

INTELLIGENT MODULES



CHAPTER 5

In This Chapter...

CLICK PLUS Intelligent Modules	5-2
C2-DCM Communication Module.....	5-2
C2-NRED Node-RED Module	5-3
C2-OPCUA OPC UA Server Module.....	5-28

CLICK PLUS Intelligent Modules

The CLICK PLUS Intelligent Modules are a series of Option Slot modules that contain onboard processing. These modules allow us to implement computationally intensive features without affecting with the performance of the ladder logic program running on the CLICK CPU.

Each intelligent module is self-contained, with dedicated processor, RAM and storage. The modules communicate with the CLICK CPU on its internal bus.

C2-DCM Communication Module

The C2-DCM module adds two more serial ports to a CLICK PLUS PLC. Each port can be configured as either an RS-232 or RS-485 connection, and can communicate with either Modbus RTU or ASCII protocols.

The C2-DCM ports are configured in the CLICK Programming software, and offer the same options as the built-in serial ports.

C2-NRED Node-RED Module

Introduction

What is Node-RED

Node-RED is an Open Source “Low-Code” programming tool for wiring together hardware devices, APIs and online services in new and interesting ways.

It provides a browser-based editor that makes it easy to wire together flows using the wide range of nodes in the palette that can be deployed to its runtime in a single-click.

What is C2-NRED

The C2-NRED is a CLICK PLUS PLC Slot Module that enables a fully autonomous processor to run Node-RED software and share memory with a CLICK PLUS PLC. This direct integration provides the following benefits:

- An Industrial package with DIN Rail mounting, power and communications shared with the CLICK PLUS PLC.
- Project Backup and Set up tools.
- A shared Backplane – enabling the C2-NRED module to directly read the CLICK PLUS memory registers. No more configuring communications with your PLC.
- Independent Processors, the CLICK PLUS PLC and the C2-NRED modules each have their own processors, so while they share memory, the application processing times on the CLICK PLUS will not be impacted by heavy data processing or communications on the C2-NRED module.

System Requirements

- **Hardware:** The C2-NRED module is compatible with slot 0 or slot 1 of any CLICK PLUS PLC with one or two option slots.
- **Software:** Requires CLICK Programming software version 3.70 or above. Compatible with Google Chrome v128+ or Microsoft Edge v128+
- **Network:** USB2.1 compatible port on Windows PC or 10 Mbps or faster Ethernet TCP/IP network

Recommended CLICK Node-RED BOM

Qty	Part Number	Notes
1	C2-01CPU or C2-01CPU-2	You'll want an ethernet port but there is limited need for the WiFi option since the C2-NRED module requires a wired connection.
1	C0-01AC	CLICK 24VDC power supply.
1	C2-NRED	The Node-Red module.
1	D0-MC-BAT	Battery to extend the memory storage time from 1 hour to a typical 3 years.
1	SE3-SW5U	Unmanaged 5 port switch.
2	C5E-STPBK-S3	3ft cat5e cables for the CLICK and C2-NRED modules.
1	C5E-STPBK-S10	10ft cat5e cable to connect to your PC.



NOTE: Don't forget I/O modules for your application!

These part numbers are a representative example. Be sure to review the cable lengths, quantities, type of ethernet switch required and so forth.

Installation



NOTE: If your CLICK PLUS PLC Firmware is below version 3.70, update the firmware in the PLC prior to installing the C2-NRED module.

Follow the instructions in the “Install the CLICK Programming Software” section of Chapter 1 to get the latest version of the CLICK Programming software installed on your PC. Refer to the CLICK PLUS application help files for more information:

[CLICK Help Version 3.70 - Introduction \(automationdirect.com\)](#)

After powering on and connecting to the CLICK PLUS PLC from the CLICK Programming Software, you must update the firmware to get both the CLICK PLUS PLC and the C2-NRED module up to the release version of firmware.

It is strongly advised to update the CLICK PLUS PLC firmware BEFORE installing the C2-NRED Module.

Follow these steps:

1. Ensure you have 24VDC wired to the CLICK PLUS PLC.
(Do not use the USB low power mode.)
2. BEFORE you install the C2-NRED module, connect to the CLICK PLUS PLC and update the firmware.
3. Power off the CPU.
4. Install the C2-NRED module, following the hardware installation instructions in the “Installing Option Slot Modules” section of Chapter 3.
5. Restore power.
6. Update the firmware again—this time it will update the C2-NRED firmware. This process can take up to 20 minutes.

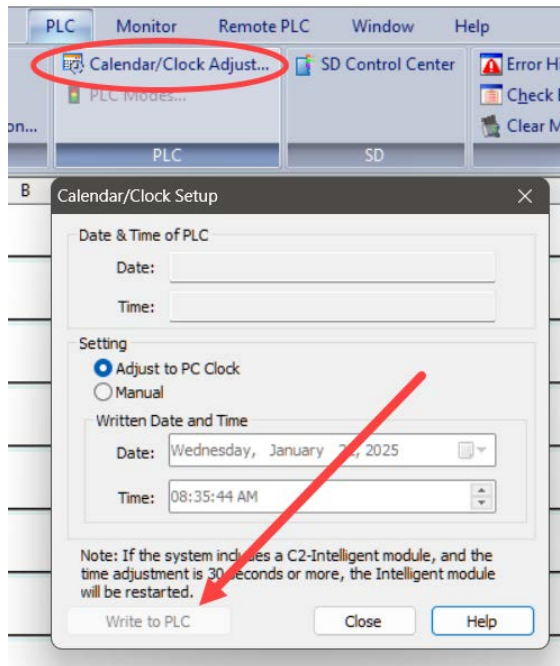


Project Configuration

Once the firmware is up to date, you must enable the C2-NRED in order to begin. See the [C2-NRED Module Configuration](#) topic in the online help for details on configuring the module.

Time (RTC)

The C2-NRED module shares a clock with your CLICK PLUS PLC. Be sure to update the time in the PLC. Click the PLC tab at the top of the CLICK programming software. Then click “Calendar/Clock Adjust”. If the time is wrong, it may impact your ability to download 3rd party modules into the Node-RED environment. Those modules are signed and if your clock is not set, you will get CERT_NOT_YET_VALID errors.



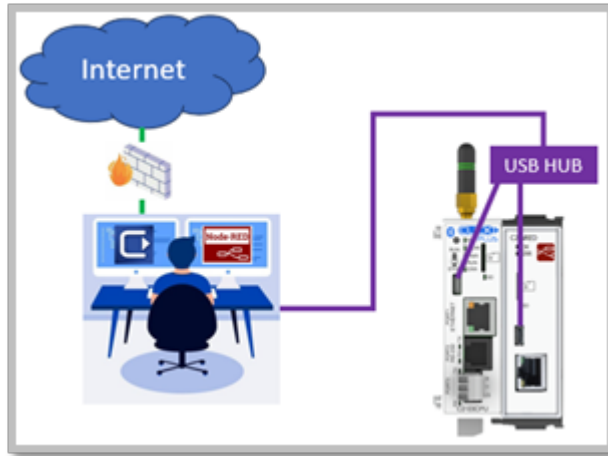
Communication

Once you have enabled Node-RED, you are able to connect to the C2-NRED module. There are 3 ways to connect:

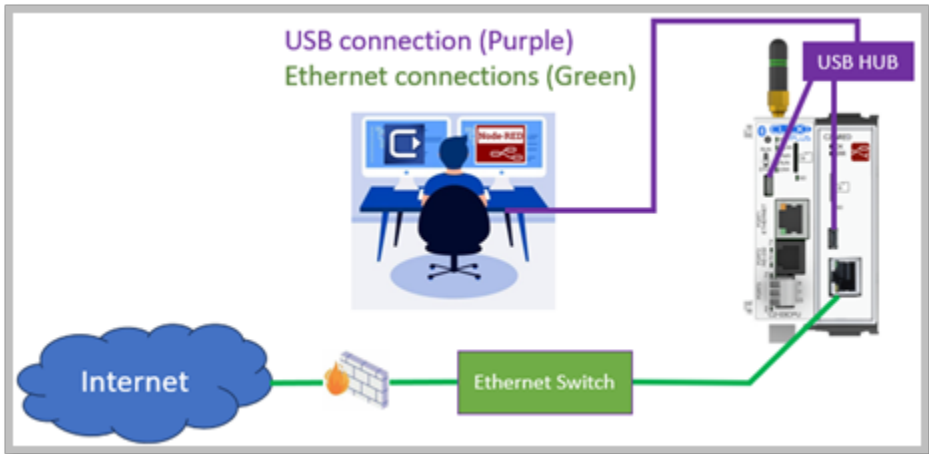
USB — Using a USB Cable, plug into the C2-NRED module. This will enable the ability to begin programming and configuring the module with the following limitations:

- Limited ability to install add on Node-RED Modules
- No ability to connect to the network using ANY Network Nodes

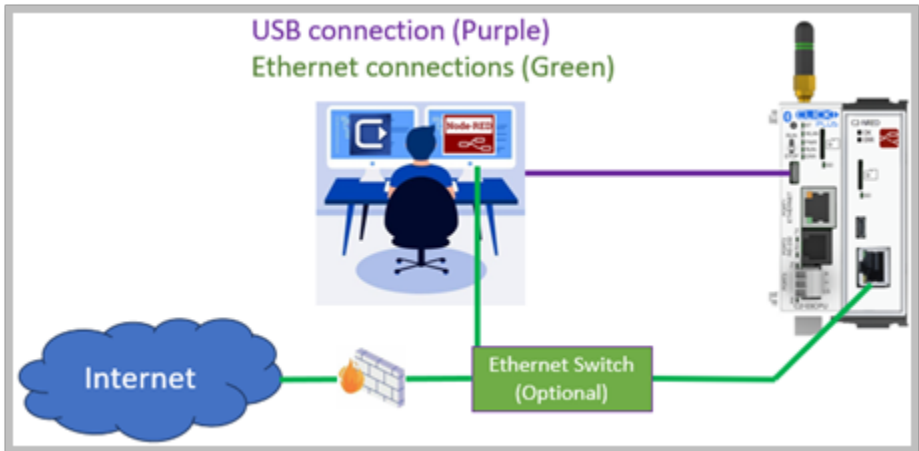
Those communication abilities require a direct connection to the network, not a bridged connection through your PC.



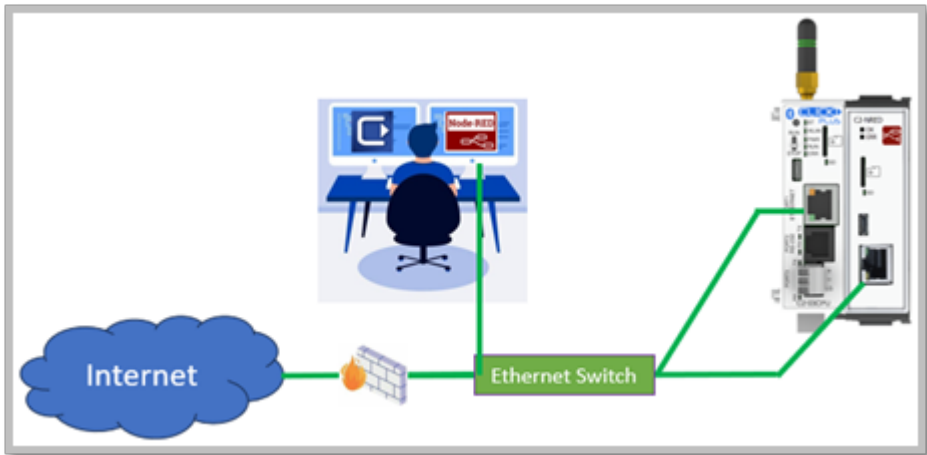
USB and Ethernet — Connect the C2-NRED module to your PC using the USB Cable and connect the C2-NRED to your network using an Ethernet cable. This will allow you to program the C2-NRED using USB, but the web server will access the network over Ethernet.



OR



Ethernet Only* — Connect your PC, and each Ethernet enabled module to a switch. All modules can be accessed over the same network and all will have access to HTTP services. If you have multiple network adapters on your PC (Wi-Fi and Wired, or multiple wired), you will not be able to connect one adapter to an external network (Internet) and the second adapter to the C2-NRED module. Node-RED will not communicate across bridged adapters or shared connections.



*This is the recommended approach due to its simplicity and speed.

Updating Firmware

After installing your C2-NRED module and ensuring you have stable communications, ensure you are using the most up to date firmware. You can check by using the Update Firmware option in the CLICK Programming software.



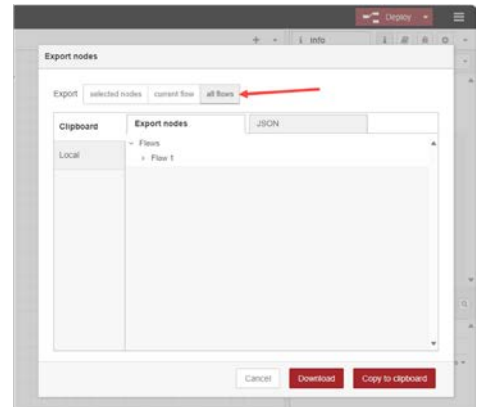
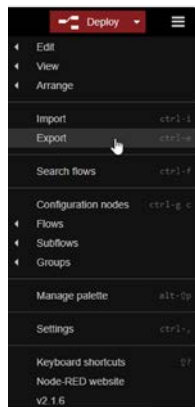
This will open a window which will show the current firmware version installed in both the PLC and the C2-NRED module. It will give you the option to update the firmware for both devices.



NOTE: Upgrading the firmware in the C2-NRED module will reset the unit to factory settings. It will erase all third party add-ons as well as your application.

Follow these steps exactly when upgrading firmware. There is no “undo” option.

1. Before you begin the firmware upgrade process:
 - a. Make a backup of your CLICK Program and settings (.CKP file).
 - b. Open Node-RED.
 - c. Open the Manage Pallet option under the menu.
 - d. Record the EXACT Names of each of the Nodes you have imported for your project.
 - e. Export your projects (Menu – Export).
 - f. The C2-NRED module must have internet access to reinstall any imported nodes, or you should ensure you have the source .tgz files for any third party modules. (See the “Getting Your Project Ready for Production” section later in this chapter.)



2. Install the latest CLICK PLC Software (Version 3.70 or later).
3. Ensure the PLC is in STOP Mode and has an external 24v power connection (not just power from the USB Connection)
4. Perform a normal firmware update of the CLICK PLUS PLC (to “Ver3.70” or later), and C2-NRED (to “1.0.0.100” or later)
Note: If updating from Version 3.60 or previous, two updates are required. First update the CLICK PLUS PLC, then detect C2-NRED and update again.
5. You may have to disconnect and reconnect to reset your IP address.
The firmware update will reset IP addresses to default.
6. Transfer your project file to the CLICK PLUS PLC. If you had previously connected to the C2-NRED, those settings will be rebuilt when you transfer the project file.
7. Make sure your C2-NRED Project Setup has “Enable Node-RED” checked. (Setup Ribbon, Slot0/1 Setting, Node-RED
8. Enable RUN Mode
9. Data View SD302-SD305 should show “1.0.0.100” or later (SD402-SD405 for Slot1)
10. Connect Web browser to the C2-NRED (USB or Ethernet)
11. Node-RED Version should be “v3.0.2”

Launch Node-RED

When you are ready to launch Node-RED, click the Node-Red Open Node-RED button on the “PLC” Ribbon:



This will open your default web Browser to the IP and port of the C2-NRED module. If you are using a USB connection to the C2-NRED programming port, it may take up to 90 seconds to initialize the USB driver the first time it is opened. Please be patient.

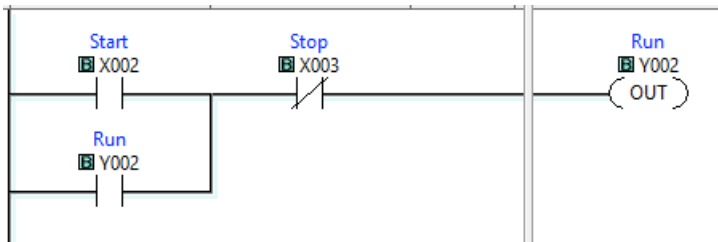
Programming

For those who have written Ladder Logic, the Graphical Programming Language (GPL) and scripting in Node-RED will seem unfamiliar. Conversely for those familiar with scripting languages like JavaScript, Ladder Logic in a CLICK PLUS PLC may feel arcane. All three languages are good and have their strengths, but the marriage of all three in the CLICK C2-NRED module will enable truly new approaches to both industrial control and maker space/commercial applications.

Let's hit a few definitions to ensure programmers understand these technologies.

Ladder Logic

Ladder Logic was developed in the 1970s as a GPL designed to look and operate like standard electrical drawings. The intent was to enable electricians with no programming experience to be able to read and write industrial control applications. The logic flows left to right and top to bottom and can be best understood by the water analogy—if water can flow through the circuit, it will. Take for example this simple latching circuit:



The vertical line on the left is called the power rail. The horizontal row is called a rung (like a rung in a ladder). If you picture water flow down the power rail, it will try to cross the rung. If X002 is an input from a momentary pushbutton, then when the button is pressed, that contact will close and the water will be able to cross that gap. At the same time, the Stop contact (X003) is normally closed, so water can cross it until that button is pressed. Once water hits the “Run” output coil (Y002), the circuit activates. When the coil is activated, the Y002 contact closes indicating it is on and the user no longer needs to hold down the button connected to X002; the circuit holds itself on, or the circuit is latched. An electrical drawing may show a symbol for a button instead of the contact, but otherwise it is the same.

Scripting languages

Scripting is a subset of programming where the program is read by the computer one line at a time. The alternative is a “compiled” application where the computer processes and optimizes the entire application ahead of time, then can run it extremely fast. Scripting languages like JavaScript and Python run slower than compiled, but they are easier to write and debug. The same logic from the example above would look like this:

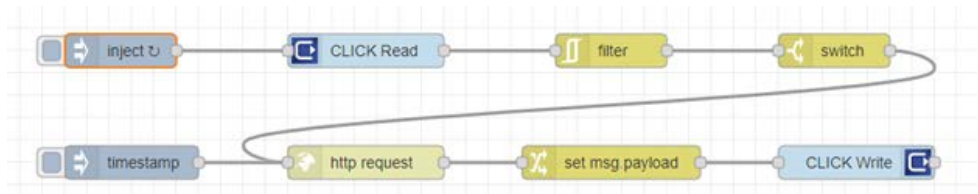
```
If (X002 or Y002) and (not X003) then {  
    Set Y002;  
}
```

When you look at a rung, ladder executes these “if then” logic sets extremely quickly. Ladder is very good at highspeed real time logic while iterative and sequential processes are sometimes easier to implement using scripting languages.

Flow editor

Node-RED introduces a 3rd type of programming. Like Ladder, it's a GPL, called a Flow editor. It works like a flow chart where each node performs a function. It can be a simple function like checking a data value, or more advanced like calling a web server to get a block of information. Under the covers, Node-RED builds a JavaScript application. It can even embed function blocks that are JavaScript subroutines.

Here is an example:



This flow will look for a start bit in the CLICK PLUS PLC (CLICK Read). When it turns “ON”, the C2-NRED will call a web site to get the weather forecast for the next week. That forecast has much more information than we need, so the “set msg.payload” node strips out only the predicted temps for the next 12 hours and writes those back to the PLC with the “CLICK Write” node.

All of this comes together because while ladder's non-sequential nature makes it great for real time control, it's harder to write programs for handling synchronous functions with long delays. Without waiting on a signal from the CLICK PLUS to check the weather, this flow could be shrunk to just 4 blocks.

Resources

JavaScript — There are thousands of websites and resources to help write JavaScript. One of the best is ChatGPT. Simply describe what you want the function to do. You'll have to test it and maybe clean it up, but it's surprisingly good at writing working code. These additional reference tools will provide a great start and, in most cases, provide all the information you need.

<https://www.w3schools.com/js/> — W3Schools is an amazing reference library with code examples and tools to test your scripts. You can Google whatever logic you want, add "w3schools" to the search and you'll find easy to follow examples.

<https://devguru.com/content/technologies/javascript/index.html> — DevGuru is more like a reference manual. It will list every command and includes useful examples for each command.

<https://jsonata.org/> — JSONata is a great tool for accessing more complex data in JavaScript. Node-RED uses this tool to help pass useful data from node to node.

Node-RED — the Node-red.org web site is a great place to look for prebuilt libraries, and a rich user community that will help you get started writing amazing Flows.

We recommend watching this YouTube playlist: [Node-RED Essentials 100](#). The videos are done by the developers of node-red. They're nice and short and to the point. You will understand a whole lot in about 1 hour. A small investment for a lot of gain.

Ladder — Automation Direct has tutorials on Ladder Logic and as always, we provide free software to help you test and get comfortable with writing Ladder Logic.

<https://library.automationdirect.com/understanding-ladder-logic/>

Data Structures

The first step to writing a program in any of these languages is understanding how they store and share data.

PLC – PLC data structures are simple data types. Each variable is a number with a prefix that specifies the datatype. You can read more in the [CLICK user guide](#).

Memory Address	Type	Range	Data Type	DataRange
X	Inputs	X001-X816	Bit	0 or 1
Y	Outputs	Y001-Y816	Bit	0 or 1
C	Internal Control Relay	C1-C2000	Bit	0 or 1
T	Timers	T1-T500	Bit	0 or 1
TD	Timer Current Value	TD1-TD500	Single word integer	-32,768 to 32,767
CT	Counters	CT1-CT250	Bit	0 or 1
CTD	Current Counter Value	CTD1-CTD250	Double word integer	-2,147,483,648 to 2,147,483,647
SC	System Control Relay	SC1-SC1000	Bit	0 or 1
DS	Small Integer	DS1-DS4500	Single word integer	-32,768 to 32,767
DD	Large Integer	DD1-DD1000	Double word integer	-2,147,483,648 to 2,147,483,647
DH	Data	DH1-DH500	Hex	0000h to FFFFh
DF	Floating point Number	DF1-DF500	Floating point	-3.4028235e38 to 3.4028235e38
XD	Input Register		Hex	0000h to FFFFh
YD	Output Register		Hex	0000h to FFFFh
SD	System Data Register	SD1-SD1000	Single word integer	-32,768 to 32,767
TXT	Text Data Register	TXT1-TXT1000	ASCII (7-bit)	Single ASCII Character

JavaScript/Node-RED – Since Node-RED is built on JavaScript, it uses the same data types as JavaScript. Variables can be the following:

Variable Type	Range
Boolean	0 or 1, True or False
Number	Any integer or floating point up to 15 digits
BigInt	Over 15 digits - Rarely used
String	An array of characters - Text
Object	A list of key: value pairs
Array	A list of any variable
Date	A date/time value

****Note** JavaScript datatypes are defined at runtime by the interpreter. They are not declared by the programmer. Any variable can change types on the fly just by assigning a new value.

Arrays and Objects are the most interesting variable types since they hold other variables.

An Array is a simple list bound by square brackets []. For example, vehicle = [“cars”, “trucks”, “boats”]

The name of the array is vehicle, it contains those 3 values. You can address it by using vehicle[1] and the result is “trucks”. Notice, the first element is vehicle[0] and the last is vehicle[2].

An Object is like an array, but instead of a simple list, you can name each of the values in the list. These are known as Key: Value pairs. Objects are bound by curly brackets {}.

For example, `person = {firstName:"John", lastName:"Doe", age:43}`

The name of the object is `person`. You can address it in two ways.

1. Key.Value e.g. `person.firstName` will return "John". This is called dot notation.
2. Key["Value"] e.g. `person["firstName"]` will also return John. This is called bracket notation.

Dot notation is more concise, but you're not allowed to use spaces in the Key. Bracket notation is useful if you want your keys to contain spaces. `Person.first Name` isn't a valid variable because of the space between first and Name, but `person["first Name"]` is fine.

When these data structures are combined, you get a structure called JSON – JavaScript Object Notation. This is where you have an object that contains a list of arrays or other objects. For example:

```
Company = {
  "employees":[
    {"firstName":"John", "lastName":"Doe"},
    {"firstName":"Anna", "lastName":"Smith"},
    {"firstName":"Peter", "lastName":"Jones"}
  ],
  "cars":[
    "Audi",
    "Volvo",
    "Ford"
  ]
}
```

This object "Company" contains two arrays, employees and cars. You can address them using dot or bracket notation. `Company.employees[1].firstName` returns "Anna" or `Company.cars[0]` returns "Audi". You can use the JSONata tool set (see reference tools earlier in this chapter) to help practice locating information in large JSON data sets.

As you start working with more data, JSON objects can get very large and complex. In fact some databases known as NoSQL databases use JSON objects to store massive datasets with billions of records. Many modern web sites make Application Programming Interfaces (APIs) available to access data sets. JSON is a standard format that allows web applications to share data using what are called RESTful web services. Amazon provides a detailed explanation here: [What is RESTful API? - RESTful API Explained - AWS \(amazon.com\)](https://aws.amazon.com/restful-api/). This is the real power of Node-Red when used with CLICK. It brings together real time control on the factory floor with business and operations data provided by modern web applications, then provides the tools to use both efficiently.

Node-Red also supports four types of variables:

1. The msg object – passes data between the nodes.
2. The context object -stores data for a node.
3. The Flow object – stores data for a flow.
4. The global object -stores data for the canvas.

Using these object inside a function node looks like this:

```
name =context.get("name"); //to retrieve a variable  
context.set("name",name); // to store a variable
```

Here is a full example script that create a counter which tells you how many times this specific node has been called:

```
var local=context.get('data') || {}; //get the value in data and assign it to local  
if (local.count===undefined)//test exists  
{  
    local.count=0;  
}  
local.count +=1;  
msg.payload="F2 "+msg.payload+" "+local.count;  
context.set('data',local);  
return msg;
```

The Flow object works the exact same way, but if you use that script in multiple functions, it would count how many times any of those function nodes have been called compared to the Context object which keeps a distinct count value for each function.

For more information check this article: [Storing Data in Node-Red Variables \(stevesnoderedguide.com\)](http://stevesnoderedguide.com)

Using CLICK with Node-Red

CLICK Read, CLICK Write, and CLICK System Info. The System info block is just a read with additional tools to understand what you are reading. All three of these blocks ONLY ACCEPT and PROVIDE Simple Arrays. You can pass in a list of Boolean, or numbers, but it must be in a simple array; [value1, value2, value3]. Even a single value must be passed between the CLICK and Node-RED as an array; [value].

Here again the JSONata tool and a JavaScript Function block are the two easiest ways to convert your information to an array. A command as simple as


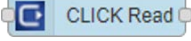
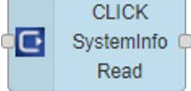
```
JavaScript Function
```

```
Age = [person.age];
```




Will convert the number in person.age to an array – simply because the brackets define the value as an array.

Working with Node-RED Custom nodes for CLICK

Inside Node-RED, there are 3 objects to share data with a CLICK PLUS PLC.

Node	Description
	This requires an array as input and writes a set number of values to the address specified.
	Accepts a starting memory address and a length. Populates an array starting with the first address. For example, X201, Len 4 will return an array [X201, X202, X203, X204].
	The same behaviors as CLICK Read, except this provides read-only access to SC bits and SD data registers. It will output an array of values. A register like RTC Day will output a simple Array containing a one-digit integer, while MAC ID will output an array with six 3-digit integers representing the MAC ID.

There are also 3 nodes which require specific configuration to work on the C2-NRED.

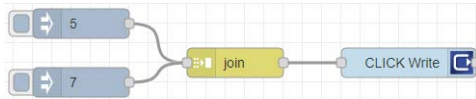
Node	Description
	This node writes a file to the C2-NRED filesystem. It has 1.5 GB available space (including your program). The file system has been locked down to prevent access or modification of any system files, but the following directory is available for user data: <code>/usr/local/nred-work/</code> In addition, you may write files to the SD Card if one has been inserted into the SD Card slot. The path to the SD card is: <code>/run/media/mmcblk0p1/</code> For example, to Write a file called Logs.txt, you would set the Filename property in the write file node to <code>/usr/local/nred-work/logs.txt</code>
	This node reads the data written to a file created by the write file node.
	This node will initiate a flow when data is written to a file by the write file node. It outputs the name of the file that was modified.

Here are some examples of data type conversions in Node-RED:

Converting data into an Array to Write to CLICK

Each of these CLICK operations accepts an array. To use them, you have to set the output of the prior operator to send an array, or you have to add a conversion element. The “Join” statement is generally the easiest, but you may use any of these functions:

- Join



Join allows a user to combine one or more inputs into an array which will be sent to the CLICK memory.

Mode: manual

Combine each: msg. payload

to create: an Array

Send the message:

- After a number of message parts: 2
- After a timeout following the first message: seconds
- After a message with the msg.complete property set

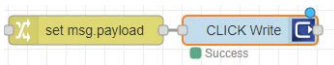
- Function



```
//msg.payload = “5” seconds. We want to send 5000ms to a timer value in the CLICK  
msg.payload=[ msg.payload*1000 ];  
return msg;  
//msg.payload now contains an array with a single value [5000]
```

The brackets convert the value to an array and even allow you to apply math inside the brackets. In this case, we’re converting the number of seconds to milliseconds before sending to the CLICK.

- Change



If the msg.payload is a constant, you can use the append function to append it to an array.

- The Change block enables the use of [JSONata](#) to perform complex JSON queries and transformations. The J: indicates a JSONata operation.

Rules

Set msg. payload

to the value J: \$append([], msg.payload)

- Allow fixed values to be Set. This sets 2 bits.

Set msg. payload

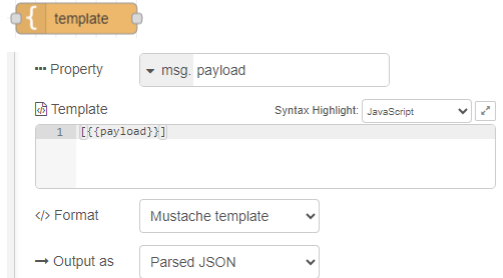
to the value {} [0,0]

- Fixed Payload

Send a JSON string that looks like an array and it will be treated like one.

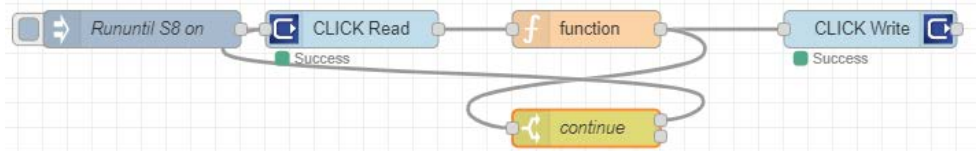
Payload

- Template



Performance

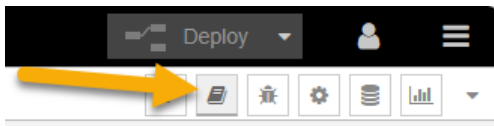
Testing basic performance with the following flow shows the time required to process a simple instruction in Node-RED:



In this flow, Node-RED reads a set of Boolean values from the CLICK, executes a simple function and based on the state of the inputs, writes to the outputs. The time elapsed between the read and the write can vary based on a number of factors but is typically 5-10ms if no other flows are running. Occasional background tasks can cause that time to be as high as 50ms, and depending where in the scan cycle the read occurs, the time can drop to as low as 1ms. Of course if the C2-NRED module is processing multiple jobs and has a lot going on, this timing can be higher than 50ms.

Default Nodes in C2-NRED

See the embedded documentation in Node-RED for help. Node-RED provides an embedded help system with detailed usage information for every Node, including the CLICK Nodes. Drag a Node onto the canvas and open the help from the right panel.

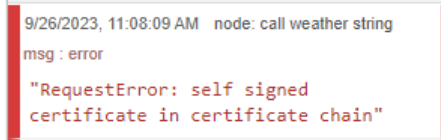


Troubleshooting

Node-RED application error:

In general, AutomationDirect cannot troubleshoot your specific application. There is a rich Node-RED community at the Node-RED.org website where you can find more resources and a supportive User-Forum. To help with your troubleshooting, however, we will include a list of common symptoms and resolutions here.

- **Symptom:** “RequestError: self-signed certificate in certificate chain”

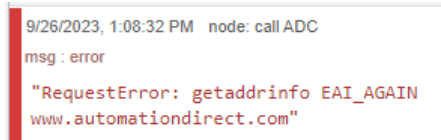


```
9/26/2023, 11:08:09 AM node: call weather string
msg : error
"RequestError: self signed
certificate in certificate chain"
```

Resolution:

C2-NRED has a problem with its time server. Sync the clocks with the CLICK PLUS Host.

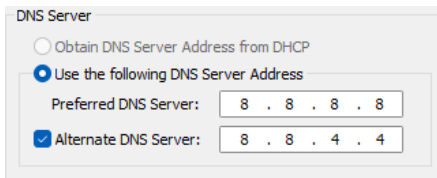
- **Symptom:** “RequestError: getaddrinfo EAI_AGAIN www...”



```
9/26/2023, 1:08:32 PM node: call ADC
msg : error
"RequestError: getaddrinfo EAI_AGAIN
www.automationdirect.com"
```

Resolution:

This means the C2-NRED cannot connect to the internet. Verify your IP address and that there is a path to the internet. It could also mean the C2-NRED DNS server isn't configured correctly. Open the C2-NRED Setup from your CLICK programming software and configure the DNS Server for the module. A Common DNS Server hosted by Google is 8.8.8.8 and the alternate address is 8.8.4.4. You may have to clear your DNS settings for this to take effect. Open the CMD prompt and enter `ipconfig /flushdns`.



- **Symptom:** Cannot access the Node-RED Dashboard

Resolution:

The Dashboard URL is case sensitive (i.e. `http://192.168.137.122:1880/ui/`).

Ensure you are not blocked by the IP Whitelist in the C2-NRED Module Configuration (Allow List).

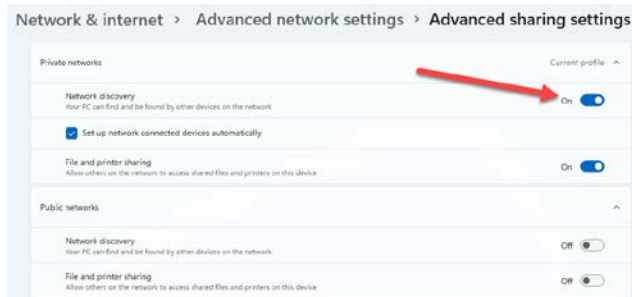
- **Symptom:** “ConnectionError: Failed to connect...”

```
11/1/2023, 11:59:15 AM node: 5c1936f0202fd51e  
msg: error  
"ConnectionError: Failed to connect to  
192.168.0.135:1433 in 15000ms"
```

Resolution:

If your network is not configured for “Network Discovery” which allows devices on the network to talk to each other, you may see this “ConnectionError”. Another possible cause is the device you are trying to connect to is not accepting connections.

To fix it, adjust the network settings:



- **Symptom:** Failed to install: [Module name]

In the debug panel, the following messages appear:

“Installation of module [module name] failed:”

“npm ERR! code CERT_NOT_YET_VALID...”

```
11/1/2023, 11:25:48 AM
msg : string(62)
"Installation of module node-red-contrib-mssql-
plus-box failed:"

11/1/2023, 11:25:48 AM
msg : string(42)
"-----"

11/1/2023, 11:25:48 AM
msg : string(320)
"npm ERR! code CERT_NOT_YET_VALID+npm ERR!
errno CERT_NOT_YET_VALID+npm ERR! request to
https://registry.npmjs.org/node-red-contrib-
mssql-plus-box failed, reason: certificate is
not yet valid+npm ERR! A complete log of this
run can be found in:+npm ERR!
/usr/local/nred/.npm/_logs/2000-01-
02T02_18_20_555Z-debug.log"

11/1/2023, 11:25:48 AM
msg : string(42)
"-----"

11/1/2023, 11:25:48 AM
msg : error
"Error: Install failed"
```

The log file will look like this:

```
2023-11-21T16:08:49.271Z Install : node-red-contrib-aedes 0.11.1
2000-01-05T02:23:26.325Z npm install --no-audit --no-update-notifier --no-fund --save --save-prefix=~ --production
--engine-strict node-red-contrib-aedes@0.11.1
2000-01-05T02:23:57.820Z [err] npm
2000-01-05T02:23:57.832Z [err] ERR! code CERT_NOT_YET_VALID
2000-01-05T02:23:57.832Z [err] npm ERR! errno CERT_NOT_YET_VALID
2000-01-05T02:23:57.944Z [err] npm
2000-01-05T02:23:57.946Z [err]
2000-01-05T02:23:57.947Z [err] ERR!
2000-01-05T02:23:57.952Z [err] request to https://registry.npmjs.org/node-red-contrib-aedes failed, reason: certificate
is not yet valid
2000-01-05T02:23:57.992Z [err]
2000-01-05T02:23:57.994Z [err] npm
2000-01-05T02:23:58.000Z [err] ERR! A complete log of this run can be found in:
2000-01-05T02:23:58.000Z [err] npm
2000-01-05T02:23:58.005Z [err] ERR! /usr/local/nred/.npm/_logs/2000-01-05T02_23_57_960Z-debug.log
2000-01-05T02:23:58.089Z rc=1
```

Resolution:

One possible cause is the RTC time in the CLICK PLUS PLC is not set correctly. Adjust the clock in the PLC to match your PC time. The C2-NRED module shares a clock with the PLC.

Fast blink on the C2-NRED Module:

The “OK LED” on the top of the C2-NRED module will blink slowly while it is starting up or after a reload of the CLICK ladder program. That is normal and does not indicate a problem. It will blink at about twice the normal rate to indicate the C2-NRED module is not enabled.

A Blinking Red Error LED indicates no project has been loaded into the PLC

C2-NRED Module Configuration (Slot0)

Network Address Configuration

- ☐ Obtain address from DHCP
- ☐ Use default fixed address
- ☒ Use the following IP address
 - IP Address: 192 . 168 . 0 . 146
 - Subnet Mask: 255 . 255 . 255 . 0
 - Default Gateway: 192 . 168 . 0 . 1

DNS Server

- ☐ Obtain DNS Server Address from DHCP
- ☒ Use the following DNS Server Address
 - Preferred DNS Server: 8 . 8 . 8 . 8
 - ☒ Alternate DNS Server: 8 . 8 . 4 . 4

☒ **Enable Node-RED** (indicated by a red arrow)

TCP Port Number (1-65535): 1880 (Typically Port No. 1880)

URL: http://192.168.0.146:1880

☐ **Enable HTTPS**

Key File:

Server Certificate File:

Security Options

- ☒ Enable Response to Ping from other devices
- ☒ Enable user to add nodes from Network
- ☐ Enable Allow List
 -

Port Management

-

Enable Node-RED Accounts

Count: 0/8

No.	Enable	Permission	User Name

Auto Logout Time(5-10,080min): 120 min
Default: 120min

Connecting your PLC over Wi-Fi:

CLICK PLUS PLCs have an extremely fast start up cycle. As a result, when you power on both the PLC and a router, the PLC will initialize before a router and may not be assigned an IP address. Power up your router first, then power up the PLC and other devices that use IP addresses. That way, the DHCP server in the router will assign addresses when those devices power up. If they power up before the router, they will get unexpected IP addresses (169.xxx.xxx.xxx). It can be more stable to disable the DHCP server in your router and assign fixed IP addresses to your components.

Be sure your connection to your Wi-Fi network is set to enable Network Discovery

On Windows 11, right click your network icon in the system tray to open– Network and Internet Settings

- Select your network connection Wi-Fi
- Select your Wi-Fi adapter properties and make sure it is set to “Private Network”
- Go back to Network and Internet
- Select Advanced network settings
- Select Advanced Sharing Settings
- Verify that Network Discovery is On

This configuration will ensure the C2-NRED module can connect to devices and your PC is discoverable on the network.

CLICK Project Loader

When you save your ladder project using the CLICK programming environment it creates a .ckp project that includes the following data:

- Ladder project
- CLICK project file (Rung Comments, and general project configuration)
- Initial data

The project loader also saves the following data to a .cklx file:

- Everything in the ckp file
- CLICK PLC Firmware (so a restore is guaranteed to have the correct firmware)
- C2-NRED configuration parameters (including IP address, allow/block lists, enable, usernames and passwords, certificate info)
- C2-NRED OS, Firmware and DB
- C2-NRED Program flows
- 3rd party Nodes

Tested 3rd party nodes for compatibility purposes:



NOTE: Testing indicates these will install correctly. ADC does not support or warrant any third party nodes.

NODE NAME	Purpose	Node 14.18.1
node-red-contrib-google-oauth2 (0.3.2)	Use OAuth2 to connect to Google APIs	Corrupts http request - do not install
node-red-contrib-aedes (MQTT Broker) (0.11.1)	Create a MQTT Broker so you don't need a separate server. This requires opening a TCP port to 1883	11 min to install
node-red-node-sqlite (1.1.0)	Install SQLite on device. (Gives a structured way to store/retrieve data collected by the PLC. Connect to this DB from a PC directly to query/retrieve the data)	Pre-Installed
node-red-contrib-cip-st-ethernet-ip (2.0.0 b3)	Create an Ethernet IP Client. Rockwell Addressing	Success
@serafintech/node-red-contrib-eip-io 1.2.1	Create an Ethernet IP Client using standard addressing	Success
node-red-contrib-airtable (0.1.2)	Connect to an Airtable Base	Success (2:20 install)
node-red-contrib-ip (1.0.1)	Return the public IP address	Success
node-red-contrib-hostip (0.0.3)	Return the local IP Address	Success
node-red-contrib-modbus tcp (1.2.3)	Communicate with a Modbus TCP Server	Success
Node-red-contrib-modbus (5.31.0)	Communicate with a Modbus Server	Success
node-red-contrib-mssql-plus-box (0.1.4)	Connect and run queries against a SQL or Azure SQL server	Success
node-red-contrib-alexa-remote2-v2 (3.10.5)	Connect to Alexa. This is the same as node-red-contrib-alexa-remote2 except it is fully configured in the Node-Red interface. This requires opening a TCP Port to 3456	Success (2:40 install)
node-red-contrib-bacnet (0.2.5)	Read and Write to a Bacnet Network	Success
node-red-contrib-fs-ops	Read and operate on the file system. *Note: Most of the folders on the C2-NRED module are locked for system stability and security. The user may access: C2-NRED memory at /usr/local/nred-work/ C2-NRED SD Card at /run/media/mmcblk0p1/	Success (1:50 install)
node-red-contrib-ui-upload	Implement a file upload control on the dashboard to get files onto C2-NRED	Success (~2 minute install)
node-red-contrib-murr-impact67pro-iolink-api	MurrElektronik	Success (~1:30 minute install)
node-red-node-serialport	Add Support for a serial port on C2-NRED	Fail - Not supported in hardware

Use the link below to identify and generate a link to the latest compatible version of any Node-RED module:

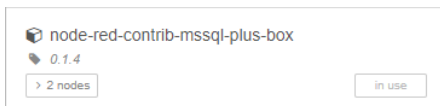
<https://automationdirect.github.io/CLICK-PLC/Node-RED/C2-NREDModuleVersionCheck/CompatibilityCheck.html>

This utility lets you enter a Node-RED module name and it will generate a download link for the best version compatible with the C2-NRED.

Getting Your Project Ready for Production

As you add external modules, follow these steps to create an offline backup of each module added to your project. This will speed up the process of restoring your Node-RED system after a firmware update, and it will protect your project from version changes if the module author modifies or deprecates functionality you are using.

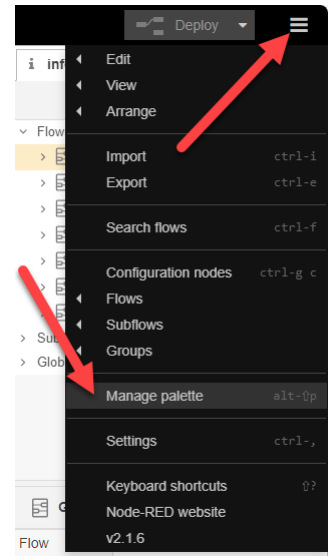
1. Backup all modules used in your project.
 - a. Find the name and version of the module you have added to your Node-RED project.
 - b. Open the “Hamburger Menu” at the top right of the screen and select “Manage Palette.”
 - c. You’ll see a list of modules included in your project.



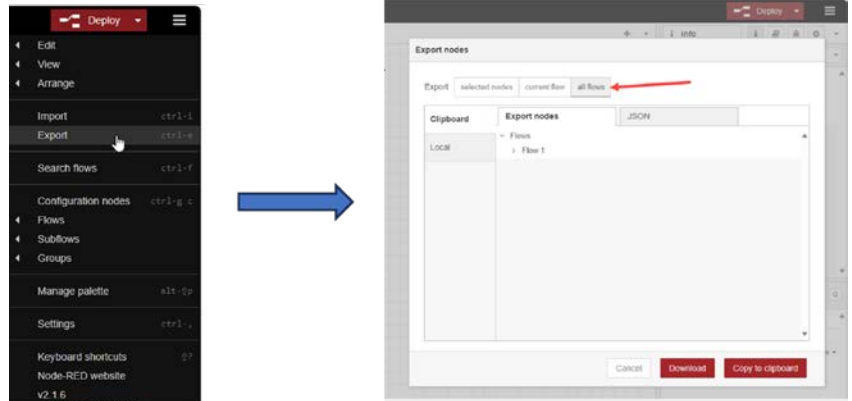
- d. Copy the name and version number of the module e.g. node-red-contrib-mssql-plus-box 0.1.4
 - e. Create a URL with the name and version – follow this pattern:
<https://registry.npmjs.org/node-red-contrib-mssql-plus-box/-/node-red-contrib-mssql-plus-box-0.1.4.tgz>

Where the **red** lettering is the name of the module, and the **blue** is the version number.

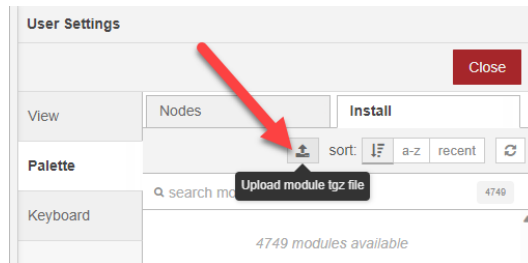
- f. Copy/Paste that URL into your browser. It will download the source file of that module to your default downloads folder. Copy the download to your project folder on your PC.



2. Make a backup of all Flows.
 - a. Open the “Hamburger Menu” at the top right of the screen and select Export.



- b. Select “all flows” and “Download”. Save the Flows to your project backup folder.
3. After a firmware update (BEFORE YOU RESTORE YOUR FLOWS), you’ll need to restore all modules you had added to your project. You can either redownload them from NPMJS or import them from your backup folder. It is recommended that you import modules from your backups to ensure you are using the same version with which you built and tested your project. You will likely need internet access to complete this step. Even if you have a backup of the modules used in your project, those modules MAY contain dependencies on additional libraries. When you install these (even from disk), Node-RED may call npm.js to load additional libraries.
 - a. To import from a backup location, open the “Hamburger Menu” and select “Manage Palette”.
 - b. Select the Install tab and click the “Import” button.



- c. Browse to your project folder where you have backed up your modules and reload each module.
4. AFTER YOU HAVE RESTORED 3rd PARTY NODES, Import your Flows.
 - a. Open the “Hamburger Menu” and select “Import”.
 - b. Click “select a file to import” and select the backup file with your flows. Click Open on the file selector and Import on the Import Nodes window.

C2-OPCUA OPC UA Server Module

Introduction

What is OPC UA

OPC UA stands for Open Platform Communications – Unified Architecture. It is not the same as the original OPC (OLE-Object Linking and Embedding for Process Control). In an OPC UA system, there are Servers that gather and collect data then serve it up to Clients that receive the data. OPC UA is a supervisory protocol. It is NOT meant for high-speed real time control systems. A fast OPC UA system will have a 100ms update rate. Machine logic should reside on the machine under control of the OPC UA servers. However, the OPC UA Client serves a very important role of providing global oversight—typically as part of a SCADA system, or system-wide monitoring system.

OPC UA (Open Platform Communications Unified Architecture) is widely used across industries that require reliable, secure, and standardized data exchange between devices, systems, and applications. Key industries and organizations that use OPC UA include:

1. **Manufacturing:** Factories and industrial automation systems, especially in sectors like automotive, aerospace, electronics, and pharmaceuticals, use OPC UA for machine-to-machine (M2M) communication, data exchange, and integration of devices on the factory floor.
2. **Oil and Gas:** Used for remote monitoring, data collection, and control of processes like drilling, extraction, and refining, allowing safe, real-time data exchange between field devices and control centers.
3. **Energy and Utilities:** Power plants, renewable energy sources (like wind and solar farms), and utility companies use OPC UA for integrating systems, monitoring operations, and optimizing energy management.
4. **Smart Cities:** OPC UA is often found in applications for building automation, traffic control, water management, and public safety, facilitating interoperability between various IoT devices and systems in smart infrastructure.
5. **Pharmaceuticals and Life Sciences:** Critical for ensuring compliance with regulatory standards by enabling precise monitoring and control of processes in drug manufacturing and other life sciences operations.
6. **Food and Beverage:** Provides standardization and real-time data exchange to improve efficiency, product quality, and traceability in production processes, from packaging to quality control.
7. **Transportation and Logistics:** Used for managing and tracking assets, monitoring vehicle data, and coordinating transportation networks, especially in smart warehouses and logistics hubs.
8. **Healthcare:** Hospitals and medical facilities use OPC UA in equipment and facility management systems for secure data transfer and integration of medical devices and systems.

As an example of where OPC UA fits, think of a railroad station. Within the station, train

location and switchgear status must be known. At a central hub the status of all systems in all stations must be visible. Due to the distance between stations, each station is connected to the public internet and the data is transmitted using TLS encryption to prevent man-in-the-middle attacks and to ensure data privacy. OPC UA offers both the performance and security necessary for this type of large, distributed monitoring and control system. Protocols such as Modbus, or EtherNet/IP are certainly fast enough but lack standard encryption. MQTTs is fast enough and secure, but since the packets are unstructured, it would take significant architectural design and any device added to the system would need custom programming to interpret or generate a data packet. OPC UA offers the security and structure that allows any OPC UA server to be easily added to the network.

There are two major benefits of using OPC UA over other communication protocols:

1. OPC UA is an Open Standard. This means that the protocol is free to use, and the specs have been published to the community. Because of this, it enables any device that is an OPC UA server to publish data, and any device that is an OPC UA client to read that data. The intent is to facilitate interconnectivity and tear down manufacturer specific walled gardens.
2. OPC UA is secure. When comparing OPC UA to other standards like Modbus and EtherNet/IP, OPC UA enables both encrypted data transmission AND client password requirements to access the data. If you look at Modbus, or EtherNet/IP, any device that you plug into your OT (Operations Technology) network can read the connection data for the network, then read and write data to your control system. While this was the generally accepted approach in the 90's, with the advent of global connections and nation state hackers, open network protocols are no longer the best method of establishing machine to machine communications.

What is the C2-OPCUA

The C2-OPCUA is a CLICK PLUS PLC Slot Module that is an OPC UA server. It can securely read all of the data registers in your CLICK PLC and provide access to those registers using the OPC UA communication standard. The C2-OPCUA Modules runs the [Embedded 2017 UA Server Profile](#). Version 1 of the C2-OPCUA does not support the full command set for OPC UA. This version supports data access features and security features; specifically authentication (either anonymous or with name and password) and security (sign and encrypt, sign only, or none). It does not support historical data access (historization), alarms and conditions (events), or UDP Pub/Sub features.

- ✓ Data Access: UA Part 8: DataAccess - 4 Concepts
- ✓ Encryption and Authentication: OPC UA Security architecture
- ✗ Historization: UA Part 11: Historical Access
- ✗ Alarms and Conditions: UA Part 9: Alarms and Conditions - 4 Concepts
- ✗ Pub/Sub: UA Part 14: PubSub - 4 Overview

System Requirements

- **Hardware:** The C2-OPCUA module is compatible with slot 0 or slot 1 of any CLICK PLUS PLC with one or two option slots.
- **Software:** Requires CLICK Programming software version 3.70 or above.
- **Network:** 10/100 Ethernet TCP/IP network

Installation



NOTE: If your CLICK PLUS PLC Firmware is below version 3.70, update the firmware in the PLC prior to installing the C2-OPCUA module.

Follow the instructions in the “Install the CLICK Programming Software” section of Chapter 1 to get the latest version of the CLICK Programming software installed on your PC. Refer to the CLICK PLUS application help files for more information:

[CLICK Help Version 3.70 - Introduction \(automationdirect.com\)](#)

After powering on and connecting to the CLICK PLUS PLC from the CLICK Programming Software, you must update the firmware to get both the CLICK PLUS PLC and the C2-OPCUA module up to the release version of firmware.

It is strongly advised to update the CLICK PLUS PLC firmware BEFORE installing the C2-OPCUA Module.

Follow these steps:

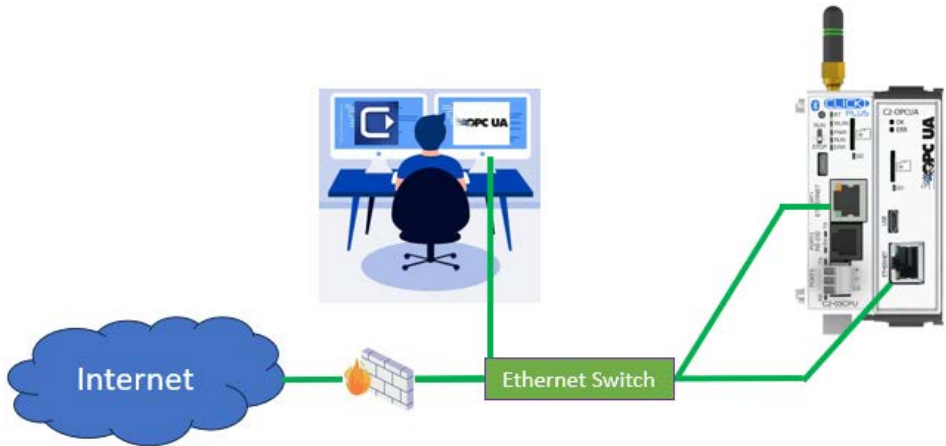
1. Ensure you have 24VDC wired to the CLICK PLUS PLC.
(Do not use the USB low power mode.)
2. BEFORE you install the C2-OPCUA module, connect to the CLICK PLUS PLC and update the firmware.
3. Power off the CPU.
4. Install the C2-OPCUA module, following the hardware installation instructions in the “Installing Option Slot Modules” section of Chapter 3.
5. Restore power.
6. Update the firmware again—this time it will update the C2-OPCUA firmware.



Communication

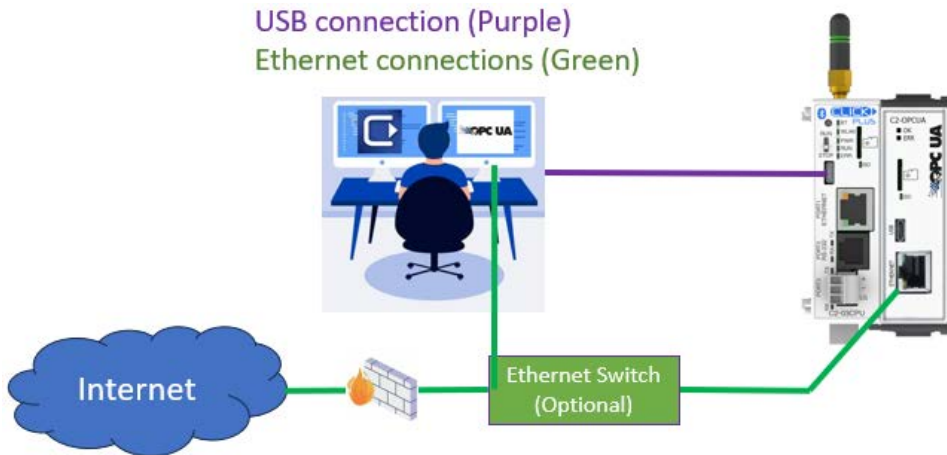
Once you have installed your C2-OPCUA module, you will need to connect to it. There are two primary connection methods:

Ethernet Only* — Connect your PC, and each Ethernet enabled module to a switch. All modules can be accessed over the same network.



*This is the recommended approach due to its simplicity and speed.

USB and Ethernet — Connect the C2-OPCUA module to your network switch and connect to the CPU using USB. This allows simple programming and configuration using USB, but the C2-OPCUA module must be connected to the network.



NOTE: The USB port on the C2-OPCUA module is used for backup/restore and factory default communications; it cannot be used for OPC UA communications.

Security

Before you get started configuring OPC UA, it is helpful to understand how the OPC UA security configuration works.

Definitions

Certificate — A luggage tag for your data. It contains information about the company and who owns the client or server. In the case of a web site, the certificate tells you who owns the web site(domain). It also contains information about how the server can encrypt data so the client and server can find a mutually agreeable method to encrypt communications.

Application Instance — An OPC UA Application (like the C2-OPCUA) installed on a single machine is called an Application Instance. Each instance must have its own Certificate which it uses to identify itself when connecting to other application instances. Each Application Instance has a unique URL.

Certificate Authority (CA) — Any publisher that generates certificates. A useful analogy is to compare certificates to a college ID. That ID is recognized by its issuer (the college) and entities on campus but not by businesses away from the college. On campus, people recognize the ID and trust it is valid. Away from campus, there is no trust, so the ID is not accepted. There are many levels of CAs ranging from Microsoft to any individual who wants to create them for their own system installations. All certificates created by any CA are valid as long as all entities in a transaction trust the CA that created those certificates.

Root-Certificate — A CA will generate a special type of certificate that can be used to authenticate all certificates they create. A root-certificate is used to verify other certificates are authentic, and unchanged.

Self-Signed Certificate — A Self-Signed Certificate is a certificate where the user is the Certificate Authority. These Certificates can be created by anyone, but they are only respected locally. Typically self-signed certificates are only useful in situations where the client and server are both internal and don't need a public CA to verify the identity of the owner of the Client or Server.

Authentication — Verify that messages received are from a known sender, and messages sent go to the intended recipient. There are various methods used to authenticate a sender. Username and password is one common way, but certificates can also be used to verify a sender. In that case, the recipient will use a root certificate from the sender's CA to validate the sender's certificate is authentic. The certificate itself will contain the senders identity.

Encryption — A mathematical method that uses a secret code known as a "key" to convert a plain text message to a coded string known as a Cipher. Encryption can use symmetric keys, where the sender and receiver use the same key to encrypt and decrypt messages. They can also use asymmetric encryption where the sender will use a public key to encode a message and the receiver will use a private key to decode the message.

Hash — A one way encryption. When a message is hashed, it cannot be decrypted. Every time the same message is hashed, it will generate an identical string. In secure communications, a hashed version of a message is included with the original message. The recipient will hash the original message and compare it to the hash sent. If they are not identical, the original message was altered and the message will be rejected.

Certificates provide two levels of security: authentication and encryption. In OPC UA, these are determined by a selection to either "Sign" (authenticate) or "Sign and Encrypt".

Below is a high level (simplified) description of the process.

1. The client makes a request to a server to start a conversation.
2. A server sends a certificate to the client. The certificate contains information about the owner of the server, and dates that the certificate is valid along with a public key.
3. The client and server then agree on an encryption method.
4. The client uses the public key from the server's certificate to encrypt messages sent to the server and sends its own certificate with the client's public key to the server.
5. The server will use the Client's public key to encrypt messages sent to the client.
6. All messages between devices are now encrypted.

Project Configuration

Once the firmware is up to date, you must enable the C2-NRED in order to begin. See the [C2-OPCUA Module Configuration](#) topic in the online help for details on configuring the module.

Time (RTC)

After the C2-OPCUA module is configured, BE SURE TO SYNC THE PLC TO YOUR PC CLOCK! The C2-OPCUA module shares a clock with your CLICK PLUS PLC. Be sure to update the time in the PLC. Click the PLC tab at the top of the CLICK programming software. Then click “Calendar/Clock Adjust”. If the time is wrong, it may impact the ability of the C2-OPCUA module to authenticate security certificates.

