

**NAME**

AttemptDevUnit — Attempt to lock a device/unit

**SYNOPSIS**

```
Failure = AttemptDevUnit(Device, Unit, OwnerName, NotifyBit);
d0                a0      d0      a1      d1.b
```

```
UBYTE  *Failure;
UBYTE  *Device;
ULONG   Unit;
UBYTE  *OwnerName;
UBYTE   NotifyBit;
```

**FUNCTION**

This function will attempt to lock the specified device/unit. It will wait a maximum of five seconds for the device/unit to become free. This delay is to allow an owner that has requested notification to get off the device when the attempt is made. This function is intended for interactive use where blocking indefinitely is undesirable.

**INPUTS***Device*

A pointer to the name of the device you wish to lock.

*Unit*

The unit number of the device you wish to lock.

*OwnerName*

A pointer to a name returned to the caller of `AttemptDevUnit()` when you refuse to give up the lock to someone else.

*NotifyBit*

Some programs, such as *Getty*, sit on a device waiting for a call to come in. It is to their advantage to know when someone else wants the device so that they can release the lock. This would allow someone to use a term program, for instance, and not have to specifically shut down the program sitting on the device.

If you wish to be notified when someone tries to lock the device/unit you own, pass a signal bit number, as returned by `AllocSignal()`, in this argument. When someone tries to lock the device/unit, your task will be signaled. Passing a zero in this argument indicates that you do not wish to be notified when someone requests the device.

**RETURNS**

If the device/unit is successfully locked, `NULL` is returned. If someone else owns the device, a pointer to their **OwnerName** is returned. If an internal error occurred, a pointer to an error message is returned.

All error messages begin with `ODUERR_LEADCHAR`. This allows one to quickly check if the returned value is an error message or owner name. The possible errors are:

`ODUERR_NOMEM . . . . .` An out of memory condition occurred while attempting to lock the device.

ODUERR\_NOTIMER... The timer.device could not be opened while attempting to lock the device.

ODUERR\_BADNAME... An invalid device name was supplied. This occurs only when **Device** is NULL.

ODUERR\_BADBIT... An invalid notify bit was supplied. This occurs only when **NotifyBit** equals -1.

ODUERR\_UNKNOWN... Not strictly an error. This occurs when the lock could not be granted, but the name of the current owner is NULL.

#### *BUGS*

None know.

#### *SEE ALSO*

`LockDevUnit()`, `exec.library/AllocSignal()`

*NAME*

AvailDevUnit — Quickly check the availability of a device/unit

*SYNOPSIS*

```
Truth = AvailDevUnit(Device, Unit);  
d0          a0          d0
```

```
BOOL      Truth;  
UBYTE     *Device;  
ULONG     Unit;
```

*FUNCTION*

This function will quickly determine if a device/unit is currently available (i.e., not locked). It does not perform a lock. It is intended for cases where one must wait for a device/unit to become free, but because of the nature of the code, waiting the five seconds `AttemptDevUnit()` takes to return is undesirable.

*INPUTS*

*Device*

A pointer to the name of the device you wish to check.

*Unit*

The unit number of the device you wish to check.

*RETURNS*

If the device/unit is currently available, `TRUE` is returned. If it is currently locked, `FALSE` is returned.

*BUGS*

None known.

*SEE ALSO*

`AttemptDevUnit()`

**NAME**

FreeDevUnit — Release a lock on a device/unit

**SYNOPSIS**

```
FreeDevUnit(Device, Unit);  
           a0      d0
```

```
UBYTE  *Device;  
ULONG  Unit;
```

**FUNCTION**

Releases a lock on a device/unit previously attained by a call to **AttemptDevUnit()** or **LockDevUnit()**. This function must be called from the same task context that the lock was attained under or the device/unit will not be freed!

**INPUTS**

*Device*

A pointer to the name of the device you wish to release.

*Unit*

The unit number of the device you wish to release.

**RETURNS**

None.

**BUGS**

None known.

**SEE ALSO**

**AttemptDevUnit()**, **FreeDevUnit()**

*NAME*

LockDevUnit — Block until a lock on a device/unit is granted

*SYNOPSIS*

```
Failure = LockDevUnit(Device, Unit, OwnerName, NotifyBit);
d0              a0      d0      a1      d1.b
```

```
UBYTE  *Failure;
UBYTE  *Device;
ULONG  Unit;
UBYTE  *OwnerName;
UBYTE  NotifyBit;
```

*FUNCTION*

This function will block until a lock on the specified device/unit is granted. It is intended for non-interactive use.

*INPUTS**Device*

A pointer to the name of the device you wish to lock.

*Unit*

The unit number of the device you wish to lock.

*OwnerName*

A pointer to a name returned to the caller of `AttemptDevUnit()` when you refuse to give up the lock to someone else.

*NotifyBit*

Some programs, such as *Getty*, sit on a device waiting for a call to come in. It is to their advantage to know when someone else wants the device so that they can release the lock. This would allow someone to use a term program, for instance, and not have to specifically shut down the program sitting on the device.

If you wish to be notified when someone tries to lock the device/unit you own, pass a signal bit number, as returned by `AllocSignal()`, in this argument. When someone tries to lock the device/unit, your task will be signaled. Passing a zero in this argument indicates that you do not wish to be notified when someone requests the device.

*RETURNS*

If the device/unit is successfully locked, `NULL` is returned. If an internal error occurred, a pointer to an error message is returned.

All error messages begin with `ODUERR_LEADCHAR`. This allows one to quickly check if the returned value is an error message or owner name. The possible errors are:

```
ODUERR_NOMEM . . . . . An out of memory condition occurred while attempting to lock the
                        device.
ODUERR_NOTIMER . . . . The timer.device could not be opened while attempting to lock the
                        device.
ODUERR_BADNAME . . . . An invalid device name was supplied. This occurs only when Device
                        is NULL.
```

ODUERR\_BADBIT . . . . An invalid notify bit was supplied. This occurs only when NotifyBit equals -1.

*BUGS*

None know.

*SEE ALSO*

AttemptDevUnit(), exec.library/AllocSignal()

*NAME*

NameDevUnit — Change the owner name value of a locked device/unit

*SYNOPSIS*

```
NameDevUnit(Device, Unit, OwnerName);  
             a0      d0      a1
```

```
UBYTE  *Device;  
ULONG  Unit;  
UBYTE  *OwnerName;
```

*FUNCTION*

This allows one to change the owner name returned by `AttemptDevUnit()`. You must own the lock on the specified device/unit or this function will do nothing.

This function is intended for programs such as *Getty*, which sit on a device and launch other programs as calls are detected.

*INPUTS**Device*

A pointer to the name of the device you wish to alter.

*Unit*

The unit number of the device you wish to alter.

*OwnerName*

A pointer to a name returned to the caller of `AttemptDevUnit()` when you refuse to give up the lock to someone else.

*RETURNS*

None.

*BUGS*

None known.

*SEE ALSO*

`AttemptDevUnit()`

*TABLE OF CONTENTS*

OwnDevUnit.library/AttemptDevUnit  
OwnDevUnit.library/AvailDevUnit  
OwnDevUnit.library/FreeDevUnit  
OwnDevUnit.library/LockDevUnit  
OwnDevUnit.library/NameDevUnit