Release Notes for Windows 10 IoT Core

Build Number 14295. March 2016

This document provides late-breaking or other information that supplements the documentation included with the Windows 10 IoT Core.

Thank you for downloading Windows 10 IoT Core. Windows 10 IoT Core is the version of Windows 10 intended for development of embedded or dedicated purpose devices and the choice for the Maker community. This package contains the bits and tools needed to install Windows 10 IoT Core on the MinnowBoard Max based on Intel® Atom E38xx series SoC (also referred to as MBM board), the Raspberry PI2 based on the ARM Cortex-A7 based SoC (also referred to as the RPI) and the DragonBoard 410c based on the QualComm Snapdragon™ 400 series processor (also referred to as the Dragon).

Privacy Statement

The privacy statement for this version of the Windows operating system can be viewed here: http://go.microsoft.com/fwlink/?LinkId=506737 (http://go.microsoft.com/fwlink/?LinkId=506737)

You can review linked terms by pasting the forward link into your browser window.

What’s new this build:

- Remote display experience is now included in the image

- Updates to the servicing engine to better support Windows as a Service
- Better support for removal of installed headless applications
- The IoTCoreImageHelper and IoTWatcher have been removed from the installation packages as their functionality has been included in the IoT Dashboard Application.
- Updated OS files including core OS bug fixes

Known issues in this build:

- This build has aggressive firewall settings enabled and is defaulting to blocking ports opened for listening. To unblock your app you need to connect to the device over SSH or Powershell and use these commands:
  - To unblock a specific port:

```
netsh advfirewall firewall add rule name=[Any name to identify rule] dir=in action=allow protocol=TCP localport=[Port number]
```

```
* For Node.JS the default port is 1337 and you can use the following command:
```

```
netsh advfirewall firewall add rule name=”Node.js UWP” dir=in action=allow protocol=TCP localport=1337
```

- Deploying a Python project from Visual Studio may result in a Visual Studio hang or “Unable to attach debugger” error. This is due to the firewall blocking the Python debugger. To enable Python development connect to the device over SSH or Powershell and run the following commands:

```
netsh advfirewall firewall add rule name="TCP5678-TCP-in" dir=in action=allow protocol=TCP localport=5678
netsh advfirewall firewall add rule name="TCP5678-TCP-out" dir=out action=allow protocol=TCP localport=5678
```

- The AllJoyn DSB Visual Studio template may not deploy to IoT Core from the latest version of Visual Studio.
- The UART/Serial (miniUART) driver for the Raspberry PI 3 onboard serial port is broken in this build.

Release Notes

Minnowboard Max Boot and Firmware Update

The MinnowBoard Max will not boot unless the firmware is version .082 or later. The minimum recommended version of the firmware is “MinnowBoard MAX 0.83 32-Bit”. Firmware updates can be downloaded from http://go.microsoft.com/fwlink/?LinkId=708613 (http://go.microsoft.com/fwlink/?LinkId=708613).

Default Administrator User Name and Password

The default administrator user name and password are hard coded in the Windows 10 IoT Core image. This is a security risk for the device, and it should not be exposed to an open internet connection until the password has been changed.

Dragonboard 410c Shutdown

On the DragonBoard, a shutdown command will not power off the board. The system will restart. Please power off the board by disconnecting the power.

Minnow Board Peripheral Support

The Windows 10 IoT Core image included in this drop supports the peripherals that are exposed on the MinnowBoard MAX board. Subsequently, Intel® will provide support of the full feature set of the Baytrail processors including the Intel Celeron™ Processors J1900/N2930/N2807 and Intel Atom™ Processors E38XX.

Raspberry Pi Video Performance

Video playback performance on the Raspberry Pi platform has not been optimized. Animated user elements including XAML-based dropdown menus may exhibit less than optimal performance.

Raspberry Pi Camera Support

With this release of Windows 10 IoT Core for the Raspberry Pi 2, support for camera peripheral devices is limited. The PiCam device directly connected to the onboard camera bus is not currently supported, as it requires GPU services that are not currently available on the Raspberry Pi because the DirectX driver has not been implemented. Modern USB webcams produce data streams that are very demanding on the USB Host controller. Even when used with low resolution settings webcams will require additional USB fine tuning and specialized control logic. We are planning to support a number of USB cameras in the near future and will publish specific information on supported devices as soon as possible.

Volume Controls

Hardware volume controls for USB microphones and speakers which depend on Windows system to change system volume are currently not supported on Windows 10 IoT Core.

USB Keyboards

Some USB keyboards and mice may not work on IoT Core. Use a different keyboard or mouse. A list of validated peripheral devices can be found on the documentation at http://go.microsoft.com/fwlink/?LinkId=619428 (http://go.microsoft.com/fwlink/?LinkId=619428).

Screen Orientation

Setting the orientation to “Portrait” may not be honored in a Universal App

Referencing Adapters with AllJoyn Templates

Attempting to add references to AllJoyn adapter projects may result in errors when using specific SDK versions. To resolve these errors, change Visual Studio’s target platform to match the current SDK version, then reload the project.

Serial Port Usage and Access on RPi2

Raspberry Pi 2 supports the serial transport for communication through the PL011 UART. This is set by default in kernel debugging scenarios. An application or device driver can use the PL011 UART to send and receive data with the PL011 device driver turning off the debugger using the following command:

```
bcedit /set debug off
```

WiFi Direct limitations on IoTCore

1. The IoTCore device has to be the connecting device – it will not work as the advertising device with another device initiating the connection.

2. Advanced pairing must be used. The sample app demonstrates how to use the advanced pairing API’s to pair the devices prior to connecting.

3. Not all wireless adapters support WiFi direct. We have tested and validated that the “Realtek RTL8188EU Wireless Lan 802.11n USB 2.0 Network adapter” works, but other adapters may not be supported.

Non-default drive mode (3890679)

On Raspberry Pi and Dragonboard, switching from a non-default drive mode to a different non-default drive mode may produce a glitch on the GPIO pin. WORKAROUND: Set drive mode once at the beginning of the application.

Application already running (1244550)

The Default startup app may conflict with itself when it is also deployed from Visual Studio. WORKAROUND: Change the default startup app to an application other than that you wish to deploy.

BackgroundMediaPlayer.MessageReceivedFromForeground may crash (2199869)

The following line of code may crash: “BackgroundMediaPlayer.MessageReceivedFromForeground += OnMessageReceivedFromForeground;”. To prevent the crash, add this code so that it is executed first “var player = BackgroundMediaPlayer.Current;”

Data breakpoints have been disabled on the Raspberry Pi2 (4266252).

No workaround at this time

Azure Active Directory Authentication Support (4266261)

The Azure Active Directory Authentication Library does not work on Windows 10 IoT Core.

Dragon Board and windbg (4710796)

The GPIO/I2C/SPI/UART drivers will be disabled when connecting to the DragonBoard with windbg.

Dragon Board headset & microphone jack (4791855)

The Dragonboard BSP has drivers for the headset jack and microphone jack, but it doesn’t have either of these jacks on board.

Dragonboard SPI runs at 4.8Mhz (5055938)

The SPI on the Dragonboard will ignore the requested speed and always run at 4.8 Mhz.

Shell Management of Application Crashes

IoT Core’s shell infrastructure monitors APPX-type applications running on the device for crashes, and restarts those applications when crashes occur. If the restarted applications continue to crash, the shell will employ a __failfast – a system critical process that causes a bugcheck and reboot in an attempt to recover. Comparable logic and handling is used to background tasks and foreground applications in a headed configuration. Crash handing and retry logic is captured below:

```
  Software\Microsoft\Windows NT\CurrentVersion\Winlogon\IoTShellExtension\CBTConfig  (or ForegroundAppConfig for headed)
    Qword:"FailureResetIntervalMs" – length of time app has to run successfully to reset failures seen to 0. – default is 0x00000000000493E0 == 5 minutes
    Qword:"BaseRetryDelayMs"  -- wait time coefficient.  Default is 0xa.
    Dword:"MaxFailureCount". Default is 10
    DWord:"FallbackExponentNumerator", default is 31.
    Dword:"FallbackExponentDenominator", default is 20


Fallback_exponent = FallbackExponentNumerator / FallbackExponentDenominator; // default is 1.55

When app crash is detected:
    if time_since_last_crash > failureresetinterval then crashes_seen = 1
    else ++crashes_seen;

    if crashes_seen > MaxFailureCount then __failfast;
    else
      delay = (dword) ((float)BaseRetryDelayMs * (crashes_seen ** Fallback_exponent))
      wait for delay and relaunch app
```

Questions/Suggestions

As always, for questions or feedback, contact us (/content/en-US/Community.htm#contact)

Edit this page on GitHub (https://github.com/ms-iot/content/blob/develop/en-

US/win10/ReleaseNotesInsiderPreview.md)

Downloads and tools

Windows 10 dev tools (http://go.microsoft.com/fwlink/p/?linkid=280676)

Visual Studio (http://go.microsoft.com/fwlink/p/?LinkId=618475)

Windows SDK (http://go.microsoft.com/fwlink/p/?linkid=616887)

Windows Store badges (http://go.microsoft.com/fwlink/p/?LinkId=534236)

Essentials

API reference (Windows apps) (http://go.microsoft.com/fwlink/p/?LinkId=534192)

API reference (desktop apps) (http://go.microsoft.com/fwlink/p/?linkid=285627)

Code samples (http://go.microsoft.com/fwlink/?LinkID=280677)

How-to guides (Windows apps) (https://msdn.microsoft.com/library/windows/apps/xaml/mt244352.aspx)

Learning resources

Microsoft Virtual Academy (http://www.microsoftvirtualacademy.com/)

Channel 9 (http://channel9.msdn.com/)

Video gallery (http://go.microsoft.com/fwlink/p/?LinkId=534190)

Windows 10 by 10 blog series (http://blogs.windows.com/buildingapps/2015/08/05/build-better-apps-windows-10-by-10-development-series/)

Programs

Get a dev account (http://go.microsoft.com/fwlink/p/?LinkId=534191)

App Developer Agreement (http://go.microsoft.com/fwlink/p/?LinkID=246847)

Windows Insider Program (http://insider.windows.com/)

Microsoft Affliate Program (https://go.microsoft.com/fwlink/p/?LinkId=529768)

English (United States) (https://dev.windows.com/locale)

Privacy and cookies (//go.microsoft.com/fwlink/p/?LinkId=521839) Terms of use (//go.microsoft.com/fwlink/p/?linkid=248155)

Trademarks (//go.microsoft.com/fwlink/p/?linkid=248154)