Microsoft

winhec

Taipei 2015

Microsoft

Driver Development Overview

Devin Wong
Software Engineer

Agenda

Universal Driver Introduction

Driver Development and Debugging

IoT Core Development Package

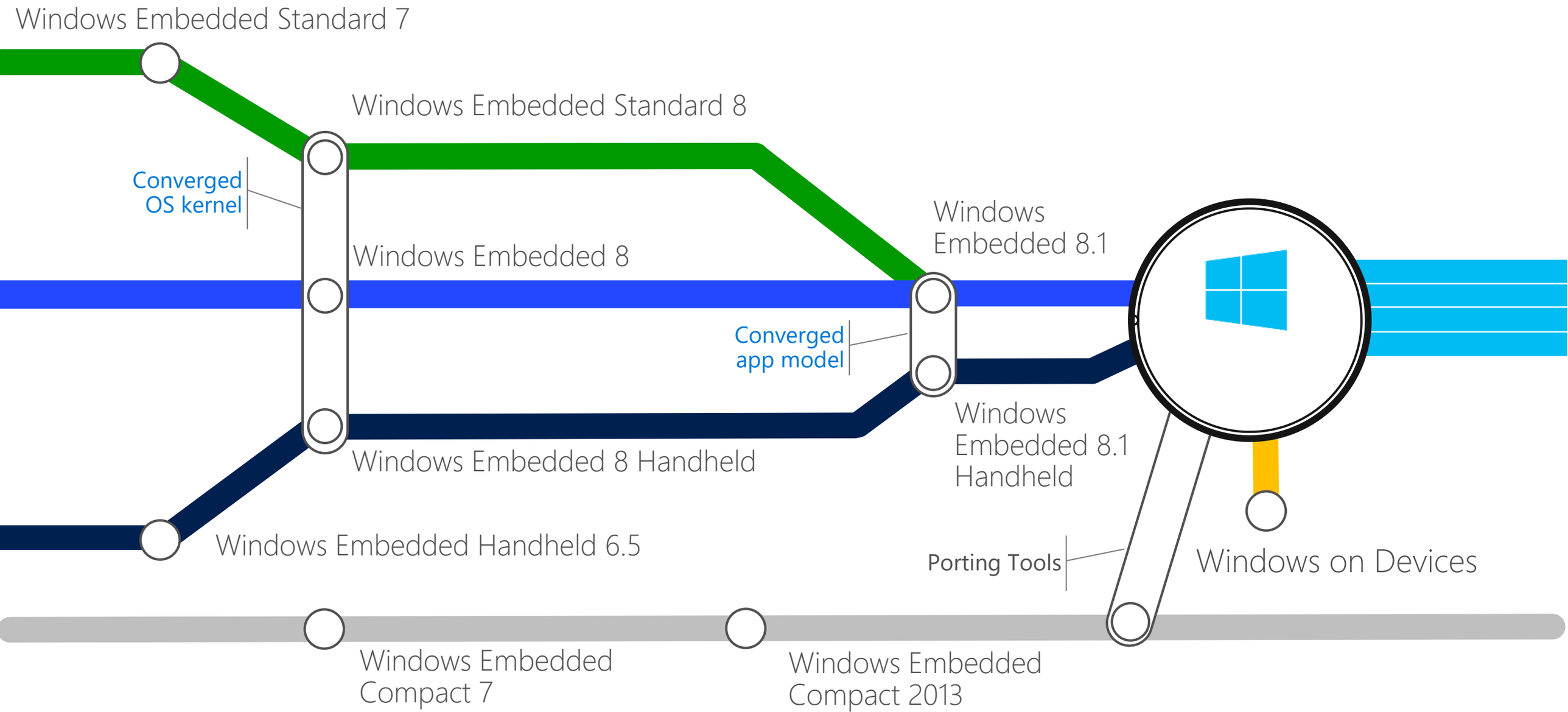
Signing, Certification and Servicing

Existing Driver Support

Other Considerations

Next Steps and Call to Action

Platform Convergence Journey



Universal Windows Platform APP

- **Converged** APIs, write **ONE** Universal App and target all Windows 10 editions
- **Scale** and get **higher ROI** by selling same App to all Windows 10 editions OEMs/ODMs
- Reuse **Windows development skills**

Universal Windows Platform
Common & Consistent APIs

Languages
- C++ /CX
- C#, VB
- JS
- Python
- Node.js

UI Frameworks
- HTML
- Xaml
- DirectX

APIs
- WinRT
- Win32
- .NET

Deployment and Execution
- APPX
- App Isolation

Tools
- Visual Studio
- PowerShell
- SSH

Windows 10 IoT “Embedded Mode”

Headless mode enabled

APIs to Access to system settings

Time zone, System Locale and Wi-Fi connection management

APIs to access buses

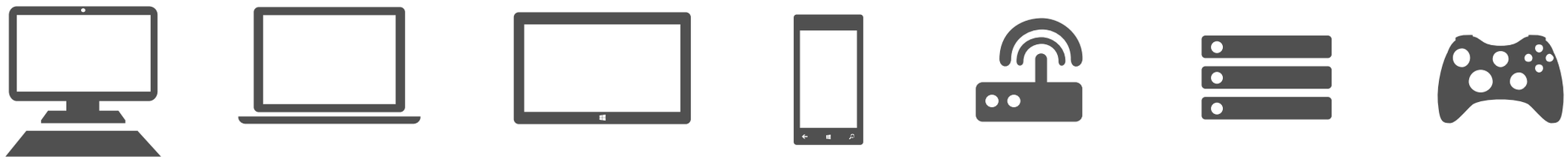
GPIO, I2C, SPI and easy access to custom hardware
http://channel9.msdn.com/Events/Build/BUILD2011/HW-747T

Background applications for long running tasks

Full control for your device, free of standard Process Lifecycle Management

Universal Windows Driver

- Re-use Knowledge, Re-use Code, Easily Scale
- We scanned over **100k drivers** to define a **universal driver API set**



Universal Windows Platform

Common & Consistent Device Driver APIs

WDF	**Location**	**NFC**
Audio	**Networking - Wired**	**Sensors**
Bluetooth	**Networking - WLAN**	**Thermal**
Buses (USB, SPB)	**Security - Biometrics**	**Touch**
HID(Retail), Buttons	**Security - Crypto**	**UEFI**
Camera	**Security - Smartcard**	**Video**
Graphics & Display	**Security - TPM**	

Jump Start Dev with Low-Cost Developer Boards

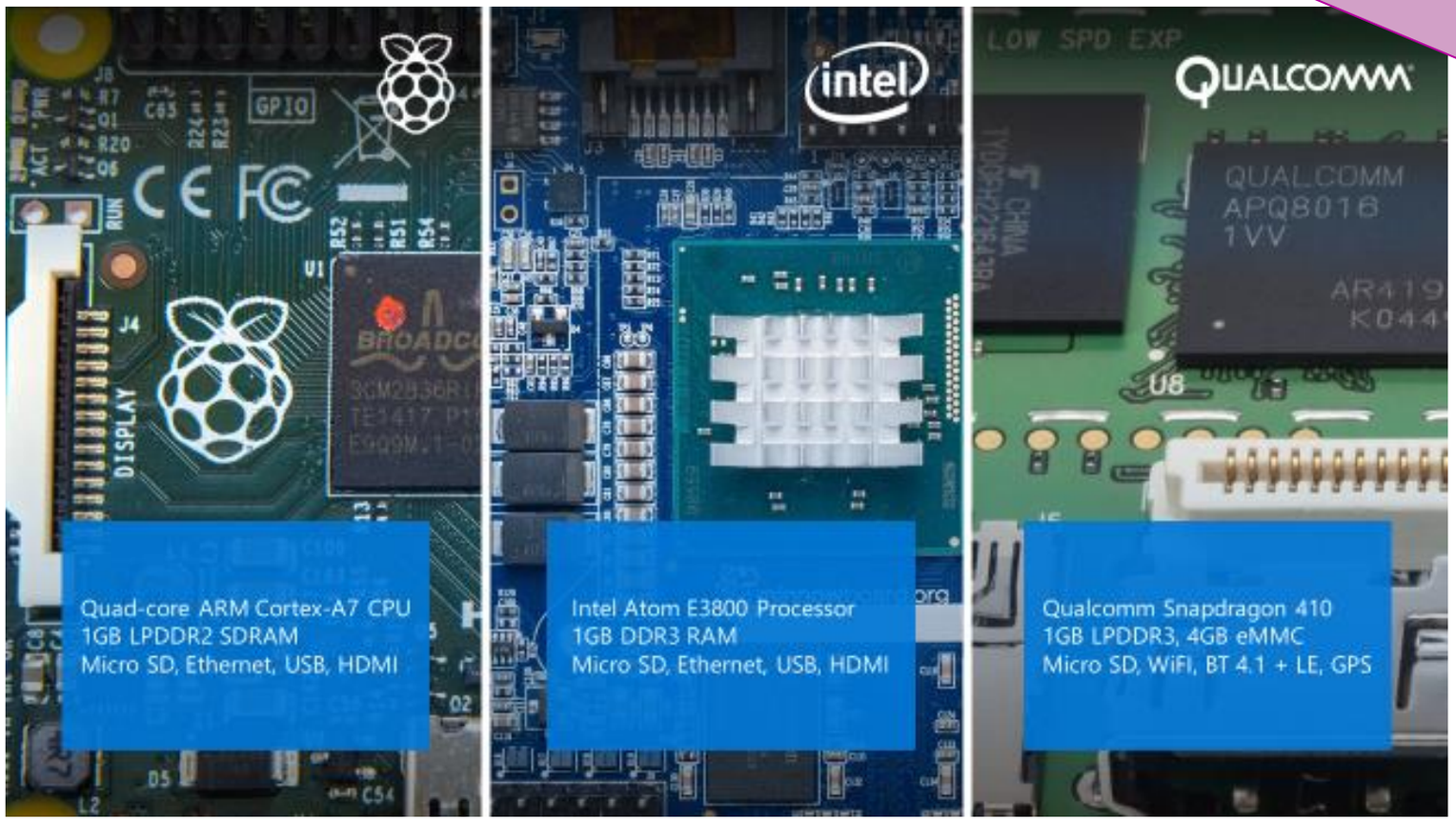
Affordable, off the shelf boards:

- www.windowsondevices.com

New in Win 10 Nov. Update

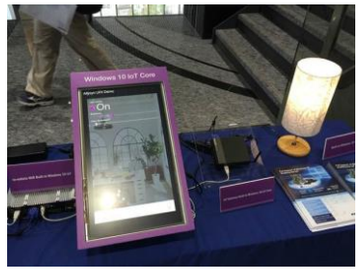
Boards Supported (blog here)

1. Raspberry Pi 2
2. Minnowboard Max
3. Qualcomm 8016-Dragonboard



Industry devices also available:

Example: Advantech Gateways



Universal Drivers

Identified set of existing driver APIs we guarantee to be supported on all UWP platforms

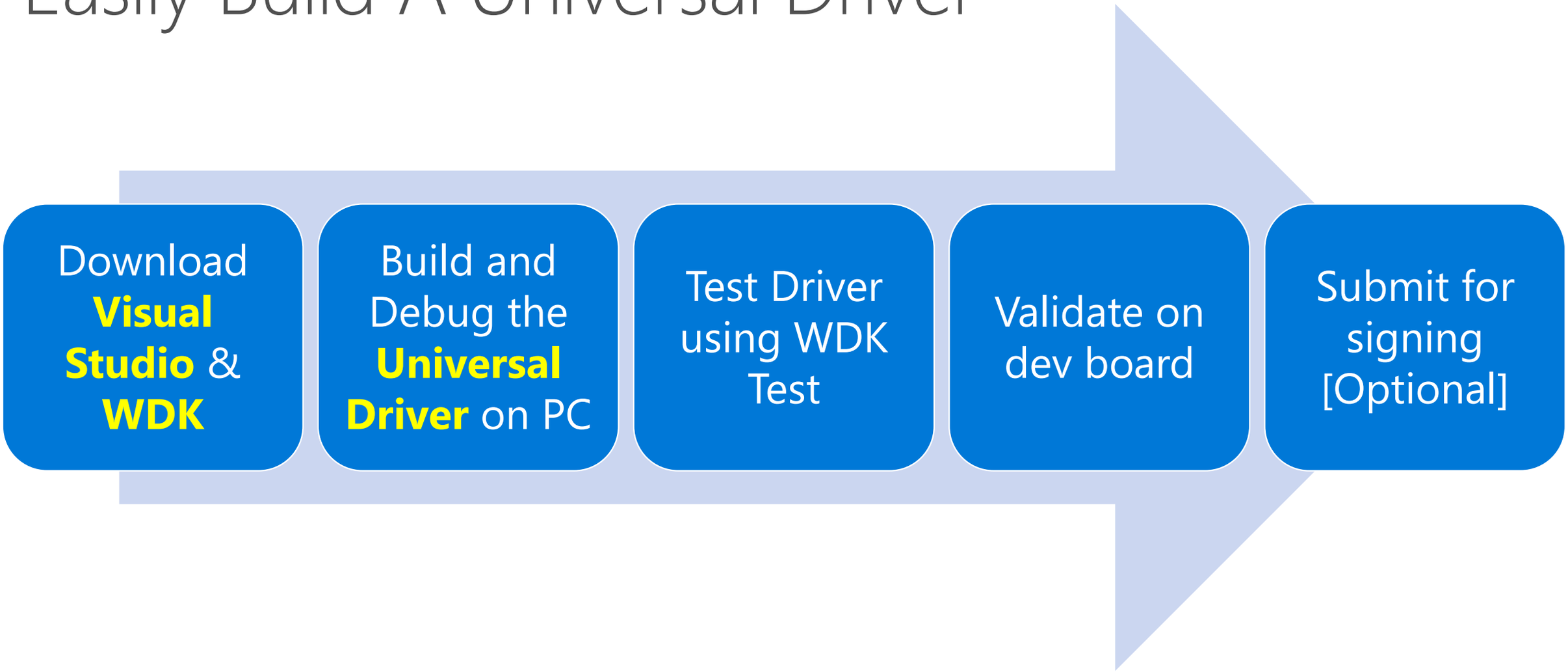
Not new APIs and not a new development model. Just the identification of common existing APIs.

Same APIs across all Windows 10 platforms

Same tools and development process end to end

Both ARM and x86

Easily Build A Universal Driver



Universal Driver samples & templates available as a starting point

Driver Development Tools

Visual Studio

Update 1 Required

Windows Driver Kit (WDK)

Driver Development

Hardware Lab Kit (HLK)

Testing

Windows Assessment and Deployment Kit (ADK)

Device Image Creation
Configuration

Driver Development Tools and Samples

https://msdn.microsoft.com/en-us/windows/hardware/dn913721

Developer Package & Mode

Why is there a developer package?

Package of development time tools and settings to make app development, device configuration, etc quick and easy when being used in a very dynamic way such as the development cycle

(Downloaded FFUs are preconfigured for VS “Developer Mode”)

What is included?

Remote Access is configured with simple preset user accounts and passwords

Command line tools to access with SSH and configure device

What services are on in the image?

PowerShell, SSH, FTP, WEBB, Remote device pinger, netsh, IoTStartup, SetBootOption, SetDisplayResolution, … and more. (List is release dependent, check the release notes for complete list.)

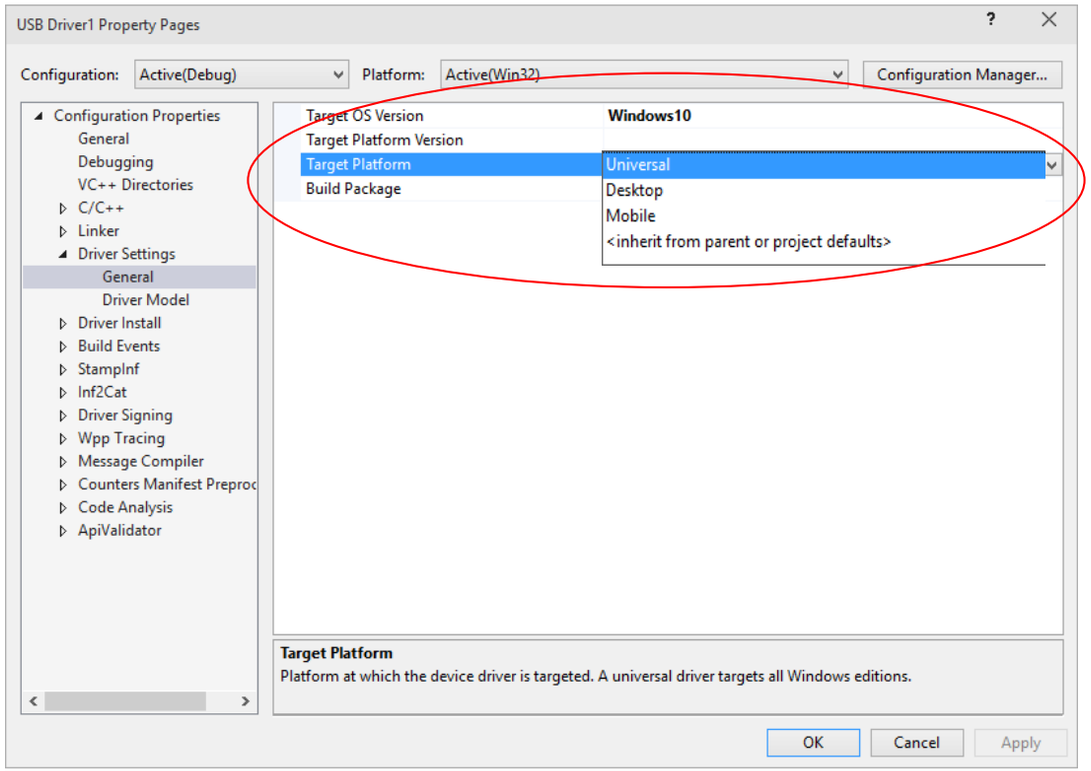
For Development ONLY!

These tools would pose a security risk if shipped without appropriate configuration and access rights.

Do not include the developer package when building and testing your near-final / final images.

Setting up VS to build Windows Universal Driver

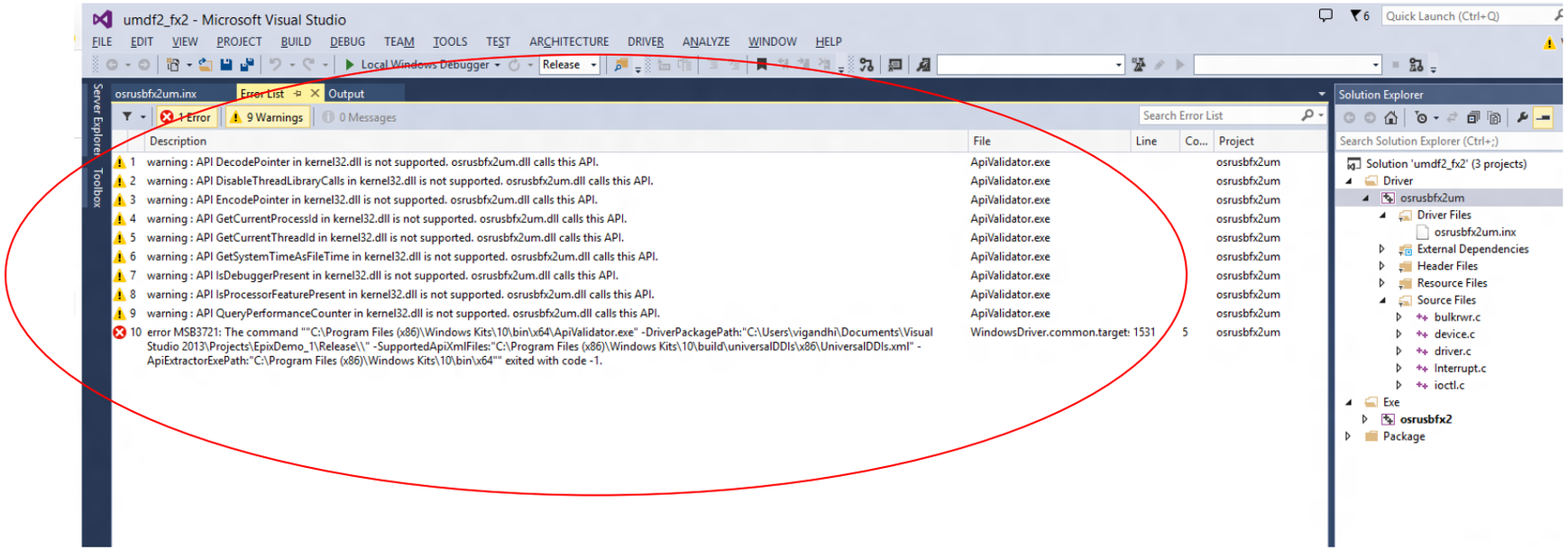
- Access Driver Setting option under Driver Properties in Visual Studio
- Set TargetPlatform == Universal



Windows Universal driver validation tool

APIValidator.exe tool

- Runs as a post build process for Windows Universal Drivers
  - Also can be run on command line
    - "C:\Program Files (x86)\Windows Kits\8.2\bin\x86\apivalidator.exe"
- Flags APIs used in the driver project that aren't part of the Universal Windows Platform



Deployment

Two Deployment Scenarios:

1. Adding a driver to an *existing image* on a device

Development F5 scenario

Provisioning via device management

2. Adding a driver to a device image at *image creation*

Included via Image Creation Designer (ICD)

Image Requirements

1. Visual Studio (Shell / App (and Driver) Development)
2. Windows Assessment and Deployment Kit (ADK)
   Contains Image Configuration Designer (ICD)
3. Latest Windows 10 IoT Core OS packages
4. Latest BSP/Drivers,
   Sample BSPs for some boards. You must contact SV for latest BSP and drivers.
5. UWP app to provide UX, if Headed device
6. Image customization XLM files

https://msdn.microsoft.com/en-us/windows/hardware/dn913721

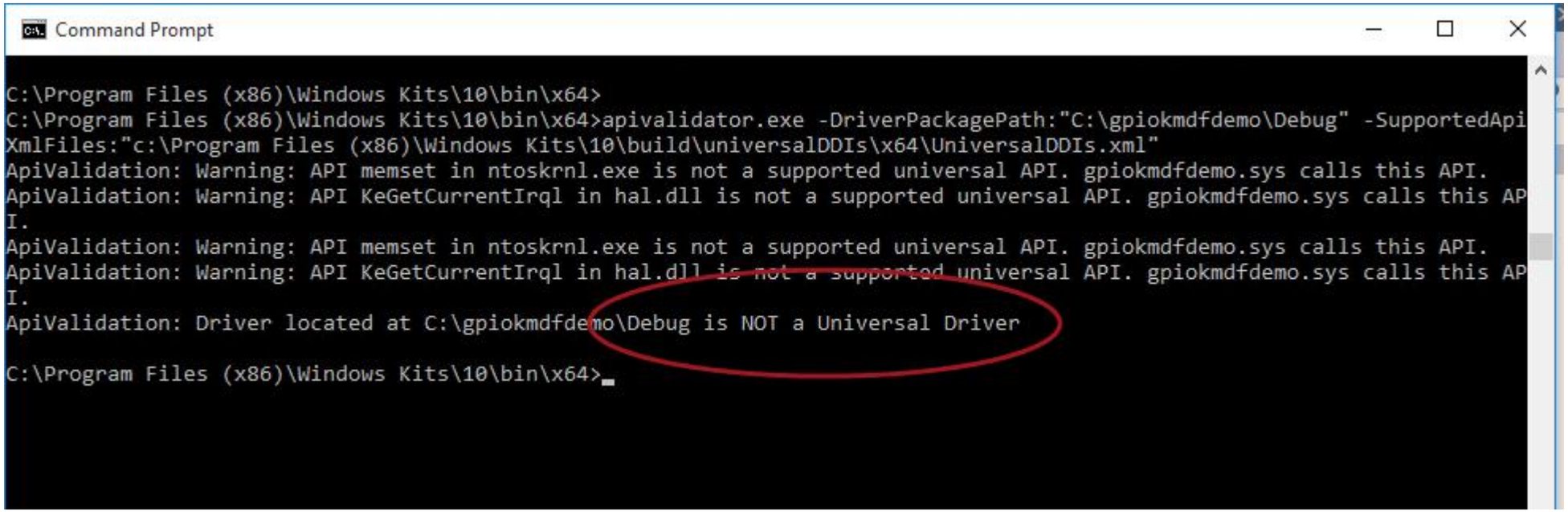
Prebuilt Developer Mode Reference Images available for Download on www.windowsondevices.com.

Demo

Validating Universal Windows Drivers with APIValidator

You can use APIValidator.exe to verify that the APIs that your driver calls are valid for a Universal Windows Driver.

APIValidator.exe is included in the WDK.



Running APIValidator against a driver that uses non-Universal APIs results in warnings indicating which APIs are not compliant with Universal APIs.

Signing

What is signing and why is it necessary?

Binary signing is to ensure the security of the platform by limiting the loading of binaries to those that are intact and trusted.

App signing

Driver signing

OS Configuration

Developer Package

Developer Mode & Cert verification options

Signing options

No Certificate Verification

Cross-Signing

Microsoft Certification Signing

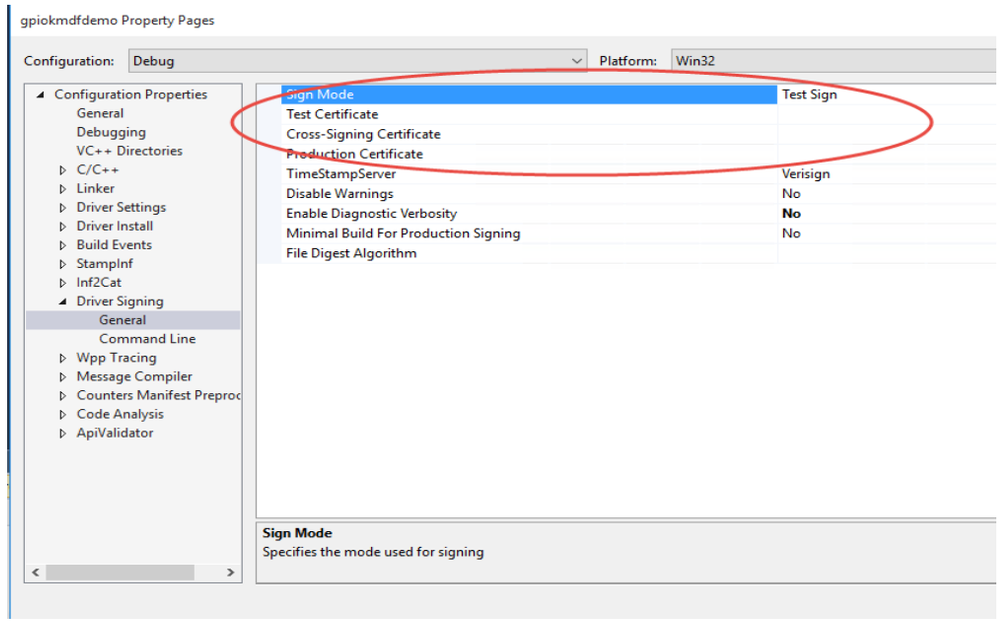
Demo
Signing a driver

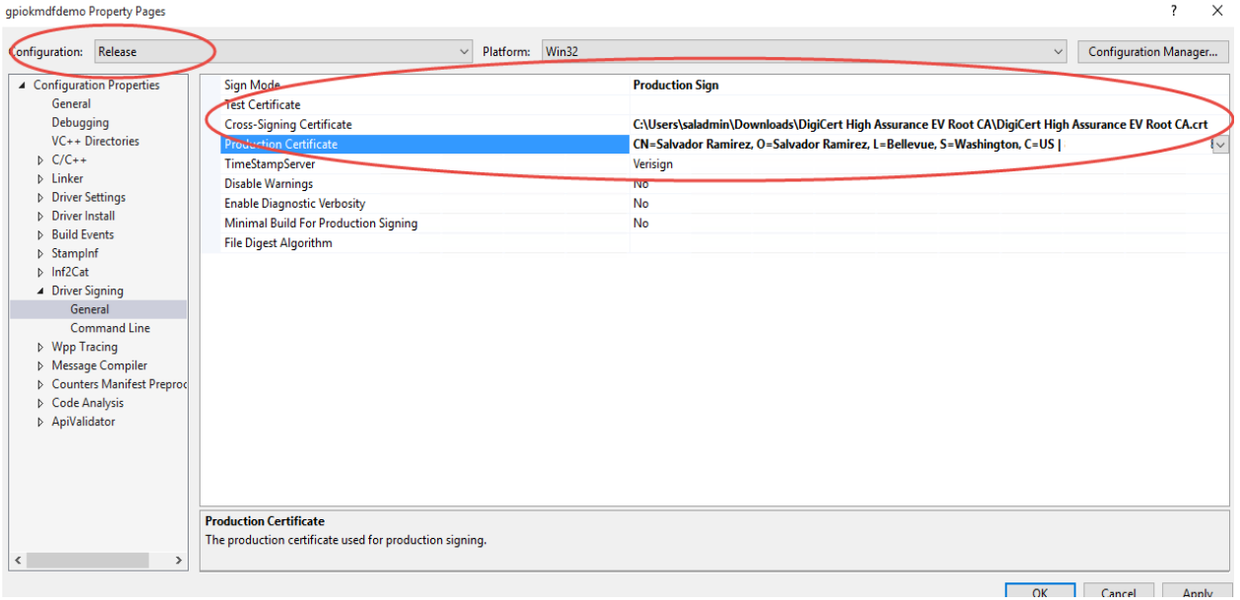
For **development purposes**, sign driver with the default test signing settings in Visual Studio. The driver will be self-signed and will be able to run on IoTCore development images.

For **public release** sign driver with (1) cross-signing certificate and (2) trusted certificate.

- Download cross-signing certificate from MSDN
- Purchase trusted code signing certificate that matches cross-signing certificate

List of Trusted cross-signing certificates: https://msdn.microsoft.com/en-us/library/windows/hardware/dn170454(v=vs.85).aspx





Extending ACPI Tables

Using ACPITABL.dat to extend support to new hardware

Create mydevice.asl

Build .dat file with asl.ext (WDK)

```
asl.exe mydevice.asl
```

Turn on test mode

```
bcdedit /testsigning on
```

Reboot

Update driver .inf to match and add to driver store

```
devcon dp_add mydevice.inf
```

Restart device

```
devcon restart acpi\VENxxxx
```
(*where VENxxxx matches the IDs in the .asl file*)

Verify device driver is running

```
devcon status acpi\VENxxxx
```

More info: https://msdn.microsoft.com/en-us/library/windows/hardware/mt186868(v=vs.85).aspx.

What about existing drivers?

How to determine if an existing driver may work?

- Analysis Tools
  - APIValidator.exe (Included with the WDK)
  - Windows 10 IoT Core API Porting Tool
    - http://ms-iot.github.io/content/en-US/win10/tools/IoTAPIPortingTool.htm

What's involved in converting to a UD?

1. Rebuild – may be a simple recompile & relink with latest libs.
2. Change Code – amount depends on amount of variation from UD API surface. Use the tools to evaluate.
3. Implement a Proxy – if source is not available and APIs can be remapped to new location.

Universal Driver Migration

If you are using	Actions to take	Why
Inbox/Class drivers	• It just works! core device types Storage, mouse, keyboard, touch, video,…	Your device automatically leverages a large ecosystem of peripherals
Kernel Mode drivers	• High backwards-compatibility for converged device areas • Make minimal changes and test	Your driver runs on more editions
User Mode drivers and services	• Note that Windows Universal Platform Win32 API surface is smaller than desktop Windows • Use replacement APIs where available • Re-design/re-implementation if APIs are not available and test • Tools available to scan and identify APIs used outside of UWP surface	Your driver runs on more editions

Porting Apps/Drivers to Windows 10 IoT Core

- UWP/UD API surface is rich but smaller compared to Windows desktop
- Use App Migration Tool to analyze compatibility of your apps

If you are using (not supported on Small Devices)	**Instead use**
App logic and code	
Win32/Native	Win32/Native in OneCoreUAP.lib (subset of Desktop API surface)
.NET libraries	.NET libraries supported in UWP App (subset of Desktop .NET APIs)
Graphic User Interface	
GDI, MFC, WinForms, WPF	XAML, DirectX, HTML

Other Considerations

Driver Certification

Not necessary for IoT Core devices

- If your driver works for your device, use it.

Advantageous because:

- Passes all necessary tests
- Microsoft Certification signing
- Can be used on multiple Windows 10 Platforms

Device UI Must be Integrated and UWP based

No UI based Installation apps

No Control Panels

No Shell Tray Apps

Summary: Building a Windows Universal Driver

1. Install **Visual Studio 2015 & WDK**
2. Pick a **Windows Universal Driver** template or sample
3. Select **driver architecture** (ARM, x86)
4. Add your code, headers, libs, etc
5. **Compile / debug /** etc.
   - **Replace non Universal APIs flagged by APIValidator.exe**
6. **Deploy** to target machine (dev board)
7. Run available **tests**

Call to Action

Download and explore Windows 10 IoT Core!

Join the pre-release program to get early access to builds

Join IoT hands-on lab tomorrow.

Give us feedback!

Questions or Feedback?

Thank you for Attending!

Microsoft

© 2015 Microsoft Corporation. All rights reserved.