New Technical Notes

Macintosh ®

Developer Support

OV 16 - Gestalt & _SysEnvirons—A Never-Ending Story

Overview

Revised by:	Rich Kubota,	July 1995
Revised by:	Rich Kubota,	April 1995
Revised by:	Rich Kubota,	February 1995
Revised by:	Rich Kubota, Sunny Singha, Joe Guo	December 1994
Written by:	Jim Friedlander	May 1987

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

Changes since April 1995: Added information on the gestaltMachineType selector for the Power Macintosh LC 5200/75, Power Macintosh 9500, and Macintosh PowerBook 5300.
Changes since February 1995: Correction - the gestaltMachineType for the Power Mac 6100/66 is 100. Added information on the gestaltMachineType selector for the Performa 611x.
Changes since December 1994: Added information on the gestaltMachineType selector for the Power PC Processor Upgrade card installed on the Quadra/LC/Performa 630, LC/Performa 475, and on the LC/Performa 57x systems. Added information on the gestaltMachineType selector for the Power Mac 6100/66, 7100/80, and for the 8100/100. Note there may be reports that the
Changes since September 1994: Added information on the gestaltMachineType selector for the Power PC Upgrade card installed on the various Quadra and Centris systems. Added the selectors for the known Performa models. Removed the machine type SysEnvirons list.

Introduction

Previous versions of this Note provided the latest documentation on new information the _SysEnvirons trap could return. Developer Support Center (DSC) 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 under System 7.0.1 it returns the *same* gestaltMachineType response as the Macintosh LC (that is, 19). However, under System 7.1 and later, the LC II responds 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 returns the gestaltMachineType response of gestaltPowerBook140 (25); under System 7.1 and all later system software versions, the value returned is 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. This requirement becomes even more important given that the PowerBook 520 and 540 and their color variants, are all identified using the same gestalt selector ID. To distinguish between the two model types, use the new Power Manager call, MaximumProcessorSpeed as documented in the developer notes for the PowerBook 520/540.

In the future, existing Gestalt ID's will be reused and it will not be possible to distinguish Macintosh models by Gestalt machine ID. Typically, the only difference between Macintosh models will be in software bundling. To go along with this change, STR# (-13695) which has traditionally stored the Macintosh name, will be altered so that each string becomes <sp><sp>Macintosh.

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 }
    gestaltPerforma200              = 23;         { Macintosh Performa 200 }
    gestaltPowerBook100             = 24;         { Macintosh PowerBook 100 }
    gestaltPowerBook140             = 25;         { Macintosh PowerBook 140 }
    gestaltQuadra950                = 26;         { Macintosh Quadra 950}
    gestaltMacLCIII                 = 27;         { Macintosh LC III }
    gestaltPerforma450              = 27;         { Macintosh Performa 450 }
    gestaltPowerBookDuo210          = 29;         { Macintosh PowerBook Duo 210 }
    gestaltMacCentris650            = 30;         { Macintosh Centris 650 }
    gestaltPowerBookDuo230          = 32;         { Macintosh PowerBook Duo 230 }
    gestaltPowerBook180             = 33;         { Macintosh PowerBook 180 }
    gestaltPowerBook160             = 34;         { Macintosh PowerBook 160 }
    gestaltMacQuadra800             = 35;         { Macintosh Quadra 800 }
    gestaltMacQuadra650             = 36;         { Macintosh Quadra 650 }
    gestaltMacLCII                  = 37;         { Macintosh LC II }
    gestaltPerforma40x              = 37;         { Macintosh Performa 40x }
    gestaltPerforma430              = 37;         { Macintosh Performa 430 }
    gestaltPowerBookDuo250          = 38;         { Macintosh PowerBook Duo 250 }
    gestaltPowerMac9150             = 39;         { Power Macintosh 9150 }
    gestaltPowerMac8100_110         = 40;         { Power Macintosh 8100/110 }
```

```
    gestaltPowerMacLC5200_75        = 41;       { Power Macintosh LC 5200/75 }
    gestaltMacIIvi                  = 44;       { Macintosh IIvi }
    gestaltMacIIvm                  = 45;       { Macintosh IIvm }
    gestaltPerforma600              = 45;       { Macintosh Performa 600 }
    gestaltPowerMac7100_80          = 47;       { Power Macintosh 7100/80 }
    gestaltMacIIvx                  = 48;       { Macintosh IIvx }
    gestaltMacColorClassic          = 49;       { Macintosh Color Classic }
    gestaltPerforma250              = 49;       { Macintosh Performa 250 }
    gestaltPowerBook165c            = 50;       { Macintosh PowerBook 165c }
    gestaltMacCentris610            = 52;       { Macintosh Centris 610 }
    gestaltMacQuadra610             = 53;       { Macintosh Quadra 610 }
    gestaltPowerBook145             = 54;       { Macintosh PowerBook 145 & 145b}
    gestaltPowerMac8100_100         = 55;       { Power Macintosh 8100/100 }
    gestaltMacLC520                 = 56;       { Macintosh LC 520 }
    gestaltMacCentris660AV          = 60;       { Macintosh Centris 660AV }
    gestaltPerforma46x              = 62;       { Macintosh Performa 46x }
    gestaltPowerMac8100_80          = 65;       { Power Macintosh 8100/80 }
    gestaltPowerMac9500             = 67;       { Power Macintosh 9500 }
    gestaltPowerBook180c            = 71;       { Macintosh PowerBook 180c }
    gestaltPowerBook520_540         = 72;       { Macintosh PowerBook 520/520c/540/540c }
    gestaltPowerMac6100_60          = 75;       { Power Macintosh 6100/60 }
    gestaltPerforma611x             = 75;       { Macintosh Performa 611x }
    gestaltPowerBookDuo270c         = 77;       { Macintosh PowerBook Duo 270c }
    gestaltMacQuadra840AV           = 78;       { Macintosh Quadra 840AV }
    gestaltMacLC550                 = 80;       { Macintosh LC 550 }
    gestaltPerforma550              = 80;       { Macintosh Performa 550 }
    gestaltPerforma560              = 80;       { Macintosh Performa 560 }
    gestaltPowerBook165             = 84;       { Macintosh PowerBook 165 }
    gestaltMacTV                    = 88;       { Macintosh TV }
    gestaltMacLC475                 = 89;       { Macintosh LC 475 }
    gestaltPerforma47x              = 89;       { Macintosh Performa 47x }
    gestaltMacLC575                 = 92;       { Macintosh LC 575 }
    gestaltPerforma57x              = 92;       { Macintosh Performa 57x }
    gestaltMacQuadra605             = 94;       { Macintosh Quadra 605 }
    gestaltMacQuadra630             = 98;       { Macintosh Quadra 630 - see LC 630 }
    gestaltMacLC630                 = 98;       { Macintosh LC 630 - see Quadra 630 }
    gestaltPerforma63x              = 98;       { Macintosh Performa 63x }
    gestaltPowerMac6100_66          = 100;      { Power Macintosh 6100/66 }
    gestaltPowerBookDuo280          = 102;      { Macintosh PowerBook Duo 280 }
    gestaltPowerBookDuo280c         = 103;      { Macintosh PowerBook Duo 280c }
    gestaltPowerMacLC475            = 104;      { Mac LC 475 & PPC Processor Upgrade Card }
    gestaltPowerMacPerforma47x      = 104;      { Performa 47x & PPC Proc Upgrade Card }
    gestaltPowerMacLC575            = 105;      { Mac LC 575 & PPC Processor Upgrade Card }
    gestaltPowerMacPerforma57x      = 105;      { Performa 57x & PPC Proc Upgrade Card }
    gestaltPowerMacQuadra630        = 106;      { Quadra 630 & PPC Processor Upgrade Card }
    gestaltPowerMacLC630            = 106;      { Mac LC 630 & PPC Processor Upgrade Card }
    gestaltPowerMacPerforma63x      = 106;      { Performa 63x & PPC Proc Upgrade Card }
    gestaltPowerMac7100_66          = 112;      { Power Macintosh 7100/66 }
    gestaltPowerBook150             = 115;      { Macintosh PowerBook 150 }
    gestaltPowerMacQuadra700        = 116;      { Quadra 700 & Power PC Upgrade Card }
    gestaltPowerMacQuadra900        = 117;      { Quadra 900 & Power PC Upgrade Card }
    gestaltPowerMacQuadra950        = 118;      { Quadra 950 & Power PC Upgrade Card }
    gestaltPowerMacCentris610       = 119;      { Centris 610 & Power PC Upgrade Card }
    gestaltPowerMacCentris650       = 120;      { Centris 650 & Power PC Upgrade Card }
    gestaltPowerMacQuadra610        = 121;      { Quadra 610 & Power PC Upgrade Card }
    gestaltPowerMacQuadra650        = 122;      { Quadra 650 & Power PC Upgrade Card }
    gestaltPowerMacQuadra800        = 123;      { Quadra 800 & Power PC Upgrade Card }
    gestaltPowerBook5300            = 128;      { Macintosh PowerBook 5300 }


{ new gestaltKeyboardType response values including those for the PowerBook models }
    gestaltPwrBookADBKbd            = 12;       { PowerBook Keyboard }
    gestaltPwrBookISOADBKbd         = 13;       { PowerBook Keyboard (ISO) }
```

```
gestaltAppleAdjustKeypad        = 14;          { Apple Adjustable Keypad }
gestaltAppleAdjustADBKbd        = 15;          { Apple Adjustable Keyboard }
                                               { includes US, ISO, and Japanese }
```

gestaltHardwareAttr Selector

The `gestaltHardwareAttr` selector has been a source of confusion for developers since originally documented in *Inside Macintosh* Volume V. 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 that 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 that 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.

One other source of confusion is bit 3 (`gestaltHasASC`). This flag was originally created to determine if the machine has the Apple Sound Chip, which was built into the Macintosh II. In the future, there will be new sound hardware that will not necessarily be the Apple Sound Chip. The question then arises whether this flag should be set when the new sound hardware is not an Apple Sound Chip. For example, if Apple were to build a Macintosh with a DSP chip instead of the Apple Sound Chip, should the `gestaltHasASC` flag be set?

Some Developers have been assuming that the `gestaltHasASC` flag determines if SndStartFilePlay or SndPlayDoubleBuffer are supported. This is a bad assumption. Currently these two Sound Manager functions are supported only on Macintosh computers that have the Apple Sound Chip, but this will change. A Macintosh Classic does not have the Apple Sound Chip but may support these two Sound Manager Functions. Additionally, the imaginary Macintosh of the future (containing a DSP) may or may not have the `gestaltHasASC` flag set, but will certainly have the ability to support SndStartFilePlay or SndPlayDoubleBuffer.

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 }
```

A Better Method to Detect (Adjustable) Keyboard Type

As documented in *Inside Macintosh* Volume VI, Gestalt simply identifies the last keyboard that was typed on. Confusion can arise when the Apple Adjustable Keyboard is attached. Gestalt can return the identifier for either the keypad or the keyboard, depending on which device was most recently used. As such, Gestalt is useful for determining which keyboard a given event came from, but may not identify all of the keyboards that are attached. The same problem would occur if one were to attach multiple keyboards of different types to any Macintosh and restart the system.

For Adjustable Keyboards there are some additional concerns - Gestalt returns the same value for all three variants of the Adjustable keyboard - US standard, International (ISO), and Japanese. For all three types of keyboards the response is 0x0F. There is also a known problem with some early releases of System Software such that using Gestalt to identify an adjustable keyboard failed with error -5550. For these reasons, an alternate method for identifying keyboard devices is provided below.

To obtain information on all keyboards attached to the machine, you can ask the ADB (Apple Desktop Bus) directly. The basic algorithm would is to:
1. Call `CountADBs` to determine the number of ADB devices that are attached.
2. For each device, `GetIndADB` and check the `origADBAddr` field to determine whether it is a keyboard device (`origADBAddr` == 0x02).
3. For keyboard devices, the `devType` field is the keyboard device type. Refer to *Inside Macintosh* Volume V for more specifics on these two calls.

Note that the `devType` field response do not exactly equate to those returned by Gestalt. The following list shows the gestalt responses and the `devType` field responses

GestaltName	Gestalt response	devType
gestaltMacKbd	1	0x03
gestaltMacAndPad	2	0x13
gestaltMacPlusKbd	3	0x0B
gestaltExtADBKbd	4	0x02
gestaltStdADBKbd	5	0x01
gestaltPrtblADBKbd	6	0x06
gestaltPrtblISOKbd	7	0x07
gestaltStdISOADBKbd	8	0x04
gestaltExtISOADBKbd	9	0x05
gestaltADBKbdII	10	0x08
gestaltADBISOKbdII	11	0x09
gestaltPwrBookADBKbd	12	0x0C
gestaltPwrBookISOADBKbd	13	0x0D
gestaltAppleAdjustKeypad	14,	0x0E
gestaltAppleAdjustADBKbd	15,	0x10
gestaltAppleAdjustADBISOKbd	15,	0x11
gestaltAppleAdjustJapanADBKbd	15,	0x12

Non-existent Gestalt Selector - 'icon'

Inside Macintosh - More Macintosh Toolbox volume, page 5-7, specifies that the `gestaltIconUtilitiesAttr - 'icon'` gestalt selector can be used to determine whether the icon utilities are present under System 7.x. Note that this selector is included in the GestaltEqu files. It turns out that this selector is not implemented until System Software v7.1.2. To check for the existence of these utilities, use the TrapAvailable code to check for the `_IconDispatch`, (0xABC9) trap. The TrapAvailable code is presented in *Inside Macintosh* VI 3-8, and as sample code in many of the snippets on the Developer CD.

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, DSC 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 that are not documented in *Inside Macintosh*; however, you should refer to *Inside Macintosh* Volume V, Chapter 1, Compatibility Guidelines, for the rest of the story. The machine type `_SysEnvirons` constants for the various Macintosh models are two less than those for the 'mach' `Gestalt` selector listed above. The previous list of machine type constants for `_SysEnvirons` has been removed to simplify the update of this Tech Note.

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) }
envAppleAdjustKeypad        = 14;          { Apple Adjustable Keypad }
envAppleAdjustADBKbd        = 15;          { Apple Adjustable Keyboard }
                                           { includes US, ISO, and Japanese }
```

Further Reference:

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