PAULA Chip serial port 0 (RS232 Break)
Bit 10	WORDSYNC	1 - Engedélyezett a szinkronizáció és start DMA a Word olvasásához. A szinkronizációs word a DSKSYNC regiszterbe (\$DFF07E) irandó.
Bit 9	WSBSYNC	1 - Engedélyezi a GCR szinkront (WSB)
Bit 8	FAST	1 - 2 mikrosec / bit (MFM)
		0 - 4 mikrosec / bit (GCR)
Bit 7-4	ATPER3-0	Hang ATTACH periódus vezérlés
Bit 3-0	ATVOL3-0	Hang ATTACH hangerő vezérlés

A GCR formátumot használják az APPLE (TM) gépei.

DSKSYNC - Disk input szinkron

A DSKSYNC regiszter az input stream szinkronizálására használatos. Ha a WORDEQUAL bit engedélyezett az ADKCON regiszterben akkor nincs adatmozgatás addig amíg a stream-ről olvasott word nem egyezik a regiszterben tárolt értékkel. A DSYNC bit az INTREQ regiszterben az egyezést jelzi.

DSKDAT és DSKDATR - Disk DMA data regiszter

Ezek a regiszterek csak teszteléshez használtak. A DSKDAT csak írható, a DSKDATR csak olvasható. Ezek a regiszterek a Disk DMA adatpufferei. 2 byte-ot (1 word) tartalmazhatnak. Az írást engedélyezi a 14. bit a DSKLEN regiszterben. A DMA vezérlő automatikusan mozgatja az adatokat ezeken a regisztereken keresztül.

A DISK MEGSZAKÍTÁSOK

A Disk vezérlő két IRQ-t generálhat:

- DSKSYNC (5.szintű, INTREQ 12.bitje) az input stream egyezik a DSKSYNC regiszterrel
- DSKBLK (1.szintű, INTREQ 1.bitje) - DISK DMA kész

A BILLENTYÜZET

A billentyűzet interfész két vonalon keresztül csatlakozik a páros című 8520-as CIA Chip-re. A két vonal a CNT (billentyűzet óra) és az SP (billentyűzet adat).

A BILLENTYÜZET ADATAINAK VÉTELE

A CNT vonal az órajelet adja a billentyűzethez, az SP vonal az adatvezeték a belső soros regiszterhez. A melléklet tartalmazza a 8520-as adatlapjait. Amikor a 8520 egy kódot vesz a billentyűzettől, IRQ-t generál. A billentyűzet várakozik a HANDSHAKE jelre, mielőtt elküldi a következő kódot. A HANDSHAKE jelet a processzor generálja, az SP vonal min. 75 mikrosec ideig tartó alacsony szintre húzásával. Ha újabb billentyűt nyomunk le azelőtt, hogy a processzor az előzőt venné, a billentyűzet processzor (6580) tárolja az újabb kódot (max.10 darabot). A billentyűzet nem ASCII kódokat generál. Összesen 128 kód létezik.

A következő felsorolásban ismertetjük a billentyűzet kódokat.

00-3F

Normál karakterek. Lásd az ábrát!

40-4F

40	Space
41	BackSpace
42	Tab
43	Enter a numerikus billentyűzeten
44	Return
45	Escape
46	Delete
4A	- a numerikus billentyűzeten
4C	Cursor fel
4D	Cursor le
4E	Cursor jobbra
4F	Cursor balra

50-5F

50	F1
51	F2
52	F3
53	F4
54	F5
55	F6
56	F7
57	F8
58	F9
59	F10
5F	Help

60-67

60	Bal oldali Shift
61	Jobb oldali Shift
62	CapsLock
63	Control
64	Bal oldali Alt
65	Jobb oldali Alt
66	Bal oldali Amiga
67	Jobb oldali Amiga

68-77

Nem használtak

F0-FF

Ezek a kódok a 6580 és a 68000-es processzorok közötti kommunikáció céljára foglaltak, karakter nincs hozzájuk rendelve.

F0	Az utolsó kód hibás, újra küldve
FA	A billentyűzet puffer megtelt
FC	Billentyűzet önteszt hiba
FD	
FE	

A BILLENTYŰZET HIBÁJA

A billentyűzet sorok és oszlopot mátrixából áll, egy billentyűt egy kereszteződés reprezentál. Bizonyos billentyű kombinációk lenyomásakor hibásan küldi ea a kódokat a HW. Ha pl. egyszerre lenyomjuk az A és S billentyűket (elküldésre kerül a két kód). Ha még lenyomjuk a Z billentyűt is, akkor a Z kódja mellett az X kódja is elküldésre kerül. Létrejön egy Fantom (Ghosting) billentyű-kód. Ez a hiba álltalában nem zavaró, mivel csak nem használt billentyű kombinációkban jelentkezik. Hat billentyű nem része a mátrixnak így azok nem okozhatnak ilyen problémákat. Ezek a Ctrl, a két Shift, a két Amiga és a két Alt billentyűk.

A PÁRHUZAMOS INTERFÉSZ

Az általános célú párhuzamos illesztő egy 25 pólusú CANON csatlakozón található a gép hátulján. Ez a csatlakozó használható CENTRONICS csatlakozójú nyomtató illesztésére. A csatlakozó bekötése a mellékletben található meg.

A SOROS INTERFÉSZ

Egy 25 pólusú CANON (aljzat) csatlakozón található a több célra is alkalmazható soros interfész. A csatlakozó bekötése a mellékletben található meg.

A SOROS ILLESZTŐ HASZNÁLATA

Az interfészt egy UART (Universal Asynchronous Receiver/Transmitter) kezeli. Az UART átviteli sebessége tetszőlegesen beállítható, az adat - bitek száma 8 vagy 9 lehet.

AZ ÁTVITELI SEBESSÉG BEÁLLÍTÁSA

A sebesség a SERPER regiszter 0-14 bitjei segítségével állítható. Ha a beírt érték N akkor a gép N+1 órajelenként ($N+1 * 279.4 \text{ nsec}$) vesz mintát a bemenetről vételkor, vagy ír ki egy bitet adáskor.

ÜZEMMÓD BEÁLLÍTÁSA

A SERPER regiszter 15.bitje állítja be. Ha magas a bit, 8+1 bites, ha alacsony, 8 bites adatszélességgel dolgozik a soros interfész. A regiszter csak írható!

A SOROS VÉTELI REGISZTER

SERDATR

Bit 15	OVRUN	Overrun bit, jelzi ha egy byte-ot azelőtt kívánnunk olvasni, mint ahogy az megjött.
Bit 14	RBF	Read Buffer Full, jelzi hogy a puffer megtelt, a byte olvasható.
Bit 13	TBE	Transmit Buffer Empty, jelzi hogy a puffer üres.
Bit 12	TSRE	Transmit Shift Register Empty, az összes bit vétele megtörtént.
Bit 11	RXD	Direkt olvasás a PAULA RXD lábáról
Bit 10		Nem használt
Bit 9	STP	STOP bit 9 bites üzemmódban
Bit 8	STP	STOP bit 8 bites üzemmódban, vagy a 9. adatbit 9 bites üzemmódban
Bit 7-0	DB7-DB0	Az adatbyte alsó 8 bitje

ADATOK ELKÜLDÉSE

A soros adó regiszterbe írjuk az adatbyte-ot. Írás után az adabyte átvitele megindul a beállított sebességgel. Amikor az átvitel befejeződött, a SERDAT nem fogad új adatot, a TBE IRQ jel bekapcsolódik. A byte biteinek átvitele a legkisebb helyiértékű bittel kezdődik. Az átvitel addig folytatódik, amíg a regiszter összes bitje nulla nem lesz. Ekkor a TSRE bit magasra vált. A folyamat újra kezdődik, amikor egy újabb nullától különböző adatbyte-ot írunk.

- írjuk az elküldendő adatot a SERDAT regiszter alsó 8 (9) bitjébe
- az adatbiteken felül írjunk még egy magas bitet is a regiszter 9. (10.) bitjébe. Ez fogja jelezni az átvitel végét.

A BILLENTYÜZET

BILLENTYÜZET ILLESZTŐ

A billentyűzet a számítógépben egy 4 eres csatlakozókábelén keresztül kapcsolódik. A négy vezeték 5V feszültséget, földet, és két jelet (KCLK billentyűzet órajel és KDAT billentyűzet adatok) biztosít. A KCLK egy irányú, és mindig a billentyűzet hajtja meg, a KDAT-ot mind a billentyűzet, mind a számítógép meghajthatja.

Billentyűzet kommunikáció:

A billentyűzet 8 bites adatszavakat visz át sorosan a fő egységhez. Mielőtt az átvitel megkezdődne, mind a KCLK, mind KDAT magas. A billentyűzet az első adatbit kirakásával kezdi meg az átvitelt (a KDAT-on), amelyet egy impulzus követ a KCLK-n (először alacsony, aztán magas). Ezután kirakja a második adatot, és elküldi a KCLK impulzus folytatva ezt addig, amíg mind a 8 adatbit el nincs küldve. Az utolsó KCLK impulzus vége után a billentyűzet KDAT-ot újra magasra állítja. Amikor a számítógép fogadta a 8. bitet, a KDAT vonalon egy legalább 75 ms-nyi alacsony impulzust kell küldenie, amelynek mind kezelési jelzés szükséges a billentyűzet számára. Mind a kód, amelyet a számítógéphez továbbít a billentyűzet, egy bittel el van forgatva. Az átvitt sorrend ezért 6-5-4-3-2-1-0-7. Ennek az az oka, hogy a be-ki jelzést utoljára vigye át a gép, ha a billentyűzetnek egy elveszett szinkront kell visszaállítani (részletesebben később). A KDAT vonal aktív-alacsony, ami azt jelenti, hogy a magas szint (+5V) 0-ként fordítódik, és az alacsony szintű jel értelme 1. A billentyűzet processzor a KDAT vonalat kb. 20 ms-nyi időre állítja be, mielőtt a KCLK-t alacsonyra venné. A KCLK alacsony marad kb. 20 ms-ig, és utána újra magas lesz. Ezután a processzor újra 20 ms-ig vár a KDAT megváltoztatása előtt. Ezért a bitsebbség az átvitel alatt kb. 60 ms bitenként vagy 17 KBit/másodperc.

Billentyű kódok:

Minden billentyűnek van egy hozzárendelt billentyű kódja. A billentyű kódok mindig 7 bit hosszúak. A 8. bit egy billentyű be-ki jelzés. A 0 (magas szint) azt jelenti, hogy a billentyű le volt nyomva, és az 1 (alacsony szint) azt jelenti, hogy a billentyű nem volt lenyomva. A CapsLock billentyű különbözik ettől, lásd később.

Pl:

- a B billentyű kódja \$35=00110101 A byte bitjeinek elforgatása miatt az átvitt bitek: 01101010 (billentyű lenyomva) - a billentyű jelen pillanatban nincs lenyomva, a billentyűkód még mindig \$35, kivéve ha a 7. bit magasra van állítva, hogy a billentyű funkciót jelezze, így a billentyű kód végeredménye \$35=10110101 lesz. Elforgatás után az átvitt bitsorozat: 01101011.

A CapsLock billentyű:

Ez a billentyű abban különbözik a többitől, hogy csak akkor general billentyüködöt, ha le van nyomva. Azonban a be-ki bitet itt is használja a rendszer. Amikor lenyomjuk a CapsLock billentyűt, ez bekapcsolja a CapsLock ledjét, a be-ki bit 0 lesz. Amikor a CapsLock kikapcsolja a lede a be-ki bit 1 lesz.

Szinkronon kívüli feltétel:

Zaj vagy más elektromos problémák a billentyűzet és számítógép közötti szinkron elvesztéséhez vezethetnek. Ez azt jelenti, hogy a billentyűzet már befejezi a kód átvitelét, de a számítógép még csak valahol középen tart. Ha ez történik, a billentyűzet nem fogja megkapni a kezelési impulzust az átvitel végén. Ha a kezelési impulzus nem érkezik meg 143 ms-on belül az átvitel utolsó KCLK jelétől számítva, a billentyűzet feltételezi, hogy a számítógép bég mindig vár az átvitel többi részére, és ezért elvesztette a szinkront. A billentyűzet ezután megpróbálja visszaállítani a szinkront, úgy hogy ún. újraszinkronizálási módba kerül. Ebben a módban a billentyűzet kibocsájt egy 1-es jelet, és vár a kezelési impulzusra. Ha 143 ms-on belül nem érkezik egy sem, akkor egy másik 1-est bocsájt ki és újra vár. A folyamat folytatódik addig, amíg a kezelés jel meg nem érkezik. Ha a szinkron visszaállítódott, azt a gép úgy érzi mintha a billentyűze egy rossz karaktert bocsájtott volna ki. Ezért van az, hogy a billentyű be-ki bitet viszi át a rendszer utoljára. Mivel a billentyűzet 1-es jelet bocsájt ki a szinkron visszaállítására, az ávtit karakter billentyűje ki-ként lesz értelmezve, ami kevésbé veszélyes, min egy billentyű leütés. Amikor a billentyűzet vesztet szinkront érzékel, feltételezi, hogy a számítógép nem tudta fogadni a billentyű kódot, amelye átvinni próbált. Mivel a számítógép képtelen rá, hogy felfedezze a vesztett szinkront, a billentyűzet felelőssége lesz, hogy a számítógépet erről értesítse. Ezt egy vesztett szinkronkód küldésével tudatja a géppel (értéke \$F9=11111001). Ezután pedig újra küldi azt a kódot ami elromlott.

Megjegyzés: A vesztett szinkron kód küldésének egyedüli oka, hogy a szofver figyelmeztesse, hogy valami nincs rendben. A vesztett szinkron kód nem segít a visszaállítási folyamatban, mert a rossz billentyű kódot egyszerűen meg lehetne ismételni anélkül, hogy megmondanánk a gépnek, hogy az elsőben hiba volt.

Bekapcsolási sorrend:

Két lehetséges útja van annak, hogy a billentyűzetet normál körülmények közöt áram alá helyezzük. A számítógépet a már csatlakoztatott billentyűzettel együtt kapcsoljuk be, vagy pedig egy már bekapcsolt számítógépre dugjuk a billentyűzetet. A billentyűzetnek és a számítógépnek mindkét esetet hiba nélkül kell tudni kezelni. Az első dolog, amit a billentyűzet a bekapcsolás esetén tesz, hogy végrehajt egy öntesztet. Ez magában foglal egy ROM ellenőrző-összeg tesztet, egy egyszerű RAM tesztet és egy valódiságot ellenőrző időzítő tesztet. Amikor a billentyűzet áram alá kerül (vagy újra indítás esetén - lásd lejjebb), nem szabad semmit sem átvinnie, amíg a számítógéppel el nem érte a szinkronizálódást. Ezt úgy teszi, hogy lassan kibocsájt 1-es biteket, ahogyan ez fent le van írva, egészen addig, amíg a kezelési jel meg nem érkezik.

Ha a billentyűzet már bekapcsolás előtt be van dugaszolva, akkor a billentyűzetnek ez a folyamat néhány percig is eltarthat, mert a számítógép az önindítással és futás kezelemmel bajlódik. A billentyűzetnek addig kell az 1-esek kibocsátását folytatni, míg el nem éri a kezelési jelet. Ha a billentyűzetet bekapcsolás után dugaszoljuk be, nyolc 1-es-re van szükség, hogy a szinkront elérjük. Ebben az esetben azonban a számítógép bármilyen állapotban lehet, de a fogadott rossz karaktereknek nem szabad befolyásolni a gépet. Azért mert nem lenyomott billentyű jelet fogad, az ejtett hiba elhanyagolható. A billentyűzet meghajtónak számítani kell erre az eshetőségre, és tudnia kell ezt kezelni is amint bármilyen egyéb alkalmazást is kell tudnia kezelni amíg csak billentyű kódokat használ.

Megjegyzés: A billentyűzetnek nem kell átvinni vesztett szinkron kódot a bekapcsolás vagy újraindítás esetében fellépő újraszinkronizálásnál, csak a kezelési jel kiesése miatt.

Miután a billentyűzet és a számítógép is szinkronban vannak, a billentyűzetnek tájékoztatni kell a gépet az öneszt eredményéről. Ha az önteszt bármilyen okból nem sikerült, akkor önteszt hiba kódot visz át (értéke \$FC=11111100). A billentyűzet nem vár kezelési jelre miután ezt a kódot elküldte. Ezután a billentyűzet processzor egy hurokba kerül, amelyben villogtatja a CapsLock led-et, hogy tájékoztassa a felhasználót a hibáról. A villanások 1-es, kettes, hármas vagy négyes csoportban fordulhatnak elő, és kb. másodpercenként egyszer villog a led.

- . 1 villanás - ROM ellenőrző összeg hiba
- 2 villanás - RAM teszt hiba
- 3 villanás - Valódiság ellenőrző időzítő teszt hiba
- 4 villanás - Egyfajta kijárási két sor vonalai között, vagy két speciális billentyű között

Ha az önteszt sikerrel jár, a billentyűzet átviszi azokat a billentyűket, amelyeket éppen lenyomtunk. Először elküld egy kezdő billentyű kódot (érték \$FD=11111101) amelyet a lenyomott billentyű kódjai követnek (a billentyű kikapcsolva minden esetben). Miután minden billentyűt elküldött (általában nem lesz egy sem), a záró billentyű kód jelenik meg (értéke \$FE=11111110). Legvégül pedig a CapsLock led-je kikapcsolódik. Ez mutatja a kezdési műveletek végét, amely után a normál működés következik.

Megjegyzés: Ezek speciális kódok, 8 bites számok, nincs billentyű be-ki hozzájuk rendelve. Az átviteli bit sorrend megegyezik a fejezet elején leírttal. Az események általános menete tehát: áram be, szinkronizálás, \$FD átvitele, \$FE átvitele.

Hidegindítás:

A billentyűzetnek a feladata az is, hogy a számítógépet reset-elje a felhasználó parancsára. A felhasználó a hidegindítást a Ctrl és a két Amiga billentyű lenyomásával érheti el. A billentyűzet erre úgy válaszol, hogy a KCLK vonalat alacsonyra teszi és elindít egy 500 ms-os időzítőt. Az 500 ms végén a processzor ellenőrzi a három billentyűt, hogy még mindig le vannak-e nyomva, és ha így van akkor újraindítja az 500 ms-os időzítőt. Ez egészen addig folytatódik amíg a három billentyű egyet el nem engedünk. Amikor egy vagy több billentyűt a három közül elengedtünk, a processzor az 500 ms végéig vár. Ezután a kezdő címre ugrik, amely elengedi a KCLK-t és újra indítja a billentyűzetet.

Speciális kódok:

KÓD

JELENTÉS

F9	Az utolsó billentyű kód rossz, a következő kód ugyanaz megismételve (akkor használatos, amikor a gép és a billentyűzet elveszti a szinkront).
FA	Billentyűzet kiviteli puffer túlcsordulás
FB	Nem használt
FC	Billentyűzet önteszt hibás
FD	Kezdő billentyű
FE	Záró billentyű
FF	Nem használt

A BILLENYÜZET HARDWARE

Ez az AMIGA billentyűzeten belüli hardware leírása. Ez csak a billentyűzet második változatára igaz, arra amely a hitelesítő időzítővel rendelkezik.

Processzor:

Típusa: Rockwell/NCR/MOS Technologies 6500/1. 2 KByte ROM-ot tartalmaz, 64 byte RAM-ot, és 4 db.8 bites I/O portot. Ezenkívül van egy 16 bites számlálója és I/O vonalán élfelismerő képességgel rendelkezik. Van még egy beépített kristály-oszcillátora amely 3 MHz-en működik, amelyet kettéosztva kapja meg az 1.5 MHz belső órajelet.

Reset áramkör:

Egy áramkör a bekapcsolás esetén reseeli a processzort. A reset impulzus kb. egy másodpercig tart, miután az áramot bekapcsolta. Az áramkör valódiság ellenőrzést is végez. Mihelyt a processzor elkezdi átfutni a billentyűzet mátrixot, a valódiság ellenőrző időzítő is működésbe lép és reseteli a processzort, ha az átvizsgálás több min 50 ms-ig áll valahol. A 15.oszlop vonala indítja a valódiság ellenőrző időzítőt.

Billentyűzet mátrix:

A billentyűzet 91 gombot tartalmaz. Közülük 84 egy 6 soros 15 oszlopos mátrixba van rendezve (6 lyukat hagy a mátrixban). Mindegyik sor egy bemenet, és felfutási ellenállással rendelkezik, amelyen a tápfeszültség esik ($R=3.3K - 11K$ -ig). Mindegyik oszlop egy-egy nyílt csalakozó, amelyen nincs felfutó ellenállás (pl.meg tud hajtani egy oszlopot alacsonyan, de magasan nem). A rendszerprogram egy oszlopot hajt meg egy- szerre, és olvassa a sorokat. A hét másik billentyű különleges shift billentyű: Ctrl, bal Shift, jobb Shift, bal Alt, jobb Alt, bal Amiga, jobb Amiga. Ezek mindegyikének van egy külön bemenete a processzoron. A portok és a billentyűk bitszámai az alább leírtak alapján alakulnak.

Portok:

Négy, egyenként 8 bites I/O port van. A következő táblázat leírja a portokat és bijeik jelentését:

A port - 6500/1 cím \$080

PA.0	be/ki	KDAT Kimenet/pozitív élfelismerő bemenet
PA.1	ki	KCLK Kimenet*
PA.2	be	Bemenet 0.sor (alacsony=kapcsoló zárva)
PA.3	be	Bemenet 1.sor
PA.4	be	Bemenet 2.sor
PA.5	be	Bemenet 3.sor
PA.6	be	Bemenet 4.sor
PA.7	be	Bemenet 5.sor

* Az előző kódról vannak kicserélve ezek a bitek, hogy kihasználják a pozitív él érzékenységet a PA.0 lábnak.(így könnyebb felismerni a ke- zelés)

B port - 6500/1 cím \$081

PB.0	be	Jobb Shift bill. bemenet (alacsony=kapcsoló zárva)
PB.1	be	Jobb Alt billentyű bemenet
PB.2	be	Jobb Amiga billentyű bemenet
PB.3	be	Ctrl billentyű bemenet
PB.4	be	Bal Shift billentyű bemenet
PB.5	be	Bal Alt billentyű bemenet
PB.6	be	Bal Amiga billentyű bemenet
PB.7	ki	CapsLock led vezérlő (magas=led bekapcsolva)

C port - 6500/1 cím \$082

PC.0	ki	0.oszlop kimenet (aktív alacsony szint)
PC.1	ki	1.oszlop kimenet
PC.2	ki	2.oszlop kimenet
PC.3	ki	3.oszlop kimenet
PC.4	ki	4.oszlop kimenet
PC.5	ki	5.oszlop kimenet
PC.6	ki	6.oszlop kimenet
PC.7	ki	7.oszlop kimenet

D port - 6500/1 cím \$083

```

-----
PD.0      ki      8.oszlop kimenet
PD.1      ki      9.oszlop kimenet
PD.2      ki     10.oszlop kimenet
PD.3      ki     11.oszlop kimenet
PD.4      ki     12.oszlop kimenet
PD.5      ki     13.oszlop kimenet
PD.6      ki     14.oszlop kimenet
PD.7      ki     15.oszlop kimenet*

```

* A billentyűzetnek csak 15 oszlopa van, 0-14-ig számozva, így azonban a processzor 16 oszlopo tud támogatni így a jövőben ezt még kihasználhatjuk.

Számláló láb (bemenet vagy kimenet)

A valódiság ellenőrző lapon a számláló láb a 15.oszlop kimenetére van kötve. A régebbi, valódiságot nem ellenőrző változatban a számláló láb nincs bekötve. Ez biztosítja a billentyűzet processzor számára azt a képességet, hogy meghatározza melyik fajta lapba van beépítve, így az új processzor a régi alapokon is tud működni (a lapon kisebb változtatásokkal).

NMI bemenet:

Ez a VCC-re van kötve és ezért sohasem kapcsolódik be.

MATRIX TÁBLÁZAT

A következő táblázat mutatja, melyik billentyűk olvashatók az A portban mindegyik meghajtott oszlopra. A billentyű kódja is megtalálható az ábrán (zárójelben, hexadecimálisan)

	5.sor	4.sor	3.sor	2.sor	1.sor	0.sor
Oszlop (7.bit)	(6.bit)	(5.bit)	(4.bit)	(3.bit)	(2.bit)	
15	tartalék	tartalék	tartalék	tartalék	tartalék	tartalék
(PD.7)	(0E)	(1C)	(2C)	(47)	(48)	(49)
14	tartalék	LEFT	CAPS	TAB	~	ESC
(PD.6)	(5D)	SHIFT	LOCK	(42)	(00)	(45)
13	tartalék	Z	A	Q	!	tartalék
(PD.5)	(5E)	(31)	(20)	(10)	(01)	(5A)
12	9	X	S	N	@	F1
(PD.4)	(N.P)	(32)	(21)	(11)	(02)	(50)
	(3F)					
11	6	C	D	E	#	F2
(PD.3)	(N.P)	(33)	(22)	(12)	(03)	(51)
	(2F)					

10 (PD.2)	3 (N.P.) (1F)	V (34)	F (23)	R (13)	\$ 4 (04)	F3 (52)
9 (PD.1)	. (N.P.) (3C)	B (35)	G (24)	T (14)	% 5 (05)	F4 (53)
8 (PD.0)	8 (N.P.) (3E)	N (36)	H (25)	Y (15)	^ 6 (06)	F5 (54)
7 (PC.7)	5 (N.P.) (2E)	M (37)	J (26)	U (16)	& 7 (07)	tartalék (5B)
6 (PC.6)	2 (N.P.) (1E)	< (38)	K (27)	I (17)	* 8 (08)	F6 (55)
5 (PC.5)	ENTER (N.P.) (43)	> (39)	L (28)	O (18)	(9 (09)	tartalék (5C)
4 (PC.4)	7 (N.P.) (3D)	? / (3A)	: ; (29)	P (19)	) 0 (0A)	F7 (56)
3 (PC.3)	4 (N.P.) (2D)	tartalék (3B)	" ' (2A)	{ [(1A)	- - (0B)	F8 (57)
2 (PC.2)	1 (N.P.) (1D)	SPACE BAR (40)	(RET) (2B)	}] (1B)	+ = (0C)	F9 (58)
1 (PC.1)	0 (N.P.) (0F)	BACK SPACE (41)	DEL (46)	RET (44)	 \ (0D)	F10 (59)
0 (PC.0)	- (N.P.) (4A)	CURSOR DOWN (4D)	CURSOR RIGHT (4E)	CURSOR LEFT (4F)	CURSOR UP (4C)	HELP (5F)

A következő táblázat mutatja, mely billentyűk olvashatók a B portban (shift billentyűk).

(6.bit)	(5.bit)	(4.bit)	(3.bit)	(2.bit)	(1.bit)	(0.bit)
LEFT AMIGA (66)	LEFT ALT (64)	LEFT SHIFT (60)	CTRL (63)	RIGHT AMIGA (67)	RIGHT ALT (65)	RIGHT SHIFT (61)

PERIFÉRIA ILLESZTŐK

A rendszer hardware akkor választja a 8520-ast (CIA-t), ha a felső 3 bit: 101. Ezen kívül a CIAA választódik ki, ha a cím 12.bitje (A12) alacsony, és az A13 magas. A CIAB választódik ki ha A12 magas, és A13 alacsony. A 8520-as eléréséhez mind byte-os, mind szavas címzést alkalmazni lehet. A byte címzéshez (ez az általános eset) az A0-nak nulla értéket kell felvenni ha a CIAA-t akarjuk, és egy értéket ha a CIAB-t. A szó címzésnél a CIAB a 8-15 adabiteken, a CIAA a 0-7 adatbiteken kommunikál (természetesen az A0 szó címzésnél mindig 0). Az A11, A10, A9, A8 biteket használjuk arra, hogy meghatározzuk melyik regisztert akarjuk elérni. Ez az egész címben *-gal van jelölve. A többi bit ezeken kívül nem számít. Így a CIAA-t a következő cím választja ki:

```
101x xxxx xx01 **** xxxx xxx0
A CIAB-t pedig:
101x xxxx xx10 **** xxxx xxx1
```

Az egyszerűség kedvéért a jövőben legyenek ezek:

CIAA = BEE*01 CIAB = BED*00

CIAB címtérkép

Byte	Adatbitek							
Címe	7	6	5	4	3	2	1	0

BFD000	/DTR	/RTS	/CD	/CTS	/DSR	SEL	POUT	BUSY
BFD100	/MTR	/SEL3	/SEL2	/SEL1	/SEL0	/SIDE	DIR	/STEP
BFD200	A port adatirány regiszter (BED000): 1 = kivitel 0*C0-ra							
BFD300	B port adatirány regiszter (BED100): 1 = kivitel 0*FF-re							
BFD400	CIAB A Időzítő alsó byte							
BFD500	CIAB A Időzítő felső byte							
BFD600	CIAB B Időzítő alsó byte							
BFD700	CIAB B Időzítő felső byte							
BFD800	Vízszintes szinkron esemény számláló					0-7	bit	
BFD900	Vízszintes szinkron esemény számláló					8-15	bit	
BFDA00	Vízszintes szinkron esemény számláló					16-23	bit	
BFDB00	Nem használt							
BFDC00	CIAB soros adatregiszter							
BFDD00	CIAB megszakítás vezérlő regiszter							
BFDE00	CIAB vezérlőregiszter A							
BFDF00	CIAB vezérlőregiszter B							
Megjegyzés: A CIAB INT6 megszakítást generál								

CIAA címtérkép

Byte	Adatbitek							
Cím	7	6	5	4	3	2	1	0
BFE001	/FIR1	/FIR0	/RDY	/TK0	/WPRO	/CHNG	/LED	OVL
BFE101	Párhuzamos port							
BFE201	A port adatirány regiszter (BFE001): 1 = kivitel 0*03-ra							
BFE301	B port adatirány regiszter (BFE101): 1 = kivitel(ki v.be)							
BFE401	CIAA A Időzítő alsó byte							
BFE501	CIAA A Időzítő felső byte							
BFE601	CIAA B Időzítő alsó byte							
BFE701	CIAA B Időzítő felső byte							
BFE801	60 Hz eseményszámláló 0-7 bit							
BFE901	60 Hz eseményszámláló 8-15 bit							
BFEA01	60 Hz eseményszámláló 16-23 bit							
BFEB01	Nem használt							
BFEC01	CIAA soros adatregiszter (billenyüzet)							
BFED01	CIAA megszakítás vezérlő regiszter							
BFEE01	CIAA vezérlőregiszter A							
BFEF01	CIAA vezérlőregiszter B							

Megjegyzés: A CIAA INT2. megszakítást generál.

ILLESZŐ JELEK

Az óra bemenőjele:

A 02 óra egy olyan TTL kompatibilis bemenőjel, amelyet belső eszközök műveleteire és mint időzítő hasonlításí értéket rendszer adatbusszal való kommunikálására használunk.

CS - Chip - válaszás bemenőjel:

A CS bemenet vezérli a 8520-as működését. A CS alacsony szintje, amíg a 02 magas, az eszközt az R/W és a cím (RS) vonalakon érkező jelekre való válaszadásra kötelezi. A CS magas szintje meggátolja, hogy ezek a vonalak vezéreljék a 8520-ast.

R/W - Olvasás/írás bemenet:

A R/W jele általában a processzor adja ki, és ez irányítja a 8520-as adatátvitelének irányát. A R/W magas szintje olvasást jelez (adatátvi- tel a 8520-ból) amíg az alacsony írást jelent (adatátviel a 8520-ba).

RS3-RS0:

A cím bemenet a belső regisztereket választja ki a regisztertérképnek megfelelően.

DB7-DB0 Adatbusz bemenet/kimenet:

A nyolc adatbusz átviteli láb a 8520-as és a rendszer adatbusz közötti információcserét végzi. Ezek a lábak magas impedanciájú bemenetek, ki- véve ha a CS alacsony ill. a 02 és a R/W magasak (hogy ezzel a rendszer olvassa az eszközt). Az olvasás alatt az adatbusz pufferek be vannak kapcsolva és a kiválasztott regiszter adatait továbbítják az adatbuszon

IRQ - Megszakítás kérelem kimenet:

Az IRQ egy nyitott kimenet amely általában a processzor megszakítás bemenetére van rákötve. Egy külső felfutási ellenálás a jelet magasan tartja, és ezzel több IRQ kimenet egybekötését teszi lehetővé. Az IRQ kimenet általában ki van kapcsolva (magas impedancia) és alacsonyan működik amint ez a funkció leírásban jelezve van.

RES - Reset bemenet:

A RES lábon megjelenő alacsony jel minden belső regisztert resetel. A port lábak bemenetei állítódhatnak, és a portregiszterek törlődnek (bár a portok olvasása minden magas bitet visszaállít passzív felfutás miatt). Az időzítő vezérlő regiszterek nullázódnak és az időzítő bezáródik. Minden más regiszter értéke nulla lesz.

Regiszter táblázat:

Mindkét 8520-asnak 16 regisztere van, amelyeket írni és olvasni lehet. Itt látható a regiszterek listája, és mindegyiknek a 8520-ashoz rendelt memóriaterületen belüli címezése.

REGISZTER

RS3	RS2	RS1	RS0	#(HEX)	NÉV	JELENTÉSE
0	0	0	0	0	PRA	Periféria adatregiszter A
0	0	0	1	1	PRB	Periféria adatregiszter B
0	0	1	0	2	DDRA	Adatirány regiszter A
0	0	1	1	3	DDRB	Adatirány regiszter B
0	1	0	0	4	TALO	Időzítő regiszter A alacsony
0	1	0	1	5	TAHI	Időzítő regiszter A magas
0	1	1	0	6	TBLO	Időzítő regiszter B alacsony
0	1	1	1	7	TBHI	Időzítő regiszter B magas
1	0	0	0	8		Esemény számláló bitek LSB
1	0	0	1	9		Esemény számláló bitek 8-15
1	0	1	0	A		Esemény számláló bitek WSB
1	0	1	1	B		Nem használt
1	1	0	0	C	SDR	Soros adatregiszter
1	1	0	1	D	ICR	Megszakításvezérlő regiszter
1	1	1	0	E	CRA	Vezérlő regiszter A
1	1	1	1	F	CRB	Vezérlő regiszter B

Szoftver megjegyzés:

Az operációs rendszer már kiosztotta a 8520-as mind a négy időzítőjét különböző feladatokra. Ha a rendszer irányítása alatt fut a gép, úgyel- jen a rendszerforrások következő kioszására.

- 8520A időzítő A: Commodore soros kommunikáció (ha nem töreńik a soros kommunikáció, az időzítő elérhető)
- 8520A időzítő B: Video nyaláb bővítő (akkor használatos, ha a BLITTER a video nyalábhoz kell szinkronizálni. Ha nincs nyalábszinkronizál BLITTER éppen működésben az időzítő elérhető.
- 8520B időzítő A: Billentyűzet (állandóan használat alatt, amikor a rendszer vezérli a gépet). - 8520B időzítő B: Valódi időzítő (a rendszer irányítása alatt állandóan használatban. Feladat bekapcsolásnál és megszakításoknál használják).

REGISZTER NEVEK

8520A	8520B	NÉV	JELENTÉS
BFE001	BFD000	PRA	Párhuzamos adatregiszter A port
BFE101	BFD100	PRB	Párhuzamos adatregiszter B port
BFE201	BFD200	DDRB	Adatirány regiszter A port
BFE301	BFD300	DDRA	Adatirány regiszter B port
BFE401	BFD400	TALO	Időzítő A port alsó regiszter
BFE501	BFD500	TAHI	Időzítő A port felső regiszter
BFE601	BFD600	TBLO	Időzítő B port alsó regiszter
BFE701	BFD700	TBHI	Időzítő B port felső regiszter
BFE801	BFD800		Eseményszámláló LSB
BFE901	BFD900		Eseményszámláló 8-15 bit
BFEA01	BFDA00		Eseményszámláló WSB
BFEB01	BFDB00		Nem használt
BFEC01	BFDC00	SDR	Soros adatregiszter
BFED01	BFDD00	ICR	Megszakításvezérlő regiszter
BFEE01	BFDE00	CRA	Vezérlő regitszer A
BFEF01	BFDF00	CRB	Vezérlő regitszer B

REGISZTER FUNKCIÓK LEÍRÁSA

I/O kapuk (PRA, PRB, DDRA, DDRB)

Az A és B kapuk egy 8 bites periféria adatregiszterből (PR) és egy-egy 8 bites adatirány regiszterből állnak (DDR). Ha a DDR egy bitje 1-re van állítva, akkor a PR-LEN neki megfelelő bitkimenetre állítódik. Ha egy DDR bit értéke 0, akkor a megfelelő PR bit bemenetként van meghatározva. Amikor egy PR regisztert olvasnak, az I/O lábak(PA0-PA7,PB0-PB7) az éppen aktuális állapotot olvassák, függetlenül attól, hogy bemenetre vagy kimenetre vannak állítva. Az A és B kapuk rendelkeznek passzív és aktív felfutási eszközökkel, így mind a CMOS mind a TTL kompatibilitást biztosítani tudják. Mindkét kapu 2 TTL meghajtó kapacitással rendelkezik. A normál I/O műveletek mellett a PB6 és PB7 kapu időzítő kimeneti funkciókat is lehetővé tesznek.

KEZELÉS (HANDSHAKING)

A kezelés olyan adatátvittelek esetén merül fel, amelyek a PC kimenő és a FLAG bemenő lábat használják. A PC alacsony lesz minden B kapu elérés utáni harmadik ciklusban. Ez a jelzés használható a B portnál az adat kész vagy a B port az adatot elfogadta jelzésként. A 16 bites adatátvittelek esetében is lehetséges a kezelés (a 16-os átvitel mind az A, mind a B kaput használja) úgy, hogy mindig az A kaput írjuk vagy olvassuk először. A FLAG egy negatív érzékeny bemenet, amely egy másik 8520 PC bemenetét fogadhatja, vagy általános ciklus megszakítás bemenet lehet. A FLAG-en bejövő bármilyen negatív átvitel bekapcsolja a FLAG megszakítás bitjeit.

REGISZTER	NÉV	D7	D6	D5	D4	D3	D2	D1	D0
0	PRA	PA7	PA6	PA5	PA4	PA3	PA2	PA1	PA0
1	PRB	PB7	PB6	PB5	PB4	PB3	PB2	PB1	PB0
2	DDRA	DPA7	DPA6	DPA5	DPA4	DPA3	DPA2	DPA1	DPA0
3	DDRB	DPB7	DPB6	DPB5	DPB4	DPB3	DPB2	DPB1	DPB0

IDŐTARTAM IDŐZÍTŐK

Valós idejű óra:

A valós idejű óra egy 24 bites bináris számlálóból áll. Pozitív él átmenetek ezen a lábon növelik a számláló értékét. A valós idejű órának passzív felfutása van. A programozható ébresztő funkció egy megszakítást biztosítást a kívánt időben. Az ébresztő funkció regiszterei ugyanazon a címen helyezkednek el mint a nekik megfelelő valós idejű óra regiszterek. Az ébresztő funkció elérése egy vezérlő regiszter bit által történik. Az ébresztő csak írható, a valós idejű óra címeinek olvasása az időt adja függetlenül az ALAPM (ébresztő funkció) bitjének állapotától. A valós idejű óra helyes beállításához és olvasásához meghatározott sorrendben kell hozzáfogni. A TOD óra automatikusan megáll, ha a regiszterekbe írunk, és nem indul újra, csak ha az esemény számláló legalacsonyabb byte-jába is írtunk már. Ez biztosítja, hogy az óra csak a megfelelő időben induljon el. Mivel az olvasó utasítás közben is történhet átvitel egyik szintről a másikba, egy kizáró funkció van beépítve, amely minde óra információt állandóan szinten tart olvasás közben. Minden óra regiszter lezáródik a legmagasabb byte olvasásáig, azonban az óra a számolást folytatja addig is, míg a regiszterek zárva vannak. Ha csak egy regisztert olvasunk nincs átviteli probléma, és a regiszter szabadon olvasható, feltéve hogy a legmagasabb byte olvasását mindig növeli a legalacsonyabb byte olvasása, hogy ezzel a lezárást kikapcsoljuk.

Bit nevek írás vagy olvasás esetén:

REGISZTER	NÉV								
8	0-7	E7	E6	E5	E4	E3	E2	E1	E0
9	8-15	E15	E14	E13	E12	E11	E10	E9	E8
A	16-23	E23	E22	E21	E20	E19	E18	E17	E16

Írás: CRB7 = 0

CRB7 = 1 ébresztő

SOROS PORT (SDR)

A soros port egy pufferrel ellátott 8 bite szinkron shift regiszter. Egy vezérlő bit választja ki a bemeneti vagy kimeneti működtetési módot.

- Bemeneti mód:

Bemeneti módban az SP lábon érkező adatok betöltődnek a shift regiszterbe a CNT lábon érkező jel emelkedő élére. Nyolc CNT impulzus után az adat a shift regiszterből átvivődik a soros adatregiszterbe, és megszakítás generálódik.

- Kimeneti mód:

A kimeneti módban az A időzítő (BAUDRATE) átviteli sebességi generátorként funkcionál. A legnagyobb elérhető átviteli sebesség a 02 órajel egynegyede, de a legnagyobb használható átviteli sebességet a vonalfeltöltést és a fogadó készülék jeleinek sebessége határozza meg. Az átvitel elindításához az A időzítőt folyamatos módra kell állítani és el kell indítani. Az átvitel akkor kezdődik, ha írtunk a soros adatregiszterbe. Az A időzítőből vett órajel a CNT lábon jelenik meg kimenetként. Az adatok a soros adatregiszterből a soros shift regiszterbe töltődnek és utána ki-shiftelődnek az SP lábra amikor CNT impulzus jelenik meg. Az A ki shiftelt adat következő CNT leeső élénél lesz érvényes, és egészen a következő leeső élíg érvényes is marad. Nyolc CNT impulzus után megszakítás generálódik jelezve, hogy a további adatokat is lehet küldeni. Ha a soros adatregisztert már a megszakítás előtt feltöltöttük új adat-tal, az új adat automatikusan bekerül a shift regiszterbe és az átvitel folytatódik. Ha a 8. CNT impulzus után már nincs több átviteli szint, a CNT magasra áll és az SP az utolsó átvitt adat szintjén marad. Az SDR adatok shiftelésénél először a legmagasabb helyiértékű bit kerül ki. A bevitelnél ugyanez a minta valósul meg.

KÉTIRÁNYÚ HASZNÁLAT

A soros port és a CNT óra kétirányú képessége lehetővé teszi sok 8520- as összekapcsolását egy közös soros kommunikációs buszon, amelyen az egyik 8520-as vezérel (adja az adatot és a shift órajelet) míg a többi chip van alárendelve. Mind a CNT és SP kimenetek lehetővé teszik egy ilyen közös busz létezését. A vezérlő/alárendelt kiválasztás menete lefolytatható a soros buszon vagy külön kijelölt kezelési (HANDSHAKE) vonalon is.

MEGSZAKÍTÁS VEZÉRLŐ REGISZTER

A 8520-ban a megszakításoknak 5 forrása lehet:

- Az A időzítő alulcsordulása (elérte a visszaszámlálás a 0-t)
- A B időzítő alulcsordulása
- Valós idejű óra ébresztőjel
- Soros port tele/üres
- A FLAG lábon érkezett megszakítás

Egy regiszter biztosítja a maszkolási és megszakítási információkat. A megszakítás vezérlő regiszter áll egy csak írható maszk regiszterből, és egy csak olvasható adatregiszterből. Bármely megszakítás az adatregiszter megfelelő bitjét fogja beállítani. Bármelyik megszakítás amely be van kapcsolva egy 1-es bittel, a maszkregiszter ugyanazon helyén bekapcsolja az IR bitet az adatregiszterben, és az IRQ lábat alacsonyra állítja. Egy több chipes rendszerben az IR biteket le lehet kérdezni, hogy meghatározzuk melyik chip generálta a megszakításkérélmét. Amikor az adatregisztert olvassák, a tartalma törlődik (0-ra állítódik), és az IRQ vonal magas állapotba tér vissza. Mivel olvasáskor törlődik, a felhasználónak biztosítani kell, hogy a megszakítás lekérdezés vagy megszakítás szolgáltatás kódja megőrződjék és képes legyen válaszolni minden bitre, amelyek be voltak állítva az adatregiszterben az olvasás időtartama alatt. A maszkregiszter 1 vagy több bitje is beállítható vagy törölhető a regiszterben lévő bármelyik másik bit érvényes állapotának befolyásolása nélkül. Ezt a legnagyobb helyiértékű bit megfelelő állapotának beállításával tehetjük meg, ezt egyébként beállítás / törlés bitnek is nevezik. A 0-6 biteken a felhasználó készít egy maszkot, amely meghatározza, hogy mely biteket akarja befolyásolni. Ezután a 7. bit használatával meghatározza hogyan befolyásolja a biteket a maszk megfelelő pozícióin.

- Ha a 7. bit 1, akkor a 0-6 bitek közül a saját maszk szavában azok, amelyek 1-re vannak beállítva, a maszkregiszter megfelelő bitjét magasra állítják. Az a bit, amely 0 volt, a maszkregiszter bitjét az eredeti állapotban hagyja.
- Ha a 7. bit 0, akkor a 0-6 bitek közül a saját maszk szavában azok, amelyek 1-re vannak állítva, a maszk regiszter megfelelő bitjét törlik. Az a bit, amely 0 volt, a maszkregiszter bitjét az eredeti állapotban hagyja.
- Ha egy meghatározott feltételtől függő megszakítás megtörténtére várunk, a megfelelő maszk bitnek 1-nek kell lennie.

Pl: Ha be akarjuk állítani az A időzítő megszakítás bitjét (bekapcsolni az A időzítő megszakítást, de közben biztosak akarunk lenni benne, hogy az összes többi megszakítás törölve van.)

Ekkor az alábbi sorrendet használja:

```
movi.b      01111110B,A0
mov.b      A0,ICR ; A legmagasabb helyiértékű bit 0 értéke azt
                  jelenti, hogy törölni kell minden bitet,
                  amely 1-re van állítva a byte további ré-
                  szében.
```

```
movi.b      10000001B,A0
mov.b      A0,ICR ; A legmagasabb helyiértékű bit 1 értéke az
                  jelenti, hogy be kell állítani minden bi-
                  tet, amely 1-re van állítva a byte további
                  részében (de ne változzon az az érték,
                  ahol a bit beírt értéke 0-ra van állítva).
```

MEGSZAKÍTÁS VEZÉRLŐ REGISZTER OLVASÁS:

REGISZTER	NÉV	D7	D6	D5	D4	D3	D2	D1	D0
D	ICR	IR	0	0	FLG	SP	ALRM	TB	TA

MEGSZAKÍTÁS VEZÉRLŐ REGISZTER ÍRÁS:

REGISZTER	NÉV	D7	D6	D5	D4	D3	D2	D1	D0
D	ICR	S/C	*	*	FLG	SP	ALRM	TB	TA

VEZÉRLŐ REGISZTEREK

A 8520-ban 2 vezérlőregiszter van: A CRA és a CRB. A CRA az A időzítőhöz tartozik, a CRB a B időzítőhöz.

Mindegyik időzítő egy 16 bites csak olvasható időzítő számlálóból, és egy 16 bites csak írható időzítő beállítóból áll. Az időzítőkbe írt adat a beállítót rögzíti a beírt időre, míg az időzítőből olvasott adat az időzítő éppen akkori tartalmát adja. A beállítót előskálázónak is nevezik, mert azt a visszaszámlálási értéket képviseli, amelyet vissza kell számolni mielőtt az időzítő egy alulcsordulási (nincs tovább számolás) állapotot érne el. Ez a beállítási érték a bejövő órajel frekvencia kettéosztó értéke. Az időzítőket használhatjuk külön-külön vagy esetleg bekapcsolva is nagyobb méretű műveletekhez. Különböző időzítő működtetési módok lehetővé teszik hosszú idejű késleltetések, különböző szélességi impulzusok, impulzus sorozatok és változó frekvenciájú hullámformák generálását. A CNT bemenet alkalmazásával az időzítők külső jel impulzusokat, frekvenciákat, impulzus szélességet, késleltetési időket tudunk mérni. Mindegyik időzítőnek van egy vezérlőregisztere, amely a következő funkciók felett független irányítást biztosít:

- Kezdés/leállítás:

Egy vezérlő bit teszi lehetővé, hogy az időzítőt a processzor bármikor indíthatja és leállíthatja.

- PB be/ki:

Egy vezérlő bit az időzítő kimenetet képessé teszi, hogy megjelenjen a B port kimenő vonalán (PB6 az A időzítőnél, és PB7 a B időzítőnél). Ez a funkció felülbírálja a DDRB-t és a megfelelő PB vonalakat automatikusan kimenetre állítja.

- Zár/Impulzus:

Egy vezérlő bit kiválasztja a B port-hoz rendelt kimenetet amíg a PB be/ki bit ki van kapcsolva. Minden egyes időzítési túlcsoordulásnál a kimenet vagy bezár, vagy generál egy egyszerű pozitív impulzust egy ciklus időtartamra. A kimenet magasan áll, ha az időzítő indul és a RES alacsonyra állítja vissza.

- Egy lefutás/Folyamatos mód:

Egy vezérlő bit választja ki mindkét fenti időzítési módot. Egy lefutási módban az időzítő a beállítási értékről visszaszámol 0-ig, megszakítást generál, újratölti a kezdőértéket és megáll. Folyamatos módban az időzítő visszaszámol 0-ig, megszakítást generál újratölti a leállítási értéket és kezdi a folyamatot előlről. Egy lefutási módban a számláló magas byte-jába való írás (5. és 7. regiszter) átviszi a beállítási értéket a számlálóba és önmagától elkezd a leszámolást függetlenül a kezdés bittől.

- Azonnali töltés:

Egy pillanat bit képessé teszi az időzítőt, beállítót, hogy betöltse a kezdőértéket az időzítő számlálóba bármikor, függetlenül attól, hogy az időzítő fut vagy pedig áll.

BEMENETI MÓDOK

A vezérlő bitek megengedik, hogy az időzítő használhassa az órát a számláló érték csökkentésére. Az A időzítő tudja a 02 impulzusokat vagy a CNT lábon bejövő külső impulzusokat számolni. A B időzítő tudja a 02 impulzusokat, CNT impulzusokat, A időzítő alulcsordulás impulzusokat, akkor is ha a CNT láb is magasan van vele egyidőben. A beállítási érték betöltődik a számlálóba minden egyes időzítő alulcsordulás esetében, azonnali töltésnél, vagy miután a számláló magas byte-jába írtunk mialatt az időzítő állt. Ha az időzítő fut a felső byte-ba való írás betölti a beállítási értéket, de ez nem kerül a számlálóba.

Bit nevek olvasó regiszterek esetén:

REGISZTER	NÉV	D7	D6	D5	D4	D3	D2	D1	D0
4	TALO	TAL7	TAL6	TAL5	TAL4	TAL3	TAL2	TAL1	TAL0
5	TAHI	TAH7	TAH6	TAH5	TAH4	TAH3	TAH2	TAH1	TAH0
6	TBLO	TBL7	TBL6	TBL5	TBL4	TBL3	TBL2	TBL1	TBL0
7	TBHI	TBH7	TBH6	TBH5	TBH4	TBH3	TBH2	TBH1	TBH0

Bit nevek írás regiszterek esetén:

REGISZTER	NÉV	D7	D6	D5	D4	D3	D2	D1	D0
4	TALO	PAL7	PAL6	PAL5	PAL4	PAL3	PAL2	PAL1	PAL0
5	TAHI	PAH7	PAH6	PAH5	PAH4	PAH3	PAH2	PAH1	PAH0
6	TBLO	PBL7	PBL6	PBL5	PBL4	PBL3	PBL2	PBL1	PBL0
7	TBHI	PBH7	PBH6	PBH5	PBH4	PBH3	PBH2	PBH1	PBH0

VEZÉRLŐ REGISZTEREK KIOSZTÁSA A KÖVETKEZŐ

A vezérlőregiszter

BIT	NÉV	FUNKCIÓ
0	START	1 = indítja, 0 leállítja az A időzítőt. A bit auto - matikusan resetelődik amikor alulcsordulás következik be egy lefutás módban.
1	PBON	1 = A időzítő kimenet a PB6 vonalon 0 = PB6 a normál működést folytatja.
2	OUTMODE	1 = lezárás 0 = impulzus (kimenet kódja)
3	RUNMODE	1 = egy lefutás 0 = folyamatos mód (futás módja)
4	LOAD	1 = azonnali törlés (ez egy pillanat bit, nem tárol adatot , a 4.bit mindig 0-val tér vissza olvasásnál, a beleírt 0-nak nincs hatása).
5	INMODE	1 = Az A időzítő pozitív CNT átmeneteket számol 0 = A időzítő 02 órajel impulzusokat számol
6	SPMODE	1 = soros port kimenet (CNT a shift órajel forrása) 0 = soros port bemenet (külső órajel szükséges)
7	TODIN	0 = 60 Hz bemeneti frekvencia 1 = 50 Hz kimeneti frekvencia

Minden használaton kívüli hatástalan az írás, és 0-t adnak vissza

A vezérlőregiszter

BIT	NÉV	FUNKCIÓ	

0	START	ld. az A időzítőnél	
1	PBON	1 = B időzítő kimenet a PB07 vonalon 0 = A PB07 a normál működését folytatja	
2	OUTMODE	ld.A időzítő	
3	RUNMODE	ld.A időzítő	
4	LOAD	ld.A időzítő	
5,6	INMODE	A CRB6 és CRB5 bitjei a négy lehetséges bemeneti mód közül egyet választanak ki a B időzítő számára a következőkből:	
	CRB6	CRB5	

	0	0	a B időzítő 02 órajel impulzusokat számol
	0	1	a B időzítő pozitív CNT átmeneteket számol
	1	0	a B időzítő A időzítő alulcsordulási impul- zusokat számol
	1	1	a B időzítő A időzítő alulcsordulási impul- zusokat számol ha a CNT láb magas
7	ALARM	1 = a valós idejű óra regisztereibe való írás az ébresztési időt állítja be 0 = az óra írása az órát állítja be.(az óra olvasása mindig az óra értéket adja ettől a bittől független	

PORT JELKIOSZTÁS

Ez a rész meghatározza, hogy a különböző jelek hogyan viszonyulnak a 8520 portjaihoz. Ez az információ képessé teszi a programozót, hogy a befojásolni kívánt külső egységeket (vagy belső vezérlő jeleket) a port címekhez tudja rendelni. Ezt a részt elsődlegesen a rendszerprogramozó tudja használni és általában az alkalmazó programozóknak nincs rászüksége. Az alkalmazói programozók inkább a rendszer dokumentációra bizzák magukat, minthogy közvetlenül a portot ériék el. Ne feledje azt sem, hogy ebben egy (multi tasking) több feladatot egy időben végre hajtó operációs rendszer. Az alkalmazó programozóknak inkább a forrás elérés megalapozott szabályait szabadna követni, hogy biztosítsák a szoftver-üknek a rendszerrel való kompatibilitást.

Címlisták:

BEERE	DATA BITS 7-0 (A12) (INT2)	
PA7.. game port 1,	pin 6 (fire button)	
PA6.. game port 0,	pin 6 (fire button)	
PA5.. RDY	disk ready	
PA4.. TKO	disk track 00	
PA3.. WPRO	write protect	
PA2.. CHNG	disk change	
PA1.. LED	led light (0=bright)	
PA0.. OVL	memory overlay bit	
SP... KDAT	keyboard data	
CNT.. KCLK		
PB7.. P7	data 7	
PB6.. P6	data 6	
PB5.. P5	data 5	CENTRONICS PARALELL INTERFACE
PB4.. P4	data 4	
PB3.. P3	data 3	DATA
PB2.. P2	data 2	
PB1.. P1	data 1	
PB0.. P0	data 0	
PC... drdy		CENTRONICS CONTROLL
E.... ack		

BEDRE	DATA BITS 15-8 (A13) (INT6)	
PA7.. com line DTR	driven output	
PA6.. com line RTS	driven output	
PA5.. com line carrier detect		
PA4.. com line CTS		
PA3.. com line DSR		
PA2.. SEL	centronics controll	
PA1.. POUT	paper out	
PA0.. BUSY	busy	
SP... BUSY	commodore	
CNT.. POUT	commodore	
PB7.. MTR	motor	
PB6.. SEL3	select external 3rd drive	
PB5.. SEL2	select external 2nd drive	
PB4.. SEL1	select external 1rd drive	
PB3.. SEL0	select internal drive	
PB2.. SIDE	side select	
PB1.. DIR	direction	
PB0.. STEP	step	
PC... not used		
E.... INDEX	disk index	