New Technical Notes

Macintosh

Developer Support

_Gestalt & _SysEnvirons—a Never-Ending Story

Overview **M.OV.GestaltSysenvirons**

Revised by: Tim Dierks September 1992
Dave Radcliffe September 1992
Written by: Jim Friedlander May 1987

This Technical Note discusses the latest changes and improvements to the `_Gestalt` and `_SysEnvirons` calls.

Changes since May 1992: Added information on the `gestaltMachineType` selector for the Quadra 950, Macintosh LC II, Powerbook 145, and future system software.

Introduction

Previous versions of this Note provided the latest documentation on new information the `_SysEnvirons` trap could return. DTS will continue to revise this Note to provide this information; however, as the `_Gestalt` trap is now the preferred method for determining information about a machine environment, this Note will also provide up-to-date information on `_Gestalt` selectors.

_Gestalt

This Note now documents `_Gestalt` selectors and return values added since the release of *Inside Macintosh* Volume VI. Please note that this is supplemental information; for the complete description of `_Gestalt` and its use, please refer to *Inside Macintosh* Volume VI.

The Macintosh LC II is identical to the Macintosh LC, except for the presence of an

MC68030 processor, so it returns the *same* `gestaltMachineType` response as the Macintosh LC (that is, 19). However, the selector will change with a future release of system software. Under System 7.1 and later, the LC II will respond to a `gestaltMachineType` selector with the value 37. Thus, there are two cases when you are on an LC II: under System 7.0.1, you will get a `gestaltMachineType` response of `gestaltMacLC` (19), but `gestaltProcessorType` will return `gestalt68030`; under future system software, `gestaltMachineType` will return `gestaltMacLCII` (37). The processor will, of course, still be a 68030.

There is a similar difficulty with the Powerbook 145. This is essentially a Powerbook 140 with a 25 MHz 68030 processor. Under System 7.0.1, it will return the `gestaltMachineType`

response of `gestaltPowerBook140` (25); under System 7.1 and all later systems, the value returned will change to `gestaltPowerBook145` (54).

Developers are reminded that the `gestaltMachineType` selector is for informational purposes only and should not be used as a basis for programmatic decisions. As always, developers are encouraged to test for the specific features they need and not to rely on any particular machine having a particular set of features.

Note: The *Macintosh PowerBook 100 Developer Notes* and the *Macintosh PowerBook 140/170 Developer Notes*, available from APDA and on the *Developer CD Series* disc and AppleLink, incorrectly document `gestaltMachineType` response values for the Macintosh PowerBook computers. The following values are, and have always been, the correct values.

Additional Gestalt Response Values

```
{ gestaltMachineType response values }
       gestaltMacLC                = 19;   { Macintosh LC }
       gestaltQuadra900            = 20;   { Macintosh Quadra 900 }
       gestaltPowerBook170         = 21;   { Macintosh PowerBook 170 }
       gestaltQuadra700            = 22;   { Macintosh Quadra 700 }
       gestaltClassicII            = 23;   { Macintosh Classic II }
       gestaltPowerBook100         = 24;   { Macintosh PowerBook 100 }
       gestaltPowerBook140         = 25;   { Macintosh PowerBook 140 }
       gestaltQuadra950            = 26;   { Macintosh Quadra 950}
       gestaltMacLCII              = 37;   { Macintosh LC II }
       gestaltPowerBook145         = 54;   { Macintosh PowerBook 145 }

{ gestaltKeyboardType response values }
       gestaltPwrBookADBKbd        = 12;   { PowerBook Keyboard }
       gestaltPwrBookISOADBKbd     = 13;   { PowerBook Keyboard (ISO) }
```

gestaltHardwareAttr Selector

The `gestaltHardwareAttr` selector has been a source of confusion for developers since originally documented in *Inside Macintosh* Volume VI . This section will try to reduce that confusion and also introduce additional information returned by the selector. But be warned that use of this selector for anything other than informational purposes should be deemed a compatibility risk. In other words, if you are dependent on the information returned by this selector to function on existing computers, you will almost certainly have problems on future systems.

The reason for this is that `gestaltHardwareAttr` returns very low-level hardware information. If you need to use this information, it implies you are too hardware dependent. So be very careful about using this information.

The principal source of confusion is bit 7, described as `gestaltHasSCSI`. What this bit really means is the machine is equipped with SCSI based on the 53C80 chip, which was introduced in the Macintosh Plus. This bit will be zero on the Macintosh IIfx and the Macintosh Quadra computers because they have a different low-level SCSI implementation. The Macintosh IIfx has a 53C80 compatible chip that also supports SCSI DMA. It reports this information using bit 6 of the `gestaltHardwareAttr` response. The Macintosh Quadra computers have yet another

SCSI implementation based on the 53C96 chip and so report different information (see below).

Another source of confusion is bit 4 (`gestaltHasSCC`). The Macintosh IIfx and Macintosh Quadra 900 have intelligent I/O processors (IOPs) that normally isolate the hardware and make direct access to the SCC impossible. Normally, these machines will report that they do not have an SCC implying, correctly, that were you to attempt to access it directly, you would fail. However, if the user has used the Compatibility Switch control panel to enable compatibility mode, `gestaltHasSCC` will report true indicating you may access the SCC directly. But remember that doing so means you are doing direct hardware access and that there may be a day when you can't access the SCC under any circumstances.

New gestaltHardwareAttr Values for Macintosh Quadra Computers

Below are the new bits supported by the Macintosh Quadra computers. Any other bits remain undocumented and subject to change.

```
gestaltHasSCSI961                   = 21; { 53C96 SCSI controller on internal bus }

gestaltHasSCSI962                   = 22; { 53C96 SCSI controller on external bus }
```

_SysEnvirons

`_SysEnvirons` was the standard way to determine the features available on a given machine. The preferred method to get this information is now `_Gestalt`; information on `_SysEnvirons` is now provided only for backward compatibility.

As originally conceived, `_SysEnvirons` would check the `versionRequested` parameter to determine what level of information you were prepared to handle, but this technique means updating `_SysEnvirons` for every new hardware product Apple produces. With system software version 6.0, `_SysEnvirons` introduced version 2 of `environsVersion` to provide information about new hardware as we introduce it; this new version returns the same `SysEnvRec` as version 1.

Beginning with system software version 6.0.1, Apple releases a new version of `_SysEnvirons` only when engineering makes changes to its structure (that is, when they add new fields to `SysEnvRec`); all existing versions return accurate information about the machine environment even if part of that information was not originally defined for the version you request. For example, if you call `_SysEnvirons` with `versionRequested = 1` on a Macintosh IIfx, it returns a `machineType` of `envMacIIfx` even though this machine type originally was not defined for version 1 of the call.

You should use version 2 of `_SysEnvirons` until Apple releases a newer version. MPW 3.0 defines a constant `curSysEnvVers`, which can be used to minimize the need for source code revisions when `_SysEnvirons` evolves. Regardless of the version used, however, your software should be prepared to handle unexpected values and should not make assumptions about functionality based on current expectations. For example, if your software currently requires a Macintosh II, testing for `machineType >= envMacII` may result in your software trying to run on a machine that does not support the features it requires, so test for specific functionality (that is, `hasFPU`, `hasColorQD`, and so on).

Warning: This test for specific functionality is particularly true of FPUs (floating-point units). Some CPUs, such as the Macintosh IIsi, may have optional, user-

installed FPUs; therefore, an application should not assume that any Macintosh with a microprocessor greater than a 68000 (for example, 68020, 68030, or 68040) has an FPU (68881/68882 or built-in for the 68040). If an application makes a conditional branch to execute floating-point instructions directly, then it should first explicitly check for the presence of the FPU.

You should always check the `environsVersion` when returning from `_SysEnvirons` since the glue always returns as much information as possible, with `environsVersion` indicating the highest version available, even if the call returns an `envSelTooBig` (–5502) error.

Calling _SysEnvirons From a High-Level Language

Due to a documentation error in *Inside Macintosh* Volume V, DTS still receives questions about how to call `_SysEnvirons` properly from Pascal and C. *Inside Macintosh* defines the Pascal interface to `_SysEnvirons` as follows:

```
FUNCTION SysEnvirons (versRequested: INTEGER; VAR theWorld: SysEnvRecPtr) : OSErr;
```

Because `theWorld` is passed by reference (as a `VAR` parameter), it is not correct to pass a `SysEnvRecPtr` in the second argument. Pascal would then generate a pointer to this pointer and pass that to the `_SysEnvirons` trap in `A0`. (The assembly-language information is essentially correct; `_SysEnvirons` really does want a pointer to a `SysEnvRec` in `A0`.) The correct Pascal interface to `_SysEnvirons` is therefore:

```
FUNCTION SysEnvirons (versionRequested: INTEGER; VAR theWorld: SysEnvRec) : OSErr;
```

In this case, Pascal pushes a pointer to `theWorld` on the stack. The Pascal interface glue then pops this pointer off the stack directly into `A0` and calls `_SysEnvirons`. Everything is copacetic.

C programmers should recognize their corresponding interface:

```
pascal OSErr SysEnvirons (short versionRequested, SysEnvRec *theWorld);
```

Inside Macintosh defines the type `SysEnvPtr = ^SysEnvRec`. It also sometimes refers to this type as `SysEnvRecPtr`. The inconsistency is insignificant because in reality MPW does not define any such type, under either name; therefore, it is never needed.

Inside Macintosh also states that “all of the Toolbox Managers must be initialized before calling SysEnvirons.” This statement is not necessarily true. Startup documents (INITs), for instance, may wish to call `_SysEnvirons` without initializing any of the Toolbox Managers. Keep in mind that the `atDrvrVersNum` field returns a zero result if the AppleTalk drivers are not initialized. The system version, machine type, processor type, and other key data return normally.

Additional _SysEnvirons Constants

The following are new `_SysEnvirons` constants which are not documented in *Inside Macintosh*; however, you should refer to *Inside Macintosh* Volume V-1, Compatibility

Guidelines, for the rest of the story.

machineType

```
envMacIIx                = 5;    { Macintosh IIx }
envMacIIcx               = 6;    { Macintosh IIcx }
envSE30                  = 7;    { Macintosh SE/30 }
envPortable              = 8;    { Macintosh Portable }
envMacIIci               = 9;    { Macintosh IIci }
envMacIIfx               = 11;   { Macintosh IIfx }
envMacClassic            = 15;   { Macintosh Classic }
envMacIIsi               = 16;   { Macintosh IIsi }
envMacLC                 = 17;   { Macintosh LC }
envMacQuadra900          = 18;   { Macintosh Quadra 900 }
envMacPowerBook170       = 19;   { Macintosh PowerBook 170 }
envMacQuadra700          = 20;   { Macintosh Quadra 700 }
envMacClassicII          = 21;   { Macintosh Classic II }
envMacPowerBook100       = 22;   { Macintosh PowerBook 100 }
envMacPowerBook140       = 23;   { Macintosh PowerBook 140 }
envMacQuadra950          = 24;   { Macintosh Quadra 950}
envMacLCII               = 35;   { Macintosh LC II }
envMacPowerBook145       = 52;   { Macintosh PowerBook 145 }
```

processor

```
env68030                 = 4;    { MC68030 processor }
env68040                 = 5;    { MC68040 processor }
```

keyBoardType

```
envPrtblADBKbd           = 6;    { Portable Keyboard }
envPrtblISOKbd           = 7;    { Portable Keyboard (ISO) }
envStdISOADBKbd          = 8;    { Apple Standard Keyboard (ISO) }
envExtISOADBKbd          = 9;    { Apple Extended Keyboard (ISO) }
envADBKbdII              = 10;   { Apple Keyboard II }
envADBISOKbdII           = 11;   { Apple Keyboard II (ISO) }
envPwrBkADBKbd           = 12;   { PowerBook Keyboard }
envPwrBkISOKbd           = 13;   { PowerBook Keyboard (ISO) }
```

Further Reference:

- *Inside Macintosh*, Volumes V and VI, Compatibility Guidelines