KODARO

Inovonics Driver Guide

February 03, 2017
This documents usage of the Inovonics driver on the Niagara AX platform.

OVERVIEW

This document serves as a guide for using of the Inovonics serial driver on the Niagara AX platform.

This driver is a passive read-only driver that listens for messages from an Inovonics wireless gateway.

System Compatibility

This driver integrates with the serial user interface of Inovonics Universal Serial Receiver. It can auto detect all [supported] devices in range and add them automatically along with their full point database.

Supported Functions

- Auto database build
- Reading values

Compatible devices:

- Connects to Universal Serial Receiver with a DB9 connector to the Tx and Rx pins.
  - Pin3 on DB9 to Rx on Receiver
  - Pin2 on DB9 to Tx on Receiver
  - Pin5 on DB9 to GND on Receiver
- Can see on network:
  - All Security End Devices
  - All Temperature/Humidity Devices
  - All Analog Input Devices

Niagara Compatibility

This software will function on all Niagara AX 3.*n.nnn* platforms.

Workflow Summary

1. Install module
2. Add Driver object
3. License Driver object
4. Add or Auto Detect devices
5. Add Points (if not auto detect used)
6. Remove unwanted Points (if auto detect)

PHYSICAL CONNECTION

The driver can communicate with the Universal Serial Module through the RS232 com port on the JACE (com1). The preferred method is to use the serial and power cable combination (ACC643) that plugs directly into the RS232 port on the JACE and the other end to the receiver. Custom cables will need to be made to communicate with the serial receiver if this cable is not used.

MODULE INSTALLATION

Install the driver module on the computer where Niagara AX Workbench will be run. To install, place a copy of the file in the modules directory of your Niagara AX installation. This is typically C:\Niagara\Niagara-3.n.nnn\modules.

Install the module on the target station. Using a Niagara AX Workbench where the module has already been installed, connect to the station's platform service. Go to the Software Manager and install.

ADDING THE DRIVER OBJECT

Open the driver palette in the engineering tool. Copy and paste the InovonicsDriver object under the Drivers node in the station database.

License the driver object.

LICENSING THE DRIVER OBJECT

Every network must be licensed. Unlicensed networks will operate in demo mode for 2 hours. After demo mode expires, the station must be restarted to resume operation, but it is otherwise safe to build a database.

Licensing is managed on an object in your database. The licensing object is located on the property sheet of the InovonicsDriver. It has the following properties.

- Product Code – Text automatically generated by the driver that is needed to generate a license key.
- License Key – Where the key to validate the license must be entered.
- Devices – The number of devices under all networks to license.

Set the number of devices. Copy the value of the "Product Code" property that is automatically generated. You should highlight the value and copy it by pressing CTRL-C. Send the product code to your Kodaro representative. They will respond with a text string for you to enter in the "License Key" property.

You must restart the station after changing the "License Key".

The exact text and case of the product code and license key are critical. Do not send screen shots. Highlight the text, copy it using CTRL-C and paste into an email.

ADDING NETWORK OBJECTS

Open the driver palette in the engineering tool. Copy and paste the appropriate network object under the driver node in the station database.

Configure the connection properties such as serial ports, addressing and credentials. The driver uses the ping monitor to determine when to try to establish a connection. Wait for the driver to connect or manually invoke the ping action.

AUTO DISCOVERY

Once the network object is connected, it is time to discover devices. The network defaults to the “Auto Add Devices” property set to true. This will learn in all devices and their points on the device’s next transmission. If you set this feature to false, then you will need to either select the “New” button on the “Inovonics Device Manager” view or drag an InovonicsDevice from the pallet into the network. You will only need to set the UID property for it to begin communication.

If there exists or will exist more devices than are licensed it is important to disable auto discovery once the desired devices have been brought into Niagara. When there are more devices than are licensed, the driver will cease communications.

MANUALLY ADDING POINTS

Once device objects have been added, double click the “Points” icon in the device manager view, or from the device property sheet. This take you to the point manager view.

When adding points, choose the correct “Type” in the add dialog. The driver adds read only points by default because there are no write capable points in these devices.

Set the point address. See Appendix A for full point registry for supported devices.

COMPONENT REFERENCE

Inovonics Driver
This is the root node of the driver. It represents a tree of Inovonics networks.

Properties:

- **License** – See the section title “Licensing”.

Inovonics Serial Network
This represents a serial connection to a serial receiver. It is a network-level component in the NiagaraAX architecture and has standard network component properties (see "Driver Architecture / Common network components" in the *NiagaraAX User Guide* for more information).

The following properties are unique or have special importance:

- **Serial Config** – Where to configure your serial port. By default this is already configured on COM1 to the correct settings (9600,8,1,none). **Do not change these settings; they are the only ones that work with the serial receiver.**
- **Message Queue** – The message queue is where serial transactions are queued up and executed in synchronous order. This object has some properties that are informational in nature only. There is also a dump action which will print the contents of the queue to console.
- **Tuning Policies** – Policies such as when to write values and polling frequency are configured here.
- **Debug** – When true, incoming serial data is written to the Station Director console. This property resets to false every time the station is restarted.
- **State** – The current state of the serial connection to the device.
- **Serial Receiver** – Read only information about the serial receiver connected to this network.
    - **Messages Since Last Check In** – How many messages the receiver has received since it last report to the network
    - **Case Tamper –** True if the case is open
    - **Supervision –** True if there has been a change in the message since the last report

Inovonics Device
This is a device-level component in the NiagaraAX architecture and has standard device properties (see "Driver Architecture / Common device components" in the *NiagaraAX User Guide* for more information).

The following properties are unique or have special importance:

- **Device Model** – The manufacturer’s device model number.
- **UID** – unique identifier of the device acts as the address.
- **First Hop** – First device to pick up the message, either a gateway or a repeater.
- **Signal Strength** – Strength of the signal.
- **Margin Level** – Signal to noise ratio of the transmission.
- **Trace Count** – Number of trace unique ID’s involved in the transmission. For future use.
- **Hop Count** – Number of hops made by the transmission.
- **Market ID** – Market identifier of the manufacturer
- **Transmission Interval** – Time in between transmissions from device.
- **Measurement Interval** – Time in between measurements by device.

Inovonics Proxy Ext

The following properties are unique or have special importance:

- **Address** – This is the name used in a point log to poll the value.
- **Device Facets** – The facets are used to map values into Niagara. For Boolean and Enum points, it is important the trueText/falseText or enum range be set.

TROUBLESHOOTING

Problems in General

- Is the driver licensed? Has the demo mode expired?
- Turn on debug and watch the console. Anything obvious?
    - Debug property is on the network property sheet.
    - Watch the Application Director console.
- Performance issues
    - Look at the Message Queue on the network property sheet
        - Discovery will be on the urgent queue. Are there a lot of messages on the urgent queue?
        - Invoke the dump action on the message queue and look at the application director to see what is on the queue.
- Try to rule out the driver.
    - Does the problem exist in other software such as HyperTerminal?

Trouble Connecting

- Check license state.
- Check the SerialConfig object on the network.
    - Is the correct com port selected?
    - Is it in use by another driver?
    - Is the baud rate correct?
- Turn on debug
    - Anything printing on console?
        - Try connecting with HyperTerminal.
        - Usually a cable problem if manually wired.
    - Does the output look wacky?
        - Verify the baud rate is correct.

Read Problems

- What is the fault cause in the point manager or on the point Proxy Ext?
- Try increasing the timeout.
- Could be a performance problem, look at the message queue (see Problems in General)

APPENDIX A: SUPPORTED DEVICES AND POINT REGISTRY

Security End Devices

Security End Device Boolean Point Registry

	Name	Clean Me	Alarm4	Alarm3	Alarm2	Alarm1	Low Battery	Tamper	Supervision	Reset
Product	**Address**	**1**	**4**	**5**	**6**	**7**	**9**	**10**	**11**	**12**
E*1210		N	N	N	N	Y	Y	Y	Y	Y
E*1210SK		N	N	N	N	Y	Y	Y	Y	Y
EN1210EOL		N	N	N	N	Y	Y	Y	Y	Y
EE1215		N	N	N	N	Y	Y	Y	Y	Y
EN1215EOL		N	N	N	N	Y	Y	Y	Y	Y
E*1235SF		N	N	N	N	Y	Y	Y	Y	Y
E*1235DF		N	N	N	N	Y	Y	Y	Y	Y
E*1247		N	N	N	N	Y	Y	Y	Y	Y
EN1249		N	N	N	N	Y	Y	Y	Y	Y
EN1260		N	N	N	N	Y	Y	Y	Y	Y
EE1261		N	N	N	N	Y	Y	Y	Y	Y
EN1261HT		N	N	N	N	Y	Y	Y	Y	Y
E*1262		N	N	N	N	Y	Y	Y	Y	Y
E*1265		N	N	N	N	Y	Y	Y	Y	Y
E*1210W		N	N	N	Y	Y	Y	Y	Y	Y
E*1212		N	N	N	Y	Y	Y	Y	Y	Y
EE1215W		N	N	N	Y	Y	Y	Y	Y	Y
EN1215WEOL		N	N	N	Y	Y	Y	Y	Y	Y
E*1216		N	N	N	Y	Y	Y	Y	Y	Y
E*1223S		N	N	N	N	Y	Y	N	Y	Y
E*1223D		N	N	N	N	Y	Y	N	Y	Y
E*1223SK		N	N	N	N	Y	Y	N	Y	Y
E*1233S		N	N	N	N	Y	Y	N	Y	Y
E*1233D		N	N	N	N	Y	Y	N	Y	Y
E*1235S		N	N	N	N	Y	Y	N	Y	Y
E*1235D		N	N	N	N	Y	Y	N	Y	Y
EN1224		N	Y	Y	Y	Y	Y	N	Y	Y
EN1224-ON		N	Y	Y	Y	Y	Y	N	Y	Y
E*1236D		N	N	Y	Y	Y	Y	N	Y	Y
E*1238D		N	N	N	Y	Y	Y	N	Y	Y
E*1242		Y	N	N	N	Y	Y	N	Y	Y

Analog End Devices

Analog End Device Boolean Point Registry					
	Name	DeltaA	Low Battery	Supervision	Reset
Product	Address	0	9	11	12
E*1702		Y	Y	Y	Y

Analog End Device Numeric Point Registry			
	Name	AI1	AI2
Product	Address	0	1
E*1702		Y	Y

Analog End Device Enum Point Registry			
	Name	Input Type	DeltaA
Product	Address	0	1
E*1702		Y	Y

Temperature/Humidity Devices

Temperature/Humidity End Device Boolean Point Registry						
	Name	Temperature	Humidity	Low Battery	Supervision	Reset
Product	Address	0	2	9	11	12
E*1210		Y	Y	Y	Y	Y

Temp/Humidity Device Numeric Point Registry			
	Name	Temperature	Humidity
Product	Address	0	1
E*1702		Y	Y

Analog End Device Enum Point Registry			
	Name	Units	DeltaT
Product	Address	0	1
EN1721		Y	Y
E*1722		Y	Y

Dual Input Temperature End Device

Dual Input Temperature End Device Boolean Point Registry						
	Name	DeltaInternal	DeltaExternal	Low Battery	Supervision	Reset
Product	Address	0	2	9	11	12
E*1723		Y	Y	Y	Y	Y

Dual Input Temperature End Device Numeric Point Registry			
	Name	InternalTemp	ExternalTemp
Product	Address	0	1
D*1723		Y	Y

Dual Input Temperature End Device Enum Point Registry				
	Name	Units	DeltaT	ExternalSensorUnits
Product	Address	0	1	2
EN1721		Y	Y	Y

High Power Receiver

High Power Repeater Boolean Point Registry						
	Name	ConfigMessage	ReceiverJammed	LowBattery	CaseTamper	Supervision
Product	Address	2	8	9	10	11
EN5040/EE5000		Y	Y	Y	Y	Y

High Power Repeater Boolean Point Registry			
	Name	Reset	OnBattPower
Product	Address	12	14
EN5040/EE5000		Y	Y

High Power Repeater Numeric Piont Registry		
	Name	NetworkId
Product	Address	0
EN5040/EE5000		Y

Submetering End Devices

Submetering End Device Point Registry Boolean Point Registry								
	Name	DeltaTotalized	RapidTransmissionMode	LowBattery	CaseTamper	Supervision	Reset	Sleep
Product	Address	0	1	9	10	11	12	14
E*1501		Y	Y	Y	Y	Y	Y	Y
E*1501-XL		Y	Y	Y	Y	Y	Y	Y
E*1501-EXT		Y	Y	Y	Y	Y	Y	Y
E*1550		Y	Y	Y	Y	Y	Y	Y

Submetering End Device Point Registry Analog Point Registry			
	Name	Count	LeakDetection
Product	Address	0	1
E*1501		Y	Y
E*1501-XL		Y	Y
E*1501-EXT		Y	Y
E*1550		Y	Y

APPENDIX B: APPLICATION NOTES

This section is intended to clarify possibly confusing data point naming convention. It is not intended to inform a user on how to build an application or how to use these or any of the data that is discovered via this driver.

SUBMETERING END DEVICES:

All submetering devices contain a point called "Leak Detection". The naming of this variable may lead somebody to think that this indicated directly when a leak was detected but in fact is a little more complex than that.

The manufactures description of this value:

Leak Detection
Monitoring fifteen-minute periods during the last 24 hours and sending information about a number of periods with flow, which could help in determining if there is a leak.

All transmitters shall determine whether or not at least one pulse count of water flow (one totalizer pulse) was detected within a leak detection period, and then report the number of periods in which this occurred. The leak detection period shall be set at 15 minutes, and the number of moving-window monitoring periods shall be set at 96 (96x15 minutes= 24 hours).

The period duration, as well as a number of periods shall be factory setup parameters. All transmitters shall support a maximum number of 96 periods.

The

The Leak Detection will always give a count of pulses based on the measuring window and the windows that had pulses, i.e. if no pulse is received for 24 hours Leak Detection will be zero and if the window is 15 minutes and pulses are continually received for a 24 hour period, the Leak Detection will be 96.

What does that all mean?

Here's an example to help clarify those descriptions.

Example:
State 1:
A pulse meter comes online and has never received a pulse or has not received a pulse in 24 hours.

Leak Detection = 0.

State 2:
The pulse meter reads some number of pulse consistently over the course of 2 hours and has a monitoring period of 15 minutes. Assuming all monitoring periods contained at least 1 pulse:

Leak Detection = 8 (2 hours = 120 minutes which is 8, 15 minute windows: 8 x15=120)

State 3:
Now the pulse meter stops receiving pulses for 24 hours.
Leak Detection = 0.

Leak Detection will decrement as pulses are not monitored over the 24-hour period and does not necessarily go from a non-zero value immediately back to zero but rather gradually goes down as the moving window continues to move without pulses being monitored.

DRIVER HISTORY

December 28, 2010: 3.5.25

- Initial release.

April 04, 2011: 3.5.25.2

- Device never going down bug fixed -

November 14, 2012: 3.6.47

- Dual temperature input device added
- Receiver object added to network
- Debug option added
- Device Model and UID added to device manager
- Point address added to point manager

November 27, 2012: 3.6.47.1

- High Power Repeater added

November 28, 2012: 3.6.47.2

- Numeric data conversion correction

February 3, 2017: 3.8.38.1

- Submetering device support added
- Rebranding to Kodaro