32 ビットおよび 64 ビット プラットフォーム用のドライバのインストール

 2007 年 12 月 17 日

機械翻訳に対する免責事項

本書は、英語で作成されたホワイト ペーパーを、機械翻訳システムにより日本語に翻訳したものであり、日本語をお使いになるお客様が英語原文を理解する上での便宜として、参考用に提供するものです。

Microsoft は和訳の正確さについて、一切保証するものではありません。また、本和訳文書の使用によって、直接または間接的に起こりうる一切の問題について、Microsoft はいかなる責任も負わないものとします。

要約

32 ビットと 64 ビットの両方のプラットフォームにデバイスをインストールする場合、プラットフォームごとにネイティブ ドライバ パッケージを提供する必要があります。ただし、単一の 32 ビット インストーラを使用して、どちらのドライバ パッケージもインストールできます。このペーパーでは、Microsoft® Windows® ファミリ オペレーティング システムに 32 ビットまたは 64 ビットのドライバをインストールする、32 ビット インストール プログラムの使用方法について説明します。

この情報は、下記のオペレーティング システムに適用されます。
 Windows Server® 2008
 Windows Vista®
 Windows Server® 2003
 Windows Server 2003 64-Bit Edition
 Windows XP
 Windows XP 64-Bit Edition

最新情報については、下記の URL で、WDK ドキュメントの「Device Installation Design Guide」を参照してください。

<http://msdn2.microsoft.com/en-us/library/ms790231.aspx>

このペーパーの最新バージョンは、次の Web サイトに掲載されています。
 <http://www.microsoft.com/whdc/driver/install/32-64bit_install.mspx>

ここで説明しているリファレンスやリソースの一覧は、このペーパーの最後に掲載されています。

目次

[はじめに 3](#_Toc189385975)

[64 ビット プラットフォームへのインストールの統合 3](#_Toc189385976)

[32 ビットおよび 64 ビット プラットフォーム用のドライバ インストール パッケージ 4](#_Toc189385977)

[ドライバ パッケージ 4](#_Toc189385978)

[DPInst インストール パッケージ 5](#_Toc189385979)

[ドライバ パッケージ インストーラ 5](#_Toc189385980)

[DIFxApp による MSI インストール パッケージ 6](#_Toc189385981)

[例 6](#_Toc189385982)

[インストーラから、正しいバージョンの DPInst を起動する 6](#_Toc189385983)

[コマンド スクリプトから、正しいバージョンの DPInst を起動する 10](#_Toc189385984)

[リソース 10](#_Toc189385985)

免責事項

本書には、今後変更される可能性のある予備情報も含まれます。

本書に記載されている情報は本書の発行時点における Microsoft の見解を述べたものです。市場ニーズの変化に対応する必要があるため、本書は記載された内容の実現に関する Microsoft の確約とはみなされないものとします。また本書に記載された情報の正確さについて、保証するものではありません。

本書は情報の提供のみを目的としており、明示または黙示に関わらず、本書について Microsoft はいかなる保証をするものでもありません。

適用される著作権法に従うことはお客様の責任です。本書の一部または全部を、電子的、機械的、複写、録音、その他いかなる手段およびいかなる形式によっても、またいかなる目的のためにも、Microsoft の書面による許可なく複製、転送、または検索システム等へ格納等することは禁じられています。ただし、これによってお客様に著作権法上認められる権利を制限するものでもありません。

Microsoft は本書に記載されている内容に対して、特許権、特許出願、商標権、著作権、またはその他の知的所有権を有する場合があります。本書は Microsoft の書面による明示的な許諾がある場合を除き、これらの特許権、商標権、著作権またはその他の知的所有権に関する権利をお客様に許諾するものではありません。

別途記載のない限り、本書中に記載されたサンプルの会社名、団体名、製品名、ドメイン名、氏名、電子メール アドレス、ロゴ、人、場所、出来事等は架空のものであり、実在の会社、団体、製品、ドメイン名、電子メール アドレス、ロゴ、人、場所または出来事との関連を示唆するものではありません。

© 2005-2006 Microsoft Corporation. All rights reserved.

Microsoft、 Windows、Windows NT、Windows Server、および Windows Vista は、米国 Microsoft Corporation およびその関連会社の米国およびその他の国における登録商標または商標です。

本書中で使用されている実在の会社名や製品名には、各権利者の保有する商標が含まれることがあります。

#### 変更履歴

|  |  |
| --- | --- |
| **日付** | **変更点** |
| 2007 年 12 月 17 日 | WDK ドキュメントの参照情報を更新 |
| 2005 年 3 月 25 日 | 初版 |

# はじめに

32 ビット インストーラで 64 ビット ドライバをインストールするのを簡略化するため、Microsoft は、Windows Server 2003 SP1 DDK で、Driver Install Frameworks (DIFx) ツールの 64 ビット バージョンを提供します。DIFx ツールは、下記のものなどです。

* The Driver Package Installer (DPInst)
* The Driver Installation Frameworks for Applications (DIFxApp)

64 ビット プラットフォーム用に別々の 64 ビット インストーラを作成する代わりに、単一の 32 ビット インストーラから、DPInst または DIFxApp の正しいバージョンを起動できます。

32 ビット ドライバのインストール パッケージを使用して、64 ビット ドライバを直接インストールすることはできません。しかし、それらのパッケージを使用して、どのアーキテクチャが動作中かを判定してから、適切なインストーラを起動することは可能です。64 ビット プラットフォームで、32 ビット インストーラは、Windows on Windows (WOW64) において動作します。WOW64 は、Win32 ベースのアプリケーションを 64 ビット Windows で実行できる、x64 エミュレータです。インストーラは、プラットフォームに対して正しいドライバ パッケージをインストールできるよう、自身が動作しているプラットフォームを検出できる必要があります。

このペーパーでは、32 ビット インストール プログラムから、Microsoft® Windows® ファミリ オペレーティング システム用の 32 ビットまたは 64 ビットのドライバをインストールする方法を説明します。

DIFx ツールについては、このペーパー末尾の「リファレンス」を参照してください。

# 64 ビット プラットフォームへのインストールの統合

64 ビット バージョンの Windows は、Win32 アプリケーションと Win64 アプリケーションの両方で、単一のソース コード ベースを開発者が使用できるよう設計されています。大半の場合、これは、32 ビットと 64 ビットの Windows ドライバにおいても当てはまります。

ユーザー モード アプリケーションの場合、64 ビット Windows には、Windows on Windows (WOW64) と呼ばれるエミュレーション レイヤが含まれています。WOW 64 は、Win32 アプリケーションを Win64 システムで実行できるようにします。WOW64 は、64 ビット カーネルに切り替える前に、必要に応じて、Win32 関数呼び出しを途中で受信し、ポインタ精度のパラメータ型を固定精度の型に変換することにより、上記のような実行を可能にします。("サンク" とも呼ばれるこの変換プロセスは、32 ビット アプリケーションの場合のみ行われます。32 ビット ドライバは、64 ビット Windows ではサポートされません。)

UpdateDriverForPlugAndPlayDevices、InstallSelectedDriver、SetupCopyOEMInf などのインストール関数は、WOW64 対応ではなく、64 ビット Windows で 32 ビット プロセスから呼び出された場合、エラーを返します。この機能は、下記の理由で有効にされませんでした。

* Windows へのドライバ インストールでは、DIFx ツールが推奨の方法です。これらのツールは、64 ビット バージョンで利用でき、将来のバージョンで機能や柔軟性が向上する予定です。
* インストール関数を WOW64 で有効にするだけでは、レガシ 32 ビット インストーラを、64 ビット Windows で動作可能にすることはできませんでした。これには、少なくとも、新しい INF モデルが必要であり、多くのレガシ インストーラをかなり変更する必要があります。
* コンピューティング モデルは、64 ビット プラットフォームの方向へ進化しつつあります。32 ビット インストール パッケージが 64 ビット ドライバを操作できるようにするフレームワークやルールのセットを作成しても、暫定的なソリューションが得られるだけです。

デバイス インストーラなど、32 ビット アプリケーションは、CreateProcess を呼び出すことにより、64 ビット実行可能ファイルを起動できます。WOW64 は、この呼び出しを途中で受信して、64 ビット オペレーティング システムへ正常に切り替えることができます。

# 32 ビットおよび 64 ビット プラットフォーム用のドライバ インストール パッケージ

「ドライバ パッケージ」は、任意の Windows プラットフォームにデバイスをインストールしてサービスするのに必要なファイルから成ります。「ドライバ インストール パッケージ」は、1 つ以上のドライバ パッケージと、ドライバ パッケージをインストールするツール (DPInst など) から成ります。

ドライバ インストール パッケージに 32 ビット ドライバ パッケージのみが含まれている場合、ユーザーがこのパッケージを 64 ビット プラットフォームにインストールしようとしたら、インストーラは、詳細なエラー メッセージを表示する必要があります。たとえば、インストーラは、新しいハードウェアの検索ウィザードで表示される下記のようなメッセージを表示する必要があります。

このデバイス用の、互換性のあるソフトウェア ドライバが見つかりません。ドライバが提供されている場合には、x64 ベース システムの Windows 上で動作するように設計されていることを確認してください。適切なドライバがない場合には、デバイスの製造元に問い合わせてください。

## ドライバ パッケージ

ドライバ パッケージは、1 つ以上のプラットフォームを対象とします。たとえば、ドライバ パッケージは、x86 または AMD64 のみを対象とすることも、x86 と AMD64 の両方を対象とすることも可能です。また、Intel Itanium ベースのシステムを対象とすることもできます。ただし、x86 ベースのアプリケーションおよびインストーラは AMD64 でも動作できるので、簡潔さのため、ここでは x86 と AMD64 を中心に取り上げます。

単一のドライバ パッケージを、32 ビット プラットフォームと 64 ビット プラットフォームの両方にインストールする場合、下記のものが含まれている必要があります。

* 32 ビットと 64 ビットでコンパイルした、すべてのカーネル モード ドライバ、デバイス インストール アプリケーション、デバイス インストーラ、クラス インストーラ、および共同インストーラ。
* プラットフォーム固有のインストール動作を制御する、デコレートされた INF セクションの使用。

場合によっては、32 ビット プラットフォームと 64 ビット プラットフォームで、別個のドライバ パッケージを提供するのが望ましいこともあります。可能な場合、プラットフォーム固有の拡張機能で、セクションを常にデコレートすることをお勧めします。つまり、可能な場合は常に、SourceDisksFiles、SourceDisksNames、DDInstall、および Models の各セクションをデコレートする必要があります。これにより、十分な知識のないユーザーが、プラットフォームと互換性がない INF を知らずにインストールするのを避けることができます。

たとえば、mydriver.inf は、x86 と AMD64 の両方を対象としています。このパッケージをレイアウトする一般的な方法は、次のとおりです。

mydriver.inf (with decorated SourceDisksFiles, SourceDisksNames, …)

mydriver.cat

AMD64

 mydriver.sys

X86

 mydriver.sys

## DPInst インストール パッケージ

DPInst のプラットフォーム固有のバージョンは、INF によってサポートされている対象プラットフォームごとに提供する必要があります。

32 ビットおよび 64 ビット プラットフォーム用の、単一のドライバ パッケージ。ドライバ パッケージ内のすべてのファイルは、DPInst.exe の両バージョンと共に、単一のディレクトリに存在する必要があります。また、DPInst.exe は、名前を変更する必要があります。\*install.exe は、後で説明するアプリケーションです。例 :

Driver32and64\
 Install.exe \* (launches either DPInst32.exe or DPInst64.exe)
 DPInst32.exe (32-bit DPInst.exe renamed to DPInst32.exe)
 DPInst64.exe (64-bit DPInst.exe renamed to DPInst64.exe)

 mydriver.inf

 AMD64

 mydriver.sys (to be installed on 64-bit platform)

 X86

 mydriver.sys (to be installed on 32-bit platform)

 …

DPInst の詳細オプションについては、ホワイト ペーパー「Using the Driver Package Installer (DPInst)」を参照してください。DPInst は、特定のサブディレクトリ内でドライバを検索するよう構成できます。その場合、ドライバは、INF と同じディレクトリに存在する必要はありません。

32 ビット プラットフォームと 64 ビット プラットフォーム用の、別個のドライバ パッケージ。 各ドライバ パッケージは、対象プラットフォーム用の DPInst.exe のバージョンと共に、別個のディレクトリに存在する必要があります。例 :

Install.exe

Driver32\
 Driver32.inf
 DPInst.exe (32-bit version)
 …

Driver64\
 Driver64.inf
 DPInst.exe (64-bit version)
 …

この例において、Install.exe は、このペーパーで後述するアプリケーションです。

## ドライバ パッケージ インストーラ

32 ビットまたは 64 ビットのドライバ パッケージをインストールするには、32 ビット インストーラは、自身がどのプラットフォームで動作しているか検出できる必要があります。64 ビット プラットフォームの場合、32 ビット インストーラは、WOW64 において動作します。

インストーラ (Installer.exe) は、下記のいずれかの方法で、自身がどのプラットフォームで動作しているか検出できます。

* プロセスへのハンドルとブール値へのポインタを使用して、IsWow64Process を呼び出します。

プロセスが WOW64 において動作している場合、**IsWow64Process** はブール値を TRUE に設定します。そうでない場合、ブール値を FALSE に設定します。

* UpdateDriverForPlugAndPlayDevices によって返された値をチェックします。戻り値が ERROR\_IN\_WOW64 の場合、32 ビット アプリケーションは、64 ビット プラットフォームで動作しています。
* GetSystemWow64Directory を呼び出します。

GetSystemWow64Directory が失敗し、GetLastErrorがERROR\_CALL\_NOT\_IMPLEMENTED を返す場合、インストーラは 32 ビット Windows で動作しています。

その後、インストーラは、CreateProcess を呼び出し、対象プラットフォーム用の DPInst を起動できます。

* 32 ビット プラットフォームの場合、インストーラは DPInst の 32 ビット バージョンを起動し、DPInst は 32 ビット ドライバ パッケージをインストールします。
* 64 ビット プラットフォームの場合、インストーラは DPInst の 64 ビット バージョンを起動し、DPInst は 64 ビット ドライバ パッケージをインストールします。

## DIFxApp による MSI インストール パッケージ

Driver Install Frameworks for Applications (DIFxApp) を使用して、ユーザー モード アプリケーションと関連付けられたドライバの MSI インストール パッケージをインストールできます。各 MSI パッケージには、対象プラットフォーム用の DIFxApp.msm マージ モジュールが含まれている必要があります。

アプリケーションと関連付けられたドライバ パッケージの場合、Microsoft Windows インストーラ (MSI) パッケージを、プラットフォームごとに作成する必要があります。

# 例

このセクションのいくつかの例は、32 ビット Windows または 64 ビット Windows 用の DPInst の正しいバージョンを起動する方法を示します。

## インストーラから、正しいバージョンの DPInst を起動する

下記のコード例は、32 ビット Windows または 64 ビット Windows 用の DPInst の正しいバージョンを、インストーラが起動する方法を示します。簡潔に記すと、この例は下記のことを行います。

1. インストーラが、32 ビット Windows または 64 ビット Windows のどちらで動作しているか判定します。

2. 32 ビット Windows 用の Dpinstx86.exe または 64 ビット Windows 用の Dpinstx64.exe を起動します。

インストーラが、32 ビット Windows または 64 ビット Windows のどちらで動作しているか判定する

下記のコード断片は、CreateProcess 呼び出し用の変数を初期化し、ヘルパー関数 IsWow64 を呼び出して、インストーラが 32 ビット または 64 ビット Windows のどちらで動作しているか判定します。

インストーラが 64 ビット Windows で動作している場合、IsWow64 は TRUE を返し、32 ビット Windows で動作している場合、FALSE を返します。この戻り値に基づき、コードは、"dpinstx86.exe" または "dpinstx64.exe" を *AppName* という変数に割り当て、次に、*AppName* の値を command という変数に割り当てます。

|  |
| --- |
| #include <stdio.h>#include <windows.h>#include <tchar.h>#include <strsafe.h>static BOOL IsWow64( void );static void PrintLastError( const WCHAR\* apiname );int \_\_cdecl wmain( int argc, wchar\_t\* argv[] ){ STARTUPINFO StartupInfo; PROCESS\_INFORMATION ProcessInformation; UNREFERENCED\_PARAMETER(argc); UNREFERENCED\_PARAMETER(argv); wprintf( L"Running dpinst depending on the underlying platform\n" ); WCHAR \* X86 = TEXT("dpinstx86.exe");// For separate driver package, WCHAR \* X86 = TEXT(“Driver32\dpinst.exe”) WCHAR \* X64 = TEXT("dpinstx64.exe");// For separate driver package, WCHAR \* X64 = TEXT(“Driver64\dpinst.exe”) WCHAR \* AppName = NULL; WCHAR \* command = NULL; if (IsWow64())  { AppName = X64; wprintf( L"We are on a 64 bit OS.\n" ); } else { AppName = X86; wprintf( L"We are on a 32 bit OS.\n" ); } command = (WCHAR\*)HeapAlloc( GetProcessHeap(), 0, (wcslen(AppName)+1)\*sizeof(WCHAR) ); if( command == NULL ) { wprintf( L"Out of memory.\n" ); return 0; }  StringCchCopy( command, wcslen(AppName)+1, AppName ); ZeroMemory( &ProcessInformation, sizeof( ProcessInformation ) ); ZeroMemory( &StartupInfo, sizeof( StartupInfo ) ); StartupInfo.cb = sizeof( STARTUPINFO );   |

CreateProcess を呼び出し、DPInst の正しいバージョンを起動する

この例は、CreateProcess を呼び出し、command を、実行するコマンド ライン (lpCommandLine) として渡します。CreateProcess が成功した場合、ゼロ以外の値を返し、新しいプロセスおよびそのプライマリ スレッドを作成します。

新しいプロセスは、指定された実行可能ファイル (Dpinstx86.exe または Dpinstx64.exe) を、呼び出しプロセスのセキュリティ コンテキストで実行します。インストーラは、プロセスが完了するのを待機し (WaitForSingleObject)、次に、ハンドルを閉じ、プロセスに割り当てられたメモリを解放して、クリーンアップします。

|  |
| --- |
| if( CreateProcess(  NULL,  command,  NULL,  NULL,  FALSE,  0,  NULL,  NULL,  &StartupInfo,  &ProcessInformation ) )  { wprintf( L"CreateProcess succeeded\n" ); } else { PrintLastError( L"CreateProcess()" );  } if( ProcessInformation.hProcess ) { DWORD Ret = WaitForSingleObject( ProcessInformation.hProcess, INFINITE );  if( WAIT\_OBJECT\_0 == Ret ) wprintf(L"Process finished.\n" ); else PrintLastError( L"WaitForSingleObject"); DWORD ExitCode; if( GetExitCodeProcess( ProcessInformation.hProcess, &ExitCode ) ) wprintf( L"Process exit code = 0x%X\n", ExitCode ); else PrintLastError( L"GetExitCodeProcess()" );  GetExitCodeThread( ProcessInformation.hThread, &ExitCode ); wprintf( L"Thread exit code = 0x%X\n", ExitCode );  CloseHandle( ProcessInformation.hProcess ); CloseHandle( ProcessInformation.hThread ); } else { wprintf( L"No process created\n"); } if( command )  HeapFree( GetProcessHeap(), 0, command );  return 0;} |

IsWow64

IsWow64 関数は、GetSystemWow64Directory を呼び出して、インストーラが 32 ビット Windows または 64 ビット Windows のどちらで動作しているか判定します。この関数が失敗し、GetLastError が ERROR\_CALL\_NOT\_IMPLEMENTED を返す場合、インストーラは 32 ビット Windows で動作しています。

|  |
| --- |
| typedef UINT (WINAPI\* GETSYSTEMWOW64DIRECTORY)(LPTSTR, UINT);BOOLIsWow64( void ){#ifdef \_WIN64 return FALSE;#else GETSYSTEMWOW64DIRECTORY getSystemWow64Directory; HMODULE hKernel32; TCHAR Wow64Directory[MAX\_PATH]; hKernel32 = GetModuleHandle(TEXT("kernel32.dll")); if (hKernel32 == NULL) { // // This shouldn't happen, but if we can't get  // kernel32's module handle then assume we are  //on x86. We won't ever install 32-bit drivers // on 64-bit machines, we just want to catch it  // up front to give users a better error message. // return FALSE; } getSystemWow64Directory = (GETSYSTEMWOW64DIRECTORY) GetProcAddress(hKernel32, "GetSystemWow64DirectoryW"); if (getSystemWow64Directory == NULL) { // // This most likely means we are running  // on Windows 2000, which didn't have this API  // and didn't have a 64-bit counterpart. // return FALSE; } if ((getSystemWow64Directory(Wow64Directory, sizeof(Wow64Directory)/sizeof(TCHAR)) == 0) && (GetLastError() == ERROR\_CALL\_NOT\_IMPLEMENTED)) { return FALSE; }  // // GetSystemWow64Directory succeeded  // so we are on a 64-bit OS. // return TRUE;#endif} |

PrintLastError

PrintLastError 関数は、この例のエラー処理コードの部分で、PrintLastError は、関数によって設定された最後のエラー コードのエラー文字列を出力します。

|  |
| --- |
| voidPrintLastError( const WCHAR\* apiname ){ DWORD error = GetLastError(); LPVOID lpvMessageBuffer; if( FormatMessage(FORMAT\_MESSAGE\_ALLOCATE\_BUFFER | FORMAT\_MESSAGE\_FROM\_SYSTEM, NULL, error,  MAKELANGID(LANG\_NEUTRAL, SUBLANG\_DEFAULT), //The user default language (LPTSTR)&lpvMessageBuffer, 0, NULL) ) { \_tprintf( TEXT("ERROR: %s: 0x%X: %s\n"), apiname, error, lpvMessageBuffer ); LocalFree(lpvMessageBuffer); }} |

## コマンド スクリプトから、正しいバージョンの DPInst を起動する

下記の例で示すように、コマンド スクリプトは、フル機能のインストーラの代わりに使用して、32 ビットまたは 64 ビットのドライバ パッケージをインストールできます。

|  |
| --- |
| @if /I "%PROCESSOR\_ARCHITECTURE%" == "AMD64" ( REM echo Installing driver from 64-bit installer on 64-bit OS dpinstx64.exe) else if /I "%PROCESSOR\_ARCHITECTURE%" == "X86" ( @if /I "%PROCESSOR\_ARCHITEW6432%" == "AMD64" ( REM echo Installing driver from 32-bit installer on 64-bit OS (WOW64) dpinstx64.exe //// For separate driver package, x64DIR\dpinst.exe) ) else ( REM echo Installing driver from 32-bit installer on 32-bit OS dpinstx86.exe// For separate driver package, x86DIR\dpinst.exe) )) |

# リソース

64 ビット ドライバ パッケージのインストールについて質問がある場合は、difxbeta@microsoft.com まで電子メール (英語) をお送りください。

Driver Install Frameworks Tools

Driver Install Frameworks Tools Overview

<http://www.microsoft.com/whdc/driver/install/difxtools.mspx>

Using the Driver Package Installer (DPInst)

<http://www.microsoft.com/whdc/driver/install/DPInst.mspx>

Using Driver Install Frameworks for Applications (DIFxApp)

<http://www.microsoft.com/whdc/driver/install/DIFxApp.mspx>

Installation

Device and Driver Installation on WHDC

<http://www.microsoft.com/whdc/driver/install/default.mspx>

Device Installation in the Windows DDK

<http://msdn.microsoft.com/library/default.asp?url=/library/en-us/install/hh/install/devinhdr_913b3860-1153-43cb-8eb9-07d07a3fc812.xml.asp>

Windows Installer in the Platform SDK

<http://msdn.microsoft.com/library/default.asp?url=/library/en-us/msi/setup/about_windows_installer.asp>

64-bit Platforms

64-bit Platform Considerations on WHDC

<http://www.microsoft.com/whdc/system/platform/64bit/default.mspx>

64-bit Issues in the Windows DDK

<http://msdn.microsoft.com/library/default.asp?url=/library/en-us/kmarch/hh/kmarch/Other_f910e5d8-a732-4faa-a8d2-d4de021dc78d.xml.asp>

INF Requirements for 64-bit Systems

<http://www.microsoft.com/whdc/driver/install/64INF_reqs.mspx>