Introduction

This document explains the mystery of master-to-master systems, and the information that must be understood to successfully deploy master-to-master systems. Most master-to-master systems can be successfully deployed using "route mode direct" and the appropriate topology. These two items are explained in detail in the subsections "Master routing" and "Topologies".

Master-to-Master

The functionality of master-to-master (M2M) consists of master routing and intersystem control. Master routing is the ability to route messages to any other master or device and is the foundation of all M2M functionality. Intersystem control allows a master, or its NetLinx program, to control and get status of any other device (or master) that is connected to any other master. The illustration below depicts a typical system of two interconnected NetLinx control systems with several devices connected to each one:
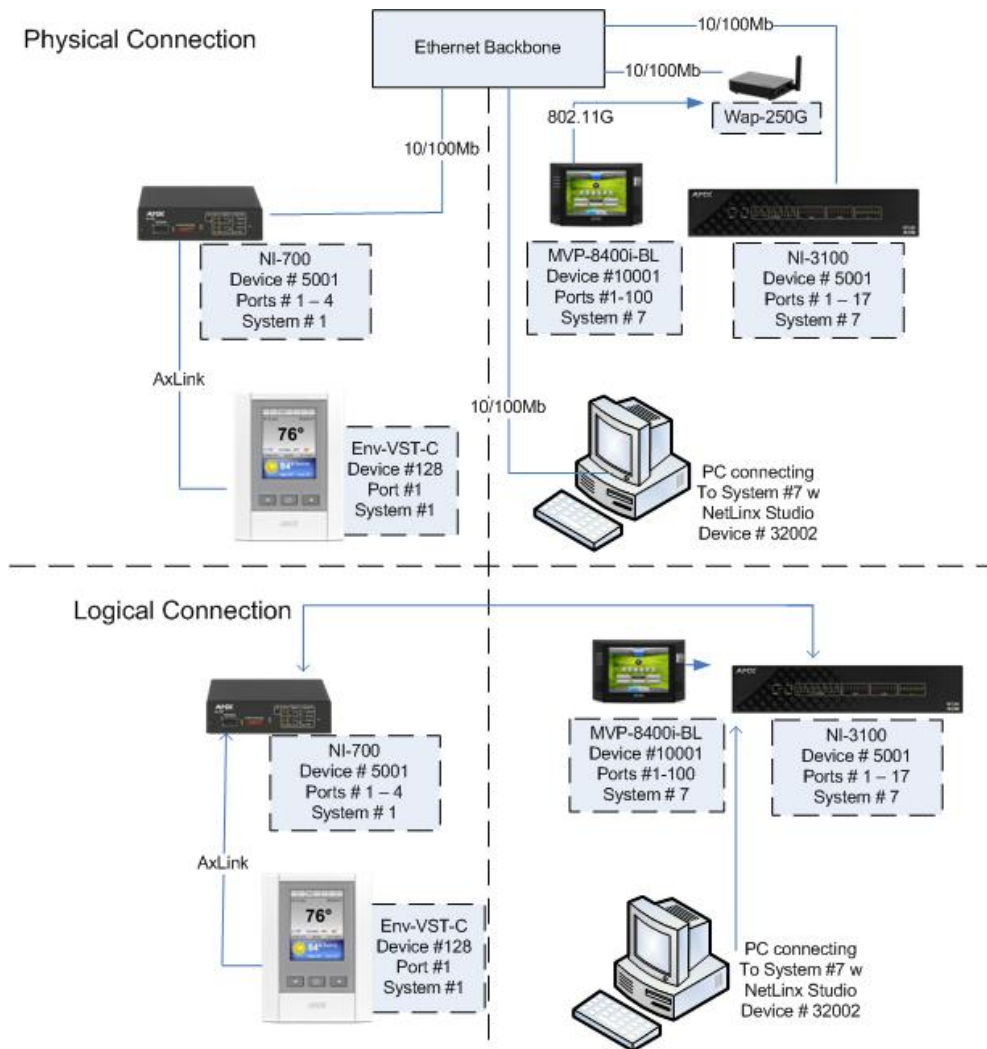

Illustration 1

The top portion of the illustration 1 shows the physical connections and the devices represented. The bottom portion shows the logical connections that have been assigned. In this example the NI-3100 will not communicate with the ENV-VST-C unless defined in the DEFINE_DEVICE section of its program code running on NI-3100 using the appropriate system number, for example 128:1:1. The first port on the MVP-8400i could be defined on system 1 using 10001:1:7, and on system 7 using 10001:1:7 or 10001:1:0.

Master routing

By design, all NetLinx masters do not automatically make a M2M connection with other NetLinx masters by virtue of being on the same network. The connection between them must be made intentionally by adding them to a list. This connection list is called the "URL List". The URL List on the NetLinx master is used to force the master to initiate a TCP connection to the specified URL/IP address*. Therefore, the first step in assembling a M2M system is to set unique system numbers on each master. Valid system numbers are 1 to 65535, system 0 is a wildcard referring to the local system and is used within DEFINE_DEVICE and NetLinx Studio connections. The next step is to configure the URL List in either of the masters, but not both, to point to the other master. For example, in *Illustration 1* NetLinx master system #1 could have its URL List configured with a single entry that contains the IP address of the NetLinx master system #7; this will establish a two-way connection. The system #7 master does not need to have a URL entry to communicate with system #1. If the system #7 master's URL List does contain the IP address for system #1 a routing loop will be created which will lead to problems.

*Note: Any TCP/IP device, including NetLinx masters, which utilize DHCP to obtain its TCP/IP configuration, are subject to having their IP address change at any time. Therefore, NetLinx master's IP address must be static unless the network supports Dynamic DNS AND a DHCP server capable of updating the DNS tables on behalf of the DHCP client. If a Dynamic DNS/DHCP server is available then the NetLinx master's host name may be used in the URL List.

| **OK**			**OK**			**WRONG**		
	System Number			System Number			System Number	
URL Entry	Master 1	Master 7	URL Entry	Master 1	Master 7	URL Entry	Master 1	Master 7
1	7		1		1	1	7	1

Once the systems are connected to each other they exchange routing information such that each master will learn about all the masters connected to each other. The implementation of master routing primarily involves the communication of routing tables between masters. The routing table is built using the entries within the local URL List, the DPS entries in the DEFINE_DEVICE section of the code, and from the routing tables exchanged between connected masters. Routing tables are exchanged between masters upon their initial connection and updates to the routing tables are exchanged periodically. Route table transmission has a certain amount of randomization built in to prevent flooding the network with routing table transmissions when a master reports online/offline. Each master in a network will add a minor random delay (1-5 seconds) so that they don't all transmit at the same time.

There is no fixed limit on the number of entries in a routing table. The number of routes is dependent on the number of systems in the network for which there is no set limit. The only limit is the memory space in each master to maintain all of the system information of the various systems throughout the network.

There are two route modes in which masters can be configured to share their routing table. The first and default is "normal", in this mode the master will share the entire routing table built from all interconnected masters. The second is "direct"; in this mode the master will share a routing table that only contains itself.

When using "direct" mode the master will only connect with the masters that are one hop away.

As a diagnostic aid, the "show route" command can be issued from a telnet session to show paths to other masters. Consider the following system of interconnected NetLinx masters:
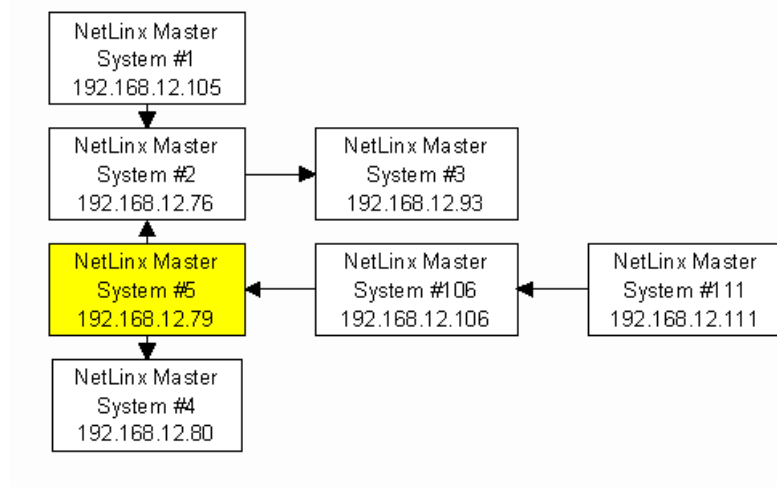
Illustration 2 – Arrows depict the direction of the initiated connection. I.e. System #1 initiated the connection to System #2 by having the IP address of System #2 in its URL List.

The following sample output is from a Telnet session connected to System #5. The connection of the NetLinx system is depicted in *Illustration 2*.

```
>show route

  Route Data:

  System Route  Metric  PhyAddress
  ------------------------------

    1       2       2       TCP Socket=18 IP=192.168.12.76 Index=3
    2       2       1       TCP Socket=18 IP=192.168.12.76 Index=3
    3       2       2       TCP Socket=18 IP=192.168.12.76 Index=3
    4       4       1       TCP Socket=16 IP=192.168.12.80 Index=1
->  5       5       0       Axlink
    106     106     1       TCP Socket=19 IP=192.168.12.106 Index=2
    111     106     2       TCP Socket=19 IP=192.168.12.106 Index=2
```

Route Data	The "Route Data:" indicates which routing mode the master is using. When the master is configured for route mode "normal", nothing additional will be presented. When the master is configured for route mode "direct", the following note will appear. "Direct Connect Only Mode"
->	The "->" to the left of system number 5 indicates that system number 5 is the local system (i.e. the system that the telnet session is connected to).
System column	The System column lists all of the systems that are in the master's routing table.
Route column	The Route column indicates which system number packets are to be routed to in order to get to their destination. For example, to send a message from system #5 to system #1 the message must be sent to/through system #2. You can see this visually in *Illustration 2* or by examining the Route entry for System #1 in the "show route" table.

Metric column	The Metric column indicates the number of system masters that the message must transverse/hop in order to get to its destination. For the example above, the metric is 2 because the message must enter system #2, then system #1. Note that a metric of 16 or "Dead" indicates a route that is expected but does not exist. Further, since the maximum usable metric is 15 there is a limit of 16 masters in the width plus height of the master topology (see section 2.1.1 Design Considerations and Constraints).
PhyAddress column	The PhyAddress column indicates the internal connection parameters used by the master to maintain the connection information.

"TCP Socket=" This is the IP socket that is used for this connection. Refer to "show TCP" for additional information.

"IP=" This is the IP address of the masters used for this connection point.

"Index=" This is the order in which the connection was established. When the master contains the entry in its URL List this often represents the order they were entered into the list. |

"Show Route" supports the "/v", verbose, parameter which will enable additional information about the routing table. This information is typically meaningful only to firmware engineering when diagnosing issues involving route table transmissions. The additional information available is described as:

Current Time	The number of milliseconds since boot.
Update Time	The milliseconds since boot when the next route table sync will occur.
Normal Update Time	The milliseconds since boot when the next route table sync will occur.
Triggered Update Time	The last time a triggered update occurred (ex. a new master came online, forcing a table update). If no triggered update has occurred, the field will say Max Time (effectively -1)
Timeout Time	The time that the next route table sync should have occurred by.
Next Update	"Normal" This indicates the next update will occur at the "Normal Update Time"
"Triggered" indicates the next update is occurring due to a triggered event.
Flags column	The Flags column indicates if the route to that master has "changed" during the last route reporting cycle. Upon the next reporting cycle with no new change, the field will be empty.

The end result of all this routing and connection data is that a device or master can communicate with other devices or masters regardless of the physical connection of the device. Note that masters may only be "connected" to each other via Ethernet/TCP/IP. As an example (using **Illustration 1**), NetLinx Studio is running on a PC that is connected to System #7 as device number 32002. The routing capabilities of the NetLinx master allow NetLinx Studio to download IR codes to the NXC-IRS4 (S=7 D=24), download a master firmware upgrade to NetLinx master #1, and download new touch panel pages to the touch panel on master #1. All of this is possible simply by having NetLinx Studio connected to a NetLinx master with M2M firmware.

Design Considerations, Constraints, and Topologies

Design Considerations

When designing a system that will utilize the M2M functionality, there are multiple points to consider.
The first thing to consider is the reason for using M2M. The most common reasons are:
- Expansion of a system to add device ports.
- Expansion of a system to an area the main system cannot reach.
- Sharing of processing load.
- Standalone capability of system areas.
- Isolation of areas for security reasons.

- Dedicate a master to common/shared devices located in a central location.
- Etc…
- A combination of the above.

The second thing to consider is the code requirements for each master:
- Masters that are only being used to add device ports must have an empty ".tkn" file loaded, otherwise the devices will not be accessible.
- Masters that are used to share the processing load or are intended to provide standalone capability must define its local devices and the specific remote devices needed on the other masters in DEFINE_DEVICE.
- Ports on remote devices declared in DEFINE_DEVICE must exist! (Example: adding touch panel port 80 when the panel file that has been loaded only specifies 20 causes errors in the negotiation)
- Events must be written for remote devices for the program to hear them. Writing events causes the master to negotiate for the transmission of these events over M2M (as reflected in SHOW NOTIFY)

The third thing to consider is the connection topology:
- Is there a main master who all other masters must connect with?
- Do all the masters need to talk to each other?
- Or is there some combination of the above?

Constraints

To properly configure the URL Lists in a multi-master system, there must be an understanding of 3 hard constraints.

The first constraint is the maximum number of 200 entries in a URL List. This limit although important will most likely never pertain as the second constraint is far more relevant.

The second constraint is the maximum number of 250 simultaneous TCP/IP connections supported by a single master. The maximum number of simultaneous TCP/IP ICSP (NetLinx device) connections supported by a single master is 200. The top ~25 of the remaining 50 are intended to be used for internal services i.e. ftp, telnet, http, etc… The next 25 are intended to be used for IP connections used in the NetLinx code via IP_CLIENT_OPEN, IP_SERVER_OPEN, and Duet modules. If there are more than 25 IP connections made from within the code they will utilize the required number of remaining 200 IP sockets which reduces the number of available socket connections and subsequently the number of available NetLinx device connections which will reduce the number of available entries within the URL List.

The third constraint is the routing metric limit of 15 usable hops on the topology of the interconnected NetLinx masters. While the limit of 15 hops may seem very limiting, this is not really the case if you carefully design the topology. Below is a visual of the 15 hop limit:

View of the 15 hop limit of Interconnected NetLinx masters.

Topologies

Chain Topology

This topology shows 16 masters connected to each other such that any master is routable to any other master.

The URL Lists would be configured like this. Please note the system number is being used for readability, the actual URL/IP address must be entered into the URL List.

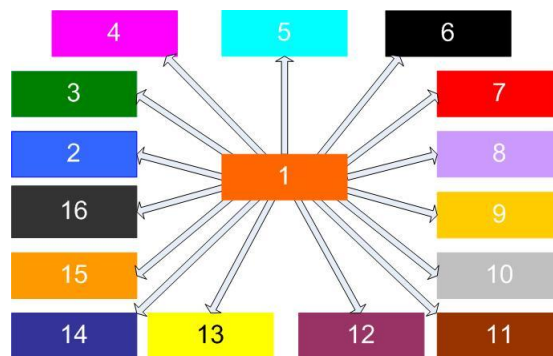| URL Entry	System Number															
	Master 1	Master 2	Master 3	Master 4	Master 5	Master 6	Master 7	Master 8	Master 9	Master 10	Master 11	Master 12	Master 13	Master 14	Master 15	Master 16
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	

Using this topology can be both network and processor intensive as a message from system 1 to a device/port on system 16 must be passed between the 14 masters. For example, a serial string sent from within the code on system 1 to 5001:1:16 will be passed to system 2, and then to 3, etc. until it reaches system 16. Therefore the single serial string results in 15 messages across the network.

With an IO pulse from system 1 to a port on system 16 the following occurs; an ON message is passed to system 2, then to 3, … until it reaches system 16, then the feedback on message sent back down the chain from system 16 to system 1, then a PUSH message from system 16 to system 1 following the same chain, then the OFF would be sent from system 1 to system 16, followed by a feedback off message from system 16 to system 1, then the RELEASE message from system 16 to system 1. Therefore that single pulse becomes 90 messages across the network.

Another drawback to this topology is if a single master loses communication than all subsequent masters will cease communicating.

Star Topology

The diagram below shows the M2M system configured in a star topology to take advantage of the fact that each NetLinx master supports multiple connections to masters:
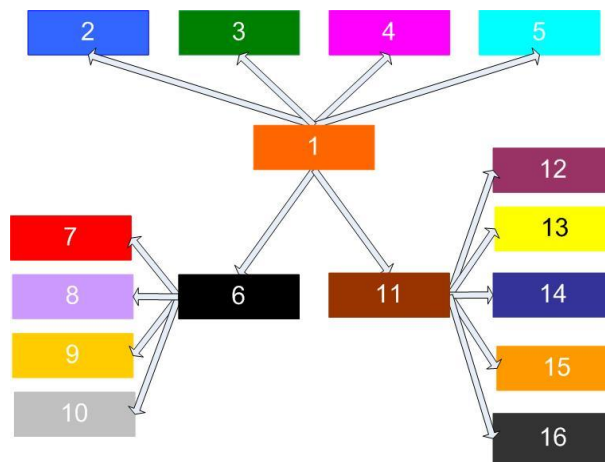

View of star topology.

The URL Lists would be configured like this. Please note the system number is being used for readability, the actual URL/IP address must be entered into the URL List.

URL Entry	System Number															
	Master 1	Master 2	Master 3	Master 4	Master 5	Master 6	Master 7	Master 8	Master 9	Master 10	Master 11	Master 12	Master 13	Master 14	Master 15	Master 16
1	2															
2	3															
3	4															
4	5															
5	6															
6	7															
7	8															
8	9															
9	10															
10	11															
11	12															
12	13															
13	14															
14	15															
15	16															

The largest drawback to this configuration is that if there is a communication issue with master 1 all other masters lose connection with each other.

Cluster Topology

Another possible connection topology is to establish communication hubs by combining the previously discussed topologies that optimize the traffic with adjacent masters but still allow connections to all other masters:

Clustered master Interconnection topology.

The URL Lists would be configured like this. Please note the system number is being used for readability, the actual URL/IP address must be entered into the URL List.
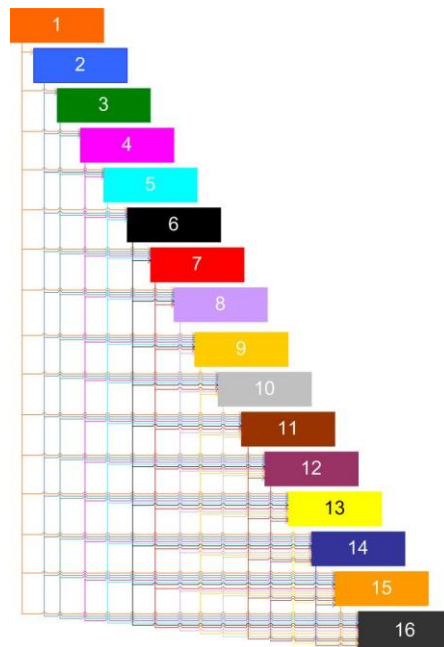
URL Entry	System Number															
	Master 1	Master 2	Master 3	Master 4	Master 5	Master 6	Master 7	Master 8	Master 9	Master 10	Master 11	Master 12	Master 13	Master 14	Master 15	Master 16
1	2					7					12					
2	3					8					13					
3	4					9					14					
4	5					10					15					
5	6										16					
6	11															

When determining the interconnection topology of many NetLinx masters, special consideration should be made to have masters that communicate a lot of information with each other to connect to each other. Thus if you have two systems that share devices, control, or information they should probably be near each other in the topology and not at opposite ends of the connection matrix where each message is forced to pass through several NetLinx masters.

Utilizing route mode direct will enable masters to isolate themselves from most traffic or to target the messages which will reduce network traffic and processor overhead.

Cascade Topology

The diagram below shows 16 masters connected to each other such that any master is routable to any other master using route mode direct.

Cascading master topology.

The URL Lists would be configured like this. Please note the system number is being used for readability, the actual URL/IP address must be entered into the URL List.
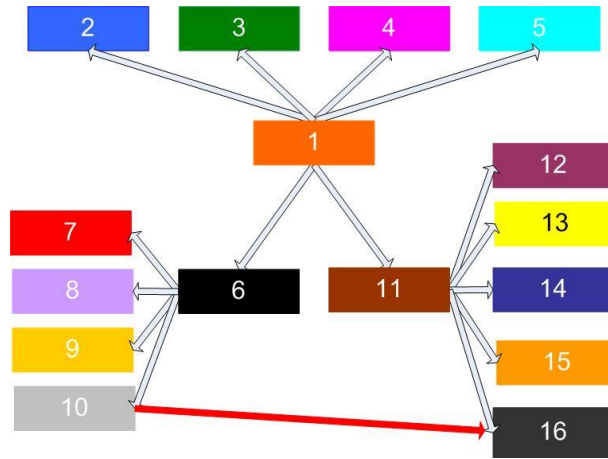
URL Entry	Master 1	Master 2	Master 3	Master 4	Master 5	Master 6	Master 7	Master 8	Master 9	Master 10	Master 11	Master 12	Master 13	Master 14	Master 15	Master 16
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	
2	3	4	5	6	7	8	9	10	11	12	13	14	15	16		
3	4	5	6	7	8	9	10	11	12	13	14	15	16			
4	5	6	7	8	9	10	11	12	13	14	15	16				
5	6	7	8	9	10	11	12	13	14	15	16					
6	7	8	9	10	11	12	13	14	15	16						
7	8	9	10	11	12	13	14	15	16							
8	9	10	11	12	13	14	15	16								
9	10	11	12	13	14	15	16									
10	11	12	13	14	15	16										
11	12	13	14	15	16											
12	13	14	15	16												
13	14	15	16													
14	15	16														
15	16															

System Number (column group header spanning Master 1–Master 16)

This topology has many advantages over the previously listed methods:

1. Each master is able to see all the other masters, with one hop
2. No passing of messages, which reduces the processing load on the master
3. Robust, if one master goes down communication is lost with only that master and the devices connected to it
4. Reduced network traffic

Cluster Topology modified

The diagram below uses the cluster concept and direct mode to link specific masters, yet remain isolated from other masters on the network.

Clustered master topology.

The URL Lists would be configured like this. Please note the system number is being used for readability, the actual URL/IP address must be entered into the URL List.

| URL Entry	Master 1	Master 2	Master 3	Master 4	Master 5	Master 6	Master 7	Master 8	Master 9	Master 10	Master 11	Master 12	Master 13	Master 14	Master 15	Master 16
												System Number				
1	2					7				16	12					
2	3					8					13					
3	4					9					14					
4	5					10					15					
5	6										16					
6	11															

Although this topology looks similar to the previous cluster topology by using route mode direct the communication connections are very specific. The masters will only be able to communicate with masters that have an arrow between them. For example the master with system 1 will only be able to communicate with masters 2, 3, 4, 5, 6, and 11, but will not connect with masters 7, 8, 9, 10, 12, 13, 14, 15, and 16. The connection, indicated with the red arrow, between master 10 and master 16 may appear to create a routing loop, but since the masters are configured to use route mode direct a loop is avoided. Master 10 will only be able to connect with masters 6 and 16.

The goal when using M2M is to minimize the amount of traffic between masters while providing the required functionality. Using route mode direct with the appropriate topology helps to accomplish this goal because it is the most efficient routing method since it will reduce network traffic and master processing of messages.

Configuring and Programming Master-to-Master systems

## Using NetLinx Studio with Master-to-Master systems

NetLinx Studio can be used to configure and diagnose M2M systems. After you have connected NetLinx Studio to the master, and you have configured the master with the proper "Network Address" information you will need to change the system number on the master using the window below.
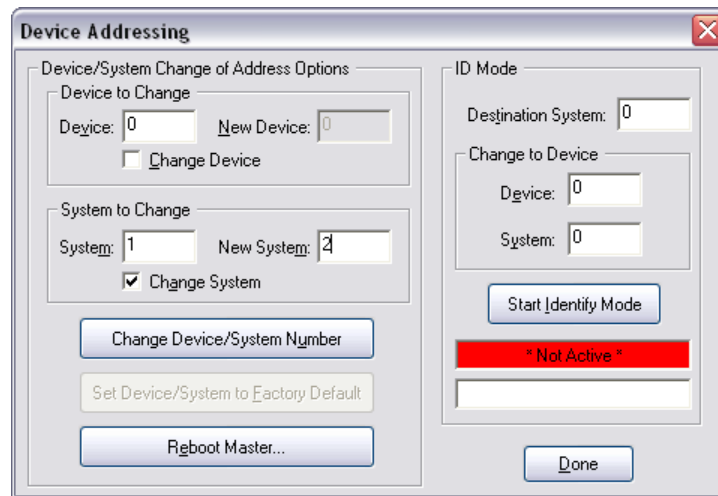
To access this window in NetLinx Studio click on the "Diagnostics" menu from the toolbar, then select "Device Addressing…", or from the "Diagnostics" toolbar select the "[IDPS icon]" icon.

Once the System Number has been changed the master must be rebooted for the change to go into effect.

The next step is to configure the URL List. This window can be accessed in NetLinx Studio click on the "Diagnostics" menu from the toolbar, then select "URL Listing…".
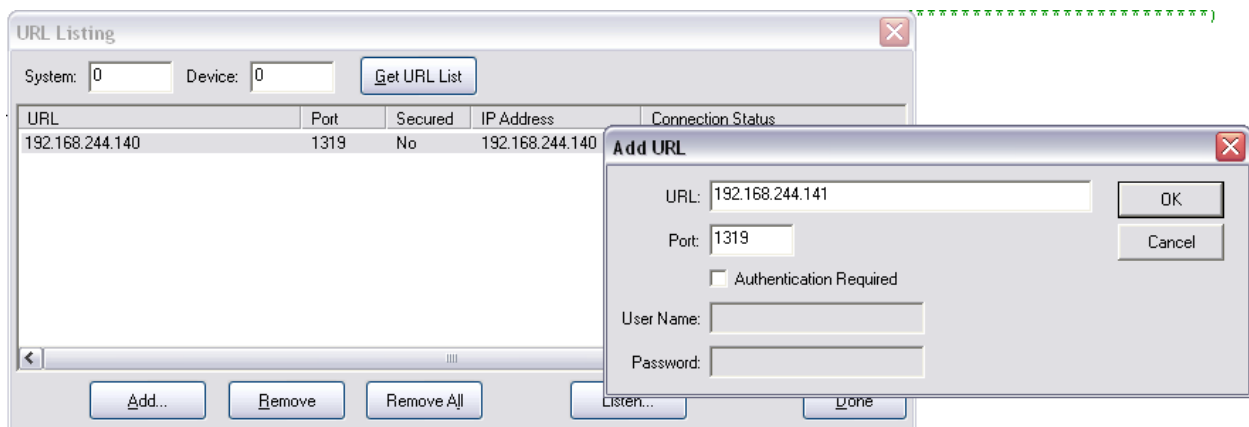
The "Get URL List" button will retrieve and display the URL List currently configured on the master which matches the "System" number specified, "0" indicates the master that NetLinx Studio used from the specified "Communication Settings". The URL List can be retrieved from other masters within the configured M2M topology. Each entry will report a "Connection Status" in the last column. The status values are "Looking up IP, Attempting Connection, and Connected"

The "Add" button will launch a window to add a new URL to the list using the appropriate authentication credentials, if required.

The "Remove" button will remove the currently selected entry from the URL List.

The "Remove All" button will remove all the entries from the URL List.

The "Listen" button will launch a window that will allow NetLinx Studio to listen for NetLinx masters on the local subnet using the port specified, default port 1319. From this view the options are to close the window or add the selected NetLinx master and its associated IP information to the "Add URL" window.

The masters and devices in a M2M system can be viewed using the "Refresh Network Online Tree" option within the "Online Tree". This function will run a recursive process that will connect to the master specified in the "Communication Settings" and gather information to populate the "Online Tree". If there are any other masters in the routing table, NetLinx Studio will then connect to those masters and get their information until the end of each branch reached.

There are some limitations in diagnosing or watching devices/ports in a M2M system using NetLinx Studio. For example, if NetLinx Studio is connected to master system 1, and a connection is established to master system 2, then only the devices on system 2 defined within the code of system 1 will be accessible to watch via "Asynchronous Notifications".

Using Telnet with Master-to-Master systems

Once the master's system number has been configured via NetLinx Studio, a telnet session can be used to configure and diagnose M2M systems. Note, when troubleshooting M2M systems NetLinx Studio and telnet connections to the master complement each other as the information from one application/interface may not be available from the other.

The first command to become familiar with is to set the routing mode on the master. The command is "route mode" followed by the desired mode, direct or normal. The routing mode on the master can be verified by sending the command "route mode" with no parameter, or with the command "show route". The "show route" command is described in a previous section of this document.

To view the entries in the URL List use the command "show url". To modify the entries in the URL List use the command "set url". Both of these commands will accept a <D:P:S> parameter to view or modify URL Lists on other masters.

The command "show system" will display all the systems and devices that are online and tracked in the device manager. The device manager tracks all devices defined in DEFINE_DEVICE or used in DEFINE_EVENT. The "show system" command supports two mutually exclusive parameters. The "<S>" parameter displays the devices on the specified system. For example, when connected to system 1 issue the command "show system 2" to display the devices on System 2. Using the "/min" parameter will limit the display to a minimal set of information.

There are two commands that are similar yet remain unique, they are "show remote" and "show notify". The "show remote" displays the devices on a remote master that are being monitored by the local master. The "show notify" displays the devices on the local master that are being monitored by a remote master. The outputs of both commands are structured similarly and are described below. The "show remote" was issued on system 1. The "show notify" was issued on system 16.

>show remote
Show Remote Device List

Device List of Remote Devices requested by this System

```
   Device  Port  System  Needs
   -------------------------------------------------------
   05001  00001  00016   Channels Commands Strings
   05001  00005  00016   Channels Commands
   33001  00001  00016   Channels Commands Strings Levels
```

>show notify
Show Notification List

Device Notification List of devices requested by other Systems

```
   Device:Port   System  Needs
   -------------------------------------------------------
   05001:00001  00001   Channels Commands Strings
   05001:00005  00001   Channels Commands
   33001:00001  00001   Channels Commands Strings Levels
```

Device column	The Device column lists the device that is being monitored.
Port column	The Port column lists the port on the device that is being monitored
System column	The System calling lists the system number that the device is connected to in the case of the "show remote". With "show notify" the system number that is watching the device will be listed.
Needs column	The Needs column contains the information that is being tracked. A device defined in "DEFINE_DEVICE" or used in "DEFINE_EVENT" will list the default needs "Channels Commands". The "Strings" need will be listed if the device is used in a "DATA_EVENT" or" CREATE_BUFFER". The "Levels" need will be listed if the device is used in a "LEVEL_EVENT" or "CREATE_LEVEL".

The command to view all of the TCP connections on a master is "show tcp". This command supports two parameters. The first parameter is "/v" which stands for verbose, this does not appear to change the results. The second parameter is "/all", this will display information about all 200 TCP/IP locations.

Control/NetLinx Language support

The features of control to M2M include channel control (PUSH/RELEASE/ON/OFF/TO), level control, send commands and send strings.

Channel controls allow one NetLinx master to PUSH/RELEASE a channel on a device of another system via the DO_PUSH/DO_RELEASE functions. Additionally, ON, OFF, TO, and feedback statements can control channels on devices of remote systems. If a channel has a characteristic modifier associated with it, that modifier still applies to the channel regardless of whether the channel is manipulated locally or remotely. For example, if a group of channels and variables is mutually exclusive then an ON to one of the channels will turn off all other channels and variables in the group prior to turning on the desired channel.

Levels, strings and commands are essentially forwarded to the destination device.

Note that control is not limited to physical devices and that NetLinx program defined virtual devices may also be manipulated by a remote system. This allows a local system to define a virtual device that can receive PUSH, RELEASE, ON, OFF, etc. and make programmatic decisions based upon that control.

Additionally, notification of control messages is not limited to "main line" functions like PUSH and RELEASE; rather all EVENT based code will operate normally regardless of the source of the original control message/function.

Design consideration and Constraints

In order to reference devices of other NetLinx systems, the devices MUST be defined in the DEFINE_DEVICE section of the NetLinx program. Conversely, only devices that are necessary should be placed in the DEFINE_DEVICE section to avoid any unnecessary network traffic between NetLinx masters.

DEFINE_LATCHING – A remote device's channel is not allowed in the DEFINE_LATCHING section.

DEFINE_MUTUALLY_EXCLUSIVE– A remote device's channel is not allowed in the DEFINE_MUTUALLY_EXCLUSIVE section.

DEFINE_TOGGLING – A remote device's channel is not allowed in the DEFINE_TOGGLING section.

The proper way to modify a channel's behavior is to use ON/OFF/TO/PULSE!

DEFINE_MODULE – As a guideline the best practice is to run a UI module on the master that the touch panel or keypad is connected to, and to run the COMM module on the master that the device is connected to. This practice should limit the number of messages across the network as the amount of messages between the UI and COMM modules is generally smaller than the amount of messages between the device and the COMM module.

Inter-master Variables

Inter-master variables are not implemented at this time. However the value of variables may be passed among the masters in the system using SEND_COMMAND or SEND_STRING to a common virtual device.

Using Virtual Devices as moderators

Virtual Devices may be used as moderators to share information between masters that may or may not be related to specific devices, like passing the values of a variable. They can also be used to minimize the network traffic by using them to distribute the information to multiple devices on other masters.

Code example of tracking online/offline state in a remote master

System 1 code:

```
DEFINE_DEVICE
SYSTEM4 = 33001:1:4


DEFINE_VARIABLE
INTEGER SYSTEM4_STATUS


DEFINE_EVENT
DATA_EVENT[SYSTEM4]
 {
   ONLINE:
   {
    SYSTEM4_STATUS = 1
   }
```

```
   OFFLINE:
  {
   SYSTEM4_STATUS = 0
   }
}
```

System 4 code:

```
DEFINE_DEVICE
SYSTEM4 = 33001:1:4
```

Modifying the URL List from within the NetLinx code

There may be times when viewing or changing the URL List from within the NetLinx code is desired. This can be accomplished using the following functions "GET_URL_LIST", "ADD_URL_ENTRY", and "DELETE_URL_ENTRY". Please refer to the NetLinx Keywords Help within NetLinx Studio for details and examples.

Master-to-Master processing queues and troubleshooting

The Route Manager queue is the message queue that receives any inbound route table messages from other masters. These messages are then processed by the Route Manager firmware to update its tables, refer to the section above labeled "Master Routing".

The Notification Manager queue is the message queue that receives notification requests for device state changes from a remote entity (ex. another master). In M2M communication, two connected masters do not blindly forward all local device state changes to the other master. Instead, they will only forward specifically requested state changes based on the remote master's needs as defined in the NetLinx code, refer to the telnet commands "show remote" and "show notify". The Notification manager queue receives these messages and then the Notification manager processes the requests and adds the information to its database, refer to the telnet commands "show remote" and "show notify", of "requested" state changes. When a state change occurs in the master, it compares the change to its database and if a remote master has requested notification of the change, it forwards the state change to the remote master.

When configuring M2M systems it may be necessary to alter the queue sizes of the above mentioned queues. This can be done using the following telnet commands "show buffers", "show max buffers", and "set queue size". To view the number of messages in each queue at a specific moment use the command "show buffers". To monitor the largest number of messages in each queue since the master has booted use the command "show max buffers".   Use the command "set queue size" to determine and set the upper limit on each queue. If the information returned from "show max buffers" is equal to the upper limit of the queue, it would be appropriate to increase the upper limit of the queue size.

General Master-to-Master Issues

When multiple masters exist within a large NetLinx installation the significance of the System number component cannot be over emphasized. Out of habit it is easy to ignore the system field within NetLinx Studio because its value has not meant anything in standalone systems. A significant source of technical support phone calls will be directly related to invalid or unintentionally incorrect settings of the system number, URL List, or route mode.

When NetLinx Studio connects to a single master, yet allows the user to access all other system masters it is inevitable that some confusion will occur. Therefore, it is a good idea to document each master's system number and the topology of the interconnections!