# disk

| COLLABORATORS | | | |
|---|---|---|---|
| | *TITLE* :<br><br>disk | | |
| *ACTION* | *NAME* | *DATE* | *SIGNATURE* |
| WRITTEN BY | | July 19, 2024 | |

| REVISION HISTORY | | | |
|---|---|---|---|
| NUMBER | DATE | DESCRIPTION | NAME |
| | | | |

# Contents

**1 disk** **1**

1.1 disk.doc . . . 1

1.2 disk.resource/AllocUnit . . . 1

1.3 disk.resource/FreeUnit . . . 2

1.4 disk.resource/GetUnit . . . 2

1.5 disk.resource/GetUnitID . . . 4

1.6 disk.resource/GiveUnit . . . 4

1.7 disk.resource/ReadUnitID . . . 5

# Chapter 1

# disk

## 1.1 disk.doc

```
    AllocUnit()
    FreeUnit()
    GetUnit()
    GetUnitID()
    GiveUnit()
    ReadUnitID()
```

## 1.2 disk.resource/AllocUnit

```
  NAME
      AllocUnit - allocate a unit of the disk

  SYNOPSIS
      Success = AllocUnit( unitNum ), DRResource
D0                    D0          A6

BOOL AllocUnit(LONG);

  FUNCTION
This routine allocates one of the units of the disk.  It should
be called before trying to use the disk (via GetUnit).

In reality, it is perfectly fine to use GetUnit/GiveUnit if AllocUnit
fails.  Do NOT call FreeUnit if AllocUnit did not succeed.  This
has been the case for all revisions of disk.resource.

  INPUTS
unitNum -- a legal unit number (zero through three)

  RESULTS
Success -- nonzero if successful.  zero on failure.

  EXCEPTIONS

  SEE ALSO
```

```
 BUGS
```

## 1.3 disk.resource/FreeUnit

```
 NAME
     FreeUnit - deallocate the disk

 SYNOPSIS
     FreeUnit( unitNum ), DRResource
    D0         A6

void FreeUnit(LONG);

 FUNCTION
This routine deallocates one of the units of the disk.  It should
be called when done with the disk.  Do not call it if you did
no successfully allocate the disk (there is no protection -- you
will probably crash the disk system).

 INPUTS
unitNum -- a legal unit number (zero through three)

 RESULTS

 EXCEPTIONS

 SEE ALSO
FreeUnit

 BUGS
Doesn't check if you own the unit, or even if anyone owns it.
```

## 1.4 disk.resource/GetUnit

```
 NAME
     GetUnit - allocate the disk for a driver

 SYNOPSIS
     lastDriver = GetUnit( unitPointer ), DRResource
     D0         A1             A6

struct DiscResourceUnit *GetUnit(struct DiscResourceUnit *);

 FUNCTION
This routine allocates the disk to a driver.  It is either
immediately available, or the request is saved until the disk
is available.  When it is available, your unitPointer is
sent back to you (via ReplyMsg).  You may then reattempt the
GetUnit.

Allocating the disk allows you to use the disk's resources.
```

```
Remember however that there are four units to the disk; you are
only one of them.  Please be polite to the other units (by never
selecting them, and by not leaving interrupts enabled, etc.).

When you are done, please leave the disk in the following state:
    dmacon dma bit ON
    dsklen dma bit OFF (write a #DSKDMAOFF to dsklen)
    adkcon disk bits -- any way you want
    entena:disk sync and disk block interrupts -- Both DISABLED
    CIA resource index interrupt -- DISABLED
    8520 outputs -- doesn't matter, because all bits will be
  set to inactive by the resource.
    8520 data direction regs -- restore to original state.

NOTE: GetUnit() does NOT turn on the interrupts for you.
      You must use AbleICR (for the index interrupt) or intena
      (for the diskbyte and diskblock interrupts) to turn them
      on.  You should turn them off before calling GiveUnit,
      as stated above.

 INPUTS
unitPtr - a pointer you your disk resource unit structure.
  Note that the message filed of the structure MUST
  be a valid message, ready to be replied to.  Make sure
  ln_Name points to a null-terminated string, preferably
  one that identifies your program.

  You need to set up the three interrupt structures,
  in particular the IS_DATA and IS_CODE fields.  Set them
  to NULL if you don't need that interrupt.  Also, set
  the ln_Type of the interrupt structure to NT_INTERRUPT.
  WARNING: don't turn on a disk resource interrupt unless
  the IS_CODE for that interrupt points to executable code!

  IS_CODE will be called with IS_DATA in A1 when the
  interrupt occurs.  Preserve all regs but D0/D1/A0/A1.
  Do not make assumptions about A0.

 RESULTS
lastDriver - if the disk is not busy, then the last unit
  to use the disk is returned.  This may be used to
  see if a driver needs to reset device registers.
  (If you were the last user, then no one has changed
  any of the registers.  If someone else has used it,
  then any allowable changes may have been made).  If the
  disk is busy, then a null is returned.

 EXCEPTIONS

 SEE ALSO
GiveUnit

 BUGS
```

## 1.5 disk.resource/GetUnitID

```
 NAME
     GetUnitID - find out what type of disk is out there

 SYNOPSIS
     idtype = GetUnitID( unitNum ), DRResource
D0              D0         A6

LONG GetUnitID(LONG);

 FUNCTION
Gets the drive ID for a given unit.  Note that this value may
change if someone calls ReadUnitID, and the drive id changes.

 INPUTS
unitNum -- a legal unit number (zero through three)

 RESULTS
idtype -- the type of the disk drive.  Standard types are
  defined in the resource include file.

 EXCEPTIONS

 SEE ALSO
ReadUnitID

 BUGS
```

## 1.6 disk.resource/GiveUnit

```
 NAME
     GiveUnit - Free the disk back up

 SYNOPSIS
     GiveUnit(), DRResource
              A6

void GiveUnit();

 FUNCTION
This routine frees the disk after a driver is done with it.
If others are waiting, it will notify them.

 INPUTS

 RESULTS

 EXCEPTIONS

 SEE ALSO
GetUnit

 BUGS
```

```
In pre-V36, GiveUnit didn't check if you owned the unit.  A patch
for this was part of 1.3.1 SetPatch.  Fixed in V36.
```

## 1.7 disk.resource/ReadUnitID

```
 NAME
     ReadUnitID - reread and return the type of drive (V37)

 SYNOPSIS
     idtype = ReadUnitID( unitNum ), DRResource
D0                 D0         A6

ULONG ReadUnitID(LONG);

 FUNCTION
Rereads the drive id for a specific unit (for handling drives
that change ID according to what sort of disk is in them.  You
MUST have done a GetUnit before calling this function!

 INPUTS
unitNum -- a legal unit number (zero through three)

 RESULTS
idtype -- the type of the disk drive.  Standard types are
  defined in the resource include file.

 EXCEPTIONS

 SEE ALSO
GetUnitID

 BUGS
```