Crimson 3.0 N-View Driver

This document explains how to configure the N-View driver within Crimson 3.0 and thereby gain access to the diagnostic information transmitted by N-Tron switches via the N-View protocol. The reader is assumed to have a working knowledge of Crimson 3.0 and of the operation and features of N-Tron switches.

Installing the Driver

Select one of the available Ethernet protocols within Crimson, and use the Pick button to select the N-Tron N-View driver. The Ethernet port needs to be enabled, but if you will not be performing any UDP or TCP communication, you may leave the port set to 802.3 Mode to avoid the need for an IP address.

Creating Devices

The N-View driver receives multicast packets from N-Tron switches and stores the associated data in one of any number of switch devices created under the driver in the Communication window. Each device may be configured to accept data from any switch on the network, or to accept it only from a switch matching a given MAC address, IP address or unit name. Specific matching is used to more easily display information about a particular switch, while the Any Switch can be used to implement a network browser function.

Device Properties

Each device has the following properties:

- The *Receive From* property defines the matching method to be used to allocate N-View packets to this device. Selecting Any Switch will allow the device to receive information from whichever switch it sees first. If multiple devices are configured to Any Switch, the system will distribute the data between the devices, keeping a specific switch matched to a given device as long as the switch continues to send packets within the timeout window. The other options allow specific characteristics of the switch to be used to select which switch's data goes where.
- The *MAC Address* properties are used to define a MAC address when MAC matching is selected. The *MAC Block* selects the Ethernet MAC allocation block in which the address is located. At the moment, only the standard N-Tron 00:07:AF block is supported. The remaining three properties define the last three bytes of the MAC address, and should be entered in hexadecimal. Note that MAC matching may fail if a switch is replaced with a new unit that has a different MAC.
- The *IP Address* property is used to define an IP address when IP matching is selected. This functionality is not available with N-Tron's monitored switches, as these devices do not have their own IP address.
- The *Unit Name* property is used to define a unit name when name matching is selected. For switches that support user-configurable names, this matching method provides the most reliable approach to selecting a switch for display, as the reconfiguration that will typically be performed when a switch is replaced will most likely redefine the name to match that of the previous unit.
- The *Data Timeout* property is used to define the period of time after which the driver will consider a switch to be unavailable. The data associated with a switch will either be cleared to zeroes, or the device will be marked as offline such that a comms error will occur. The required behavior can be configured via the driver's *Unused Devices* property.

Data Representation

Each device exposes a number of data values via a set of 32-bit registers called D registers. Registers with an address less than 1000 refer to the state of the entire switch, while registers from 1000 onwards refer to the state Port 1 on the switch, registers from 2000 onwards refer to the state Port 2 on the switch and so on.

Switch Data

The following data refer to the state of the entire switch:

Register	Value
D0	1 if the switch data is valid or 0 otherwise.
D1	The number of ports detected on the switch.
D2	The IP address of the switch, if one is configured.
D3	The ring status flags: Bit 0 = Set if switch is N-Ring Manager. Bit 1 = Set if switch is N-Ring Member. Bit 3 and 2 = 00 = Ring Broken. Bit 3 and 2 = 01 = Ring Half-Broken Low. Bit 3 and 2 = 10 = Ring Half-Broken High. Bit 3 and 2 = 11 = Ring Healthy.
D10 – D15	The MAC address of the switch.
D50 – D65	The unit name of the switch.

Port Data

The following data refer to the state of port x:

Register	Value
Dx000	The chip ID of the port.
Dx001	The port ID of the port.
Dx002	The port status flags: Bit 7 = Set if port is high speed. Bit 6 = Set if port is full duplex. Bit 5 = Set if port is active. Bit 4 = Set if port is enabled. Bit 3 = Reserved. Bit 2 = Reserved. Bit 1 = Reserved. Bit 0 = Set if port is paused.
Dx003	The speed of the port in megabits per second.
Dx100 – Dx149	The port MIB as described below.

As noted above, x should be replaced with 1 to 8 depending on the port to be accessed.

Port MIB

The port MIB contains the following data:

Offset	Member Name	Notes
0	tx_octets	64 bits stored in two registers.
2	tx_drop_pkts	
3	tx_qos_pkts	
4	tx_broadcast_pkts	
5	tx_multicast_pkts	
6	tx_unicast_pkts	
7	tx_collisions	
8	tx_single_collision	
9	tx_multiple_collision	
10	tx_deferred_transmit	
11	tx_late_collision	
12	tx_excessive_collision	
13	tx_frame_in_disc	
14	tx_pause_pkts	
15	tx_qos_octets	64 bits stored in two registers.
17	rx_octets	64 bits stored in two registers.
19	rx_under_size_pkts	
20	rx_pause_pkts	
21	pkts_64_octets	
22	pkts_65to127_octets	
23	pkts_128to255_octets	
24	pkts_256to511_octets	
25	pkts_512to1023_octets	
26	pkts_1024to1522_octets	
27	rx_over_size_pkts	
28	rx_jabbers	
29	rx_alignment_errors	
30	rx_fcs_errors	
31	rx_good_octets	64 bits stored in two registers.
33	rx_drop_pkts	
34	rx_unicast_pkts	

35	**rx_multicast_pkts**	
36	**rx_broadcast_pkts**	
37	**rx_sa_changes**	
38	**rx_fragments**	
39	**rx_excess_size_disc**	
40	**rx_symbol_error**	
41	**rx_qos_pkts**	
42	**rx_qos_octets**	64 bits stored in two registers.

Mapping Tags

The most efficient way to access switch name data is to map a single string tag to D10, configuring a length of 16 and a packing mode of None. MIB data is best accessed by mapping an array of 50 entries to register Dx000 and then accessing the individual values using the array indices listed above. If 64-bit values are to be manipulated, the Crimson 64-bit math functions can be used, but only to limited effect. These larger values, which refer almost entirely to byte counts, are thus not particularly useful.