



Intel Unite® Plugin Reference Design for Power Control



Legal Disclaimers & Copyrights

All information provided here is subject to change without notice. Contact your Intel representative to obtain the latest Intel product specifications and roadmaps.

Intel technologies' features and benefits depend on system configuration and may require enabled hardware, software or service activation. Performance varies depending on system configuration. No computer system can be absolutely secure. Check with your system manufacturer or retailer or learn more at intel.com.

You may not use or facilitate the use of this document in connection with any infringement or other legal analysis concerning Intel products described herein. You agree to grant Intel a non-exclusive, royalty-free license to any patent claim thereafter drafted which includes subject matter disclosed herein.

No license (express or implied, by estoppel or otherwise) to any intellectual property rights is granted by this document.

The products described may contain design defects or errors known as errata which may cause the product to deviate from published specifications. Current characterized errata are available on request.

Intel disclaims all express and implied warranties, including without limitation, the implied warranties of merchantability, fitness for a particular purpose, and non-infringement, as well as any warranty arising from course of performance, course of dealing, or usage in trade.

Intel, the Intel logo, Intel Unite, Core, and vPro are trademarks of Intel Corporation in the United States and/or other countries.

*Other names and brands may be claimed as the property of others

© 2015 Intel Corporation. All rights reserved.



Table of Contents

1	Power Control Reference Design	4
1.1	Description	4
1.2	Interfaces.....	4
1.3	Project Structure and Setup –	5



1 Power Control Reference Design

1.1 Description

This plugin reference design provides power control orchestration for hub Intel Unite® app. Other plugins can be written to interface with this plugin, which will then allow the screen to wake automatically and stay awake when a room is marked as occupied. Communication with this plugin is achieved over a RESTful API. This allows other software, not just plugins, to interface with the power control plugin.

The plugin takes these values and calculates the percentage of “reporters” that are indicating room occupancy to determine whether or not to report the room as occupied.

1.2 Interfaces

This plugin is designed with a REST interface so that you can add multiple points of occupancy for greater accuracy. For example, you could add a motion detection and sound detection, and require that both or either must be true before occupancy is set.

Constants

PERCENT_TRUE_REQUIRED_FOR_OCCUPANCY (double) - Represents the % of “reporters” that must report as occupied before the plugin will also report occupied and alter the date of the display.

TIME_FOR_DISPLAYS_TO_STAY_ON_BEFORE_SLEEPING_WITHOUT_INTERACTION (Integer)
- The amount of time (in seconds) the system should be idle, including the room being reported empty by various plugins and pieces of software, before turning the displays off to save power.

127.0.0.1:8000/Unite/RoomOccupied

Returns True/False based upon plugin's analysis of data received from other reporters.

127.0.0.1:8000/Unite/OccupancyDetected/{guid}

Accepts a guid so the value provided can be adjusted at a later time. This call marks the guid provided as showing the room as occupied, based upon whatever criteria you determine makes the room occupied.

Note: If the guid contains the string **8675309**, the reporter is believed to be 100% accurate and automatically causes the room to report occupancy. For instance, someone connected to the HUB or someone presenting could be considered 100% accurate and should use these key string value in its guid.



127.0.0.1:8000/Unite/OccupancyNotDetected/{guid}

Accepts a guid so the value provided can be adjusted at a later time. This call marks the guid provided as showing the room as unoccupied, based upon whatever criteria you determine makes the room unoccupied.

1.3 Project Structure and Setup –

There are three files in the project: *IUnitePowerPluginServices.cs*, *PowerControlPlugin.cs*, and *Routing.cs*.

IUnitePowerPluginServices.cs

This file defines the RESTful API service provided by the plugin, specifying the details surrounding calls: What URI, what format the body is in, what format the response is in, and what values are taken in by the functions. This service must be implemented by *PowerControlPlugin.cs*.

Routing.cs

This simple static class contains constants for the routing strings to ensure consistency throughout the project where the need to specify the URI exists.

PowerControlPlugin.cs

This file has two internal classes related to monitor control. This file also implements the service specified by *IUnitePowerPluginServices.cs* and the interface specified by *Intel.CFC.Plugin.CFCPlugin*, the interface needed to be a Unite plugin.

In order to create a RESTful service, there are a few references that need to be added to the project: *System.Net.Http*, *System.ServiceModel*, *System.ServiceModel.Routing*, *System.ServiceModel.Web*, and *System.Xml*. With these references added, you should be able to add the code required for a RESTful API.