Windows 10 IoT Enterprise 2016 LTSB
Overview, Setup & Configuration

Martin Grossen, Franchise Manager Microsoft Embedded/IoT Europe
martin.grossen@silica.com
Microsoft MVP
Table of contents

03 Windows 10 IoT Enterprise Channel / Licensing Description
13 Windows 10 IoT Enterprise 2016 Standard Installation
18 The Device Lockdown Features
26 DEMO: Keyboard Filter
30 DEMO: Shell Launcher
36 DEMO: Unbranded Boot
40 DEMO: Custom Logon
46 DEMO: Unified Write Filter
57 Avnet IoT Toolkits
62 Q & A
Windows 10 IoT Enterprise 2016
Channel / Licensing Description:
- CLA
- Professional / Enterprise
- CB / CBB / LTSB
- PKEA / ePKEA
- OPK
Microsoft Industrial Channel Rebranding

Windows Embedded

Windows IoT
Microsoft Industrial Channel

Windows

Windows IoT
Microsoft Windows Embedded: Embedded OEM CLA

CLA: „CUSTOMER LICENSE AGREEMENT“ → MICROSOFT EMBEDDED OEM CONTRACT
- Agreement between OEM and Microsoft to get access to Embedded/IoT licenses (COAs)
- Agreement with electronic signature
- Territorial dependence: EMEA, US/Latam, Japan, China, Hong Kong, Taiwan
- Cost free
- No quantity commitment, OEM doesn’t need to buy any licenses
- OEM has to bundle HW with OS including COA and Application → Industrial solution
- OEM is required to support their solution
- OEM has worldwide export rights
- OEM is allowed to produce recovery / update and upgrade media for end customers
- OEM can define „Outsource Manufacturers“. An OM can purchase and install OEM licenses in the name of the OEM.
- OEM can define „3rd Party Integrators“. A TPI can build the OS image in the name of the OEM
## Definition of CB / CBB / LTSB

<table>
<thead>
<tr>
<th><strong>CB: CURRENT BRANCH</strong></th>
<th><strong>CBB: CURRENT BRANCH FOR BUSINESS</strong></th>
<th><strong>LTSB: LONG TERM SERVICING BRANCH</strong></th>
</tr>
</thead>
</table>
| – Security updates and patches and new functions will be installed direct at availability and can not be switched off.  
  ➔ Updates must be installed every month, else store will not run any more. | – For Windows 10 Pro, Enterprise and Education alternative to get security updates and patches at availability. New functions with a timely delay.  
  ➔ Important updates must be installed every 4 month, can be handled by Enterprise IT. | – Updates are available but customer don’t need to install them.  
  – 10 years after release security updates and patches.  
  – No new function updates, no store, no edge, no Cortana.  
  – Microsoft will designate a long term support rollup every 2-3 years. |
# Definition of CB / CBB / LTSB

<table>
<thead>
<tr>
<th>Available Branches</th>
<th>Update Possibilities</th>
<th>Channel Availability</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Windows 10 Home</strong></td>
<td>Current Branch</td>
<td>Windows Update</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Direct OEM</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Retail/ESD</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Free upgrade</td>
</tr>
<tr>
<td><strong>Windows 10 Professional</strong></td>
<td>Current Branch</td>
<td>Windows Update</td>
</tr>
<tr>
<td></td>
<td>Current Branch for Business</td>
<td>Windows Update for Business</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Direct OEM</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Retail/ESD</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Volume Licensing</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Free upgrade</td>
</tr>
<tr>
<td><strong>Windows 10 Education</strong></td>
<td>Current Branch</td>
<td>Windows Update</td>
</tr>
<tr>
<td></td>
<td>Current Branch for Business</td>
<td>Windows Update for Business</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Direct OEM</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Retail/ESD</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Volume Licensing</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Free upgrade</td>
</tr>
<tr>
<td><strong>Windows 10 Enterprise</strong></td>
<td>Current Branch</td>
<td>Windows Update</td>
</tr>
<tr>
<td></td>
<td>Current Branch for Business</td>
<td>Windows Update for Business</td>
</tr>
<tr>
<td></td>
<td>Long Term Servicing Branch</td>
<td>Windows Update</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Direct OEM</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Retail/ESD</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Volume Licensing</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Free upgrade</td>
</tr>
<tr>
<td><strong>Windows 10 IoT Enterprise</strong></td>
<td>Long Term Servicing Branch</td>
<td>Windows Update</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Windows Update for Business</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Direct OEM</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Retail/ESD</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Volume Licensing</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Free upgrade</td>
</tr>
</tbody>
</table>
Definition of PKEA and ePKEA

**PKEA: PRODUCT KEY APPLICATION**

Every single machine has its own license number on license sticker (COA) and must be installed with this number and will be activated under this number.

**ePKEA: EMBEDDED PRODUCT KEY APPLICATION**

Embedded OEM gets an OEM license number from Microsoft per e-mail and he can use the same OEM license number for every device. The ePKEA is a multiple activation key (MAC).
Windows 10 IoT Enterprise 2016 / Redstone Licenses

2016 Version = Codename “Redstone” = “Windows 10 Anniversary Update”

Microsoft has changed the vertical license approach (POS / ThinClient / Tablet) to a CPU performance model. No difference in features, same installation media!

3 licenses are available for Windows 10 IoT Enterprise 2016 LTSB and CBB:

“HighEnd”: For high end systems based on an Intel i7 or higher CPU
→~ 150 USD for small quantities

“Value”: For mid range industrial systems with the power of an i3, i5 or Celeron processor.
→ ~ 82 USD for small quantities

“Entry”: For low end systems based on an Intel Atom CPU.
→~ 39 USD for small quantities
Windows 10 IoT Enterprise – What you get

Microsoft OEM Preinstallation Kit = OPK
Multi Language User Interface = MUI
OPK only English + MUI (24 languages at the moment)
→ Use DISM to de-install English language package if not used
Language Interface Pack = LIP (for other languages than MUI)

Attention: License key in OPK is for deployment only! It can not be activated.
Windows 10 IoT Enterprise – What you get

COA (License Sticker) needs to be affixed to the device
Windows 10 IoT Enterprise 2016
Standard Installation
WAYS TO INSTALL WINDOWS 10 IOT ENTERPRISE

- Just burn the OPK ISO file as bootable DVD and install from DVD
- Create bootable USB Stick (min. 8GB) and install from USB
  - Prepare bootable NTFS USB stick:
    ✓ Diskpart
    ✓ List Disk
    ✓ Select Disk 5
    ✓ **Clean**
    ✓ Create Partition Primary
    ✓ Active
    ✓ Format fs=ntfs quick
    ✓ Assign
- Just copy the DVD ISO file content 1:1 to the USB Stick
Windows 10 IoT Enterprise Standard Installation
Windows 10 IoT Enterprise Standard Installation
Windows 10 IoT Enterprise Standard Installation
The Device Lockdown Features

- Overview
- Activate Lockdown Features
- How to Configure the Features
Device Lockdown Features Overview

Overview of the Device Lockdown possibilities in Windows 10 IoT Enterprise 2016 LTSB and CBB.

- **Write Filters and Overlays**: Easily create read only devices. Improve system uptime.
- **USB Filter**: Only allow approved USB peripherals.
- **Dialog and Notification Filters**: Block Pop-up Dialog Boxes and system notifications.
- **Input Filters**: Block hotkeys and edge gestures to prevent system access.
- **AppLocker and Layout Control**: Control which apps are visible and can run.
- **Shell and App Launcher**: Enable single Win32 or Modern app experience on device.
# Lockdown Features Comparison

<table>
<thead>
<tr>
<th>Capability / Embedded Feature</th>
<th>WES7</th>
<th>Industry 8.1</th>
<th>Windows 10 IoT Enterprise 2016</th>
</tr>
</thead>
<tbody>
<tr>
<td>Protect devices physical storage media</td>
<td>EWF / FBWF</td>
<td>EWF / UWF</td>
<td>UWF (Unified Write Filter)</td>
</tr>
<tr>
<td>Boot fast to a known state on the device (RAM Image boot from Hibernate)</td>
<td>HORM</td>
<td>-</td>
<td>HORM</td>
</tr>
<tr>
<td>Suppress Windows UI Elements during Windows logon and shutdown</td>
<td>Embedded Logon</td>
<td>Embedded Logon</td>
<td>Embedded Logon</td>
</tr>
<tr>
<td>Block edge gestures</td>
<td>-</td>
<td>Gesture Filter</td>
<td>Group Policies</td>
</tr>
<tr>
<td>Block hotkeys and other keys / key combinations</td>
<td>Keyboard Filter</td>
<td>Keyboard Filter</td>
<td>Keyboard Filter</td>
</tr>
<tr>
<td>Launch a API32 desktop application on login</td>
<td>Shell Launcher</td>
<td>Shell Launcher</td>
<td>Shell Launcher</td>
</tr>
<tr>
<td>Launch a Universal Windows (modern style) app on login and lock system</td>
<td>-</td>
<td>Application Launcher</td>
<td>Assigned Access</td>
</tr>
<tr>
<td>Suppress system dialogs</td>
<td>Dialog Box Filter</td>
<td>Dialog Filter</td>
<td>Group Policies</td>
</tr>
<tr>
<td>Suppress toast notifications</td>
<td>-</td>
<td>Toast Filter</td>
<td>Group Policies</td>
</tr>
<tr>
<td>Control processes that can run</td>
<td>AppLocker</td>
<td>AppLocker</td>
<td>AppLocker</td>
</tr>
<tr>
<td>Restrict USB devices / peripherals on system</td>
<td>Group Policies</td>
<td>USB Filter</td>
<td>Group Policies</td>
</tr>
<tr>
<td>Suppress Windows UI elements displayed during boot</td>
<td>Embedded Boot</td>
<td>Embedded Boot</td>
<td>Unbranded Boot</td>
</tr>
<tr>
<td>Custom brand during boot</td>
<td>-</td>
<td>UEFI BIOS custom logo</td>
<td>UEFI BIOS custom logo</td>
</tr>
<tr>
<td>Suppress Windows UI elements displayed during logon / logoff</td>
<td>Embedded Logon</td>
<td>Embedded Logon</td>
<td>Custom Logon</td>
</tr>
<tr>
<td>Configure lockdown / embedded features</td>
<td>ICE</td>
<td>ELM</td>
<td>None ☺️ → Contact Silica! ☺️</td>
</tr>
</tbody>
</table>
How to activate the Device Lockdown features?

Using the Control Panel to install the Device Lockdown features.

- Just go to the **Control Panel / Programs** and click on **Turn Windows features on or off** and select the **Device Lockdown features** you need. Click **OK** to install and **Restart**
Win 10 IoT Lockdown: Unified Write Filter

**SECTOR BASED PROTECTION**
- Create read only devices
- Protect system against write operations

**REGISTRY EXCLUSION**
- Improve system up-time
- Reduce IT support & improve compliance
- Secure system

**FILE & FOLDER EXCLUSION**
- System must be designed for UWF filter
- Attention: Can increase boot-time
Win 10 IoT Lockdown: USB Group Policy

- Prevent installation of all devices
- Allow users to install only authorized devices
- Prevent installation of prohibited devices
- Control read and write permissions on removable media
- Secure system
- Implemented in local system group policy
Win 10 IoT Lockdown: Granular UX Control

Fully customize the Start Menu, Start Screen taskbar to a desired layout

Suppress toast notifications
Configuration of the Device Lockdown features

To configure the Device Lockdown features you have several possibilities depending on the feature:

- **Image Configuration Designer (ICD) with “Provisioning Package”**
- **System Image Manager (SIM)**
- **Group Policy Editor** to change policy settings
- Command line management tools e.g. “uwfmgr.exe” for the **Universal Write Filter**
- **Registry Editor** to change settings in the registry
- **PowerShell** scripts
- **Windows Management Instrumentation (WMI)**

**Note:** find documentation about customizing an Enterprise Desktop System at Microsoft: [https://msdn.microsoft.com/en-us/library/windows/hardware/mt571991(v=vs.85).aspx](https://msdn.microsoft.com/en-us/library/windows/hardware/mt571991(v=vs.85).aspx)
DEMO: Keyboard Filter
DEMO: Keyboard Filter

Check if your Keyboard Filter service is running:

⚠️ When the Keyboard Filter is not working please check if the “Microsoft Keyboard Filter” Service is configured to “Startup Type Automatic” and if the service is “Running”.

![Services screenshot](image)
DEMO: Keyboard Filter

You can also check with the Registry Editor the Keyboard Filter Settings and make changes if necessary.

HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows Embedded\KeyboardFilter
ATTENTION: There is a Breakout-KeyScancode defined per default to exit to the login screen when pressing 5 times the defined key!

Default: ScanCode: 5b -> Windows Key

Set this value to 0 if not used!

HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows Embedded\KeyboardFilter
DEMO: Shell Launcher
Shell Launcher is only working for Win32 Apps and not for the new Universal Apps. For this you can use the Option Assigned Access within the user configuration of Windows 10.

You can use PowerShell to configure Shell Launcher. You can configure different shells for different users e.g. one user is using your self developed user interface and an administrator is using the standard Explorer shell.
DEMO: Shell Launcher

The following script will set the “default shell” to “cmd.exe”, will set “iexplore.exe” for a standard user with the name “Silica” and will set “explorer.exe” for all users in the group “Administrators”.

```powershell
# Set the command prompt as the default shell, and restarts the device if the command prompt is closed.
$ShellLauncherClass.SetDefaultShell("cmd.exe", $restart_device)

# Display the default shell to verify that it was added correctly.
$DefaultShellObject = $ShellLauncherClass.GetDefaultShell()

# Set Internet Explorer in Kiosk Mode as the shell for "Silica", and restart Internet Explorer if closed.
$ShellLauncherClass.SetCustomShell($Silica_SID, "C:\Program Files\Internet Explorer\iexplore.exe -k http://www.avnet-silica.com", ($null), ($null), $restart_shell)

# Set Explorer as the shell for administrators.
$ShellLauncherClass.SetCustomShell($Admins_SID, "explorer.exe")

# Enable Shell Launcher
$ShellLauncherClass.SetEnabled($TRUE)

# View all the custom shells defined.
Get-WmiObject -namespace $NAMESPACE -computer $COMPUTER -class WESL_UserSetting | Select Sid, Shell, DefaultAction
```

# Create a handle to the class instance so we can call the static methods.
$ShellLauncherClass = [wmiclass]"\$COMPUTER\$NAMESPACE\WESL_UserSetting"

# Create a function to retrieve the SID for a user account on a machine.
function Get-UsernameSID($AccountName) {
    $NTUserObject = New-Object System.Security.Principal.NTAccount($AccountName)
    return $NTUserSID.Value
}

# Get the SID for a user account named "Silica". Rename "Silica" to an existing account on your system to test this script.
$Silica_SID = Get-UsernameSID("Silica")

# Define actions to take when the shell program exits.
$restart_shell = 0
$restart_device = 1
$shutdown_device = 2
```

# This example sets the command prompt as the default shell, and restarts the device if the command prompt is closed.
$ShellLauncherClass.SetDefaultShell("cmd.exe", $restart_device)

# Display the default shell to verify that it was added correctly.
$DefaultShellObject = $ShellLauncherClass.GetDefaultShell()

"nDefault Shell is set to " + $DefaultShellObject.Shell + " and the default action is set to " + $DefaultShellObject.defaultaction

# Set Internet Explorer in Kiosk Mode as the shell for "Silica", and restart Internet Explorer if closed.
$ShellLauncherClass.SetCustomShell($Silica_SID, "C:\Program Files\Internet Explorer\iexplore.exe -k http://www.avnet-silica.com", ($null), ($null), $restart_shell)

# Set Explorer as the shell for administrators.
$ShellLauncherClass.SetCustomShell($Admins_SID, "explorer.exe")

# Enable Shell Launcher
$ShellLauncherClass.SetEnabled($TRUE)

# View all the custom shells defined.
Get-WmiObject -namespace $NAMESPACE -computer $COMPUTER -class WESL_UserSetting | Select Sid, Shell, DefaultAction

# Enable Shell Launcher
$ShellLauncherClass.SetEnabled($TRUE)

# View all the custom shells defined.
Get-WmiObject -namespace $NAMESPACE -computer $COMPUTER -class WESL_UserSetting | Select Sid, Shell, DefaultAction

# Enable Shell Launcher
$ShellLauncherClass.SetEnabled($TRUE)

# View all the custom shells defined.
Get-WmiObject -namespace $NAMESPACE -computer $COMPUTER -class WESL_UserSetting | Select Sid, Shell, DefaultAction
```

# Enable Shell Launcher
$ShellLauncherClass.SetEnabled($TRUE)

# View all the custom shells defined.
Get-WmiObject -namespace $NAMESPACE -computer $COMPUTER -class WESL_UserSetting | Select Sid, Shell, DefaultAction

# Enable Shell Launcher
$ShellLauncherClass.SetEnabled($TRUE)

# View all the custom shells defined.
Get-WmiObject -namespace $NAMESPACE -computer $COMPUTER -class WESL_UserSetting | Select Sid, Shell, DefaultAction

# Enable Shell Launcher
$ShellLauncherClass.SetEnabled($TRUE)

# View all the custom shells defined.
Get-WmiObject -namespace $NAMESPACE -computer $COMPUTER -class WESL_UserSetting | Select Sid, Shell, DefaultAction
```
When you have saved your PowerShell script you can copy it to your target System where the Shell Launcher feature is already activated.

Open an administrative “PowerShell ISE” session and “open” your created script.
Before you can execute your PowerShell scripts you have to allow that to your system. Execute in the following command in the administrative Power Shell and then execute the script.

**Set-ExecutionPolicy Unrestricted**

```
PS C:\Windows\system32> Set-ExecutionPolicy Unrestricted
PS C:\Windows\system32> C:\Users\Avnet\Desktop\Lockdown_Features\3_Configure_Shell_Launcher_PowerShell.ps1
Default Shell is set to cmd.exe and the default action is set to 1
Current settings for custom shells:

Sid    Shell                     DefaultAction
---    ------                     ----------
S-1-5-21-485977005-4227812786-691694588-1001 C:\Program Files\Internet Explorer\iexplore.exe -k http://www.avnet-silica.com explorer.exe 0
S-1-5-32-544

Enabled is set to True
```

PS C:\Windows\system32> |
DEMO: Shell Launcher

Result when the user is logged on:

USER AVNET  

USER SILICA
DEMO: Unbranded Boot
With Unbranded Boot You can suppress Windows elements that appear when Windows starts or resumes and can suppress the crash screen when Windows encounters an error that it cannot recover from.
Using BCDEDIT to configure the Unbranded Boot feature on your System during runtime.

- To disable the F8 key during startup to prevent access to the Advanced startup options menu:
  • `bcdedit.exe -set {globalsettings} advancedoptions false`

- To disable the F10 key during startup to prevent access to the Advanced startup options menu:
  • `bcdedit.exe -set {globalsettings} optionsedit false`

- To suppress all Windows UI elements (logo, status indicator, and status message) during startup:
  • `bcdedit.exe -set {globalsettings} bootuxdisabled on`
DEMO: Unbranded Boot

Result when the your System is powering up:

BEFORE BCDEDIT COMMAND

AFTER BCDEDIT COMMAND
DEMO: Custom Logon
DEMO: Custom Logon

You can use the Custom Logon feature to suppress Windows 10 UI elements that relate to the Welcome screen and shutdown screen. For example, you can hide the logon UI for an AutoLogon user or hide buttons from the Welcome screen like the Switch user button or the power button.
DEMO: Custom Logon

- In the Registry Editor navigate to the Key:
  `HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows Embedded\EmbeddedLogon`

- Change the REG_DWORD “HideAutoLogonUI” from “0” to “1”.

- Reboot your System
 DEMO: Custom Logon

Using the Registry to hide the logon UI for an user where AutoLogon is specified.

- To specify **AutoLogon** for a specific User. Logon to your System with an administrative account and **right click** to the **Start Menu**, select **“Run”** and type in **“netplwiz”**.
Deselect “**Users must enter a user name and password to use this computer**” and click on “**Apply**”. In the Automatically sign in window type is the “**User name**” and the “**Password**” for the user you want to sign in automatically.
Now the UI during the logon phase of the user is completely hidden. You will see directly the specified shell.
DEMO: Unified Write Filter
You can use the Unified Write Filter (UWF) feature on your device to help protect your physical storage media, including most standard writable storage types that are supported by Microsoft Windows, such as physical hard disks, solid-state drives, internal USB devices, external SATA devices, and so on.

You cannot use UWF to protect external removable drives, USB devices or flash drives.

**Note:** UWF fully supports the NTFS file system; however, during device startup, NTFS file system journal files can write to a protected volume before UWF has loaded and started protecting the volume.
Universal Write Filter in the version of Windows 10 IoT Enterprise 2016

- Overlay could be configured as RAM or DISK overlay.
- Exclusions could be specified.
- HORM (Hibernate Once Resume Many) is back.
The first time you enable UWF on your device, UWF makes the following changes to your system to improve the performance of UWF:

- Paging files are disabled.
- System restore is disabled.
- SuperFetch is disabled.
- File indexing service is turned off.
- Fast boot is disabled.
- Defragmentation service is turned off.
- BCD setting `bootstatuspolicy` is set to `ignoreallfailures`.
Configuring UWF with the Command Line tool “uwfmgr.exe”.

- Open an administrative command prompt.

- Just type “uwfmgr.exe” and all options and parameters will be displayed.
DEMO: Unified Write Filter

```
Administrator: Command Prompt

(c) 2016 Microsoft Corporation. All rights reserved.

C:\Windows\system32\uwfmgr
Unified Write Filter Configuration Utility version 10.0.14393
Copyright (C) Microsoft Corporation. All rights reserved.

uwfmgr[.exe] parameter [commands] [arguments]

Description:
Configures Unified Write Filter (UWF) lockdown options.

Parameter List:
- filter  Configures and displays UWF settings such as filtering state.
- overlay Configures and displays overlay settings.
- volume  Configures and displays volume filtering settings.
- file    Configures and displays file exclusion settings.
- registry Configures and displays registry key exclusion settings, commit registry changes.
- servicing Configures and displays servicing mode settings.
- get-config Displays all configuration information for both the current and next sessions.
- help or ? Displays help for basic parameters.

Examples:
  uwfmgr.exe get-config
  uwfmgr.exe filter enable
  uwfmgr.exe volume protect c:

For more information about these UWFMGR parameters and their arguments, specify a parameter immediately before help (e.g. uwfmgr.exe filter help).
```
We will do the following settings in an example.

- We will “enable” the “UWF”
- We will “protect” the Volume “C:\”
- We will “exclude” the directory “c:\test” from the filter
- We will change the “Overlay Type” from “RAM” to “DISK”
- We will Check the “settings” and test the filter
DEMO: Unified Write Filter

See current configuration of UWF.

uwfmgr get-config
DEMO: Unified Write Filter

Enable UWF

`uwfmgr filter enable`

Protect Volume C:

`uwfmgr volume protect c:`
DEMO: Unified Write Filter

Exclude folder “c:\test”
uwfmgr file add-exclusion c:\test

Change overlay type from RAM to DISK
uwfmgr overlay set-type disk
DEMO: Unified Write Filter

See current configuration and changes after reboot.

uwfmg get-config
Avnet IoT Toolkits
Different Microsoft Windows IoT Toolkits from AVNET

IoT Configuration Manager

Windows IoT Toolkit
- DISMUI
- Recovery Creator
- Recovery Wizard
- Media Creator
- Windows Deployment Tool
- Windows Offline Configurator
- Windows Online Configurator
AVNET Windows IoT Configuration Manager

- Easy-to-use configuration center for all Lockdown Features

- Works with features from Windows 8.1 and later

- Easy activation of features
AVNET Windows IoT Toolkit Suite

- DISMUI
- Recovery Creator
- Recovery Wizard
- Media Creator
- Windows Deployment Tool
- Windows Offline Configurator
- Windows Online Configurator
AVNET Windows IoT Toolkit Suite: Benefits

- High Cost Savings For Small, Medium & Big Businesses
- One Tool For Every Step In The Development Cycle
- No Deep Technical Knowledge Needed
- Familiar User Interfaces
- Enormous Time Savings
Thank you!