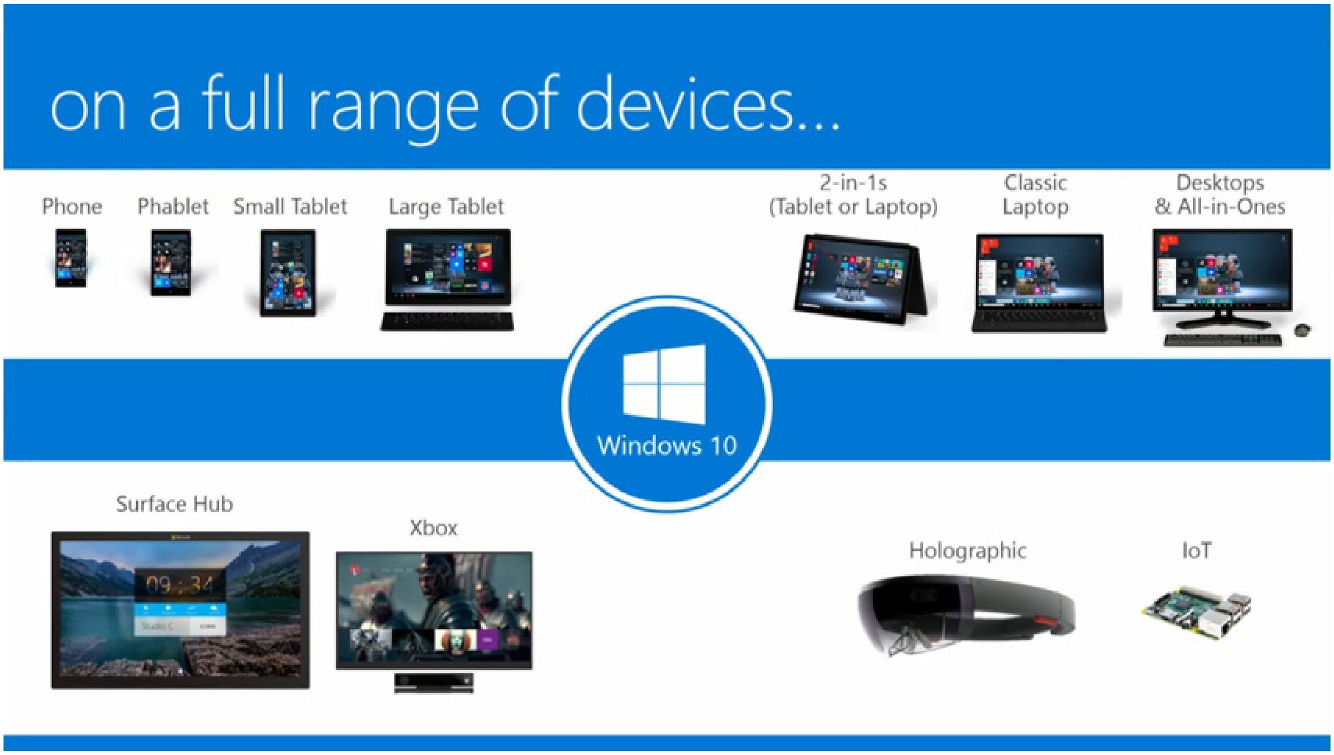
FROM WINDOWS CE TO AN IoT RTOS

Microsoft's Evolution And The Rise Of Real-Time Applications

IntervalZero

OVERVIEW

As the Internet of Things (IoT) continues to grow in popularity and need, the systems that enable it must evolve as well. Microsoft's transition to a one Windows world certainly benefits IoT systems, but is that enough? To meet the real-world requirements of this new ecosystem, organizations that build IoT apps need a real-time operating system (RTOS). A real-time deterministic solution ensures that your applications' performance is not only maintained but enhanced, without losing memory or the capability to run other applications without slowing down your system. This article will examine an existing real-time Windows IoT solution and why it is critical for creating IoT applications that outperform the competition.



on a full range of devices...

A Short History Of Windows Compact (CE)

Although Microsoft has made no direct comment regarding the future of Windows Compact (CE), it is clear that CE is giving way to Windows 10 IoT Core. The interesting thing with CE is that it had two distinct uses: products that needed real-time processing and those that did not, or at least did not need tightly bounded determinism. In fact, in the beginning, CE was a simple general-purpose operating system that was different than Windows, but provided a similar programming interface (Win32 subset) as well as a similar overall user look and feel. The original target for CE was small single-purpose devices.

However, once CE went into use, users quickly recognized that in many cases, the devices that would benefit from using CE also had a real-time, deterministic requirement; for example, an environmental control system for a building or a controller for brewing systems. Microsoft recognized this need for real-time processing in the OS and decided to work with industry experts, including VenturCom (now IntervalZero) to provide it as an inherent part of the product. So, in version 2.12, Microsoft introduced a real-time scheduler into the Windows CE kernel.

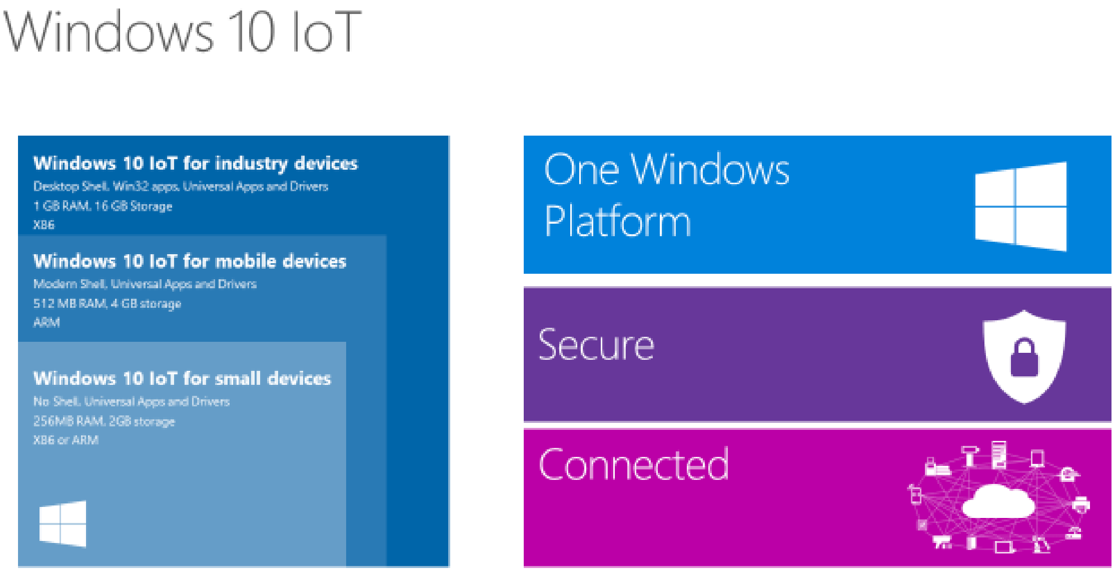
Windows 10 IoT Core must now address the same two use cases as its predecessor. It can certainly meet the non-real-time needs with flying colors, but in spite of the real-time priority class, the other is a real problem for Microsoft. How can they keep real-time users in the boat and attract new customers?

From Windows CE to Windows 10 IoT Core

Windows CE ultimately failed to capture a huge market share because of its disjointed relationship with standard Windows. While there was a fairly good amount of overlap at the top of CE and bottom of WES, there were also some pretty big gaps. For example, you may have needed some amount of determinism but really needed the full power and ease of use of standard Windows. Additionally, while Windows CE was similar, it was different enough in user interaction and development tools to be distinctly "not the same" as standard Windows.

Much of this issue stemmed from the fact that CE is a distinctly different kernel and subset of the programming interfaces. This was a big turn off to many companies. Although developers could use a good amount of code for each system, there still had to be code sections that were specific for CE platforms and standard Windows platforms, including WES systems.

In contrast, Microsoft has refactored Windows 10 in such a way that it can build the various OS products, including IoT versions, from a core set of functionality which scales up to a given device/ platform. This was a vital effort in providing a truly scalable OS platform. The result is known as OneCore.



Windows 10 IoT

Windows 10 IoT for industry devices
Desktop Shell, Win32 apps, Universal Apps and Drivers
1 GB RAM, 16 GB Storage
X86

Windows 10 IoT for mobile devices
Modern Shell, Universal Apps and Drivers
512 MB RAM, 4 GB storage
ARM

Windows 10 IoT for small devices
No Shell, Universal Apps and Drivers
256MB RAM, 2GB storage
X86 or ARM

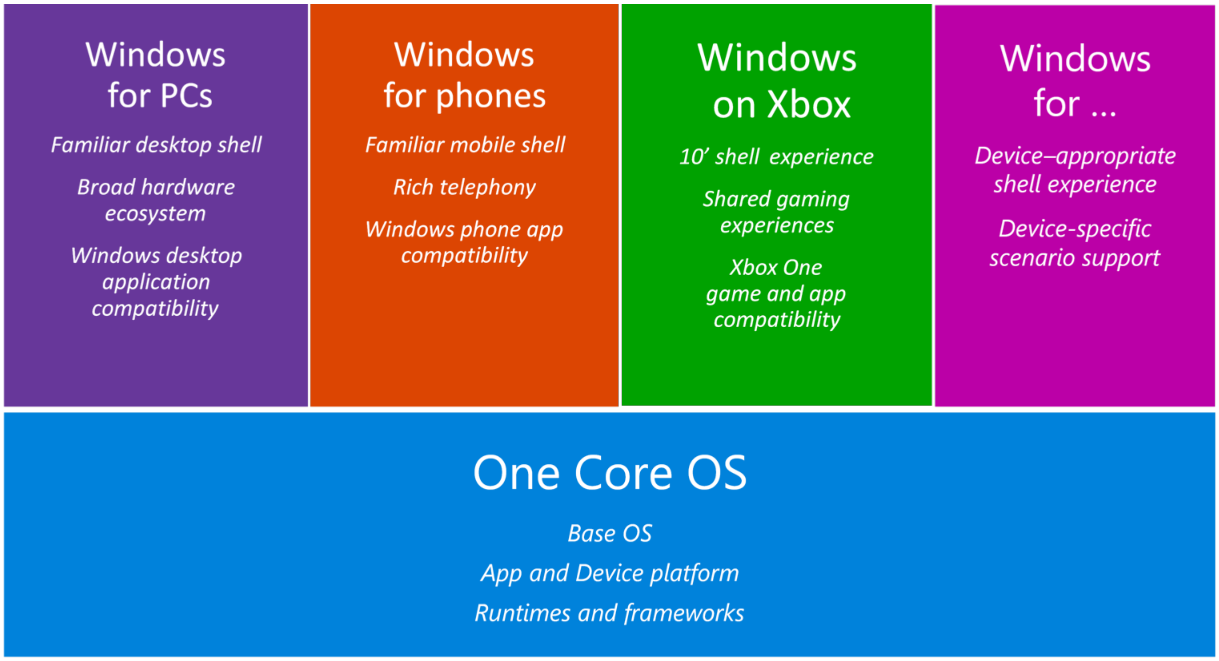
One Windows Platform

Secure

Connected

The OneCore Foundation

OneCore exists at the bottom of all Windows 10 platforms, including IoT SKUs. Out of it comes IoT Core, which can be built to run on x86, x64 and ARM devices. IoT Enterprise, like WES of the past, is an x86/x64 only OS. In fact, IoT Enterprise is exactly the same product as the Windows 10 Enterprise LTSB (long term service branch) with a different license model than the IT Enterprise SKU. This is actually great news for IoT/embedded developers, as it ensures changes to the OS, including updates, only occur at relatively long-spaced intervals, unlike the frequently updated commercial version.



Windows for PCs

Familiar desktop shell

Broad hardware ecosystem

Windows desktop application compatibility

Windows for phones

Familiar mobile shell

Rich telephony

Windows phone app compatibility

Windows on Xbox

10' shell experience

Shared gaming experiences

Xbox One game and app compatibility

Windows for ...

Device–appropriate shell experience

Device-specific scenario support

One Core OS

Base OS

App and Device platform

Runtimes and frameworks

UWP (Universal Windows Platform) and .NET Core take OneCore to the next level by providing developer services and application programming interfaces that work across platforms. This now addresses one of this bigger flaws with CE. If you design to .NET Core, your application will run on the family of IoT devices from Core to Enterprise. This empowers scalability.

At the system level, universal drivers like UWP and .NET Core are a set of programming interfaces that, if adhered to during driver development, can produce drivers from the same source code that can run on x86, x64, and ARM systems. This is huge for device vendors that provide hardware that can work on many different systems. It may not be binary compatibility, but for device drivers, write-once-target-many is a powerful tool.

Introducing real-time processing

Microsoft has addressed almost all of the previous major challenges to bridge the gaps between Windows CE and WES. The last and most important is the real-time processing component. Windows 10 IoT is great as-is for many products, such as kiosks, ATMs, vending machines, etc. But there is a huge number of devices and systems out there that require at least a minimum amount of hard real-time processing.

Beyond those more purpose-specific devices, the ability to create large-scale deterministic processing systems is now being realized. To replace rigid hardware systems with the power of Intel architecture, a solution such as the combination of Microsoft Windows 10 IoT and a real-time platform like RTX64 is both needed and available today. These systems push the envelope in terms of multi-function processing, from taking advantage of the Windows ecosystem to meeting hard real-time requirements on the same system at the same time.

RTX64
Real-time 64-bit
Windows
IntervalZero

With the combination of Intel's low-powered, small form factor ATOM processors, Microsoft's consolidation of the Windows 10 OS for IoT devices, and a real-time platform like IntervalZero's RTX64, there is a straightforward solution to the CE legacy. In fact, it has evolved tremendously. Imagine the ability to develop single real-time solutions that can work across a full range of small devices up to enterprise-level systems without any binary modifications.

The ability to build these systems from a common foundation using higher and higher functional components is extremely powerful. It enables a Windows 10 IoT platform with real-time capability, topped by functional modules such as programmable logic controllers (PLCs), vision processing engines, and motion control. These individual, "plug-and-play" modules can be easily integrated by the platform to create a fully functional, flexible, and scalable system. As a result, organizations leveraging the IoT gain a completely integrated solution wrapped in UX-driven software – in short, the extensibility and scalability you need to implement IoT solutions now and in the future.