Motion Control Products

Application note
Connecting CP600 to motion products via Modbus TCP

AN00199

Rev D (EN)

Seamless high speed Ethernet communication between HMI and motion products

Introduction

The CP600 range of intelligent HMI panels is able to communicate with other peripherals (e.g. AC500 PLCs, ABB motion products) via a selection of communication protocols. This application note details how these HMIs can interface with ABB motion products via Modbus TCP. For general guidance on the use of Panel Builder 600 please refer to ABB manual 2CDC159007M0201.

To configure a CP600 HMI to communicate with an ABB motion control product via Modbus TCP requires Panel Builder 600 version 1.80.00.34 (or later). Please contact your local Sales office if you need to update your existing version of this software.

Integrated Modbus TCP support is available on the following active ABB motion control products:

- NextMove e100
- MicroFlex e190
- MotiFlex e180

Refer to application note AN00198 for further details on the operation of Modbus TCP on these products.

Modbus TCP uses an Ethernet-based physical medium. If the CP600 is to be connected to a NextMove e100 that is also the manager of an Ethernet Powerlink (EPL) real-time network then it is necessary to use one of the connected MicroFlex e190 or MotiFlex e180 drives as an EPL router between the HMI and e100 controller (see Application Note AN00247 for further details).. This ensures Modbus TCP packets are processed in the asynchronous portion of the Ethernet Powerlink communication cycle.

It is recommended that Workbench build 5854 (or later) is used to follow the content of this application note.

Motion product configuration

Integrated Modbus TCP operation is included with the following firmware versions:

- NextMove e100 : firmware version 5633 (or later)
- MicroFlex e190 : firmware version 5850 (or later)
- MotiFlex e180 : firmware version 5809 (or later)

NextMove e100 configuration

When using Modbus TCP the NextMove e100 uses the standard Modbus port 502. If the NextMove e100 is not operating as an EPL network master the IP address used by Modbus TCP is set by the rotary address switches on the front of the e100 product; 192.168.100.x (where x is the switch setting).
If the NextMove e100 is operating as an EPL master and a MicroFlex e190 or MotiFlex e180 is being used as an EPL router then Network Address Translation (NAT) can be used to access the NextMove e100 via an alternative IP address (see Application Note AN00247 for further details).

All Modbus parameters are configured via the MODBUSPARAMETER Mint keyword.
Before enabling Modbus operation it is necessary to set the correct byte and word order to suit the connected Modbus client (master), in this case the CP600 HMI, and to configure how Modbus registers in the received data packets are mapped to internal data areas in the Mint controller (see also application note AN00198).

As ABB PLC products use big endian byte order and big endian word order the Mint program needs to ensure the relevant Modbus parameters are set accordingly (via the MODBUSPARAMETER keyword, typically as part of the Mint Startup module) to ensure connectivity between the CP600 HMI and any other networked ABB Modbus TCP devices (e.g. an AC500 PLC):

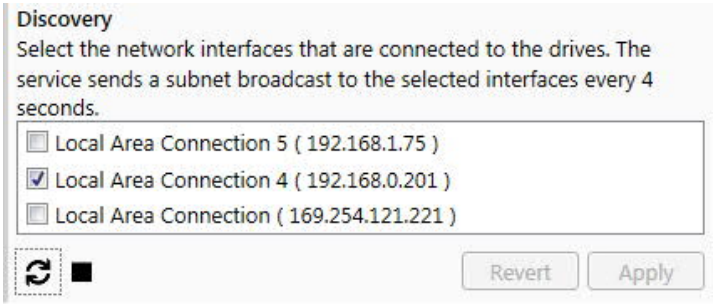
Example Mint code – Mint Modbus TCP slave connected to CP600 using Netdata array

```
ModbusParameter (_busETHERNET, _mpBYTE_ORDER)   = 0  'Use big endian byte order
ModbusParameter (_busETHERNET, _mpWORD_ORDER) = 0  'Use big endian word order
ModbusParameter (_busETHERNET, _mpREGISTER_MAPPING) = _rmNET_DATA
ModbusParameter (_busETHERNET, _mpENABLE) = 1
```

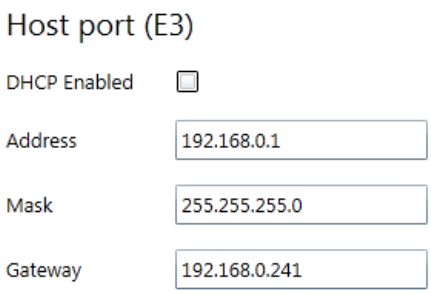
Most new applications using NextMove e100 are likely to utilise NetData as there are 1000 of these (as opposed to 99 Comms locations) and 32 NetData events (as opposed to only 10 Comms events).

MicroFlex e190 and MotiFlex e180 configuration

The IP address for these products is set via the 'Network' page as part of the 'Configuration' menu within Workbench.
Click 'Configuration' in the Workbench Toolbox and then select 'Upload configuration from controller' to retrieve the current communication configuration. Note that in order to display the Configuration screen it must be possible for the drive to be "discovered" by the Mint Sidebar/HTTP server…if Workbench will not display the Configuration screen check that the HTTP server is configured to scan the PC's network adaptor in use and that there are no firewalls or other anti-virus software preventing access to the drive.

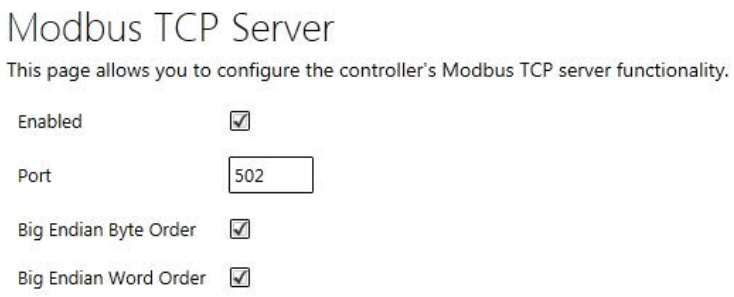


Once uploaded select 'Network' from the list of available options. The screenshot below shows the section from the 'Network' screen responsible for setting the drive's generic Ethernet address and subnet mask (the screenshot it taken from a MicroFlex e190 connection, this is identical for a MotiFlex e180).



If you need to edit/change the drive's IP address edit these entries as required and then click on 'Apply' at the bottom of the screen Note that the Gateway must use the same subnet (e.g. 192.168.0.x) as the drive (if you change any of the first 3 octets you will of course loose communications with the drive and will need to adjust your PC's network adaptor to be able to communicate again).

Now click on the 'Modbus TCP Server' option on the Configuration screen….



Ensure the 'Enabled' check box is ticked to allow the CP600 to communicate with the drive via Modbus TCP. The default port number for Modbus TCP is 502 (to match the CP600) so this should not be changed.

It is necessary to set the correct byte and word order to suit the connected Modbus client (master), in this case the CP600 HMI. As ABB PLC products use big endian byte order and big endian word order the 'Configure Communication Interfaces' page of the Configuration wizard provides settings (as shown above) to allow the MicroFlex e190 or MotiFlex e180 to operate this way.

Again click on 'Apply' to save any changes made (although the default values are set to allow communication with a CP600 to work 'out of the box' so you should find you don't have to make any changes).

MicroFlex e190 and MotiFlex e180 inherently map Modbus registers onto Net Data (so there is no requirement to use the MODBUSPARAMETER keyword to initialise this mapping). It is not possible to target the Comms array via Modbus when using these drives.

Register Mappings

All CP600 Modbus functions target a common data area in the Mint controller (as set by the Mint keyword ModbusParameter (_mpREGISTER_MAPPING) if using a NextMove e100 or fixed as Netdata when using a MicroFlex e190 or MotiFlex e180). These data areas have a fixed mapping with respect to the Modbus registers used by the HMI as shown by the table below (equivalent AC500 addresses are also shown for reference):

Server Modbus register	AC500 address		Mint Comms array (Comms=Real, Commsinteger = DWord)		Mint Netdata array (Netfloat = Real, Netinteger = DWord)	
0	%MW0.0	%MD0.0	Invalid	Invalid	Element 0 MSW	Element 0
1	%MW0.1		Invalid		Element 0 LSW	
2	%MW0.2	%MD0.1	Element 1 MSW	Element 1	Element 1 MSW	Element 1
3	%MW0.3		Element 1 LSW		Element 1 LSW	
4	%MW0.4	%MD0.2	Element 2 MSW	Element 2	Element 2 MSW	Element 2
5	%MW0.5		Element 2 LSW		Element 2 LSW	

…	---	---	---	---	---	---
198	%MW0.198	%MD0.99	Element 99 MSW	Element 99	Element 99 MSW	Element 99
199	%MW0.199		Element 99 LSW		Element 99 LSW	
200	%MW0.200	%MD0.100	Invalid	Invalid	Element 100 MSW	Element 100
201	%MW0.201		Invalid		Element 100 LSW	
202	%MW0.202	%MD0.101	Invalid	Invalid	Element 101 MSW	Element 101
203	%MW0.203		Invalid		Element 101 LSW	
…	---	---	---	---	---	---
1996	%MW0.1996	%MD0.998	Invalid	Invalid	Element 998 MSW	Element 998
1997	%MW0.1997		Invalid		Element 998 LSW	
1998	%MW0.1998	%MD0.999	Invalid	Invalid	Element 999 MSW	Element 999
1999	%MW0.1999		Invalid		Element 999 LSW	

LSW – Least Significant Word : MSW – Most Significant Word

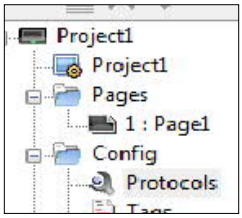
HMI IP address

The HMI needs to be setup to have an IP address where the first 3 octets match the network to which it will be connected (e.g. if the HMI is connecting directly to a MicroFlex e190 drive with IP address 192.168.0.1 then the HMI could be set to IP address 192.168.0.2, if the HMI is communicating with a NextMove e100 that is controlling local axes only and has an IP address of 192.168.100.5 then the HMI could be set to 192.168.100.67).

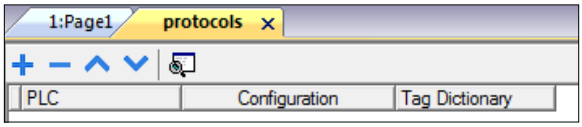
Power up the HMI and wait for it to finish booting. Press and hold any part of the HMI touch screen until the context menu appears. Select the ‘Show System Settings’ menu option. A rotating menu will appear, click on ‘Next’ or ‘Back’ until the ‘Network’ option is highlighted.
Click on the ‘Network’ button, select the ‘Specify an IP address’ option and use the on-screen keyboard to enter the required IP address. Set the subnet to 255.255.255.0. Click on the small ‘OK’ button at the top right corner of the dialog when complete.
Click on the ‘x’ button at the top right of the ‘System Settings’ screen to return the HMI to its operating screen.

HMI protocol configuration

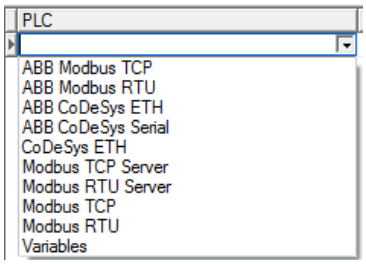
Having started Panel Builder 600 and launched a new project you will be presented with a blank screen representing the first page of your HMI application. At the left of the screen is the “ProjectView” which shows a tree structure of the available functions within the HMI project.
Expand the “Config” folder if necessary and then double-click the “Protocols” icon…



Now click on the “+” button to add a protocol to the HMI project…



A dropdown control appears under the PLC heading, click on this to display the list of available protocols…

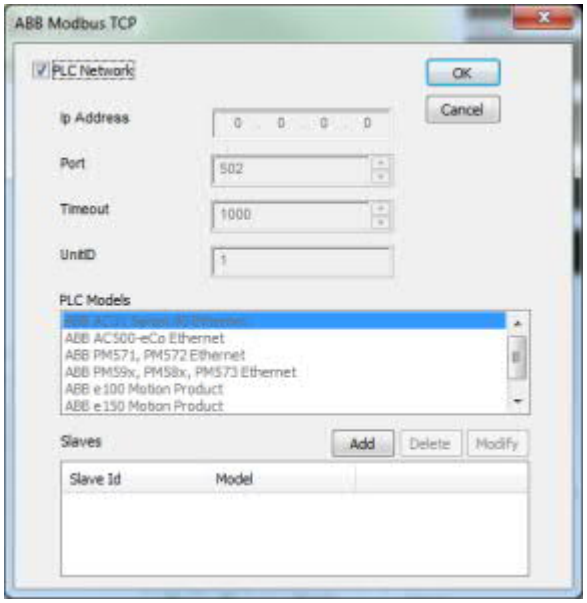


We need to select the ABB Modbus TCP protocol. This is a client (master) protocol that allows the CP600 HMI to communicate with both ABB PLCs and motion control products. The ABB versions of Modbus protocols differ from the generic Modbus protocols available in two ways:

1. The ABB Modbus protocols use big endian word order for data encoded into the Modbus data packets
2. The ABB Modbus protocols allow PLC specific addresses to be utilised in preference to generic Modbus registers

Note that there are no ABB specific versions of the Modbus Server protocols (so the CP600 HMI would usually be used as a client/master device when connecting to ABB products).
Having selected ABB Modbus TCP the software will now ask us to configure the connected devices…

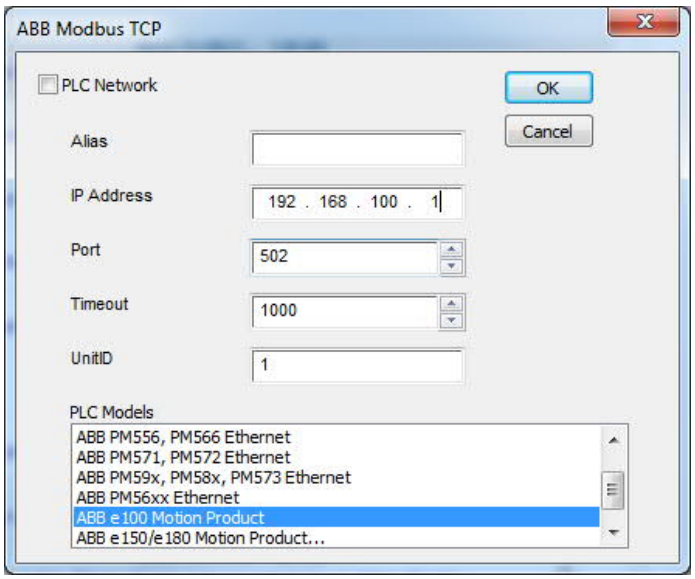
If the HMI is connected to a single Modbus TCP slave device (e.g. MicroFlex e190 or AC500 PLC) there is no need to select the 'PLC Network' check box. If there are multiple slave devices connected to the HMI then it is essential this box is selected (for the purposes of this application note we will select this option).



The dialog changes slightly and now includes an additional area showing the slave devices that have been added to the Modbus TCP network.

Click on the 'Add' button…the software now asks the user to enter details about the connected slave device….

Our ABB motion product (NextMove e100 for this example) has an IP address of 192.168.100.x (where x is set by the rotary switches on the front of the product). If we were connecting to an EPL master node via an EPL router drive we would enter the Ethernet side address the router has mapped to 192.168.100.240 at this point (refer to application note AN00247 for further details about the operation of the EPL router drive).



As detailed earlier, Modbus TCP uses Port number 502, this has been completed for us already. The timeout is the amount of time (in ms) the HMI will wait for a response from the connected devices before deciding there is a communication error. The default setting of 1 second (1000ms) is adequate.

The UnitID is used by some Modbus systems when routing data packets from Modbus TCP (Ethernet) through to Modbus RTU devices (serial). For ABB products this setting is not relevant, but the Unit ID needs to be unique for each device added to the Modbus TCP network. Therefore it is recommended this value is set to match the last octet of the IP address (1 in our example). Finally we need to select a PLC model. As we're using a NextMove e100 we've selected the 'ABB e100 motion product' PLC model. PLC Models are available for e100, e150 and e180 drive products (select e150/e180 if using a MicroFlex e190).

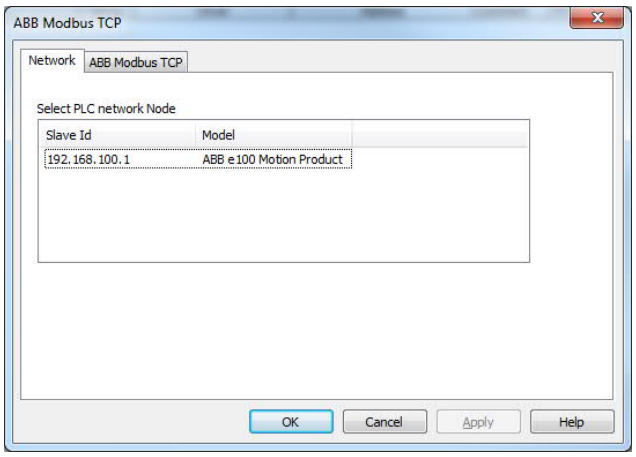
The software returns to the previous screen and now shows our configured node in the list of slave controllers. To add another slave device repeat the above process. To modify any settings, highlight the slave controller and click on the "Modify" button. Once all slave controllers have been added the protocol configuration process is complete.

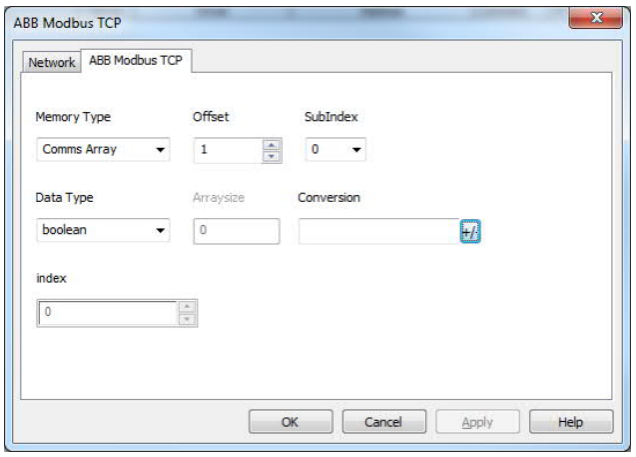
Creating Tags

Having configured the Modbus TCP protocol we can now start to create Tags to use throughout the HMI project. ABB motion products do not support Tag Export functions (unlike the PLC products) so Tags must be entered manually.
Double-click the "Tags" icon in the Project View window…

The Tag list screen now appears in the right hand pane. A filter at the top of this screen allows the user to select whether they wish to view Tags associated with a specific protocol or all Tags in the project (for example, if the HMI is being used as a Gateway between Modbus RTU and Modbus TCP there will be two protocols in use and there will be Tags associated with each of these protocols).

Click on the "+" button to create a new Tag…
If we selected "PLC Network" earlier when configuring our protocol the software will ask us to select which of the connected controllers the Tag relates to (in this example we only have a single controller)…





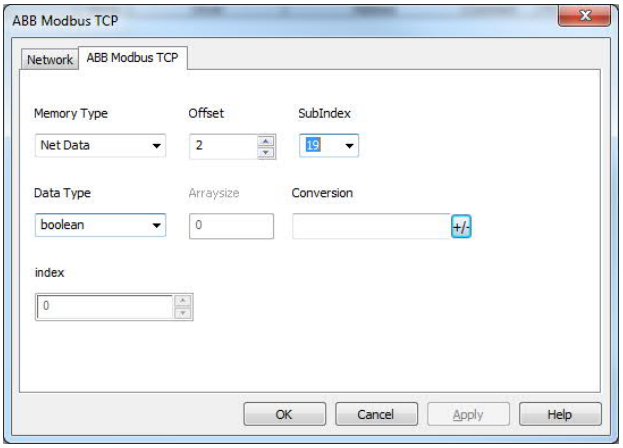
The field tab (labelled 'ABB Modbus TCP' in this case) lets us program which memory location in the motion controller the Tag relates to. The available 'Memory Type' selections are specific to the PLC model selected as shown by the table below:

PLC Type	Comms Array	Net Data
e100 motion product	Yes	Yes
e150/e180 motion product	No	Yes

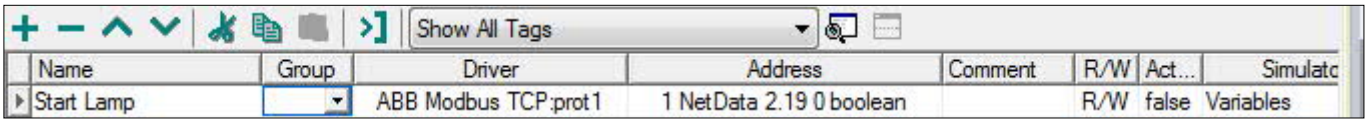
You should ensure that all tags used in the HMI project are setup to use a single Memory Type (i.e. do not attempt to mix memory types in the same project).
There is a third Memory type available (Node Override IP)….this can be used to modify at run-time the IP address being used as the slave's IP address (refer to the Help file within Panel Builder 600 for further details).
The other entries on this dialog are as follows:

Setting	Description
Offset	This relates to the index into the Comms Array or Net Data area - e.g. 3 to access Netinteger(3) or NetFloat(3)
Data type	Select from Boolean, Byte (signed 8 bit integer), Short (signed 16 bit integer), Int (signed 32 bit integer), unsignedByte, unsignedShort, unsignedInt, Float (32 bit IEEE) or String
Subindex	This entry varies depending on the data type. For Boolean (bit) level data the subindex can be 0 to 31 (corresponding to the 32 bits in a commsinteger or netinteger location). For Byte level data the subindex can be 0 to 3 (where 0 is the least significant byte). For Word level data the subindex can be 0 or 1 (where 0 is the least significant word)
Arraysize	Only used if String data type selected. Specifies the number of characters/bytes to be used by the string. A commsinteger/netinteger location can store up to 4 characters so if an array size of more than 4 is specified then subsequent data locations are used to store the additional characters - e.g. If a tag was configured to use Comms Array Offset 1 as a String and "ABCDEF" was to be stored this would result in 0x41424344 ("ABCD") being stored in Commsinteger(1) and 0x45460000 ("EF") being stored in Commsinteger(2)
Conversion	This entry allows the user to add a translation to (e.g. word swap) the data Index – this setting is not used

So as an example, if we needed a Tag to use in conjunction with a lamp in the HMI project (i.e. a Boolean/bit value) and we wanted this to relate to Bit 19 of NetData 2 in our ABB e100 motion product (e.g. NextMove e100) we would setup our Tag as shown below:



If we now click OK the software allows us to enter a name for our newly created Tag…



We can now click on the "+" button again to continue to add Tags to the project.

Using Modbus data in Mint programs on ABB motion products

The table below shows how various data types are likely to be used by a Mint program:

Data Type	Comms Array	NetData Array
Boolean	CommsInteger	NetInteger
Byte (signed or unsigned)	CommsInteger	NetInteger
Short (signed or unsigned)	CommsInteger	NetInteger
Int (signed or unsigned)	CommsInteger	NetInteger
String	CommsInteger	NetInteger
Float	Comms	NetFloat

The simplest way to access data with widths less than 32 bits (i.e. bits, bytes and words) in a Mint program is via the Mint BITFIELD keyword.

If we use our previous example where we configured an HMI tag related to Bit 19 of NetData location 2, our Mint program could contain the following code to read this bit from the HMI…

```
Bitfield BitData
        DoubleWord As 0 to 31
        Bit0 As 0
        Bit1 As 1
        Bit2 As 2
        Bit3 As 3
        Etc…
        Bit19 As 19
        Bit20 As 20
        Etc…
End Bitfield
Dim HMIBitData As BitData
HMIBitData.DoubleWord =  NETINTEGER(2)  'Read all 32 bits into a bitfield variable
OUTX(0) = HMIBitData.Bit19  'Set output 0 according to the value written by the HMI
```

…and we could then use the following code to write to this bit…
HMIBitData.Bit19 = INSTATEX(1) 'set bit 19 to reflect the state of input 1 on the controller
NETINTEGER(2) = HMIBitData.DoubleWord

Similar BitField types could be used to encode Byte and Word level data…

```
BitField ByteData
        DoubleWord As 0 to 31
        Byte0 As 0 to 7
        Byte1 As 8 to 15
        Byte2 As 16 to 23
        Byte3 As 24 to 31
End BitField

BitField WordData
        DoubleWord As 0 to 31
        Word0 As 0 to 15
        Word1 As 16 to 31
End BitField
```

For controllers not supporting the BITFIELD keyword (e.g. NextMove ESB-2 running firmware version 5424) data less than 32 bits wide must be extracted using the logical OR, AND, NOT functions for example.

Examples:

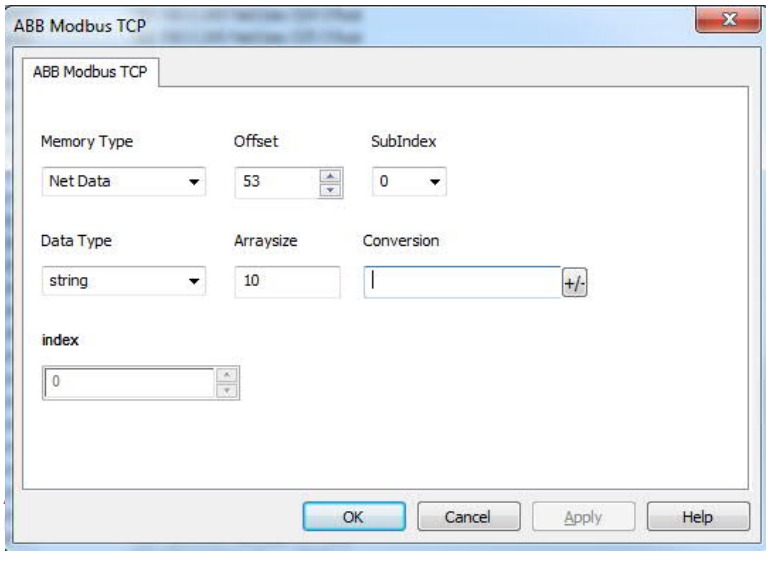
```
Dim nNetData2Bit19 As Integer
nNetData2Bit19 = ((NETINTEGER(2) & 0x00080000) > 0)

Dim nNetData2Word1 As Integer
nNetData2Word1 = SHIFT((NETINTEGER(2) & 0xFFFF0000), 16)
```

Accessing 32 bit data (int or Float) is much simpler, the Mint program just needs to utilise COMMS, COMMSINTEGER, NETFLOAT or NETINTEGER according to the setting of Modbus parameter _mpREGISTER_MAPPING (when relevant) and the programmed HMI data type.

For string data either COMMSINTEGER or NETINTEGER data should be used. Strings are made up of character data where each character is an eight bit (byte) value. Therefore each COMMSINTEGER or NETINTEGER location is capable of storing 4 characters. If the HMI Tag has been programmed to store more than 4 characters then successive locations are utilised as required.

Example:
A label on our HMI screen needs to display text up to 10 characters in length. The screenshot below shows how a Tag could be programmed to allow this. In this example we've started our string from NETINTEGER(53) and 2 sequential locations (inclusive of NETINTEGER(53) will be used to store our 10 characters…



The ‘ArraySize’ field in the dialog determines the number of characters the string Tag can store. In this case 10 characters means
the HMI will access NETINTEGER(53) through to NETINTEGER(55). The table below illustrates some example short strings…

NetInteger (53)	NetInteger (54)	NetInteger (55)	HMI String Display
0x41424344 (hex)	0	0	“ABCD”
0x41424344 (hex)	0x45460000 (hex)	0	“ABCDEF”
0x41424344 (hex)	0x45464748 (hex)	0	“ABCDEFGH”
0x41424344 (hex)	0x45464748 (hex)	0x494A0000 (hex)	“ABCDEFGHIJ”
0x41424344 (hex)	0x45464748 (hex)	0x494A7691 (hex)	“ABCDEFGHIJ”
0	0x45464748 (hex)	0x494A0000 (hex)	“”
0x41424344 (hex)	0	0x494A0000 (hex)	“ABCD”
0x41420044 (hex)	0x45464748 (hex)	0x494A0000 (hex)	“AB”

You can see from the above table that the bottom 16 bits of NetInteger(55) are not used. Also, as soon as a NULL (ASCII value 0) is encountered in the data the string is terminated, regardless of the contents of the remainder of the NetInteger locations.

Mint events

It is possible to associate Mint events / interrupts with Comms or NetData locations.

If using NextMove e100 it is possible to utilise Comms events and / or Netdata events. Comms events (1 to 10) and NetData events (0 to 31) are raised whenever the data in the associated location is changed. Writing the same value to one of these locations will not raise an event in Mint (unless using Workbench or the Mint ActiveX control to write values to Comms locations).

If using MicroFlex e190 or MotiFlex e180 it is only possible to utilise Netdata events. NetData events (0 to 31) are raised whenever the data in the associated location is changed. Writing the same value to one of these locations will not raise an event in Mint.

Physical Connection

The table below shows the physical connection possibilities for active Mint products supporting integrated Modbus protocols. AC500 and CP600 products are included for reference.

Connection Type	NextMove e100	NextMove ESB-2	MicroFlex e190	MotiFlex e180	AC500	AC500 Eco	CP600
RS232	Yes	Yes (by variant)	No	No	Yes	No	Yes
2 wire RS485	No	No	No	No	Yes	Yes	Yes
4 wire RS422	Yes	Yes (by variant)	No	No	No	No	Yes
Ethernet	Yes	No	Yes	Yes	Yes	Yes (by variant)	Yes

When using Modbus TCP / Ethernet straight-through or crossover cables may be used. If the HMI needs to connect to more than one product, wire the network in a ‘star’ type configuration using an Ethernet switch between the HMI and the connected server(s). If a NextMove e100 is part of an Ethernet Powerlink (EPL) network then it is also necessary to include a MicroFlex e190 or MotiFlex e180 drive acting as an EPL router between the switch and the NextMove e100.

Contact us

For more information please contact your

local ABB representative or one of the following:

new.abb.com/motion
new.abb.com/drives
new.abb.com/drivespartners
new.abb.com/PLC