Distributed automation with 2pcs NetControl

Application Note

rev. 1.0

03.06.2022

CONTENTS

Document versions

Version	Date	Brief description of the changes introduced
1.00	06/03/2022	Initial version of the document

Legend:

ⓘ *The text contains additional and useful information that explains specific situations and features.*

⚡ *The text contains information of essential importance which you must get to know well!*

1. Introduction

In practice, it is often necessary to "transfer" a mechanical contact condition (eg button, alarm) to a remote location in the form of a mechanical state again (eg relay output). At close distances, this is easily done with the help of wires, wireless modules, etc. When the distances are long or it is difficult to run cables, this simple case turns out to be problematic.

The implementation of automation, in which the input sensors and the control output are distant from each other, is another aspect of the same problem. For example, a level sensor and a pump that are not located in the same place.

NetControl devices can help you here thanks to their network connectivity, which allowa a signal to be transported to a virtually unlimited distance.

The SNMPv1 protocol, which is supported as standard, is used for the transport protocol. SNMPv1 uses UDP for data transmission, which, although not as reliable as architecture, is sufficiently reliable at the current level of network systems.

In each device there is a possibility as a step from Macros to send a command "snmpset" with a value of 0 or 1 to the ioValue.0 objects on each of the channels of another ***NetControl*** .

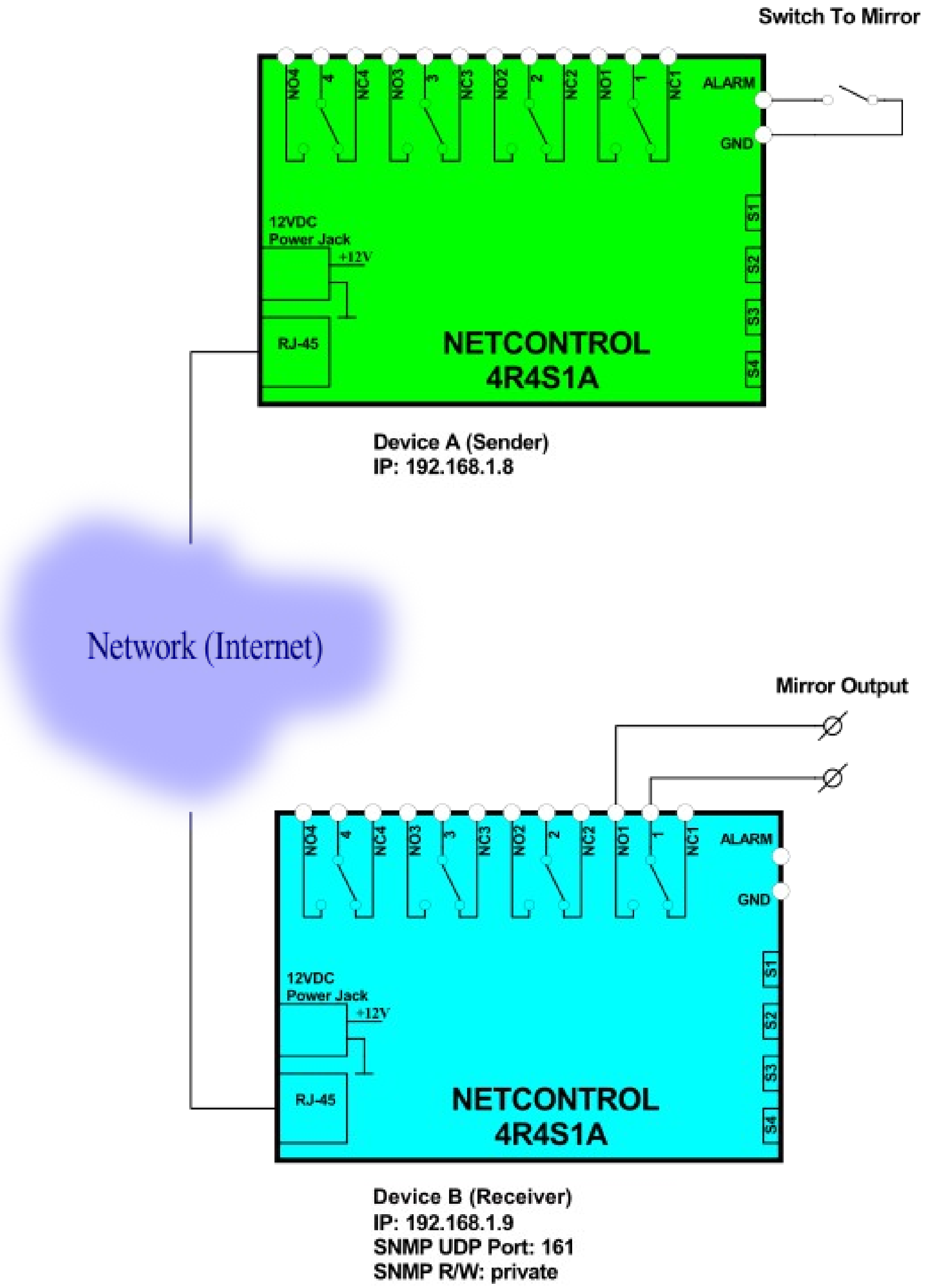
2. Transfer of input state with two NetControls

2.1. Concept diagram

The following figure shows the configuration in which the status of the alarm input ("Switch to mirror") of device "A" is to be "transferred" to the output Line1 ("Mirror Output") of device "B".

Both devices are in a local network and for this we have assigned them IP addresses from one range (network 255.255.255.0). Of course, all kinds of network settings are possible, as long as device A has access to the SNMP UDP port of device B.

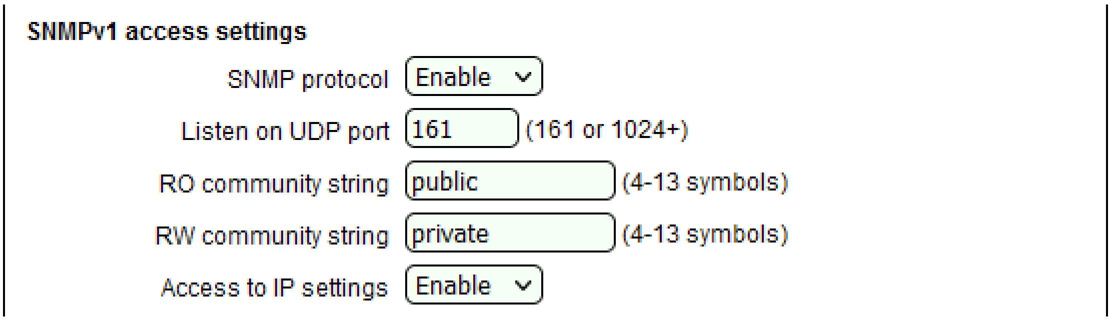
Mirror equation:
"Device A" Switch Open => "Device B" Output Open
"Device A" Switch Closed => "Device B" Output Closed



2.2. Device "B" settings

Device "B" does not need any specific settings. But you should note what its settings are in the "IP Settings / SNMPv1 access settings" section. The following picture shows the factory settings of each device, but the important thing is:

1. “SNMP protocol” = Enable
2. Pay attention to the UDP port and the RW password (since the device will receive the command 'snmpset' to change its output requires W (rite) access)



SNMPv1 access settings

SNMP protocol: Enable

Listen on UDP port: 161 (161 or 1024+)

RO community string: public (4-13 symbols)

RW community string: private (4-13 symbols)

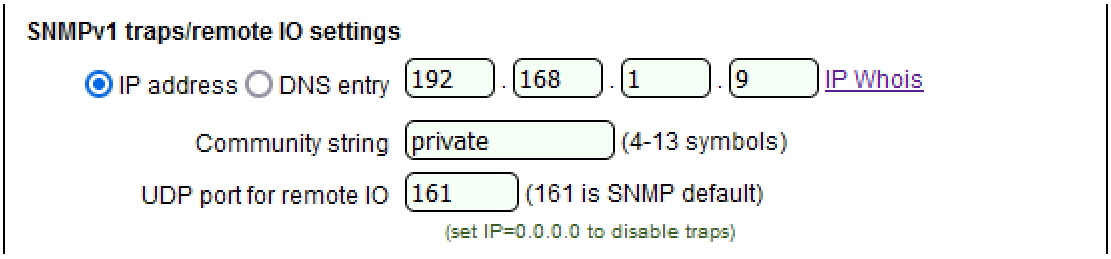
Access to IP settings: Enable

2.3. Device "A" settings

In this device it is necessary to make more settings in different sections.

2.3.1 SNMP settings for "Remote IO"

First you need to configure the parameters for access via SNMP to the other device (IP Settings / SNMPv1 traps / remote IO settings). Make sure that the three parameters correspond to those of device "B", which we specified in the previous section.



SNMPv1 traps/remote IO settings

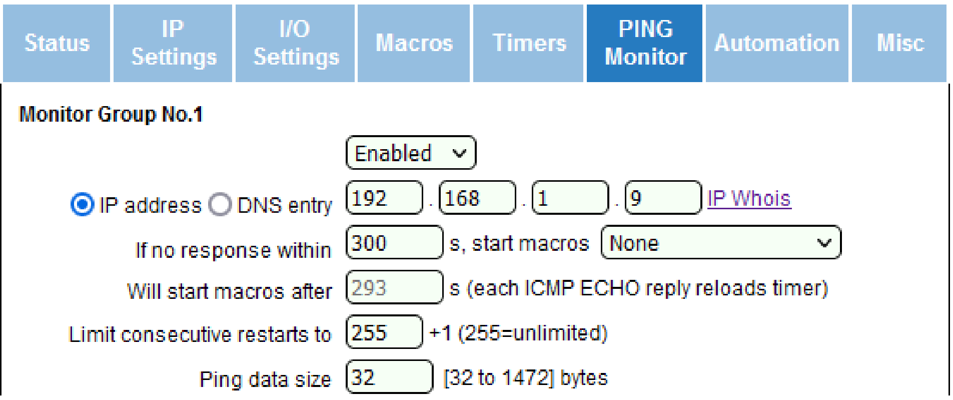
● IP address ○ DNS entry: 192 . 168 . 1 . 9 IP Whois

Community string: private (4-13 symbols)

UDP port for remote IO: 161 (161 is SNMP default)

(set IP=0.0.0.0 to disable traps)

If you do not have network diagnostic tools, you can check if device A "sees" B by setting the PING monitor block in "A" to PING the IP address of B. Check the "Will start macros after" field several times - if the value periodically returns to 300s - the device B corresponds to PING.



Monitor Group No.1

Enabled

● IP address ○ DNS entry: 192 . 168 . 1 . 9 IP Whois

If no response within: 300 s, start macros: None

Will start macros after: 293 s (each ICMP ECHO reply reloads timer)

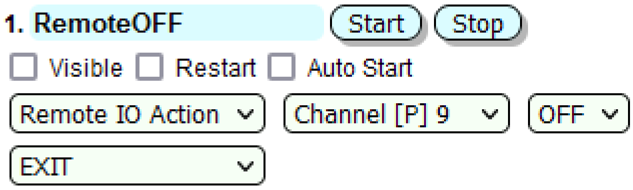
Limit consecutive restarts to: 255 +1 (255=unlimited)

Ping data size: 32 [32 to 1472] bytes

2.3.2 Creating macros to turn on/off Line 1 of "B"

We are now defining two macros that will issue a SNMPv1 command to the remote device "B" to change the state of Line 1. There are two macros, as we need to define separately to turn the output on and off (note that we define exactly macro 1 and 9 pair).

Macros 1



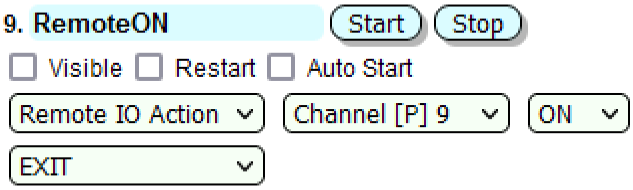
1. RemoteOFF

□ Visible □ Restart □ Auto Start

Remote IO Action | Channel [P] 9 | OFF

EXIT

Macros 9



9. RemoteON

□ Visible □ Restart □ Auto Start

Remote IO Action | Channel [P] 9 | ON

EXIT

At this stage you can perform a manual test with the Start buttons for both macros: you must register the change in the state of the relay of Line 1 on device "B". If this does not happen - there is a problem in the network and / or SNMP settings.

Pay attention to the channel number of the device "B" - [P] = 9. This is one of the settings that should be paid special attention to and most often the reason is that the function does not work properly. In the macro settings you have access to channels 1 to 24 - all channels supported by the software, but each model supports only PART of this complete set.

To determine for a specific model what are the values of [P] that correspond to Line 1, 2, etc. it is necessary to refer to the "User Manual" for the ***NetControl*** (available at www.ipnetcontrol.net). Find the section that describes the specific model - in our case "NetControl 4R4S1A" and the sub-section "5.1. Relationship between SNMP index and channel type". Part of the table is shown below for Line X outputs:

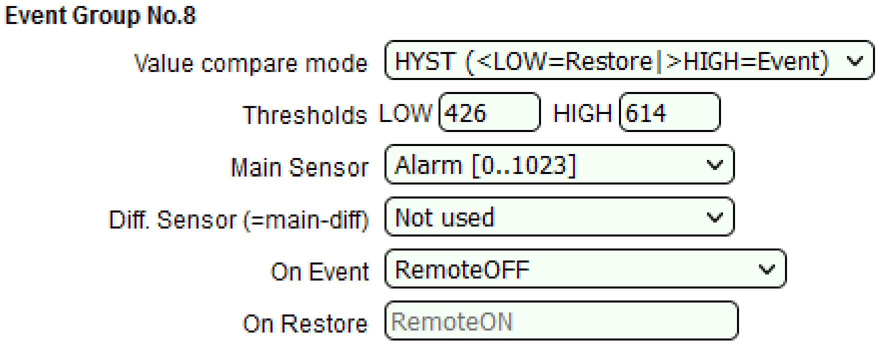
Channel name	Number [P]	Access ioValue [P]	Notes
Line1	9	R / W	0 (Low) = relay off (NC circuit CLOSED) 1 (High) = relay on (NC circuit is OPEN)
Line2	10	R / W	
Line3	11	R / W	
Line4	12	R / W	

From this table it can be seen that for this model Line 1, which is the goal of our macro, is numbered [P] = 9 - for this in the "Remote IO Action" of the macro select "Channel [P] 9".

2.3.3 Automation setting for alarm input

All you have to finally specify that the configured macros start when the status of the alarm input changes.

You choose a free "Event Group" from the "Automation" menu and set it up as follows (usually for alarm inputs there is such a group factory-defined and you only need to specify the macros for it):



Event Group No.8

Value compare mode: HYST (<LOW=Restore|>HIGH=Event)

Thresholds LOW 426 HIGH 614

Main Sensor: Alarm [0..1023]

Diff. Sensor (=main-diff): Not used

On Event: RemoteOFF

On Restore: RemoteON

The idea of this setting is that when the value of the alarm input (in practice it is an analog input with a value from 0 to 1023) exceeds 614 (this is equivalent to an **open** contact) the macro "Remote OFF" will be launched once. The relay of device "B" will switch off and its normally open contact on Line 1 will be **open** .

In the opposite direction: if the Alarm contact of device "A" is **closed** , the value of the channel will fall below 426 and this will lead to a one-time execution of the "On Restore" macro "Remote ON". It will send a command to "B" and it will **close** the normally open contact of Line 1.

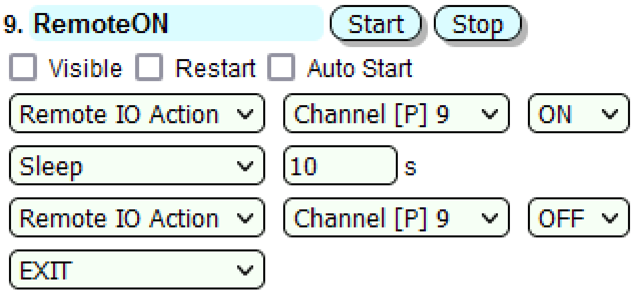
2.4. Additional options in the settings

You can modify the behavior of the overall operation of the two ***NetControl*** with different settings

In "A" you can specify a longer time on the input filter "Alarm" in the menu "IO Settings" - this will add a delay in the response of Automation pain and, accordingly, the transmission of the command to device "B".

In device "B" you can set Line 1 to be in "Impulse Output" mode (from the "IO Settings" menu). The macros can then be modified so that each time the alarm input (closed state and Remote ON macro) is activated, Line 1 output in "B" is triggered only for the time of "Impulse" and then switches itself off.

The same can be done in the "Remote ON" macro itself, without changing the settings for Line 1 to "B". However, in this case there is a risk (albeit theoretical) of data loss in the network and an undefined state for Line 1. The idea in this macro can also be used to duplicate the command to the remote device to increase reliability.



9. RemoteON

□ Visible □ Restart □ Auto Start

Remote IO Action | Channel [P] 9 | ON

Sleep | 10 s

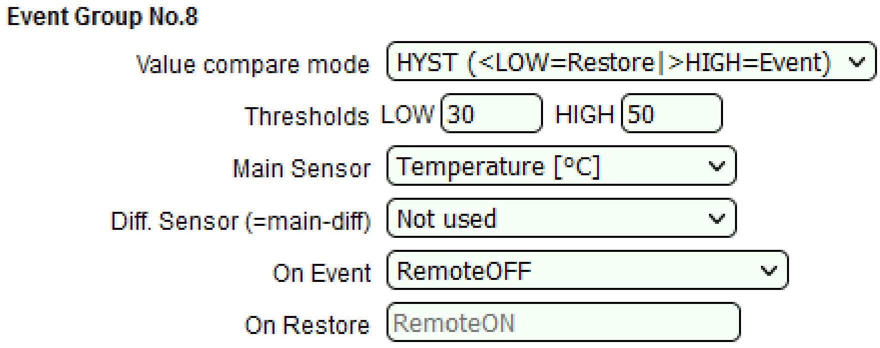
Remote IO Action | Channel [P] 9 | OFF

EXIT

3. Distributed thermostat with two devices

In the example from the previous section, instead of the alarm input of device "A", let's use the input for sensor S1, on which we place a temperature sensor TDS300. Our goal is to implement a thermostat, while the heater is connected to Line 1 of device "B".

All basic settings remain the same - IP, SNMP, Macros, only we need to define the required Automation block in device "A":



Event Group No.8

Value compare mode: HYST (<LOW=Restore|>HIGH=Event)

Thresholds LOW 30 HIGH 50

Main Sensor: Temperature [°C]

Diff. Sensor (=main-diff): Not used

On Event: RemoteOFF

On Restore: RemoteON

If the temperature exceeds 50°C, the macro "RemoteOFF" will be executed, which will send an SNMP command to device "B" and turn off the heater connected to its Line 1. Conversely, if the temperature drops below 30°C - executes macro "Remote ON" and the heater turns on.

It is easy to reverse the logic by swapping macro actions to control a fan rather than a heater.