

Mint^{Multi-Tasking} MT Application Note

AN00140-003: Using Wago CANOpen IO

Overview

Baldor now offers WAGO Modular IO Systems for use with Mint Motion Controllers using the CANOpen field bus interface. The WAGO units provide a very flexible solution to applications requiring extended or distributed IO.

A CANOpen "Coupler" is provided as the interface, to this an assortment of input and output "slices" can be added to configure a device providing exactly the appropriate type and quantity of IO for a given application.

Introduction to WAGO IO Devices

The CANOpen coupler is compatible with a whole range of individual IO modules. These include digital inputs, digital outputs, analogue inputs and outputs etc. Each module simply clips to the previous one producing a stack. Power and communications are then routed through contacts on the side of each "slice" and on to the next one.

This application note explains how to make a connection to the WAGO coupler and how to write simple MINT code read and write to the devices. The following table shows a list of available modules.

Part number	Description
750-338	CANopen Coupler
750-348	ECO CANopen Coupler
750-400	2-Channel Digital Input Module DC 24V
750-402	4-Channel Digital Input Module DC 24V
750-430	8-Channel Digital Input Module DC 24V
750-456	2-Channel Analog Input Module +/-10V
750-461	2-Channel Input Module for RTDs
750-468	4-Channel Analog Input Module 0-10V
750-480	2-Channel Analog Input Module 0-20 mA
750-492	2-Channel Analog Input Module 4-20 mA
750-501	2-Channel Digital Output Module DC 24V
750-504	4-Channel Digital Output Module DC 24V
750-530	8-Channel Digital Output Module DC 24V
750-550	2-Channel Analog Output Module 0-10V
750-552	2-Channel Analog Input Module 0-20 mA
750-554	2-Channel Analog Input Module 4-20 mA
750-556	2-Channel Analog Output Module +/-10V
750-600	End Module
750-601	Supply Module DC 24V
750-614	Field Side Connection Module 0-230V

Supported Controllers

- NextMove**PCI
- NextMove**PCI-2
- NextMove**BX^{II}
- NextMove**ST
- NextMove**ES
- NextMove**ESB
- NextMove**E100
- Microflex**E100
- MintDrive**^{II}
- Flex+Drive**^{II}

Note – **Flex+Drive^{II}** is a Slave only. There must be another master node in the Network e.g. **MintDrive^{II}**

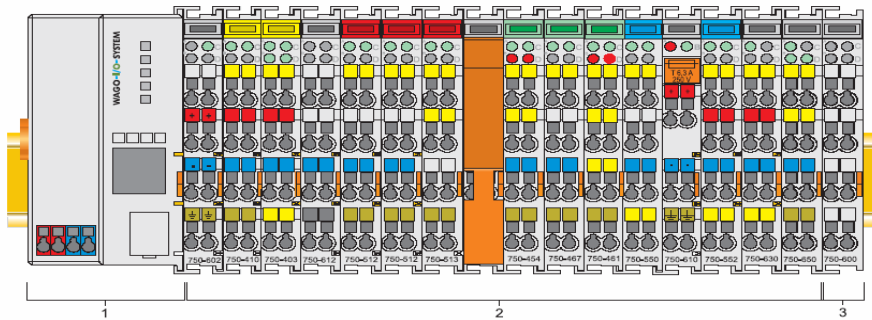
Relevant Keywords

- NODESCAN**
- CONNECT**
- REMOTEOUT**
- REMOTEOUTX**
- REMOTEIN**
- REMOTEINX**
- REMOTEDAC**
- REMOTEDACD**

Mint^{MT} Multi-Tasking Application Note

Configuring a WAGO CANOpen coupler

There are two essential settings that must be set on the WAGO units before communications can be established, these are the "baud rate" and the "node address". As with all networked systems the baud rate must be set the same on all nodes of the network and the node addresses must all be different. For a CANOpen network Node Address 1 is reserved for the Master Network Controller, in a Baldor system a Mint Controller will always be the CAN Master, so address 1 should not be used for the Wago Coupler.



The DIP switches are used for setting both the baud rate of the fieldbus coupler and the module ID (Node Address). 24Vdc must be applied to the power terminals of the couple unit during the configuration process.

Setting the Baud Rate

The bus coupler supports 9 different Baud rates set as a 4 bit code. The DIP switches are used to set the baud rate. To change the bus coupler into the Baud Rate configuration mode set all DIP switches off and power cycle the unit.

The currently set baud rate will be displayed by slow flashing of the top group of four LEDs (STOP, RUN, Tx- Overflow, Rx-Overflow), where STOP = Switch 1, RUN = Switch 2, Tx-Overflow = Switch 3 and Rx-Overflow = Switch 4.

The table below can be used to determine which baud rate is currently set for a given LED sequence, and which DIP switch settings are required to set a new baud rate. The default baud rate is 125kB/s, Baldor Mint controllers usually use 500kB/s for CANOpen devices.

Dip	Function	1 MBit	800 kB	500 kB	250 kB	125 kB	100 kB	50 kB	20 kB	10 kB	is displayed by LED
1 (LSB)	Baud rate	0	1	0	1	0	1	0	1	0	STOP
2	Baud rate	0	0	1	1	0	0	1	1	0	RUN
3	Baud rate	0	0	0	0	1	1	1	1	0	Tx-Overflow
4 (MSB)	Baud rate	0	0	0	0	0	0	0	0	1	Rx-Overflow
5											
6											
7											
8	Acceptance	'off' -> 'on' : Accepting the configuration settings									

Mint^{MT} Multi-Tasking Application Note

The new baud rate setting can now be selected using the DIP switches. To save the new configuration turn DIP8 to 'ON'. After saving the new baud rate is displayed by a steady light on the corresponding LEDs. (The baud rate of 1MBaud, is displayed by all 4 LEDs flashing)

NOTE - No data exchange via CAN is possible whilst setting the Baud Rate.

Setting the Node Address

Once the baud rate setting is completed, switch off the unit and set the DIP switches to the required Node ID address, this should be a number higher than 1 and not equal an address used by any other node connected on the same network.

The Node Address is set as a binary value on the DIP switches, i.e. the Node ID 2 is set by DIP2 = ON, module ID 8 by DIP4 = ON, etc. The node IDs must be in the range 2 to 127.

Connecting a Wago IO unit.

The CAN connector on the WAGO unit is a male 9 way D type, the connector on the Mint controller depends on the controller type, it is either an RJ45 or a male 9 way D Type. Some Mint controllers require that the isolated CAN inputs are powered externally this means 24V has to be supplied into the CAN cable from a separate source. The following table shows the connection requirements for each Mint controller type

Mint Controller	Connector Type	24V required.	Network Termination
Nextmove PCI	DB9 male	Link on breakout unit	Jumper available
Nextmove PCI-2	DB9 male	Link on breakout unit	Jumper available
Nextmove BX	RJ45	Jumper available	Jumper available
Nextmove ESB	RJ45	Yes, pins 4 & 5	Jumper available
Nextmove ES	RJ45	Yes, pins 4 & 5	Jumper available
Nextmove E100	DB9 male	Yes, pins 6 & 9	External resistor
Microflex E100	DB9 male	Yes, pins 6 & 9	External resistor
Mint Drive II	RJ45	No	Dip switch available

A Flex Plus II drive cannot be used on its own with CANOpen devices as it cannot be a CAN bus master.

Mint^{MT} Multi-Tasking Application Note

Diagram showing connectors to a DB9 connector

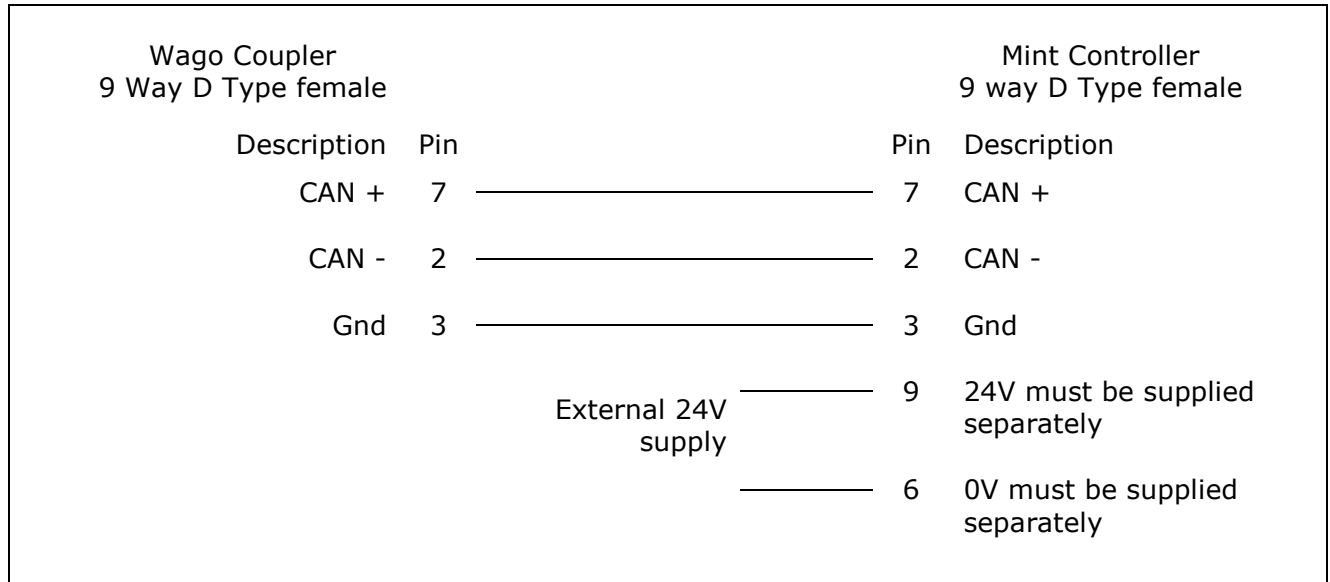
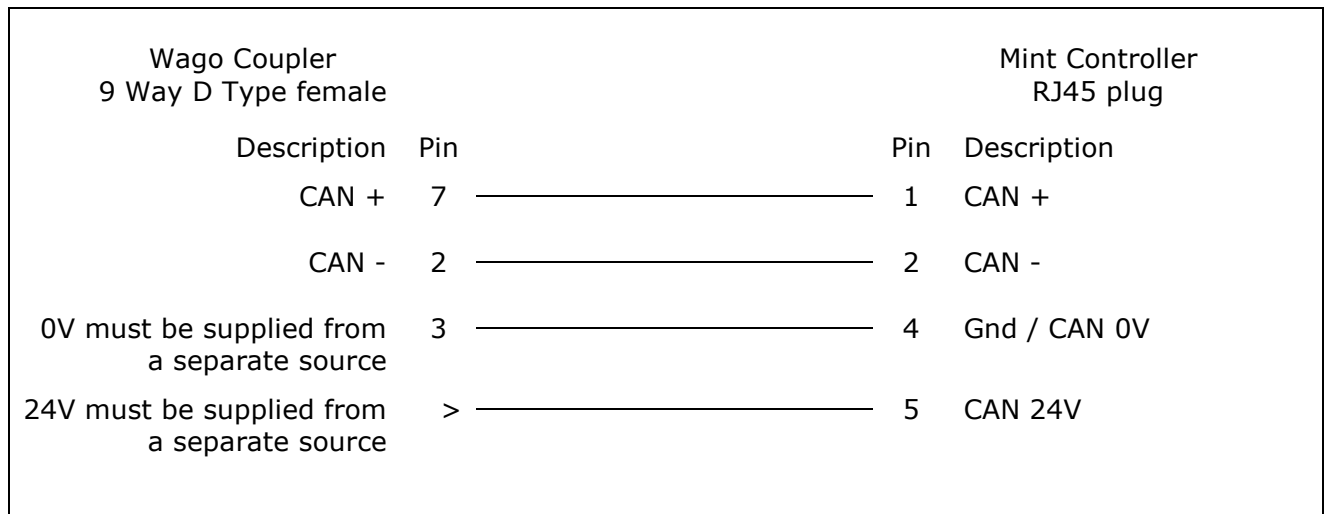


Diagram showing connections to an RJ45 connector

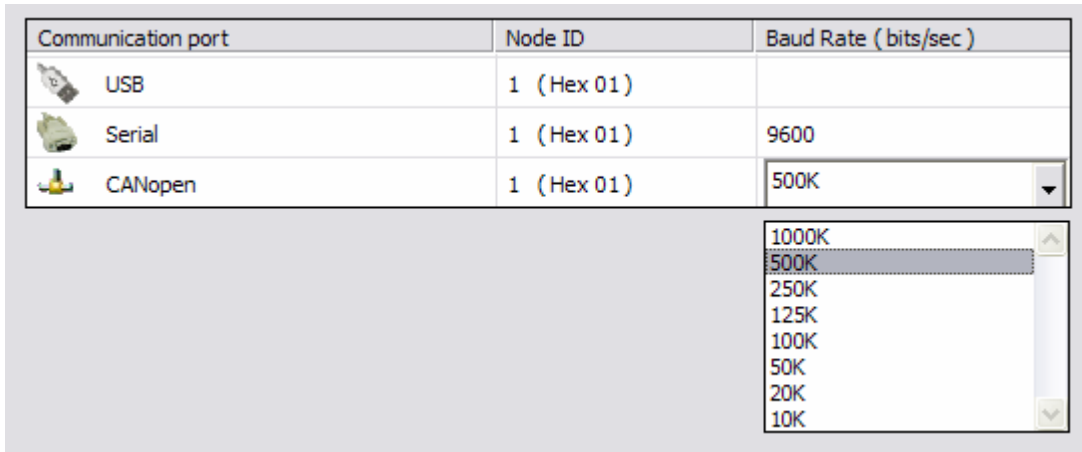


Note: A terminating resistor (120 ohm 0.5W) should be fitted across CAN+ and CAN- at each end of the CANopen network (see the previous connection table for details on which controllers provide an inbuilt method of including this resistor via Jumpers or Dip Switches and which controllers require the manual additional of an external resistor).

Mint^{MT} Multi-Tasking Application Note

Initializing Communications

It is possible to test the communication link using only Workbench v5 or v5.5. When connected to the Mint Controller make sure that the Node address for the CANOpen port is set to Node 1, it is also recommended that a baud rate of 500k bits/s is used. (The Node Address and baud rate can be set using the Network or Connectivity tool in the left hand tool bar of workbench).

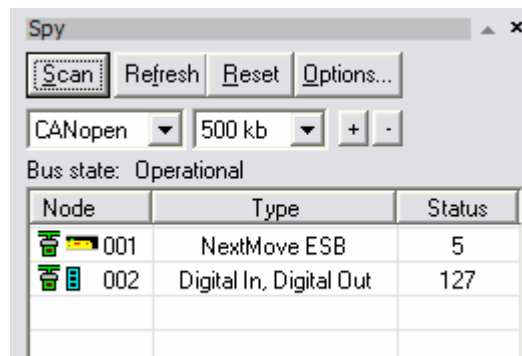


The Spy window on the right side of the Workbench screen has a CAN tab, select this to make a quick connection to the IO node.

Make sure that the CAN cable is connected and that the Wago coupler is powered correctly.

- Select the CANOpen port and 500k/s
- Press the Reset button
- Press the Scan button

When scanning is complete two nodes should be identified, the Mint controller (Node 1), and the Wago unit.



Mint^{Multi-Tasking} MT Application Note

Using Mint commands to read and write to a Wago unit

A remote node can also be detected using the following Mint commands

```
BUSRESET(1)  
NODESCAN(1,x)
```

(where x is the node address of the Wago unit)

Before reading or writing to the IO a software connection must be established, for this you must use the CONNECT command

```
CONNECT(1,1,x) = 1
```

(where x is the node address of the Wago unit)

At this point the status value of the Wago unit, as displayed in the CAN spy window, should have changed from 127 to 5, showing that the connection is now operational.

To create a MintMT program to establish a connection to a remote IO unit the following code is often used (assume the Wago unit is configured as Node 2)

```
BUSRESET(1)  
NODESCAN(1,2)  
TIME = 0  
PAUSE NODELIVE(1,2) or TIME>3000  
IF NODELIVE(1,2) = 1 THEN  
    CONNECT(1,1,2) = 1  
ELSE  
    PRINT "Connection to IO Unit failed"  
END  
END IF
```

Once a connection has been set up then the Mint remote IO commands can be used to read and write to the IO module.

To write to a bank of outputs use:	REMOTEOUT(1,2) = 3
To write to an individual output use:	REMOTEOUTX(1,2,0) = 1
To read from a bank of inputs use:	PRINT REMOTEIN(1,2)
To read from an individual input use:	PRINT REMOTEINX(1,2,0)
To write to an analog output channel use:	REMOTEDAC(1,2,0) = 1
To read from an analog input channel use:	PRINT REMOTEADC(1,2)
In all cases the first parameter is the bus, in this case CANOpen is always bus = 1. The second parameter is the IO Node address. The third parameter, where applicable, is the IO channel number. Data written to or read from analogue channels is as a percentage of maximum output.	

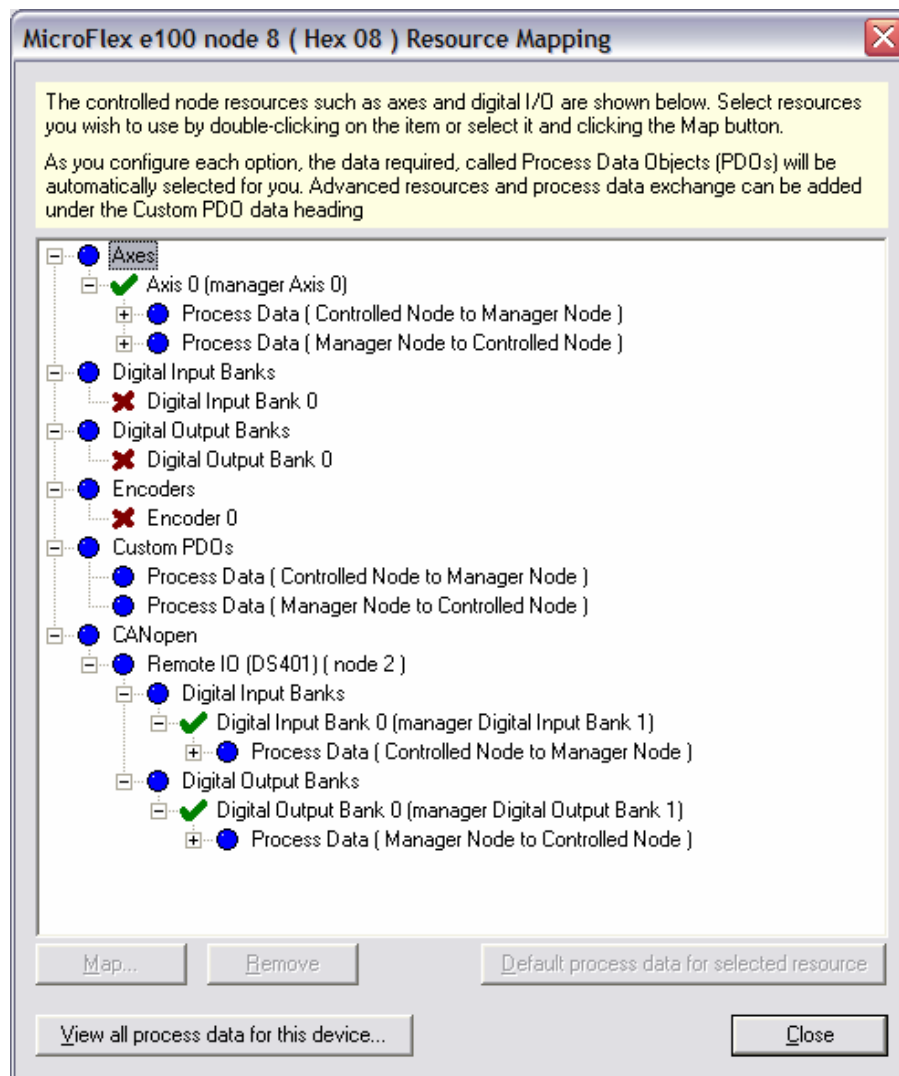
Further information on all of these commands can be found in the MintMT help file within Mint Workbench.

Mint^{MT} Multi-Tasking Application Note

Mapping a WAGO IO unit into an E100 system

Currently a Nextmove E100 does not support the mapping of remote IO but the controller will automatically scan the CAN port to detect any connected remote IO devices. It will then make the necessary connections. Therefore the NODESCAN and CONNECT instructions are not required in a Nextmove E100 controller. REMOTE_{xxx} commands shown above can be used immediately.

A Microflex E100 drive connected to a Nextmove E100 over EPL can be used to connect to Remote IO and then map the IO back to the Nextmove E100 so that it can be handled much like local IO. As with the Nextmove there is no requirement to include any connection instructions or settings into the Microflex drive. To map the IO back to the Nextmove Master node the appropriate settings need to be made in the Device Configuration File (DCF) of the Nextmove.



It is advised to use Mint Workbench v5.5 build 5513 or later is used for this operation (upgrades are available for download from www.baldormotion.com).

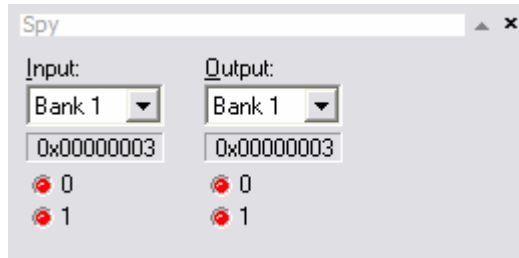
Mint^{Multi-Tasking} MT Application Note

During the creation of the DCF, using the System Configuration Wizard, the drive must be mapped to the Nextmove, as part of this process the CANOpen IO can also be mapped. Firstly by adding the Remote IO node under the CANOpen heading. Then by adding the relevant Input and Output banks to the new Remote IO node, specifying which local bank they will be referred to. Each bank can have a maximum of 32 channels and each node can have a maximum of 2 banks. This allows up to 64 channels per node.

Configuring Mapped IO

Once the appropriate mappings have been made the remotely connected IO unit can be read as if they are local IO wired directly into the Nextmove. This means that they can be configured as interrupts or axis IO such as LIMITFORWARDINPUT, STOPINPUT etc. (HOMEINPUT sensors must be wired directly to the local inputs on the MicroflexE100 drive.)

Remotely mapped IO will also appear in the Digital I/O page of Workbench, here it can be configured as active high or low or inputs can be set as edge triggered. This screen can also be used to define which inputs are used as Axis inputs.



The IO spy window, on the right of the Workbench screen, can be used to test the IO by selecting the appropriate bank. Bank 0 is the real local IO so the first remotely mapped IO is likely to be Bank 1. Outputs can be turned on and off by clicking on the output channel led, the state of inputs will be shown by the input led indicators.

When reading a remotely mapped IO channel using INX or OUTX commands the input or output number will be defined as:

IO Channel number = (Bank Number x 32) + IO Channel.

So for example for Input 5 on Bank 2 the input number will be (2 x 32) + 5 = 69

So to read the input the command is INX(69)