TECHNOLOGY IN SYSTEMS

Embedded Windows: The Next Generation

Windows 7 Goes Embedded

It has been almost a decade since a new Windows Desktop OS migrated to the embedded space, and Windows Embedded Standard 7 does not disappoint.

John R. Malin and Sean D. Liming
SJJ Embedded Micro Solutions

Windows Embedded Standard 7 (WES 7) takes the latest Windows desktop operating system, Windows 7, into the embedded space just as its predecessors, Windows XP Embedded and Windows NT Embedded did for Windows XP and Windows NT before that.

WES 7 still offers the ability to create custom Windows OS images and continues the use of embedded enabling features that go beyond the normal desktop capability. WES 7 does take a departure from the development of the predecessors, but the benefits of these tools help to support the product life cycle. Best of all WES 7 supports creating 32-bit and 64-bit Windows OS images. Applications can still be developed with the latest Visual Studio and .NET Framework SDK development tool. All the application and custom driver development work can take place on the desktop before spending time or money with the WES 7 tools.

The three driving factors for using Windows embedded have been familiarity with the Windows desktop operating systems, familiarity with the Windows application development tools that provide access to a rich graphical user interface (GUI), and the open PC architecture with readily available off-the-shelf hardware. These driving factors apply to WES 7 as well, but there are several deeper reasons why WES 7 is an improvement over its predecessors: new, from the ground-up development tools, shorter development time, easier target deployment and better servicing.

The .NET Framework Bet

Windows XP Embedded has found its way into a variety of embedded systems such as gaming machines, medical test equipment, set top boxes, in-vehicle computing, shipping equipment, test and diagnostic systems, kiosks, ATM machines, think clients, digital displays, vending machines and security systems just to name a few. The ability to quickly develop a graphical user interface (GUI) application has been the main reason for this success.

Certainly there are different GUI development packages available for different operating systems, but the ease of programming and the ability to find the resources is very important. Java's popularity sparked an idea to re-think the way Windows applications were developed. .NET Framework launched several years ago made writing Windows applications faster without having to deal with mundane tasks like memory management. Once accepted by Windows programmers, new capabilities to create rich and dynamic applications have been added with the latest .NET Framework releases. The big bet on .NET Framework is now paying dividends. Because Java has not advanced as rapidly as .NET Framework, we have had a few developers looking to use .NET Framework and Windows rather than Java and Linux. In essence, Windows is the support structure for .NET Framework application development.

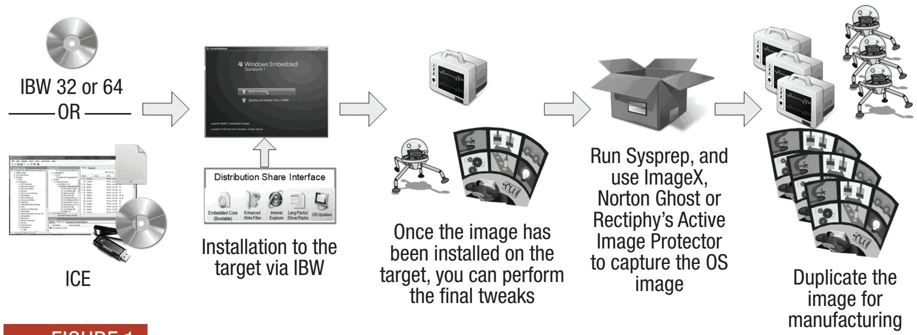

IBW 32 or 64
OR
ICE
Distribution Share Interface
Installation to the target via IBW
Once the image has been installed on the target, you can perform the final tweaks
Run Sysprep, and use ImageX, Norton Ghost or Rectiphy's Active Image Protector to capture the OS image
Duplicate the image for manufacturing

FIGURE 1

The new WES 7 development process addresses the needs for development and manufacturing. The new tools shorten the development time.

New Building Blocks

Windows XP Embedded had over 15,000 components with which to build a custom operating system. Around 1,500 were for operating system-specific capabilities, and the rest were device drivers. Some components were for only a single file like OLE32.dll. The advantage was that small image sizes could be created. The disadvantage was the difficulty of supporting all these individual build files in the field. In the end, due to component dependencies, many of these individual components got included in the image anyway.

In order to address the needs of the full product life cycle, the WES 7 building blocks and tools are derived from the Windows 7 OEM Pre-installation Kit. WES 7 is still a modularized Windows 7 operating system, but the concept of components has been replaced by larger modules known as feature packages. WES 7 has a common core that includes the kernel, HAL and critical boot drivers needed to ensure that the system will always boot, eliminating any worries about BSOD Stop 0x7B issues. Feature packages store the rest of the operating system features such as IIS, Media Player, MSMQ, Internet Explorer, etc. These packages are signed .CAB files that are stored in a distribution share. Windows XP Embedded required a database engine to manage the individual components. A distribution share frees you from the component database and the requirement for a database engine, making backup and/or sharing of the distribution share easier. Best of all, WES 7 comes with support for building both 32-bit and 64-bit Windows images, and a separate distribution share exists for each.

Bigger packages built on top of a common core produce bigger operating system builds. One could build a 40 Mbyte image using Windows XP Embedded, but WES 7 is a little bigger. The minimum 32-bit kernel is about 570 Mbyte, with typical builds running between 900 Mbyte to 2 Gbyte depending on the feature set installed. The image size for a 64-bit target is approximately double that of an equivalent 32-bit image. Size might be a drawback for some, but the real treat comes in the form of the new development process and life-cycle support that starts with the new development tools.

Development Tools

WES 7 introduces a change in development direction from previous Windows Embedded versions. The Windows XP Embedded image had to be built on a development machine and finished on the target hardware. WES 7 just installs the OS on the target like a desktop OS.

WES 7 provides two paths for installing the operating system. The first is to install directly from DVD. The WES 7 toolkit comes with 32-bit and 64-bit WES 7 installation Image Builder Wizard (IBW) DVDs. These DVDs boot to WinPE and run a wizard, which allows you to select the features and functions for the image. IBW gives you a fast method for installing operating systems, but it doesn't give you the most control over what gets installed and feature configuration (Figure 1).

The second path is the Image Configuration Editor (ICE), the IDE tool that provides a more advanced build method similar to Target Designer for Windows XP Embedded (Figure 2). ICE is not an update of Target Designer used by Windows XP Embedded, but a re-coding of the Windows System Image Manager used in the Windows Application Installer Kit. ICE works in conjunction with the Image Builder Wizard (IBW). The output of ICE is not a WES 7 OS image but an answer file that is used by IBW to select and configure the desired feature packs and install them on the target hardware. The answer file includes information about the packages you select, but also any settings and custom applications / drivers. ICE lets you create a custom IBW disk based on the answer file, which includes the selected feature packages, custom settings, any custom software, and WinPE boot files. ICE also manages the distribution share, where the OS packages are stored. Either path makes for a straightforward development process, significantly shortening the time required to develop the image.

Once the OS has been installed, you can then perform any changes and further setups to the image. Once completed, the image can be packaged up for manufacturing. Sysprep and ImageX (Figure 1) are a couple of tools that come with WES 7 to manage the manufacturing process.

The Sysprep tool makes it possible to clone a WES 7 image for volume production deployment. WES 7 requires each system to have a unique Security ID (SID), which plays an important role in Windows networking and NTFS file permissions. Sysprep rolls back a WES 7 image creating a master image for production. Each production deployment of the master image generates a new SID on first boot.

Finally, ImageX.exe allows you to capture and deploy images for production or in the field. ImageX creates WIM files that store the OS image. The WIM files can be mounted locally to apply any updates or modified for country-specific localization.

Embedded Enabling Features

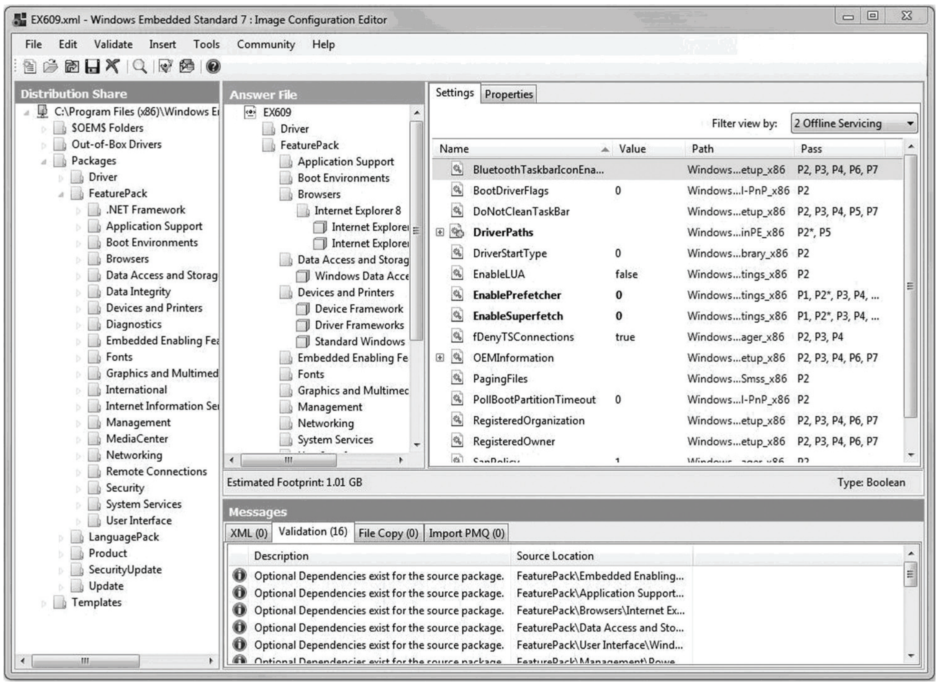
The Embedded Enabling Features (EEFs) offer unique capabilities that address the needs of an embedded system / appliance versus a desktop platform. The EEFs are what differentiate WES 7 from the Windows 7 desktop operating systems.

WES 7 provides write filters that protect the WES 7 OS image from direct access and protect the life of flash memory systems. The Enhanced Write Filter (EWF) provides write filtering for a disk partition or an entire disk volume. When activated, EWF redirects writes to the protected partition or volume to a RAM overlay. The File Based Write Filter (FBWF) provides write filtering on a file-by-file basis. As with EWF, FBWF redirects writes to the protected files to a RAM overlay. The Registry filter preserves certain registry keys when a RAM overlay is used by EWF or FBWF to protect the registry.

Time to boot is always a concern for embedded systems, and WES 7 provides a mechanism to shorten boot time: Hibernate Once, Resume Many (HORM). The system boots and saves the hyberfil.sys file, once. Then the system is subsequently booted and resumed from the saved hyberfil.sys file in subsequent boots, thus reducing the system boot time.

WES 7 provides the Bootable Windows USB Stack feature pack that supports booting from USB flash disks or USB hard drives. The target system must support USB 2.0 boot and have the USB hard drive option in the BIOS.

Embedded systems typically run 24/7 and often run unattended. Even with the best designed systems errors can occur, which can cause error message dialog boxes to appear. WES 7 provides the Message Box Default Reply EEF to handle these occurrences. Message boxes can be automatically intercepted and given a default response, i.e. 'OK' or 'Cancel', and the Message Box error events can be logged.



Filter view by: 2 Offline Servicing

Name	Value	Path	Pass
BluetoothTaskbarIconEna...		Windows...etup_x86	P2, P3, P4, P6, P7
BootDriverFlags	0	Windows...l-PnP_x86	P2
DoNotCleanTaskBar		Windows...etup_x86	P2, P3, P4, P5, P7
DriverPaths		Windows...inPE_x86	P2*, P5
DriverStartType	0	Windows...brary_x86	P2
EnableLUA	false	Windows...tings_x86	P2
EnablePrefetcher	**0**	Windows...tings_x86	P1, P2*, P3, P4, ...
EnableSuperfetch	**0**	Windows...tings_x86	P1, P2*, P3, P4, ...
fDenyTSConnections	true	Windows...ager_x86	P2, P3, P4
OEMInformation		Windows...etup_x86	P2, P3, P4, P6, P7
PagingFiles		Windows...Smss_x86	P2
PollBootPartitionTimeout	0	Windows...l-PnP_x86	P2
RegisteredOrganization		Windows...etup_x86	P2, P3, P4, P6, P7
RegisteredOwner		Windows...etup_x86	P2, P3, P4, P6, P7

Estimated Footprint: 1.01 GB

Type: Boolean

Description	Source Location
Optional Dependencies exist for the source package.	FeaturePack\Embedded Enabling...
Optional Dependencies exist for the source package.	FeaturePack\Application Support...
Optional Dependencies exist for the source package.	FeaturePack\Browsers\Internet Ex...
Optional Dependencies exist for the source package.	FeaturePack\Data Access and Sto...
Optional Dependencies exist for the source package.	FeaturePack\User Interface\Wind...

FIGURE 2

Image Configuration Editor is the graphical tool to develop custom images. ICE is a modified version of the Windows System Image Manager.

Like system error message boxes, applications can open dialog boxes that require responses. The Dialog Box Filter has been provided to block certain applications or known windows from running. A Dialog Filter Editor Tool has been provided that identifies all windows that are open on a test system, allows selection of the windows that are to be suppressed in the target embedded system, and creates a configuration file for the Dialog Box Filter. When the Dialog Box Filter and the configuration file are added to the WES 7 image, the windows and applications designated in the configuration file will be automatically blocked from running.

Servicing

Product maintenance is an important part of a product's complete life cycle. Field service has always been a challenge with Windows NT Embedded and Windows XP Embedded. With smaller components comprised of files and registries, Windows XP Embedded images were difficult to maintain. WES 7 uses packages instead of components, which makes the difference in servicing support. With WES 7, you can update a single package instead of individual files and registry keys.

WES 7 provides a new servicing tool for the development phase called Windows Embedded Developer Update (WEDU). The WEDU tool automatically checks the development environment to be sure that the latest product revisions are installed. WEDU downloads and installs updates as they become available to update the distribution shares.

For the OS images themselves, there are new tools that are built in, such as the Deployment Image Servicing and Management (DISM.EXE) and Windows Update Standalone Installer (WUSA.EXE). The powerful command line tool, DISM, installs, uninstalls, configures and updates features and packages for a WES 7 image. DISM.exe builds into every WES 7 image. DISM can be used to update images in the factory for manufacturing servicing or in the field for field support servicing. The images can either be online or offline providing multiple scenarios to update the image. WUSA is a simpler command line utility. Both tools can be scripted to apply updates to images in the field. How the update gets to the target is still up to you.

Real-Time Support

WES 7 is not a real-time ("deterministic") operating system like other solutions in the embedded market. Windows

CE (Windows Embedded Compact) is the embedded Windows offering designed to be a small real-time operating system. This doesn't mean WES 7 is limited in this area. There are some unique add-ons that help add real-time support to WES 7.

If you have no investment in a real-time kernel now, then a product like TenAsys' INtime should be considered. INtime is an add-on kernel to Windows. The INtime kernel runs as a process in Windows memory. INtime applications run at a higher priority level, while Windows itself is held at a lower priority. INtime comes with an SDK that integrates into Visual Studio. The result is a dual kernel solution that addresses both real-time and non-real-time applications. In a multicore system, you can assign each operating system to use a different core. The interesting result of the combined architectures is that you can create Windows applications that call real-time threads. The Windows graphical applications and real-time threads can pass information back and forth between each other.

If you already have a significant investment in a real-time kernel, then a product like TenAsys' embedded Virtual Machine (eVM) should be considered. TenAsys' eVM for Windows runs on multithreaded Intel processors that have VT-x support and VT-d support. eVM allows any real-time OS like iRMX, QNX, VxWorks, etc. to run alongside Windows on the same hardware platform. You can communicate between Windows and the OS running in the virtual machine via shared memory, virtual COM ports, or virtual Ethernet ports. You can keep your investment in your current real-time operating system and take advantage of Windows as a front-end user interface.

WES 7 is based on the new popular desktop operating system, Windows 7, but WES 7 goes beyond the desktop so you can integrate the rich desktop features and functionality into unique embedded systems. The development tools provided with WES 7 are new from the ground up. Because of this, WES 7 offers quicker development time and better servicing options than prior embedded Windows versions. Best of all, you can take advantage of the off-the-shelf software and hardware available for the PC. .NET Framework can be used to create feature-rich applications and add-ins can be used to support real-time requirements. With all of these advantages, WES 7 should be given serious consideration for your next embedded project. ◢

SJJ Embedded Micro Solutions
Yorba Linda, CA.
(714) 970-7523.
[www.sjjmicro.com].
[www.annabooks.com].

TenAsys
Beaverton, OR.
(503) 748-4720.
[www.tenasys.com].