

RTU32M – Plant Applications

Management of Distributed and Remote I/O

Application Note

5 Sep 2020 Doc. 13021



Plant applications require lots of I/O – but not all in one place as shown here...

Introduction

The latest generation modular Brodersen RTUs not only improve the functionality available for our traditional small-medium sized remote site applications, but they also meet the requirements of plant style applications. Our previous generation of brick style RTUs have always supported I/O expansion and allowed for creation of distributed I/O solutions, but their physical size, power requirements and limited functionality of the I/O modules meant that such solutions did not meet expectations, or match capabilities of large PLCs.

This application note describes how the RTU32M series I/O modules can be used in plant applications where I/O is both distributed within a cabinet and located remotely around the plant on an Ethernet LAN.

RTU32M – with advanced PLC functionality suitable for plant applications:

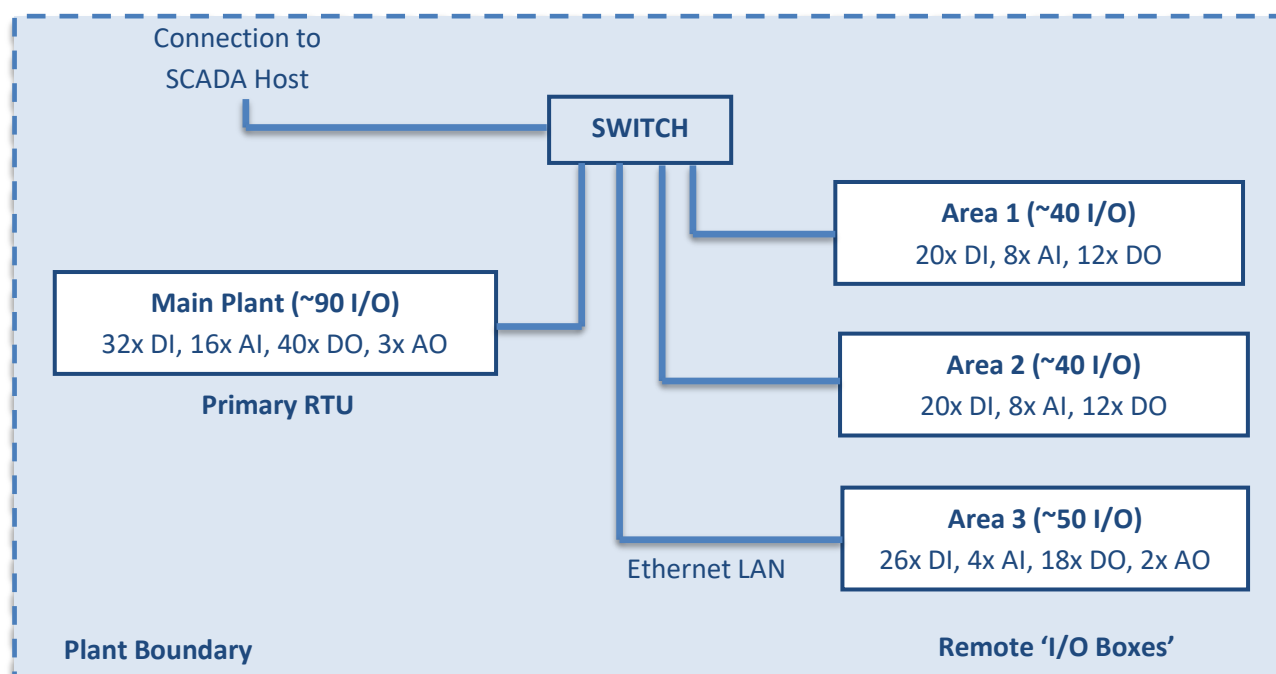
- Secure local and remote access to configuration and diagnostics
- Small footprint, high density modular I/O
- Failsafe, hot swap, smart I/O
- Online logic changes (including application restart with outputs held)
- Distributed I/O (segmented bus)
- Remote I/O (serial and LAN based I/O - remote CPUs have no logic app or options)
- Redundancy options (multiple power supplies, CPUs, I/O, LANs etc)
- Global variable 'binding' allows fast setup of a single plant wide logic application
- Future proof (option to include logic/smarts in the remote RTUs, when needs change)
- Modern platform, ensuring longevity and continued development of firmware and hardware functionality (including new module types).

Most RTUs are well suited for remote site applications (typically <200 I/O)

In the past - when asked to highlight the differences and benefits of using an RTU versus a PLC, the benefits of using RTUs focused on I/O isolation and SCADA protocols that are not well supported by PLCs, such as IEC60870, IEC61850 and DNP3. These protocols allow for prioritisation of data, logging of time stamped events, inclusion of data quality and described data formats, hence they are well suited for wide area SCADA systems with geographically dispersed remote sites. As the majority of the remote sites had <200 I/O, physical size and management of distributed I/O did not need to be considered – so the I/O isolation and protocol functionality was enough to make the RTU the best choice.

Not all RTUs suit plant applications (typically >200 distributed/remote I/O)

Plant applications have large amounts of I/O that need to be distributed remotely because of the various separated areas that form the plant. The physical size of the I/O is important, but the remote I/O must also be managed by the primary/central PLC/RTU ie. as an extension of its own physical I/O.

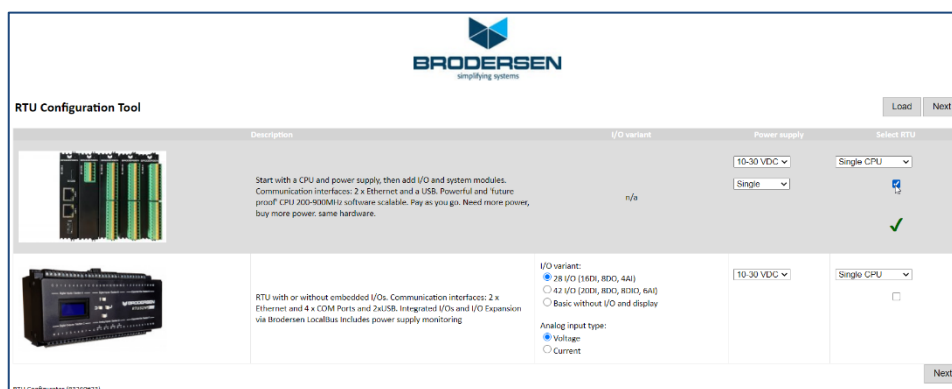


Example Distributed I/O Plant Application

Online RTU Configurator – Design what you need!

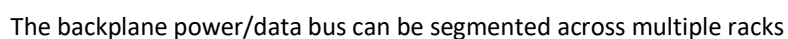
Our RTU Configurator allows creation of different RTU setups. Add power supplies, CPU, I/O and comms modules to create appropriate RTU layouts.

It's a bit different to a 'stacking bricks' TETRIS game, but still fun!



<https://brodersen.com/RTUConfigurator/>

Each RTU32M supports up to 60 I/O modules. The overall backplane bus length can be up to 40m (assuming that appropriate bus cables are used).



Plant solutions need local and remote I/O to be managed in a single application

Most plant applications have a combination of distributed and remote I/O. An important consideration is management of the remote I/O. It is much easier to manage the plant automation in a single/central logic application if the remote I/O is 'connected' by a background process that ensures remote I/O is referenced in the same way as local I/O. This also avoids needing logic applications in the remote I/O sites.

Build it and they will come ...

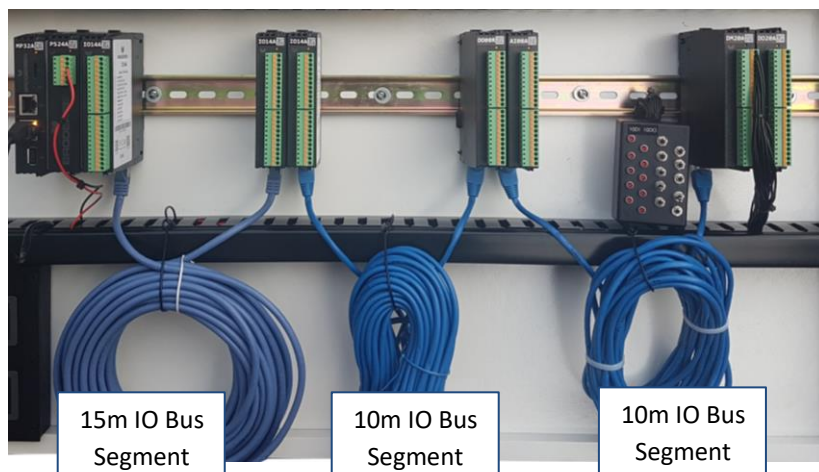
It is very tempting to completely fill a test rack with hundreds of RTU modules – just because we can!

Instead, we loaded it with four RTUs to represent the main plant and 3x remote area locations of the example plant application shown below.

Objective – prove one logic application can manage all I/O!



The Test Rack – 4x RTUs



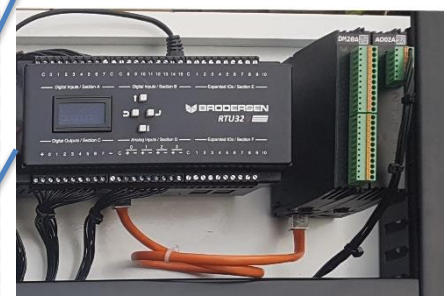
Main Plant – the central RTU/PLC logic application



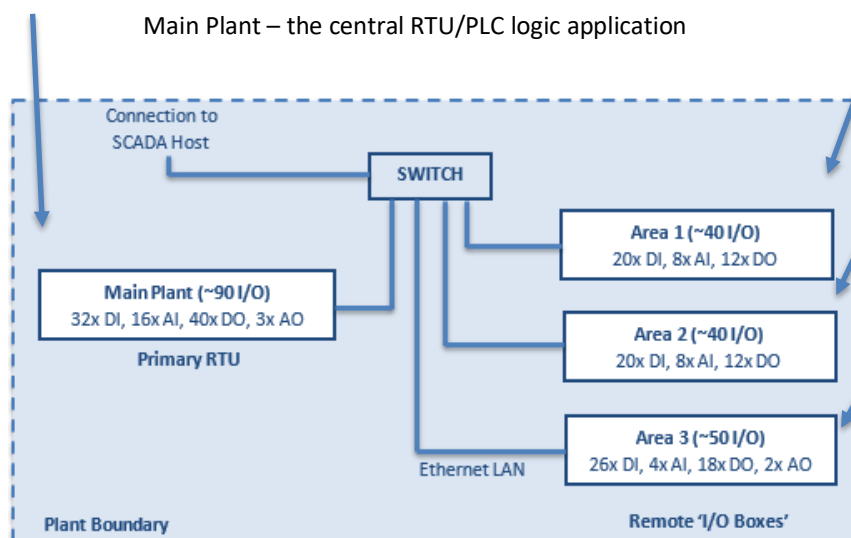
Area 1 – remote I/O



Area 2 – remote I/O



Area 3 – remote I/O



Example Distributed I/O Plant Application

WorkSuite – managing the setup of multiple I/O locations

The Brodersen WorkSuite software makes it easy to setup and manage the sites. Each site has a 'project' that is listed in a common Workspace area as shown below eg. 'IO Boxes 1-3' and the 'IO Box Master'.

The screenshot shows the Brodersen WorkSuite interface. The left pane displays the 'Workspace' tree with 'IOBox1' selected. The main area shows a rack of modules: RTU32M, PS24A, DI20C, AI08A, and DO12R. A text box explains: 'Each IO Box project has variables associated to the local I/O (created using the IO wizard). A logic program is not required (but can be added later if additional functionality is required at this location).' Below the rack, a table lists the variables for the DI20C module.

Name	Value	Symbol	Type	Channel
Art-No	DI20C	IOBox1_Slot2_DI00	Input Bit	0
Description	20 Digital Inputs (2 High Sp...	IOBox1_Slot2_DI01	Input Bit	1
		IOBox1_Slot2_DI02	Input Bit	2
		IOBox1_Slot2_DI03	Input Bit	3
		IOBox1_Slot2_DI04	Input Bit	4
		IOBox1_Slot2_DI05	Input Bit	5
		IOBox1_Slot2_DI06	Input Bit	6
		IOBox1_Slot2_DI07	Input Bit	7
		IOBox1_Slot2_DI08	Input Bit	8
		IOBox1_Slot2_DI09	Input Bit	9

Download one/all projects

Select the project to edit

The screenshot shows the Brodersen WorkSuite interface with the 'IOBox_Master' project selected in the workspace. A dropdown menu is open, showing the list of projects: IOBox1, IOBox2, IOBox3, and IOBox_Master. Arrows point from the text labels to the 'Download' icon and the dropdown menu.

The Global Binding Editor – mapping tables make association of remote I/O easy...

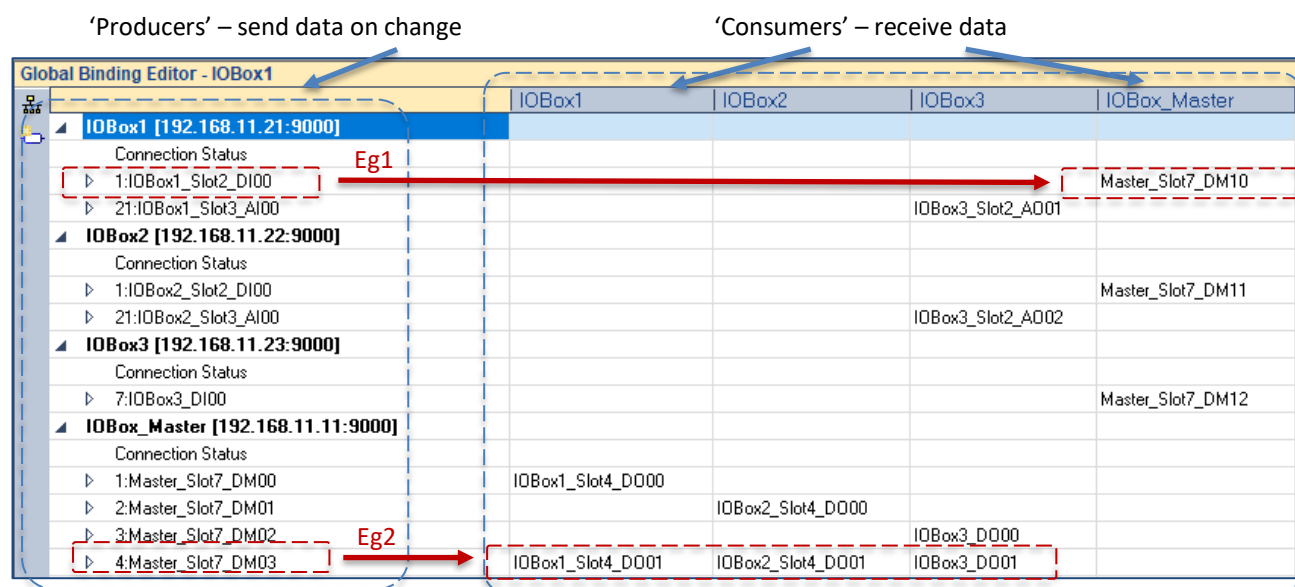
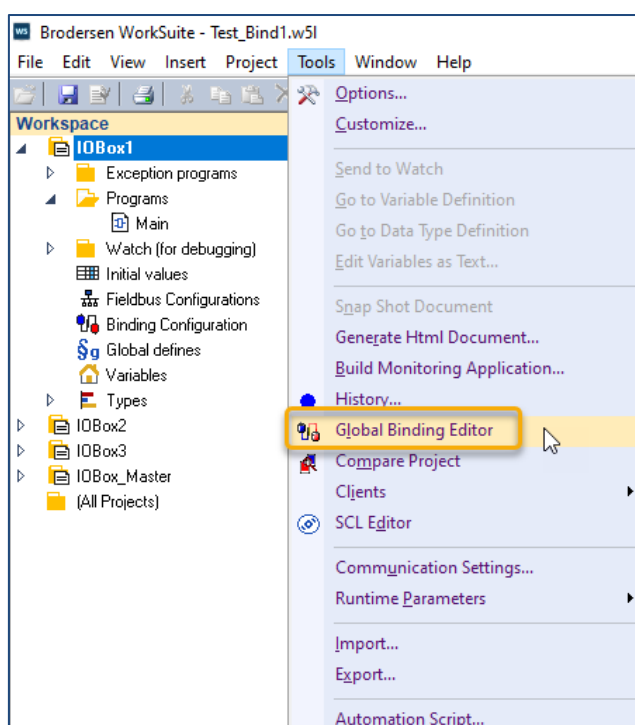
The Global Binding Editor allows the association of tags from the various projects in the Workspace to be made by dragging variables in to a table.

Communications between sites then occurs if any of the producer data values change (without needing to setup logic or comms tasks to send/poll data).

An initial 'proof of concept' setup example.

Before creating a more complex solution, first look at the simple example below. Here a few input points from each location are 'produced' and sent on change to remote consumer output points.

Note: this simple example is moving 'physical I/O' ie. AI and DI values from one site to AO and DO values at another site.



Eg1. Input DI00 on IOBox1 activates output DM10 on the IO Master

Eg2. Input DM03 on the IO Master activates outputs DO01 on each of the 3x remote IO Boxes

In most plant applications the primary function of the binding protocol is to allow the master to have an image of all remote variables (updates to input images are received from the remote site – writing to an output image sends an update to the remote site). The variables associated to/from the remote sites can be any variable type (not just physical I/O).

The Global Binding Editor – link and point information, tag debounce/hysteresis

Each communications link reports its connection status and each point has additional status fields that include error status and date and time stamps. Each point also has a hysteresis option to manage debounce and significant change (minimises excessive event reporting).

	IOBox1	IOBox2	IOBox3	IOBox_Master
IOBox1 [192.168.11.21:9000]				
Connection Status				IOBox1_Link_Fail=FALSE
1:IOBox1_Slot2_DI00				Master_Slot7_DM10=TRUE
Error Status				IOBox1_rec_error=0
Date Stamp				IOBox1_rec_date=29 August 2020
Time Stamp				IOBox1_rec_time=t#6h56m17s237ms

Remote Point
Value (Producer)

Image of the Remote
Point Value (Consumer)

Point Status

ie. the master knows when each point
last changed and the link status

Link Identifier: 21

Variable: IOBox1_Slot3_AI00

☐ Create consumer variables

Hysteresis

☒ Hysteresis

Positive: 50

Negative: 100

☐ Symmetric

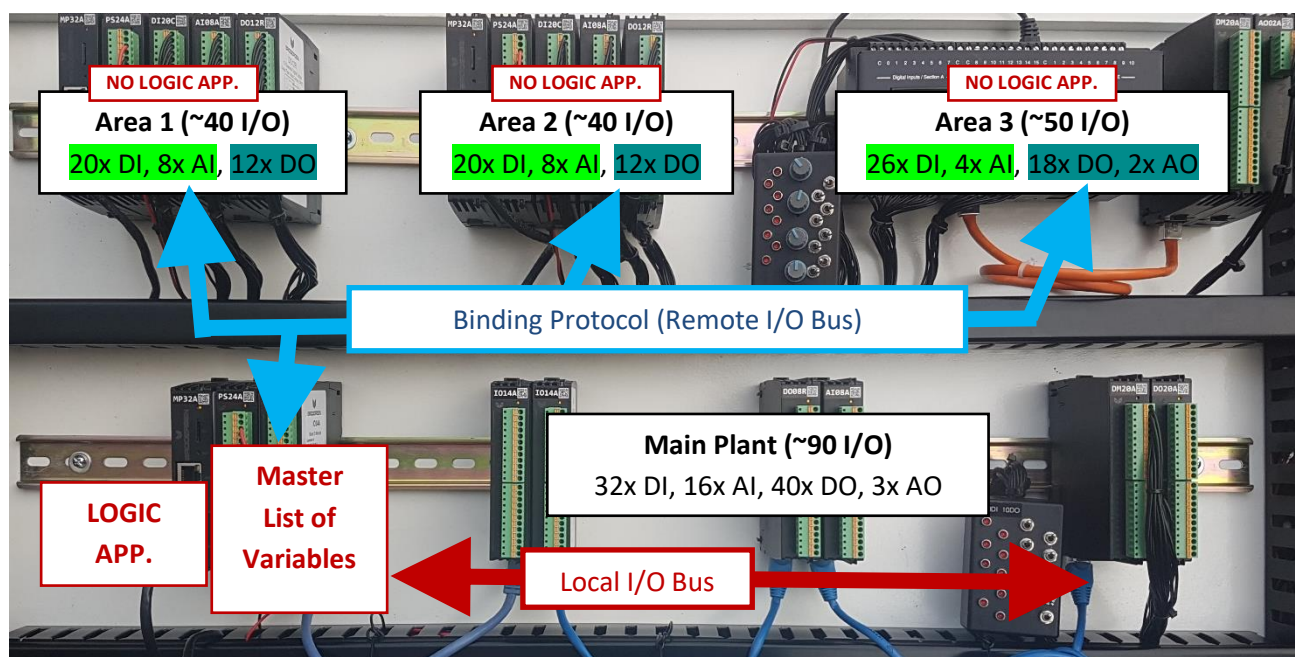
OK Cancel

Point properties

ie. when creating a produced 'public' variable by dragging a tag in to the destination/consumer field, a consumer variable can be created and debounce/hysteresis values set.

Back to the example plant application

The Master RTU has ~90 of its local I/O variables and ~130 remote I/O variables ('tags of interest') mapped from/to the remote sites. The logic application interacts with the master list of variables (local/remote I/O).



Mapping of Local and Remote I/O variables - Example Plant Application

Note: the master has an additional 98 local I/O tags associated to its own physical I/O.

Plant Application – the Binding Editor maps variables from/to the remote sites

The Binding Editor adds a column for each site. The first column holds producer variables. In the example plant application, each remote site produces its physical inputs, as listed below.

IOBox1 [192.168.11.21:9000]	IOBox2 [192.168.11.22:9000]	IOBox3 [192.168.11.23:9000]
Connection Status	Connection Status	Connection Status
1:IOBox1_Slot2_DI00	1:IOBox2_Slot2_DI00	1:IOBox3_AI00
2:IOBox1_Slot2_DI01	2:IOBox2_Slot2_DI01	2:IOBox3_AI01
3:IOBox1_Slot2_DI02	3:IOBox2_Slot2_DI02	3:IOBox3_AI02
4:IOBox1_Slot2_DI03	4:IOBox2_Slot2_DI03	4:IOBox3_AI03
5:IOBox1_Slot2_DI04	5:IOBox2_Slot2_DI04	5:IOBox3_DI00
6:IOBox1_Slot2_DI05	6:IOBox2_Slot2_DI05	6:IOBox3_DI01
7:IOBox1_Slot2_DI06	7:IOBox2_Slot2_DI06	7:IOBox3_DI02
8:IOBox1_Slot2_DI07	8:IOBox2_Slot2_DI07	8:IOBox3_DI03
9:IOBox1_Slot2_DI08	9:IOBox2_Slot2_DI08	9:IOBox3_DI04
10:IOBox1_Slot2_DI09	10:IOBox2_Slot2_DI09	10:IOBox3_DI05
11:IOBox1_Slot2_DI10	11:IOBox2_Slot2_DI10	11:IOBox3_DI06
12:IOBox1_Slot2_DI11	12:IOBox2_Slot2_DI11	12:IOBox3_DI07
13:IOBox1_Slot2_DI12	13:IOBox2_Slot2_DI12	13:IOBox3_DI08
14:IOBox1_Slot2_DI13	14:IOBox2_Slot2_DI13	14:IOBox3_DI09
15:IOBox1_Slot		
16:IOBox1_Slot		
17:IOBox1_Slot		
18:IOBox1_Slot		
19:IOBox1_Slot		
20:IOBox1_Slot		
21:IOBox1_Slot		
22:IOBox1_Slot		
23:IOBox1_Slot		
24:IOBox1_Slot		

Remote Input Variables

PRODUCED VARIABLES - 86x Remote INPUTS:

Each remote IO Box project has variables associated to its own local I/O.

The remote input values are mapped using the global binding editor to 'internal' variables in the master from I/O input variables in the remote I/O Box.

If a remote input value changes, a message is sent to the master.

In the example plant application, the master produces its images of the remote physical outputs, as listed below.

IOBox_Master [192.168.11.11:9000]	IOBox1	IOBox2	IOBox3
Connection Status			
1:IOBox1_Slot4_DO00	IOBox1_Slot4_DO00		
2:IOBox1_Slot4_DO01	IOBox1_Slot4_DO01		
3:IOBox1_Slot4_DO02	IOBox1_Slot4_DO02		
4:IOBox1_Slot4_DO03	IOBox1_Slot4_DO03		
5:IOBox1_Slot4_DO04	IOBox1_Slot4_DO04		
6:IOBox1_Slot4_DO05	IOBox1_Slot4_DO05		
7:IOBox1_Slot4_DO06	IOBox1_Slot4_DO06		
8:IOBox1_Slot4_DO07	IOBox1_Slot4_DO07		
9:IOBox1_Slot4_DO08	IOBox1_Slot4_DO08		
10:IOBox1_Slot4_DO09	IOBox1_Slot4_DO09		
11:IOBox1_Slot4_DO10	IOBox1_Slot4_DO10		
12:IOBox1_Slot4_DO11	IOBox1_Slot4_DO11		
13:IOBox2_Slot4_DO00		IOBox2_Slot4_DO00	
14:IOBox2_Slot4_DO01		IOBox2_Slot4_DO01	
15:IOBox2_Slot4_DO02		IOBox2_Slot4_DO02	
16:IOBox2_Slot4_DO03		IOBox2_Slot4_DO03	
17:IOBox2_Slot4_DO04		IOBox2_Slot4_DO04	
18:IOBox2_Slot4_DO05		IOBox2_Slot4_DO05	
19:IOBox2_Slot4_DO06		IOBox2_Slot4_DO06	
20:IOBox2_Slot4_DO07		IOBox2_Slot4_DO07	
21:IOBox2_Slot4_DO08		IOBox2_Slot4_DO08	
22:IOBox2_Slot4_DO09		IOBox2_Slot4_DO09	
23:IOBox2_Slot4_DO10		IOBox2_Slot4_DO10	
24:IOBox2_Slot4_DO11		IOBox2_Slot4_DO11	
25:IOBox3_Slot1_DM11			IOBox3_Slot1_DM11
26:IOBox3_Slot1_DM12			IOBox3_Slot1_DM12
27:IOBox3_Slot1_DM13			IOBox3_Slot1_DM13
28:IOBox3_Slot1_DM14			IOBox3_Slot1_DM14
29:IOBox3_Slot1_DM15			IOBox3_Slot1_DM15
30:IOBox3_Slot1_DM16			IOBox3_Slot1_DM16
31:IOBox3_Slot1_DM17			IOBox3_Slot1_DM17
32:IOBox3_Slot1_DM18			IOBox3_Slot1_DM18

Master 'image' of Remote Output Variables

Remote Output Variables

PRODUCED VARIABLES - 36x Remote OUTPUTS:

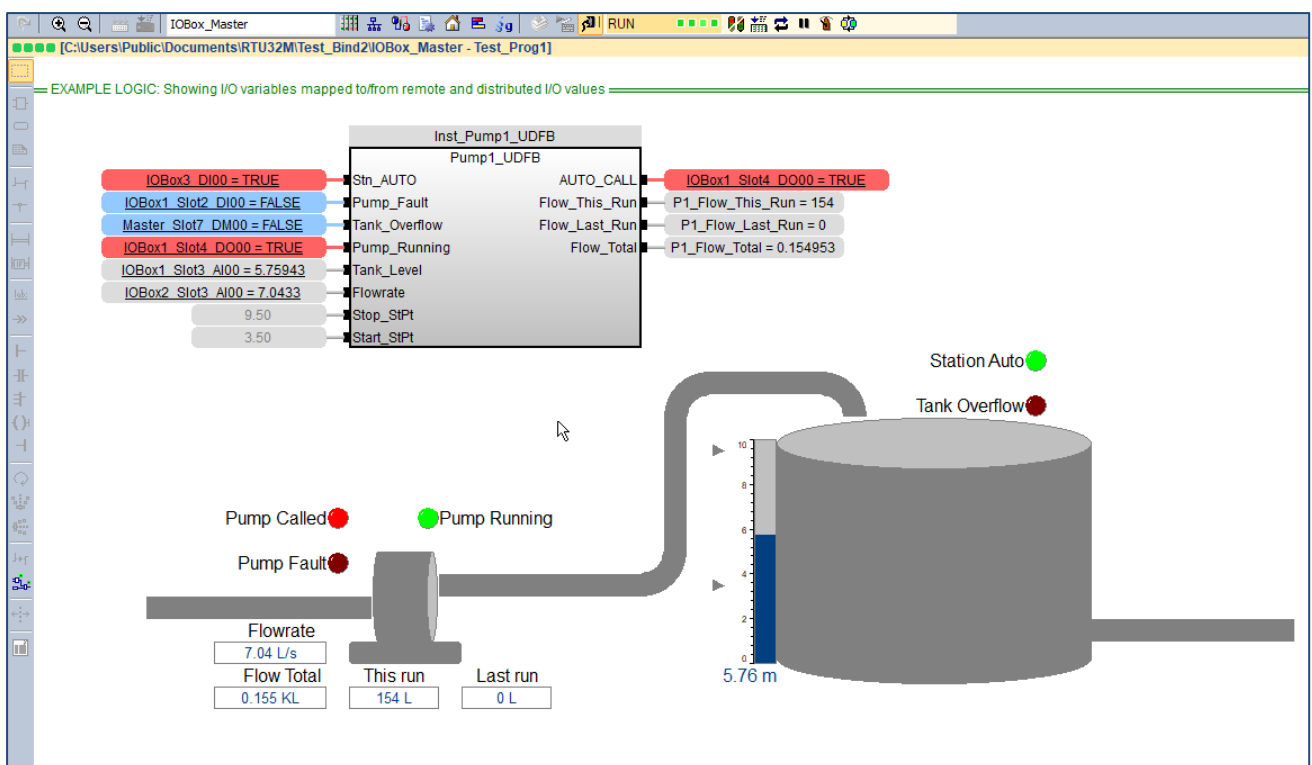
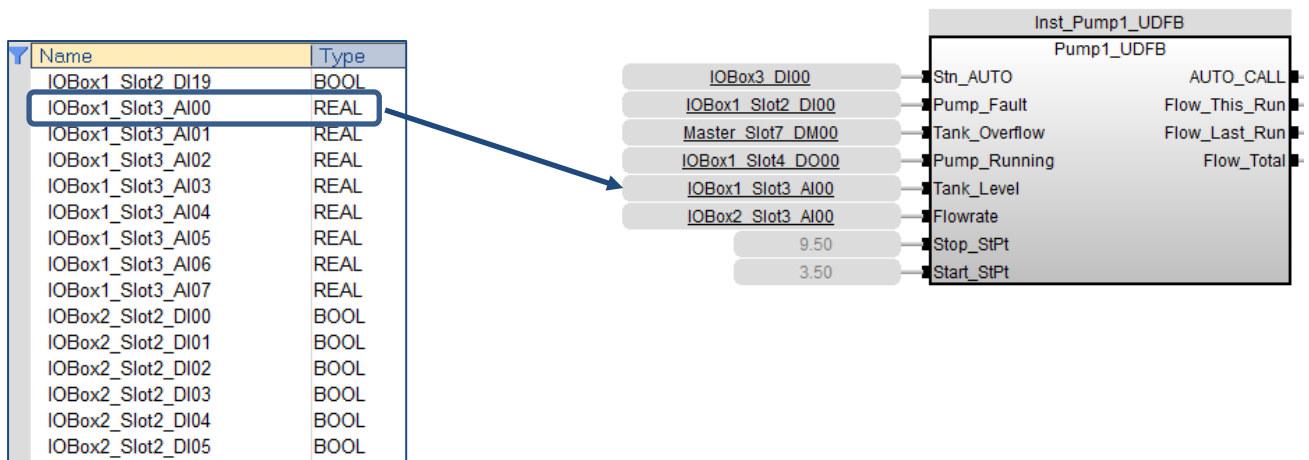
Each remote IO Box project has variables associated to its own local I/O.

The remote output values are mapped using the global binding editor from internal variables in the master to I/O output variables in the remote IO Box.

If a master output value changes, a message is sent to the remote IO Box.

Plant Application – proving that it works

A selection of local and remote I/O variables have been mapped to a simple pump controller application to demonstrate/prove the concept.



Example Plant Application Logic – online with some remote I/O

Get on the Brodersen Bus – local and remote I/O solutions!

Very remote I/O – far far away in Melbourne Australia...

Watch our 10m video to find out if the test application worked.

<https://brodersen.com/training/>

Additional product information is available from our website,
or from the authorised distributor in your region.

<http://www.brodersen.com>

