

# Windows Embedded Standard 7 Technical Overview

## Introduction

Windows Embedded Standard 7 is the next generation platform in the product family that includes Windows XP Embedded and Windows Embedded Standard 2009. Windows Embedded Standard 7 delivers the power, familiarity and reliability of the Windows 7 operating system in a highly customizable and componentized form, enabling OEMs in retail, hospitality and other markets to focus on their core competencies and create product differentiation. By default you get compatibility with Win32 and .NET applications and integration with Microsoft Enterprise Server and tools. Windows Embedded Standard 7 can be deployed to off-the-shelf hardware and drivers, and the platform supports x86 and x64.

## New Windows 7 Features in Windows Embedded Standard 7

Many new and interesting features in Windows 7 transfer to Windows Embedded Standard 7 and can be used in embedded scenarios for specialized devices. These include the following:

- Improved performance through shorter boot times and SuperFetch, which improves the response times of active applications.
- Better security- through BitLocker and BitLocker to Go, which provide data protection on media (including removable media such as a USB key), AppLocker which is a tool that lets you define policies that allow only certain applications to run (which is especially applicable on specialized devices), and also access controls to the device through the biometrics framework. There are several other features in the kernel that enhance security, such as Windows Service Hardening which allows for services to access only those resources that they have been granted permission to access. This makes it more difficult for services to be hijacked to gain access to system resources.
- Reduced power consumption through the TimerCoalescingAPI and the idle process.
- Rich user experiences. Features such as Windows Aero, Windows Touch and Windows Presentation Foundation enable developers to create rich user experiences, and with Windows Sensor and Location platform these applications can be aware of the content of the device itself.

## Development Model

We built the Windows Embedded Standard 7 toolkit to target the embedded device development model, from concept through software and platform decisions, embedded-specific requirements, drivers and peripherals, to user experience and servicing of the device. The aim is to make prototyping and redesign that happens throughout this life cycle easier. We also wanted to minimize the ramp-up time that you need to learn our tools. Therefore, we reused as much of the existing set of Windows tools, such as the Automated Installation Kit (AIK), as possible. Many IT professionals and system administrators are already familiar with these tools. For those scenarios that required embedded-specific functionality, we improved the Windows tools and provided additional ones.

The three main tools that are used for creating and maintaining OS images are Image Builder Wizard (IBW), Image Configuration Editor (ICE) and Deployment Image Servicing and Management (DISM).

1. IBW runs on the device interactively and is suitable for fast prototyping, or for situations where minimal customization is required. It presents a set of wizard pages that developers go through to select features and drivers. Then dependency resolution happens automatically and the OS image is built on the device.
2. ICE is suitable for more advanced embedded development scenarios that require more customization and control of settings. This tool runs on the development computer and provides more powerful functionality and better integration with source control of images.
3. DISM enables developers to install features to an image either while it is running or offline.

The experience of building an image by using IBW is a new interactive experience that was not offered in previous generations of Windows Embedded Standard. There are three main steps that are involved:

- Build bootable media containing Windows Preinstallation Environment (PE), Image Builder Wizard and a collection of componentized features and driver packages.
- Restart the device, install and customize OS image. After booting to Windows PE, IBW automatically starts and presents a wizard page that lets you select drivers, features and languages. IBW will automatically detect drivers that are needed on the hardware and will map those drivers to available driver packages, conduct dependency checking and build the image on the device. After logon custom software can be installed or settings can be changed.
- Generalize and deploy. After you create a master image that has all the settings and software that you want, you can run Sysprep to generalize the image. Then capture the image to a WIM file by using a tool named ImageX. This WIM file can be deployed to multiple computers.

Using ICE to build an image offers some similar experiences as previous generations of the platform, but also offers some new ones. ICE is installed on the development computer, and it is used in scenarios where images must be reproducible and are also more customized. The steps for building an image using ICE are as follows:

- ICE generates a file that is named Unattend.xml that contains information about the features, drivers, languages, and updates that you want to include in an image. You can also use scripts to include custom software and applications, as well as preconfigure registry key settings.
- Create bootable media in ICE, either a CD or a USB key that contains Windows PE, IBW, the distribution share where the OS and software files are stored and the unattend.xml file. Restart the target device.
- At this point IBW will take over to build the image, but in this case IBW will consume the information in the Unattend.xml file and will install the OS without user interaction (so you will not be presented with any wizard pages).

- Generalize and deploy –After a master image has been created that has all the settings and software that you want, you can run Sysprep to generalize the image. Then capture the image to a WIM file by using a tool named ImageX. This WIM file can be deployed to multiple computers.

## Image Building Blocks

New to Windows Embedded Standard 7 is the concept of Embedded Core. This is a bootable entity that contains the most common pieces of functionality that are required for any image to be bootable (such as the kernel or networking stack). It is automatically included as the base building block of all images.

In addition to Embedded Core you make selections for the other functional building blocks:

- Feature packages
- Driver packages
- Language Packs
- Embedded enabling feature packages
- Third-party software
- OS updates

All building blocks are contained within a distribution share (DS), which is similar to the component database and repository folders that previous generations of the platform have. All resources for all building blocks are present in this share. The distribution share folder can be shared so that multiple developers can use the same source to create images.

As soon as you make your selections the Image Build Engine processes the information and assembles the embedded OS image on the device.

Let us look at more in-depth information for some of these building blocks-

### *Embedded Core (eCore) –*

A collection of functionality that is needed for booting. It includes the kernel, boot-critical drivers (except for SCSI adapter drivers which have a large footprint and can be added later), WinLogon, NetLogon, File systems (NTFS, UDF and so on), command shell, servicing stack, basic networking and RPC. Embedded Core is language neutral, so it can be booted without the footprint of the back-up language (English). It is ideal as a minimal platform for application and driver testing - if an application or driver can run on eCore, then it is virtually assured that critical dependencies for the application are satisfied, and it will run on a larger image.

### *Feature Sets and Packages –*

This is a new concept in Windows Embedded Standard 7. A feature set is a collection of components for a functional area, such as Explorer Shell. A feature set is composed of one or more packages that are selectable by a user. These packages also have dependency relationships with other packages, both in the feature set itself and packages in other feature sets.

Packages are signed by Microsoft, which means the contents cannot be changed. If the contents of a package, such as files and registry keys, are changed then the package cannot be installed because the system will be unable to verify the signature and will assume the package is corrupted. Settings for any package can be changed if the package exposes configurable settings through the ICE user interface. Signing packages enables Microsoft to service the features. After installation, if a resource is removed from a package, it will be brought back onto the device as soon as the feature is serviced.

There are about 150 feature sets in Windows Embedded Standard 7. Another new feature of the product is that feature sets can be installed on the image after it has been built on the device. This makes maintaining, updating and customizing an image in the field much easier because you do not have to re-deploy the image in order to make changes.

#### *Driver Packages –*

Windows Embedded Standard 7 has all drivers that are included with Windows 7, grouped by their INF files into driver packages. In some cases, such as with printer drivers, where the size of the package is too big we have broken out packages by manufacturers. We have about 500 driver packages in the platform. Driver packages are also language neutral.

#### *Language Packages-*

Our goal is to eventually support about 40 languages and Language Interface Packs (LIPS), some at release to manufacturing (RTM) and some shortly after RTM. MUI resources have been packaged into language packs and images can be localized by adding one or more languages.

The first set of languages that will be shipped at RTM are as follows:

- Chinese – Traditional
- Chinese- Simplified
- English
- French
- German
- Italian
- Japanese
- Korean
- Russian
- Spanish

Language packs can be installed post-build, after the device is deployed in the field.

#### *Embedded Enabling Features –*

Embedded Enabling Features enable several scenarios that are important for specialized devices that may not be important for PCs. For most specialized devices, for example, it is desirable that they are locked down, in a known state and take only authorized changes. Windows Embedded

Standard 7 offers three write filters, which were also present in previous generations of the platform. Enhanced Write Filter (EWF) protects the system at the partition level by preventing writes to disk and redirecting them to an overlay cache in RAM or another partition. These writes can be discarded at reboot, restoring the system to a known state. File Based Write Filter (FBWF) protects at the file level, and redirects writes to a RAM overlay cache, but it allows for exceptions. This means that explicitly defined folders and files will persist writes to disk but all other writes will be redirected to RAM and can be discarded at reboot. Both EWF and FBWF support an API to let a developer to control flushing cached contents to disk. Registry Filter works with both of the other filters to enable the persistence of certain registry keys even when the write filters are turned on.

Windows Embedded Standard 7 includes USB Boot, which was also available in earlier versions, but also offers two new boot features-VHD Boot (shipped in RTM) and SD Boot (will be introduced after RTM).

One area that we have made quite a number of enhancements is with creating custom experiences. Generally, it is not desirable that specialized devices display dialog boxes or system messages that require user input, so we have created a more extensive infrastructure for blocking these through the Dialog Filter and Message Box Auto Reply features. We are also enabling OEMs to better develop unbranded startup screens and have custom logon desktop background. We have also allowed a custom shell to be included without requiring Explore Shell which provides a more seamless custom experience with smaller footprint.

## Image Deployment

It is possible to customize an image after it has been built on the device, such adding additional software applications, language packs or drivers, and then generalize a master image using Sysprep so it can be deployed to multiple machines. Sysprep removes system-specific data from the image. ImageX or DISM can then be used to capture the image to a .wim file for deployment. Diskpart can be used to create volumes on the device, BCDBoot to create the boot manager, and BCDEdit used to manage multiple operating systems.

There are several deployment options available with Windows Embedded Standard 7:

- Windows Deployment Services (WDS)- this is used for remote installation. Once the image has been created the image can be placed on the WDS Server and devices can connect to the server and download the image file.
- CD/DVD/USB flash – device can be booted to WinPE from the media and the .wim file installed using ImageX.
- System Center Configuration Manager- this can be used to deploy the .wim file to devices in an enterprise and also to manage updates to the device on the network.

## Managing and Servicing the Image

In Windows Embedded Standard 7 we have enabled a number of servicing scenarios. OEMs can still have full control of the servicing of the device- they can download updates to their distribution

share, rebuild the image and redeploy it or directly apply the update to the image using DISM to an image offline or while it is running. However, they may want to allow their end user to do automatic updating of the image- in this case they can include Windows Update in their image that can be configured to allow applicable embedded updates to be downloaded to the device. Note that Windows desktop updates are not applicable to embedded images- only embedded update packages can be applied.

Servicing through WSUS and System Center Configuration Manager is still enabled by Windows Embedded Standard 7. There is better integration with Active Directory, Group Policy and Microsoft management tools like System Center Operations Manager and System Center Configuration Manager, and we are looking at integrating with more third-party servicing and management tools.

### Conclusion

Windows Embedded Standard 7 is a high-performance and highly reliable platform that enables OEMs to innovate and deliver next generation devices with rich user experiences and seamless connection to the world of Windows.

Windows Embedded Standard 7 will come in two SKUs: WS7E and WS7P. For additional information, visit the Windows Embedded Standard web site:

<http://www.microsoft.com/windowseembedded/en-us/products/westandard/component-library.msp>