Windows CE 6.0

New Features Summary

Copyright © 2008 CodeConcept
All rights reserved

Prepared by:	**CC Team**
Last update:	**1st February 2012**
Revision:	**1.01**

Revision	*Description*	*Date*	*By*
1.01	First Draft	01 February, 2009	CC Team

2008 is the year that this unpublished work was originally created. CodeConcept sp. z o.o., hereinafter referred to as "CodeConcept", owns all rights to this work and intends to maintain this work as confidential so as to maintain this work as a trade secret. CodeConcept may also seek to maintain this work as an unpublished copyright. In the event of an inadvertent or deliberate publication, CodeConcept intends to enforce its rights to this work under copyright laws as a published work. Those having access to this work may not copy, use or disclose the information in this work unless expressly authorized by CodeConcept to do so.

Table of Contents

1. Introduction

Windows Embedded CE 6.0 (codenamed "Yamazaki") is the sixth major release of Windows Embedded Operating System targeted to enterprise specific tools such as industrial controller to consumer electronics devices like digital cameras. Windows Embedded CE 6.0 features a completely redesigned kernel, which supports over 32,000 processes, up from 32 process support of the previous versions. Each process receives 2 GB of virtual address space, up from 32 MB. Windows Embedded CE 6.0 was released on November 1, 2006, and includes partial source code. Windows Embedded CE 6.0 is also the basis of Windows Mobile 7, codenamed "Photon".

1.1. Purpose

The purpose of the paper is to present differences between Windows Embedded CE 6.0 and previous version of the Windows CE OS.

1.2. Definitions, Acronyms and Abbreviations

SDB	Standard development board
BSP	A board support package (BSP) is software that implements and supports an operating system (OS) on SDB.
KITL	Kernel Independent Transport Layer
OEM	Original Equipment Manufacturer
OAL	OEM adaptation layer

1.3. References

Explore The New Features In Windows Embedded CE 6.0
http://msdn.microsoft.com/msdnmag/issues/06/12/windowsce/

Windows CE
http://en.wikipedia.org/wiki/Windows_CE

Windows CE 6.0
http://en.wikipedia.org/wiki/Windows_CE_6.0

Windows CE6.0 Architecture
http://download.microsoft.com/documents/australia/medc2006/Windows_CE6_Architecture_Boling.ppt

2. New features and changes

To run Windows CE, a CPU must support two privilege levels. The higher privilege level is referred to as kernel mode; the lower privilege level is known as user mode. Any memory that is allocated for either code or data is assigned to one of these two modes. Putting code in user mode helps the overall environment run more robustly and more securely. Some System components (such as filesystem, gwes – graphics, windowing and events subsystem, device driver manager) have been **moved to** the **kernel space**. The system components which now run in kernel have been **converted from EXEs to DLLs**, which get loaded into kernel space. **New Device Driver Model** that supports both User Mode and Kernel Mode Drivers.

The new kernel brings a completely new memory architecture that eliminates the previous limits on available processes and address space size. **New Virtual Memory Model** - the lower 2GB is the process VM space and is private per process. The upper 2GB is the kernel VM space. The 32 process limit has been raised to **32,768 processes**. The 32 megabyte virtual memory limit has been raised to the total virtual memory (Up to 2GB of private VM is available per process).

The **Platform Builder IDE** is integrated into **Microsoft Visual Studio 2005**, allowing a single development environment for both platform and application development.

New security infrastructure. The redesigned one-tier security model is compliant with the Trustworthy Computing Security Development Lifecycle (or SDL). This ensures that only trusted applications can run on embedded device.

UDF 2.5 filesystem for storing files on optical media. Support for Microsoft's upcoming **exFAT** file system. 802.11i (WPA2) and 802.11e (QoS) wireless standards, and multiple radio support.
CE 6.0 works with x86, ARM, SH4 and MIPS based processor architectures. **New Cellcore** components to enable devices to easily make data connections and initiate voice calls through cellular networks.

3. Development Environment

By adding Platform Builder to Visual Studio 2005, Windows Embedded CE 6.0 helps deliver a superior unified embedded developer experience. Developers can now use one common familiar environment to develop both application and operating system software.

3.1. Visual Studio 2005

Platform Builder for Windows Embedded CE 6.0 has been created as a plug-in for Visual Studio 2005 and is customized for developing embedded operating systems and components.

3.2. Application Development Options

Developer can go native or managed by choosing preferred Microsoft development technology—Win32, MFC, or the .NET Compact Framework (C#, VB).

3.3. Device Emulation

The new ARM-based device emulator included with Platform Builder makes it easy to configure, build, and test operating system images.

3.4. Editors

Coding and building are faster. Windows Embedded CE 6.0 editors now incorporate Intellisense technology, syntax highlighting, and graphical bib and registry file editors.

3.5. Improved Compilers

Based on Visual Studio 2005, the latest compilers increase C++ language conformance; provide better libraries; support CRT, ATL, and MFC; and offer improved run-time security checks (/GS).

3.6. Source Code Access

Access to CE 6.0 source code helps developers debug, test, and make changes to an OS image. It also allows to modify the operating system software to create differentiated features while maintaining control over intellectual property.

Source: http://www.microsoft.com/windows/embedded/eval/wince/default.mspx.

4. Application Development with an Imported SDK

An application developer can then use the SDK to create applications that run on the Windows CE–based run-time image from which the SDK was exported.

4.1. Windows CE 5.0

Developer can import an SDK created by Platform Builder into eMbedded Visual C++ 4.0 or Visual Studio 6.0. http://msdn2.microsoft.com/en-us/library/ms859574.aspx.

4.2. Windows Embedded CE 6.0

Developer can import an SDK created by Platform Builder into the following Visual Studio 2005 SP1 and greater for the following editions:

- Standard
- Professional
- Team Edition
- Team Suite
- Team Foundation Server

Source: http://msdn2.microsoft.com/en-us/library/aa907963.aspx.

5. Application Development Tools

Developer can re-use most of existing investments in user interfaces and applications developed within previous versions of Windows Embedded CE. CE 6.0 provides continuity of key features and functionality including .NET Compact Framework 2.0 for managed application development and Win32, MFC, ATL, WTL and STL for native application developers.

Source:
http://download.microsoft.com/download/0/f/1/0f1dce86-dd37-4ebb-ac4f-be6d9ee078b1/WindowsEmbedded-CE6-Datasheet.pdf.

6. Middleware

Microsoft provides operating system building blocks for an array of devices to meet the requirements of vertical markets. This lets developer focus his/her development resources on adding product innovation, differentiation and value. For example, the Windows Media Connect and Digital Video Recording components include built-in middleware that provide the networking, device drivers and codecs needed to develop a networked media device, while helping to minimize time-tomarket and development cost.

Source:
http://download.microsoft.com/download/0/f/1/0f1dce86-dd37-4ebb-ac4f-be6d9ee078b1/WindowsEmbedded-CE6-Datasheet.pdf.

7. Microsoft Technologies

Today's embedded devices are increasingly connected, and applications in devices need to be interoperable with network infrastructure. Embedded devices with an operating system based on CE 6.0 can connect with other Microsoft technologies including:

- SQL/Everywhere
- XML Web Services
- Web Browsing
- Media Playback
- Networking

Source:
http://download.microsoft.com/download/0/f/1/0f1dce86-dd37-4ebb-ac4f-be6d9ee078b1/WindowsEmbedded-CE6-Datasheet.pdf.

8. Kernel enhancements

Fundamental enhancements to the Windows CE kernel are said to include:

- **Multitasking enhancements** - CE 6.0 supports up to 32,000 processes with 2 GB of virtual memory per process, versus CE 5.0's maximum of 32 processes having a maximum of 32 MB virtual memory each.
- **Improved performance** - OS processes are moved to kernel space; GWES, device driver manager, and file system manager are in kernel space for improved performance.
- **Addition of kernel mode drivers** - CE 5.0 only supported user mode drivers; kernel mode drivers offer performance, user mode drivers provide stability; additionally, there can be multiple instances of the user mode driver manager.
- **Strict partitioning of user mode and kernel mode**.

Basically, the Platform Builder toolset functionality from earlier versions of Windows CE had become a plug-in component for Visual Studio (VS), the popular Microsoft IDE (integrated development environment) that is used for Windows application development. This eliminates the need for developers to use two different toolsets during device development: Platform Builder for developing platform-level technologies like drivers and BSPs (board-support packages), and Visual Studio for developing application-level software.

Source: http://www.windowsfordevices.com/news/NS2632317407.html.

9. Device driver architecture changes

CE 6.0 implements both kernel mode device drivers and user mode device drivers. The use of kernel mode drivers provides enhanced security and robustness. OEMs can prevent drivers from gaining accesses to kernel resources by third party drivers, and hence can offer more security to his installation. OEMs can ship products with kernel mode drivers for all of the peripherals they supply, and for add on peripherals they can allow third parties to load kernel drivers only if they are signed by them, otherwise the drivers are restricted to run in user mode. However, user mode drivers are restricted to use Virtual Copy for only the memory space defined in the registry. In contrast, Windows CE 5.0 allowed user mode drivers to Virtual Copy any memory region other than the ones mentioned in the registry for them.

One reason that this new architecture is likely to have the biggest impact on device drivers is that existing Windows CE device drivers often use functions like MapPtrProcess to read or write directly in an application's address space. In CE 6.0, drivers that still need to access data in an application's address space need to run in kernel mode. This is accomplished by linking to the appropriate set of kernel libraries (k.coredll.dll) instead of their user-mode counterparts.

In CE 6.0, Filesys.exe, device.exe, and GWES.exe -- which were earlier part of user mode -- have been moved in to kernel mode. Calls to SetKMode and setting process permissions, so that other processes can be accessed, are not possible due to the reasons mentioned above. Consequently, drivers that use these calls have to be rewritten.

Source: http://www.windowsfordevices.com/articles/AT9457847627.html.

10. Compatibility

So what does the new architecture do for application compatibility for existing CE 5.0 based applications? - the majority of "Well Behaved" applications should just run in their unmodified binary form - there are two types of application supported on Windows CE, native, and managed - native applications are assembled, compiled, and linked into a binary application that runs on the processor it's been targeted at (ARM, MIPS, SH4, x86), a managed application is built against the .NET Compact Framework and is written either in C# or VB.

Managed applications are already abstracted from the underlying operating system by the Compact Framework and therefore should just run (unless the application is using platform invoke to call any of the deprecated (no longer supported) APIs) - native applications may not work if they are using any memory/API "tricks", there are a number of deprecated APIs, these include SetKMode, SetProcPermissions etc...

In the now defunct slotted address scheme, the currently running process is mapped into slot zero. This was done by fixing up all pointers to reference memory in the range of zero to 32MB. By calling a function like MapPtrProcess, the currently running process can take a pointer from another process and access that memory directly in the process slot (in the range 32MB-64MB for slot 1, 64MB-96MB for slot 2, and so forth). With the implementation of a private process address space, no application process can look into the address space of any other application process. Instead, developer will need to use the standard Win32 shared memory APIs to enable two processes to read and write to the same memory.

How does developer determine whether an application will run on CE 6.0 or not? The good news is that Microsoft introducing a new command line tool called CeAppCompat which can be used to examine a single application/DLL binary or a complete folder full of applications/DLLs and produces an HTML report listing which binaries might have a problem. CeAppCompat takes two command line parameters - the command line looks like: CeAppCompat -i \FolderPath -o Foo,
where Foo will be the name of the HTML report Foo.html. The CeAppCompat tool simply examines the import tables for each of the binaries and maps these back to the APIs exposed on the operating system (developer can examine the import tables on a binary using the command line tool DumpBin - like so, Dumpbin /Imports MyApp.exe) or developer can use the UI based application Depends.exe to do the same thing.

Dependency Walker is completely free to use. However, developer may not profit from the distribution of it, nor may bundle it with another product. It can be downloaded from this address: http://www.dependencywalker.com/.

Source: http://blogs.msdn.com/mikehall/archive/2006/10/12/ce-5-0-application-compatibility-on-ce-6-0.aspx.

11. Usefull links

Windows CE 6.0 Embedded requirements: http://www.microsoft.com/windows/embedded/eval/wince/techspecs.mspx.

Windows Embedded Home: http://msdn.microsoft.com/virtuallabs/winxpembed/default.aspx.

Windows Mobile Home: http://msdn2.microsoft.com/en-us/windowsmobile/default.aspx.

Real Time Systems with MS Windows CE: http://www.microsoft.com/technet/prodtechnol/wce/plan/realtime.mspx.