Simple Device Discovery Protocol

Implementation

Process

Contact Us

Control4 Corporation
11734 S. Election Road
Salt Lake City, UT 84020 USA
http://www.control4.com

1.0 Understanding SDDP

SDDP is a device discovery protocol developed by Control4. It leverages other protocols and can work on almost any physical layer. Currently, SDDP is used for discovery on IP devices. SDDP assumes that there is just one “controller” device that is running SDDP server in a system and one or more “controlled” devices with SDDP client implemented.

On IP networks, SDDP defines a discovery protocol similar to SSDP, which is the discovery protocol used by Universal Plug and Play (UPnP). In the protocol, devices/services make announcements on a multicast address when they become available and immediately before they go away.

An SDDP client implementation is required on any IP device that you want to be automatically discoverable and identified, and to have the correct driver downloaded and installed in Control4 installations. SDDP also allows for your IP device to use a UUID string rather than a static IP address when identified in a Control4 system. This means that it will work well in a DHCP environment and be much less problematic for Control4 dealers/installers to work with and support.

The basic benefits of auto device discovery, UUID device connectivity, auto driver download, and unique device identification are extremely valuable to the more than 3000 active Control4 dealers installing home automation solutions in more than 75 countries worldwide and will help differentiate your IP device in the dealer’s selection and purchase decisions.

2.0 Pre-Implementation Considerations

As a developer implementing SDDP, it is very useful for you to have access to several Control4 services. Registering for these services will allow for your SDDP implementation to leverage all of the hardware/ software functionality and support provided by the SDK.

It is also recommended that you have an actual Control4 controller such as the C4-HC250-BL. Testing your SDDP development work against a Control4 controller is the best approach in validating your implementation. The controller you use should be updated to a Director version of 2.2 or greater. This will ensure that the SDDP server functionality on the controller is running.

Finally, for SDDP to function properly, you must have a functional Control4 device driver posted in the Control4 driver database cloud. You can develop a driver for your device using the DriverWorks SDK and development tools which are not discussed in the scope of implementation of SDDP or this SDK, but are available from your Control4 contact as needed. Note: You will see the device driver referred to as the “.c4i file” throughout this document.

If you do not have access to a Control4 controller, or find that you need help with registering for any of the services described below, or would like access to the Control4 DriverWorks SDK, please send an email detailing the assistance you need to: SDDP-Support@Control4.com

Registration with Control4
You, or someone at your company who owns the relationship with Control4, should have a dealer or end-user account on http://my.control4.com/

If your company does not already have an account on my.control4.com you can generate an end user account without going through the dealer approval process at the following website: https://my.control4.com/Registration.aspx
Please be sure to select "C4iQ Partners" as your dealer of record when completing the Consumer Account form.

You can also choose to sign up to become a Control4 dealer by accessing the following website: http://www.control4.com/dealers/
Click on the "Sign Up to Become a Dealer" link and complete/submit the online form. After approval, you will be provided with a Dealer login that will grant you access to software updates that will support your SDDP implementations now and in the future.

If you have difficulties finding your company registrations credentials or generating an account, please email us requesting support at SDDP-Support@Control4.com

ComposerPro Credentials
In order to validate your implementation of SDDP, you should have a valid ComposerPro license. This license will allow you to use your ComposerPro credentials to login to a local copy of ComposerPro software on your Microsoft Windows machine. ComposerPro software provides the interface that will allow you to validate your implementation for both the "Auto Add" and "Add on Identify" features of SDDP. Note that you will need a copy of ComposerPro version 2.3 or greater to validate your SDDP implementation.

If your company has a dealer account, released versions of ComposerPro software can be downloaded through Dealer area of the Control4 website:
http://dealer.control4.com/assets/common/themes/Control4Dlr/login.aspx

If you do not have access to valid ComposerPro credentials or you do not have access to ComposerPro version of 2.3 or greater, please email SDDP-Support@Control4.com to request assistance.

Login access to Control4's External File Server
Control4 has provided an external file server for SDDP SDK users on the following website:
https://files.control4.com/action/login

Access to the SDDP folders on this FTP site is useful as the latest SDK and supporting documentation can be downloaded there. Access to these folders is only provided to authorized users. To request access, please send an email to SDDP-Support@Control4.com

3.0 Implementing SDDP

3.1 Implementation Overview

The process outlined in this document represents a simplified way to implement Control4's SDDP protocol onto a device. It is the recommended path for an SDDP adopter to follow. Control4 realizes that this process may not be applicable for all device implementations. Prior to using the SDK, we recommend reviewing all of the supporting documents delivered with the SDK to identify the best approach to implement SDDP onto your device. These documents include the SDDP Specification document as well as the SDDP API document.

Implementation follows a few basic steps that are explained in the in the following sections of this document. To implement SDDP, you will:

- Access the software provided in the SDK (sddp-1.0.tar.gz)
- Modify the example .conf file
- If needed, generate a unique host name for your device
- Compile sddpd to verify there are no compilation errors
- Include the compiled sddpd and modified .conf file into your device software build
- Implement an SDDP announcement with some physical button press (or UI selection) on your device

3.2 Modifying the Example Configuration file

The SDDP SDK includes a sample configuration file called Example-Receiver.conf. It is located in the sddpd-1.0.tar.gz file. The purpose of this file is to not only serve as an example, but it can also be used as template to create your own device configuration file. You will also find an hc800.conf file in the SDK. We are providing this file for reference as it is a real, working copy of successfully created .conf file.

This section will detail each line of the example configuration file and define the values they require. The first step to implementing SDDP is to replace the example content with your device's information. Note that you will need to specify the complete path to location of the config file when compiling your code. Otherwise, the SDK will display an error that resembles: "Incomplete config file `xxx`."

The Example-Receiver.conf file contains the following lines:

```
Example-Receiver.conf
Type = acme:receiver:ACME-100
PrimaryProxy = receiver
Proxies = receiver,tuner,tunerXM,dvd
Manufacturer = ACME
Model = ACME-100
Driver = acme_receiver_ac100.c4i
```

```
Config-URL = http://<host>/netconf/
MaxAge = 1800
```

Each of the lines is defined below:

`Type = acme:receiver:ACME-100`
The Type string uniquely identifies your device's type or model. This must identically match the `<search_type>` value found in the device's .c4i file.

`PrimaryProxy = receiver`
This is your device's primary proxy type. This must match the `<OnlineCategory>` value found in your device's .c4i file.

`Proxies = receiver,tuner,tunerXM,dvd`
The Proxies string lists all proxies implemented by your device's driver. If the driver supports only one proxy, this value will be the same as the PrimaryProxy value. This must match the `<proxies>` value found in your device's .c4i file.

`Manufacturer = ACME`
The manufacturer of the device. This must match the `<manufacturer>` value found in your device's .c4i file.

`Model = ACME-100`
The model of the device. This must match the <model> value found in the device's .c4i file.

`Driver = acme_receiver_ac100.c4i`
This the actual file name of your device's .c4i file as it is stored in the online driver database.

`Config-URL = http://<host>/netconf/`
This is the URL to the device's http accessible management UI. If the device does not have an http accessible management UI, this can be omitted.

`MaxAge = 1800`
This value is the announcement interval in seconds. In the example above, the time interval is 30 minutes. This value represents the amount of time between sending NOTIFY messages. Control4 recommends that you do not modify this value.

Once you have correctly modified the example configuration file with your device's information, customizing the .configuration file is complete.

By default sddpd will load /etc/sddpd.conf. However, you can change this location by specifying another path from the command line. See sddpd command line help (-h) for more information.

3.3 Generating a Unique Host Name

Note: The following step is only required if your device does not already use a Unique Host Name.

Generally speaking, devices that have UPnP implemented (or almost any other device discovery protocol) will already have a unique host name. A simple way to test this is to call `hostname` from the command line on any Linux device. This will report a hostname. You must ensure that the hostname reported is a unique hostname for your device. If your device does not use a unique host name, please follow the steps in the remainder of this section.

As described above, a globally unique host name is required for your device and SDDP to work together. In the SddpServer.c file (delivered in the SDK) the SddpSetHost function is called. You will use this function to set your device's unique host name.

You will need to modify the SddpServer.c file to call SddpSetHost with a globally unique string. This string could be a UUID, MAC address or some other globally unique identifier. As a suggestion, Control4 products concatenate the name using the following format: device type-MAC address. For example: `home-controller-800-000fff123456`.

3.4 Compiling sddpd

For purposes of review, at this point you should have a modified configuration file for your device. It is possible that you have also modified the SddpServer.c file so your device's unique host name is called by the SddpSetHost function.

Control4 now recommends using the makefile supplied in the SDK to test the compilation of sddpd. After successful compilation, you are now ready to integrate sddpd and the modified .conf file into your build system.

3.5 SDDP Device "Identify" Capability

Critical to the success of SDDP implementation is your device's ability to broadcast an "identify message" when an action has taken place to identify the device. The action to identify is typically a physical button push or sequence of button pushes on your device. As a secondary, less desirable option, you may choose a menu selection from the UI of your device to serve as an identification action.

Regardless of means, your device must have the ability to be identified. Specifically, when an identify action takes place on your device, you will need to send a UNIX signal `(SIGUSR1)` to the sddpd process. For example:

```
/ > killall –USR1 sddpd
```

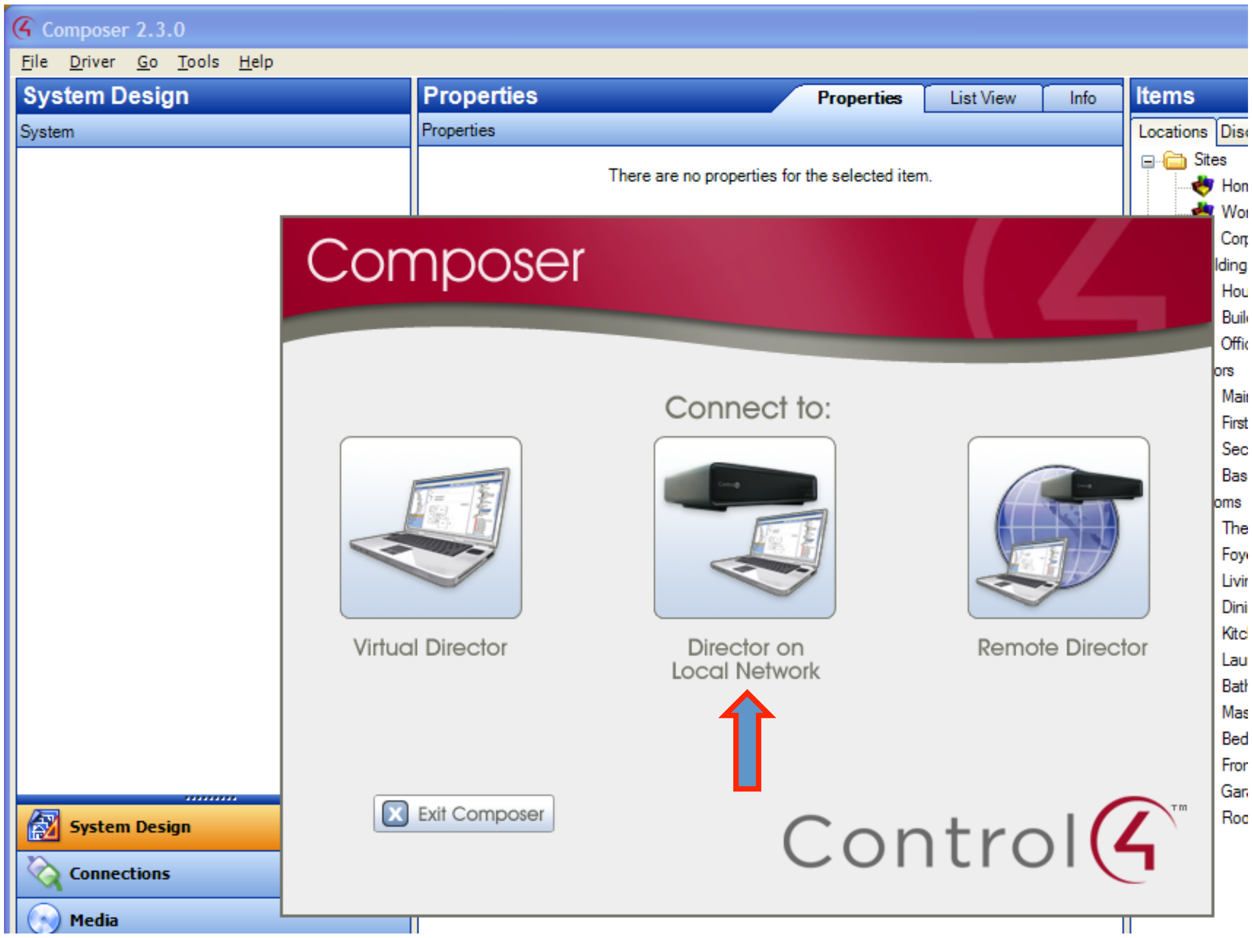
4.0 Testing SDDP with Composer Pro

4.1 Create a simple project

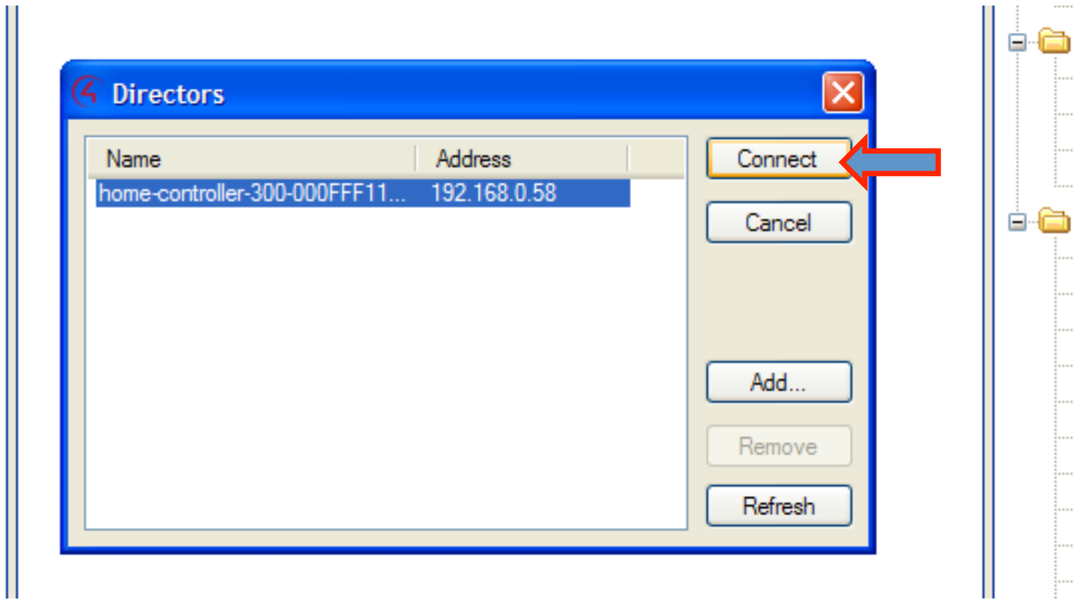
Install ComposerPro 2.3 (or later) in a MS Windows environment. Once launched, you will be asked to enter your Composer login credentials. If you have not received these credentials from us, please send an email to SDDP-Support@control4.com to request them.

Once authenticated, ComposerPro will load to the connection screen shown below. Please make sure that the computer running ComposerPro is on the same local network as your controller and that your controller has been powered on and given time to fully boot. This should take no longer than five minutes.

Next, select "Director on Local Network" from the "Connect to:" options

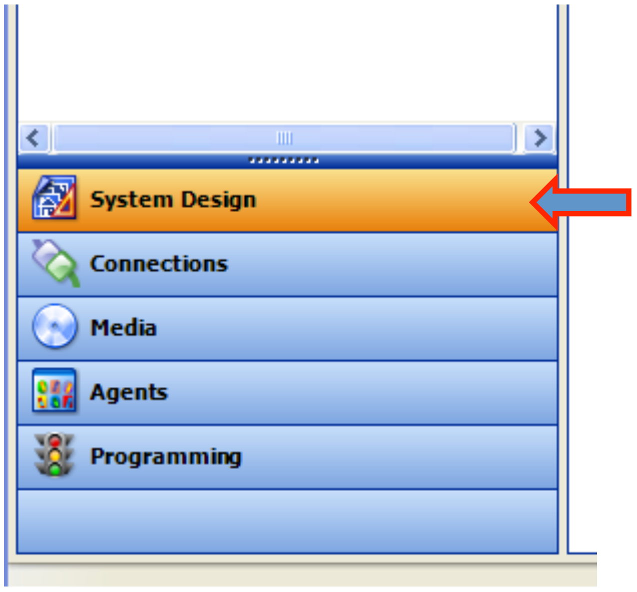


Your controller should be shown in the list of Director Devices along with its IP address. Please select your controller and click the "Connect" button:

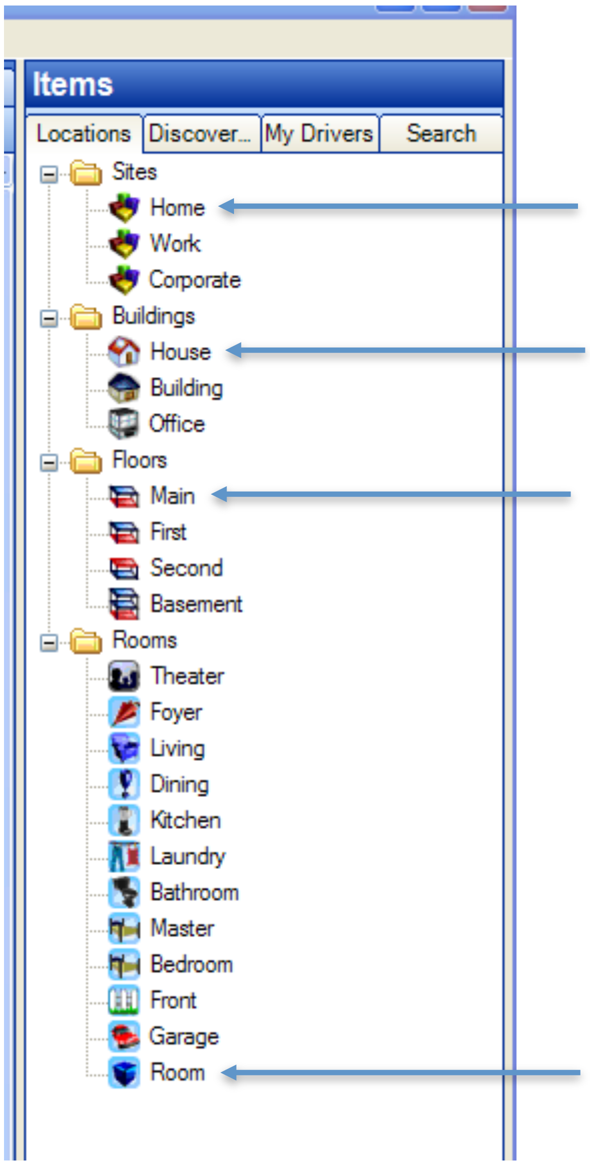


This will connect ComposerPro to the Director software running on your Controller. Once connected, a blank, generic project will be opened.

On the bottom, left hand side of the Composer screen, you should see the "System Design" bar highlighted (if not, please click on this bar):



The right side of the Composer screen should be showing the "Locations" tab with the choices available to define the physical environment where devices will be located in your project:



To build a simple project, double-click on the following location in the order below:

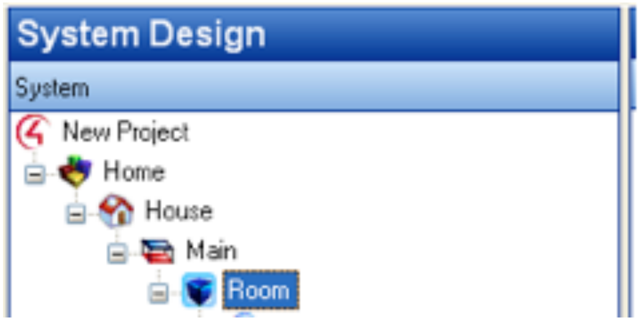
1. Double-click on "Home"

2. Double-click on "House"

3. Double-click on "Main"

4. Double-click on "Room"

This will build a very basic project structure that you should see on the top-left side of the Composer screen:



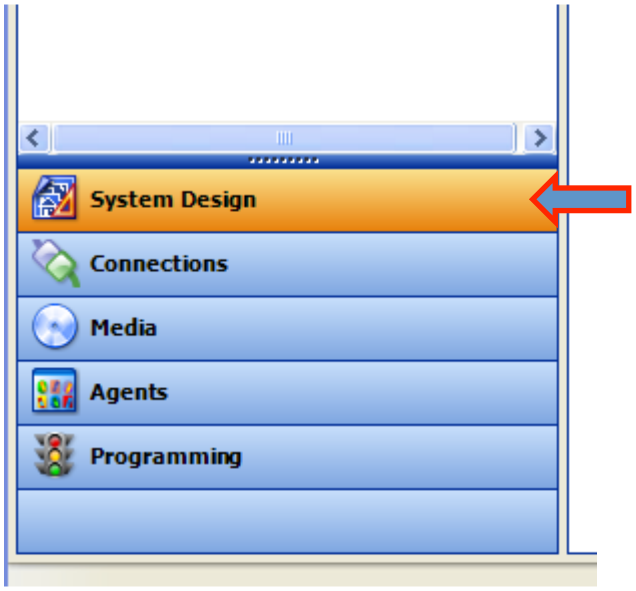
You are now ready to verify that the "Auto add" and "Add on Identify" functionality of SDDP is working on your device.

4.2 Verify the "Auto Add" feature

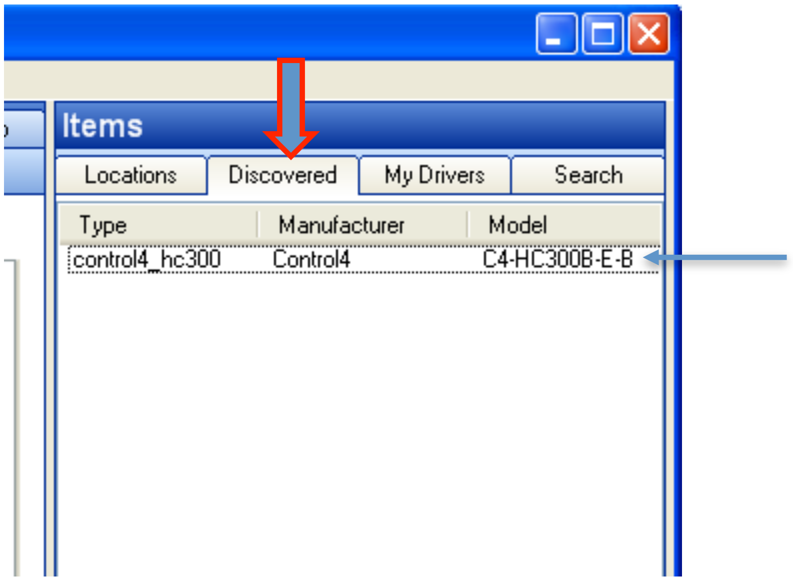
Before proceeding, please ensure that your SDDP enabled device is connected to the same local network as your controller and the computer running ComposerPro.

The "Auto Add feature of SDDP supports an SDDP enabled device to be discovered and displayed in ComposerPro. The device can then easily be added to a project through the Discovered device tab.

On the left hand side of the Composer screen, towards the bottom, you should see the "System Design" bar highlighted (if not, please click on this bar):



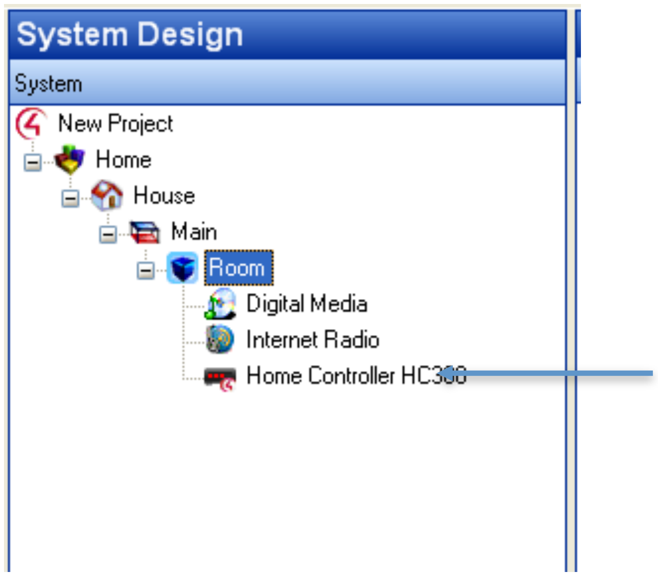
On the right hand side of the Composer screen, please select the "Discovered" tab. This tab shows all available SDDP enabled devices on the same local network as your controller. Note that once a device has been successfully "identified" into the project, it will no longer appear under this tab. Once your device is fully booted, you should see it appear with the appropriate Type, Manufacturer, and Model information as a discovered SDDP device that has not yet been identified into the project:



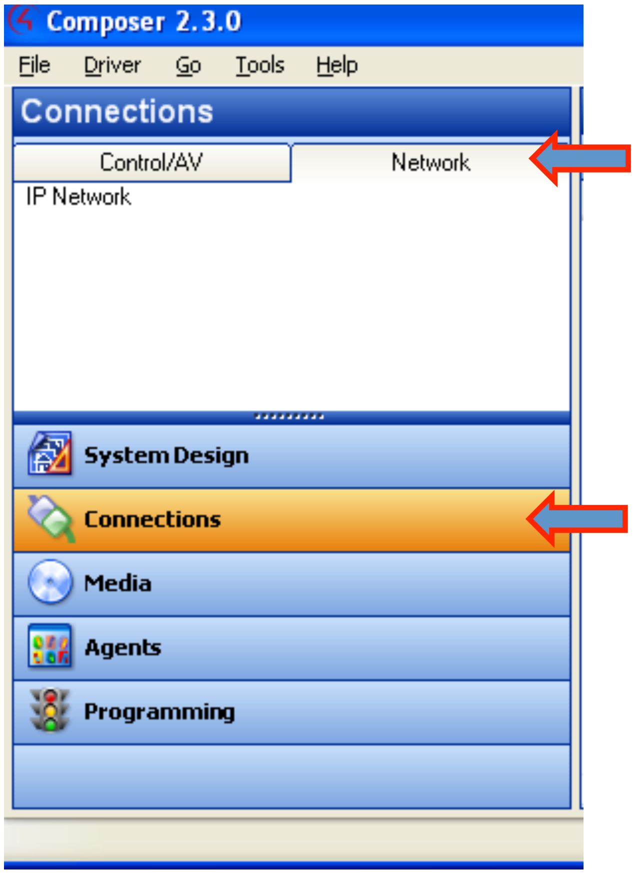
Type	Manufacturer	Model
control4_hc300	Control4	C4-HC300B-E-B

To validate the auto-add feature, simply double-click on your device from the list of devices showing under the "Discovered" tab as show above.

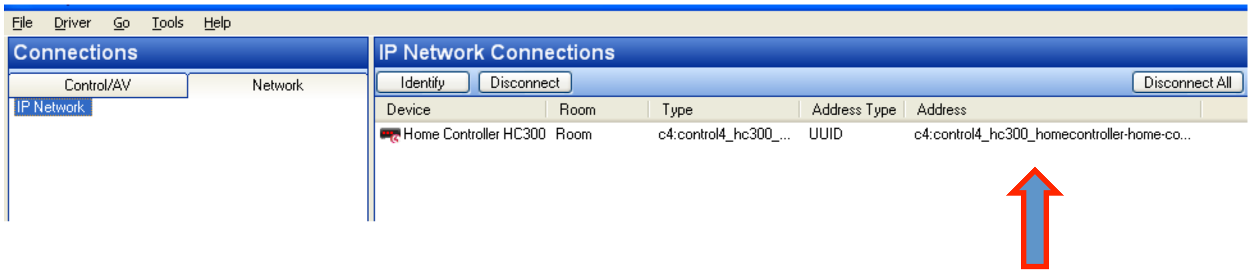
You should see your device automatically appear on the top-left side of the Composer screen in the "Room" of your generic project. If you "mouse over" your device, you should see the name of your driver:



On the left side of the Composer screen toward the bottom, please select the "Connections" bar and then toward the top of the screen, please select the "Network" tab.



You should see that your device has automatically been identified into the project with its UUID:



Device	Room	Type	Address Type	Address
Home Controller HC300	Room	c4:control4_hc300_...	UUID	c4:control4_hc300_homecontroller-home-co...

To prepare to test the "add on identify" functionality, you should now select your device in the "IP Network Connections" window shown above and right click to select "Disconnect Device". This should make your device show up (once again) under the "Discovered" tab in System Design.

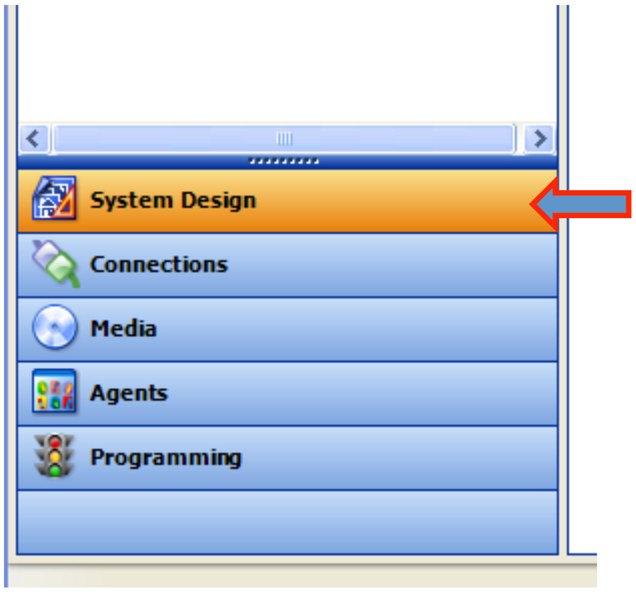
4.3 Verify "Add on Identify" feature

Before proceeding, please ensure that your SDDP enabled device is connected to the same local network as your controller and the computer running ComposerPro.

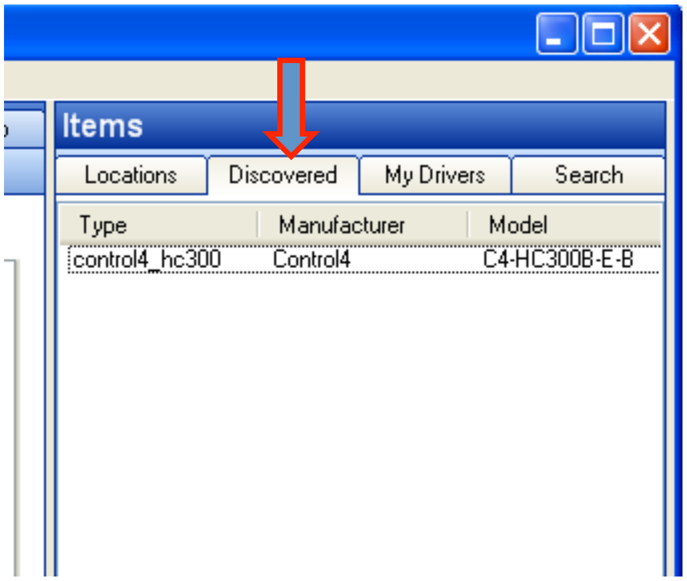
The "Auto on Identify" feature of SDDP supports an SDDP enabled device to be discovered and displayed in ComposerPro. As in the "Auto Add" functionality, the device is displayed in the Discovered tab in ComposerPro. However, a check box called "Auto Add identified devices" is populated and the device is physically identified by means of a button push or a selected UI menu option. The device is then added to the project in the appropriate location.

This feature is particularly useful when a project has several devices that are identical, such as TVs. It will allow for a TV to be discovered and added to a family room as well as the same model TV to be discovered and added in a bedroom.

On the left hand side of the Composer screen, towards the bottom, you should see the "System Design" bar highlighted (if not, please click on this bar):

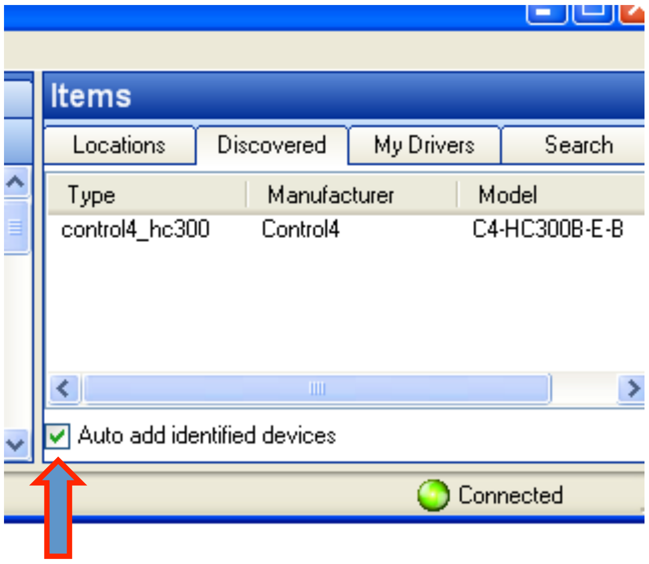


On the right hand side of the Composer screen, please select the "Discovered" tab. This tab shows all available SDDP enabled devices on the same local network as your controller. Note that once a device has been "identified" into the project, it will no longer appear under this tab. Once your device is fully booted, you should see it appear with the appropriate Type, Manufacturer, and Model information as a discovered SDDP device that has not yet been identified into the project:



Type	Manufacturer	Model
control4_hc300	Control4	C4-HC300B-E-B

Please check the box toward the bottom of the "Discovered" tab window that is labeled: "Auto add identified devices":

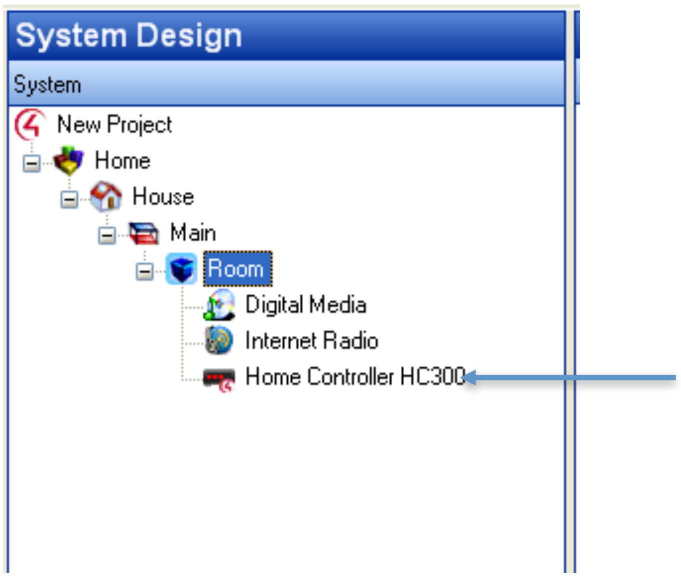


Type	Manufacturer	Model
control4_hc300	Control4	C4-HC300B-E-B

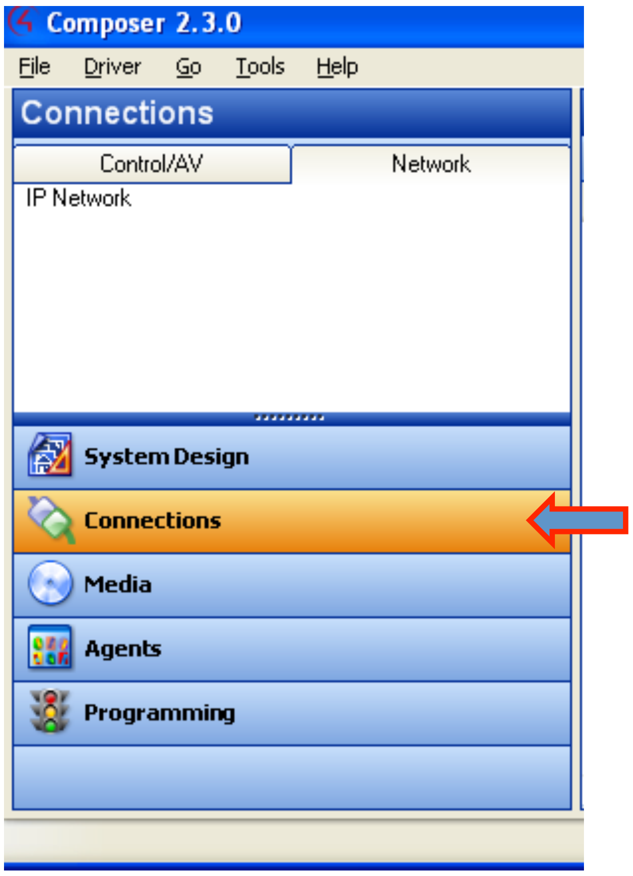
☑ Auto add identified devices

To validate the add-on-identify feature, you will now need to activate the "identify" function of your device. This is the physical action required to identify your device. This may be a physical button press on the device or selectable a menu option in the device UI.

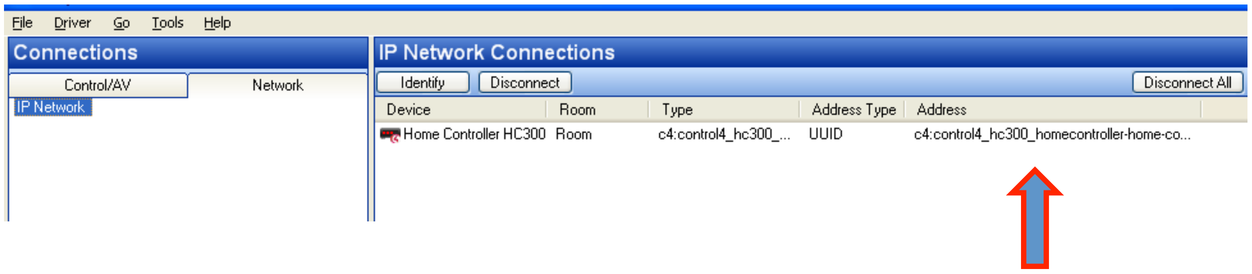
You should see your device automatically appear on the top-left side of the Composer screen in the "Room" of your generic project and if you "mouse over" your device, you should see the name of your driver:



On the right side of the Composer screen toward the bottom, please select the "Connections" bar and then toward the top of the screen, please select the "Network" tab.



You should see that your device has automatically been identified into the project with its UUID:



Device	Room	Type	Address Type	Address
Home Controller HC300	Room	c4:control4_hc300_...	UUID	c4:control4_hc300_homecontroller-home-co...

If you wish to repeat either of these validations, you should now select your device in the “IP Network Connections” window shown above and right click to select “Disconnect Device”. This should make your device show up (once again) under the “Discovered” tab in System Design.

4.4 Testing the sddpd.conf file with Packet Transfers

If a properly configured sddpd.conf file exists for the device it will be possible to run the sddpd daemon and initiate communication. This can be tested by using SDDP functions. Note that SDDP uses port 1902.

Director Start Up: When director starts up, it will multicast a SEARCH request as defined in section 4.1 of this document. It will also multicast this message periodically, when exactly depends on the MaxAge of each discovered device.
Sddpd should respond to this request with a direct unicast to director (not a multicast or broadcast) as defined in section 4.2 of this document.

Start Up/NOTIFY ALIVE: When sddpd starts up, it will multicast a NOTIFY ALIVE message as defined in section 5.2.

Shut Down/NOTIFY OFFLINE: When sddpd is shut down (when the system shuts down), it will multicast a NOTIFY OFFLINE message as defined in section 5.4

Identification/NOTIFY IDENTIFY: When device identification is triggered, sddpd should multicast a NOTIFY IDENTIFY message as defined in section 5.3. Note that the identification “action” could be performed in numerous ways. For example, the device may have an identification button that must be pressed or a menu item may be selected from a configuration U. Also, the identification process may be initiated manually by sending a SIGUSR1 signal to sddpd by command line. For example: `kill -SIGUSR1 `pidof sddpd’`