Windows Vista 時代のデバイス・ドライバ開発

関連データ

第8回 デバイス・ドライバのデバッグ手法

日高 亜友，川出 智幸，相良 徹

今回からはWindowsデバイス・ドライバのデバッグ手法を取り上げる．Windows Vistaを開発のターゲットとするデバッグ環境の構築方法とツールの使い方を，WDKの提供機能に範囲を限定せず，紹介する．これらの多くは従来のシステムでも利用可能である．（筆者）

1. Windowsデバイス・ドライバのデバッグ方法いろいろ

前回までの解説は，Windows Vistaのリリース導入に伴って，Microsoft社がデバイス・ドライバ開発者向けに提供した新しい開発キットであるWDKに関連する話題が中心でした．実はWDKもその前身であるDDKも開発キットとは言え，デバイス・ドライバをデバッグするためのツールはほとんど含まれていません．

Windows XPの頃まではSoftICEという強力なデバッグ・ツールが市販されていて，筆者もドライバ開発では必ず利用していました．しかしWindows Vistaに対応しないということで現在では販売が終了し，2007年4月にはサポートも終了しました．そこで今回は，Windows Vista以降のデバイス・ドライバのデバッグ手法に注目してみます．

ソフトウェア，特にデバイス・ドライバやオペレーティング・システム（以下OS）のデバッグ手法というのは進化が遅いものです．Windows Vista以降のデバッグとはいっても，基本的な機能はWindows 2000の頃からほとんど変わらず，以前の環境でも利用できるものがほとんどです．Windowsのデバイス・ドライバを開発するときのデバッグ方法を方式別に，つまりデバッグ機能を実装する仕組み別に整理してみると，以下のようになります．

- 埋め込みデバッグ（デバッグ・メッセージ出力）
- WPP（Windows Software Trace Preprocessor）トレーシング
- ソース・コード・デバッグ
- 事後デバッグ
- そのほかのツールを使用

また，マシン構成別にデバッグ環境を考えると，次のように分けることができます．

- スタンドアロン環境
- ターゲット・ホスト接続環境
- 仮想マシン環境

埋め込みデバッグとWPPトレーシングは，どちらも主にスタンドアロン環境でデバッグ対象のプログラムを実行し，あらかじめ埋め込んである情報を基にして動作を観察する目的で利用します．このような方法をローカル・デバッグと呼びます．これらの2者では動作する仕組みとソース・コードの記述方法が異なります．

ターゲット・ホスト接続環境では2台のマシンを使用し，デバイス・ドライバ開発におけるMicrosoft社提供の主力デバッグ・ツールであるWinDBG（ウィンデバッグ）を利用します．また，このような手法をリモート・デバッグとも呼びます．

仮想マシン環境は，物理的には1台のマシンを使用しますが，2台のマシンを使用するターゲット・ホスト接続環境と実質的には同じ構成です．最近ではマシン・パワーやリソースが豊かになるとともに，仮想マシン環境を実現するソフトウェアの機能も向上し，なおかつ入手しやすくなり，環境が簡単に構築できるようになりました．また，ドライバ開発に仮想マシン環境を利用する場合はほかのメリットもあるので，これについては次回以降で取り上げます．

事後デバッグとは，何らかの原因でシステムが停止した場合の原因究明です．主にWinDBGを使用してクラッシュ・ダンプを解析する作業のことです．

そのほかのツールとしては，本連載第4回（2007年8月号，pp.173-178）で取り上げたPrefastやDTMのほか，SDV（Static Driver Verifier）のようなツールもこれに相当します．デバッグを助けるというよりも，一通りデバイス・ドライバが動作するようになった後で動作を検証するためのテスト・ツールの位置付けのものが多くあります．

KEYWORD ── デバイス・ドライバ，デバッグ，WPPトレーシング，WinDBG，DbgPrint，DebugView，DebugMon，TraceView，MsConfig

表1
そのほかのデバッグ・ツールの例

名　前	説　明	入手先・備考
ChkINF	INF ファイルのシンタックス・チェッカ	WDK: tools¥chkinf
Device Path Exerciser	通常想定しないパラメータでドライバを呼び出してテストするドライバのセキュリティ・チェッカ	WDK: tools¥dc2wmiparser
PnpDTest	PnP 環境のシミュレーション・テスト・ツール	WDK: tools¥other¥<arch>
PoolMon	メモリ獲得と解放をチェックするモニタ・プログラム	WDK: tools¥other¥<arch>
PREfast	ソース・コード中のバグや書式をチェックするソース・コード・パーサ	WDK: tools¥prefast
PwrTest	パワー・マネジメント環境テスト・ツール	WDK: tools¥acpi¥pwrtest
Static Driver Verifier	ソース・コードをインタープリタで実行することで，実際のドライバを起動せずに Driver Verifier 相当の各種チェックを実行	WDK: tools¥SDV
Driver Verifier	ドライバ・テスト時の異常環境の構成状態チェック	Windows¥System32
DTM	ロゴ取得用テスト	WDK, WLK (Windows Logo Kit)

これらの手法やツールの状況を**表1**にまとめます．このほかにも，後述する DebugView のようにサード・パーティからもデバッグに欠かせないツールが提供されています．特に Sysinternals の Web サイトからは開発を支援する有用なツールが入手可能なので，一度ご覧になることをお勧めします（下掲の**コラム1**を参照）．

2. 埋め込みデバッグ

デバッグ方法の説明で最初に挙げた「埋め込みデバッグ」について簡単に説明します．これはいわゆる，ソース・コード中に DbgPrint などによるメッセージ出力文や ASSERT 文などのデバッグ機能をあらかじめ組み込んでおき，デバッグ時にツールで表示や実行を制御する方法です．アプリケーション・プログラムの printf に相当します．

Windows カーネルがサポートしているこれらのデバッグ用機能の多くは，wdm.h の中でマクロとして定義されているので，これを利用します．**表2**に代表的なデバッグ用命令とマクロの一覧を示します．

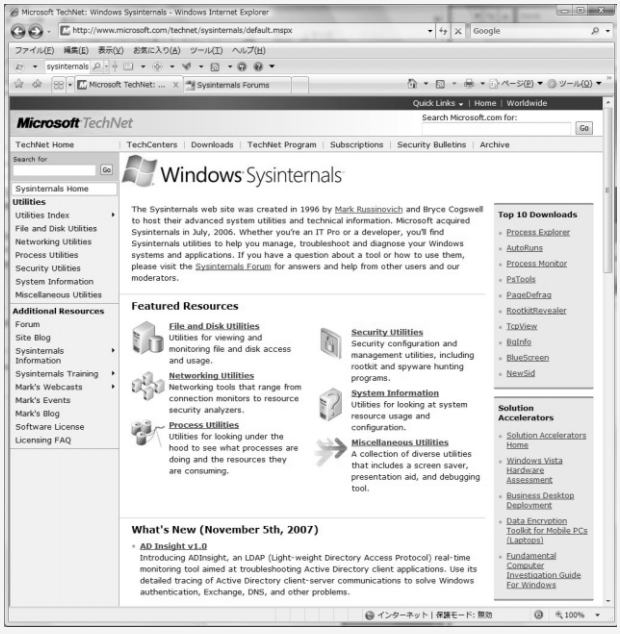
これらのデバッグ用命令を使用して，実行時に外部から

コラム1　Sysinternals

Sysinternals とは，古くは Mark Russinovich 氏と Bryce Cogswell 氏が共同で運営する Winternals Software 社の Web サイトでした（**図A**）．元々インサイド・シリーズの書籍や各種 Windows ユーティリティの公開で有名でしたが，2006 年に Microsoft 社に買収されて，彼らはまさしく“中の人”になっています．無保証ながら買収後もツール類の公開と開発は継続されていて，新しい情報を随時発信しています．

本稿の中で紹介している DebugView のほかにも，Ms Config の強力版でスタートアップを管理する Autoruns，プロセス管理ツールの Process Explore といった知る人ぞ知る定番ツール，そしてトラブル対策用のユーティリティをすべてまとめた Sysinternals Suite なども入手可能です．更新も頻繁でユーザも活発に活動しているので，定期的に訪問することをお勧めします．

- Syinternals
  http://www.microsoft.com/technet/sysinternals/
- Sysinternals フォーラム
  http://forum.sysinternals.com/



図A　Sysinternals の Web ページ