SCADA DESIGN AND ENTERPRISE CONNECTIVITY FOR A WATER PROCESSING SYSTEM

HAMK
UNIVERSITY OF APPLIED SCIENCES

Bachelor's thesis

Degree Programme in Automation Engineering

Valkeakoski 18.11.2011

Stanley Twumasi

HAMK
UNIVERSITY OF APPLIED SCIENCES

ABSTRACT

Valkeakoski
Automation Engineering

Author Stanley Twumasi **Year** 2011

Subject of Bachelor's thesis SCADA design and enterprise connectivity for a water processing system.

.

ABSTRACT

This thesis is a development project designed to monitor and control a laboratory water processing system for Hamk University of Applied Sciences Automation Engineering Department.

The main areas of this thesis can be categorized into four main sections. The first part deals with the technical documentations of the water processing system. The second section involves the installation and configuration of the Beckhoff (PLC) input and output cards used as a remote terminal unit.

The third section involves graphical programming and the development of the human machine interface with LabVIEW to acquire or generate data from or to these sensors and actuators of the process through Beckhoff's input and output cards by utilizing ACTIVE X control and AdsOCX standards or protocol.

The final part deals with enterprise integration demonstrating how process data is captured from the process and circulated or made available in the local area network (LAN) of Hamk University of Applied Sciences

Keywords: process, control, configuration, active x, enterprise integration

Pages 58p+ appendices 3p.

CONTENTS

1 **INTRODUCTION** 1

1.1 Terminology 1
1.2 Objectives 2
1.3 Problem description 2
1.4 Motivation 3

2 TECHNICAL DOCUMENTATION 3

2.1 Process structure 3
2.2 Main sensors and actuators 4
2.2.1 Water pump 4
2.2.2 Control valve 4
2.2.3 Magnetic flow meter 5
2.2.4 Magnetic solenoid valves 6
2.2.5 Level transmitters 6
2.2.6 Limit switch/float switch 7
2.3 Process and instrumentation diagram 8
2.4 Loop function principle 9
2.5 Process circuit diagram 10
2.6 Instrumentation specification sheet 11
2.7 I/O reservation table 11

3 BUS COUPLER AND BUS TERMINALS 12

3.1 Current to voltage signal conversion 14

4 CONFIGURATION OF BECKHOFF I/O SYSTEM (RTU/PLC) 15

4.1 Scanning devices automatically 16
4.2 Scanning boxes automatically 17
4.3 Scanning terminals automatically 18
4.4 Scanning devices manually 19
4.5 Scanning boxes manually 21
4.6 Scanning terminals manually 22

5 LABVIEW AND TWINCAT DATA PATH IMPLEMENTATION 24

6 GRAPHICAL PROGRAMMING. 25

6.1 Inserting an a active X object 27
6.2 Adding a property 28
6.3 Adding AdsAms Server NetId and AdsAms Server Port Properties 29
6.4 Entering data for AdsAms Server NetId 30
6.5 Entering data for AdsAmsServer Port 31
6.6 Adding a flat sequence structure 32
6.7 Creating a while loop 33
6.8 Coding binary input terminals 35
6.9 Coding binary output terminals 37
6.10 Coding analog input terminals 39

6.11 Coding analog output terminals 40
6.12 Adding constants to coded terminals 41

7 INDICATORS, DATA DISPLAYERS AND ALARMS 45

7.1 Pump fault indicator and alarm 45
7.2 Water Tank1(WT1) limit switch and alarm 46
7.3 Tank2 and Tank3 filling - draining valves switches, and pump switch 47
7.4 Tank 2 and Tank3 level transmitters, and flow meter transmitter 48
7.5 Pump Speed and control valve 50
7.6 X-scale offset and multiplier 52
7.7 Wait until millisecond multiplier and iteration 53
7.8 Process control principle 53

8 ENTERPRISE CONNECTIVITY 55

8.1 Process data access 55

9 CONCLUSIONS 57

9.1 Recommendation 57

SOURCES 58

Appendix 1 Control cabinet box and field layout

Appendix 2 Controller and power supply

1 INTRODUCTION

1.1 Terminology

Acronym SCADA stands for supervisory control and data acquisition. These are normally computer based systems used to control and monitor a process. These industrial or infrastructure processes can be monitored and controlled by computer systems. (National Communication System 2004).

SCADA systems were introduced about six decades ago and they have evolved with technological advancement. SCADA systems have also shifted from closed proprietary systems to an open system allowing designers to choose equipment that can help them monitor their unique system using equipment from mixed vendors. (National Communication System 2004). These SCADA systems utilize common communication protocols such as Ethernet and TCP/IP to transmit information or data from field devices to the master control unit. (National Communication System 2004).

The main components to a SCADA system are the instrumentation and control devices, RTUs (Remote terminal units) or PLCs (Programmable logic controllers), communications systems, and the master station HMI (Human machine interface) (Figure 1). (National Communication System 2004).

The RTU/PLC provides an interface to the instruments and control devices of the process. The communication system provides the path for data transmission between the RTU/PLC and the master station. (National Communication System 2004).
These communication systems can be wire connected or wireless. The HMI displays this information in an easily understood graphical form, saves the data received, transmits alarms and permits operator control as required. (National Communication System 2004).

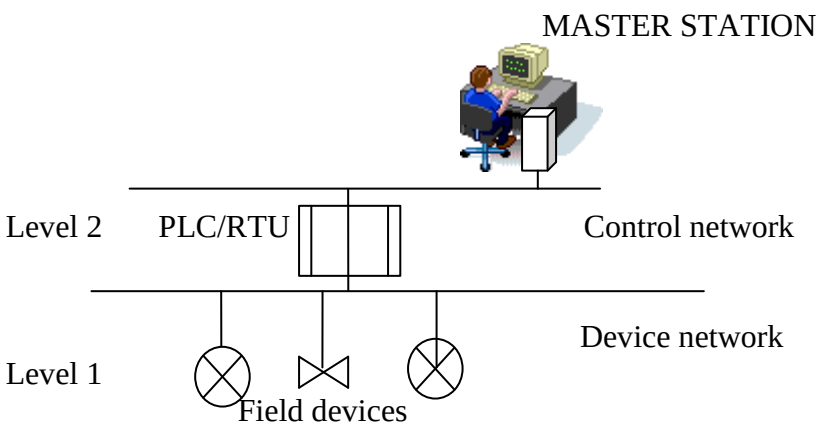


Figure 1 The main components of a SCADA system. (Bailey. D & Wright 2003, 70).

1.2 Objectives

The objectives of this thesis were to develop a SCADA system that would monitor and control a water processing system, gather the process information and disseminate this information to the appropriate work station located in the local area network of Hamk University of Applied Sciences.

1.3 Problem description

At an earlier development phase of the water processing system at Hamk University of Applied Sciences the system utilized a control panel equipped with knobs, lights, and switches (Figure 2).

To implement supervisory control, the operator must manually push the switches and turn the knobs. Data acquired from the process is trapped internally making it impossible to share.

Figure 2 Earlier phase control panel

A modern processing system must have an open network infrastructure that is able to disseminate data from the field device network all the way up to the execution and business level allowing the hardware of different manufacturers to communicate between themselves.(Berge 2011,3).

Likewise a state-of-the-art processing system must also have an open software infrastructure beginning from the control network all the way up to the business network, allowing the software from different developers to communicate between themselves. (Berge 2011, 3).

1.4 Motivation

The motivation for this research came from a current trend of software and hardware applications in the automation industry. These highlight and the increasing importance of data distribution within an enterprise between software and hardware from different software developers as well as automation device manufacturers.

2 TECHNICAL DOCUMENTATION

2.1 Process structure

The process structure is made of anodized aluminum profile. It is mounted on four castors so that it can be moved from one station to another. The frame comes with a working surface to accommodate computers and other devices. A laminated timber slab is fitted between the frame ends. All the connection devices that are needed are mounted on this slab: power sources, terminal connectors and a converter.

The process consists of a reservoir tank made of transparent acrylic 700x400x500mm and two transparent acrylic vertical tanks. There are pipes made of plastic, except the ones from the pump to the T-piece, which is made of 38mm steel pipe, and from there onto a 25mm steel pipe of the magnetic valve. (Figure 3).

Figure 3 Process structure

2.2 Main sensors and actuators

The water process further consists of two level transmitters (hydrostatic pressure), a magnetic flow meter, a control valve, a pump equipped with an internal frequency converter, and four magnetic solenoid valves.

2.2.1 Water pump

A Grundfors water pump is equipped with an internal frequency converter, two indicator lights and a warning or fault signal output terminal block built into the MLE terminal box. The warning or fault signal output can be wired to indicate normal operation as well as warning or fault. (Figure 4).

The main function of the pump in the process is to pump water from Tank1 into either Tank 2 or Tank3.

Figure 4 Grundfors water pump

2.2.2 Control valve

The control valve is made up of two components. There is a valve coupled to an actuator (AQM24-05) with a control signal ranging between 0-10V DC. The actuator has an automatic calibration function for the calibration of the starting point and stroke. This allows a simple adaptation to various valves. (Figure 5).

The function of the control valve in the process is to control the amount of water that is flowing back from either Tank 2 or 3 into Tank1.The opening and closing of the valve can also be done manually.

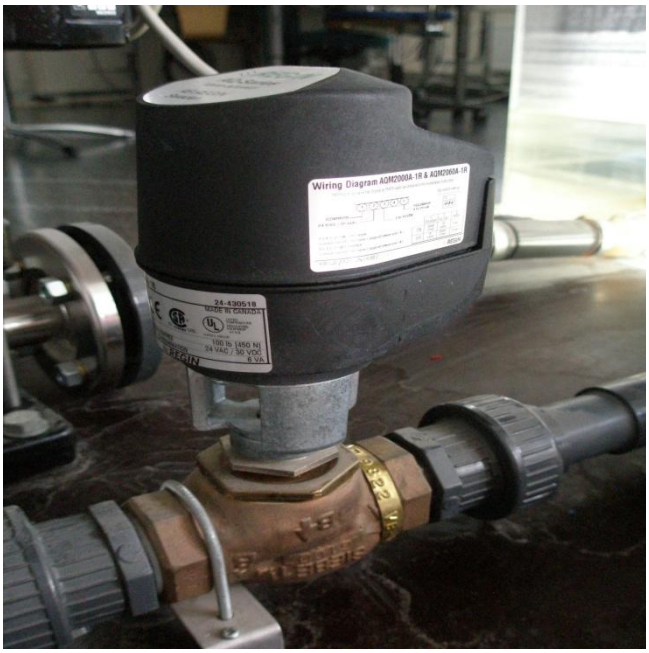


Figure 5 Control valve

2.2.3 Magnetic flow meter

A magnetic flow meter measures the amount of water that is flowing into Tank 2 and Tank3. (Figure 6).

Figure 6 Valmet flow meter

2.2.4 Magnetic solenoid valves

There are two filling and two draining magnetic valves. The functions of the filling valves are to allow pumped water to flow into Tank 2 and Tank 3 when opened. When opened the draining valves allow water from Tank 2 and tank 3 to flow back into Tank 1. (Figure 7).

Figure 7 Magnetic solenoid valves

2.2.5 Level transmitters

Hydrostatic pressure level transmitters mounted at the base of Tank 2 and Tank 3 measure the level of water inside the tanks. (Figure 8).

Figure 8 Hydrostatic pressure transmitters

2.2.6 Limit switch/float switch

The switch consists of a micro switch inside a polystyrene cover, from which a 3wire cable with acrylic sheath protrudes. It has a counter weight which is fixed to the cable at a desired distance from the float. In the process the limit switch defines a minimum level of water in Tank 3. (Figure 9).

Figure 9 Float switch

2.3 Process and instrumentation diagram

The process and instrumentation diagram illustrates how different components (piping, instrument and equipment) of the process are connected together and act as the basis for all design activities (Figure 10). (Battikha 2007, 253).

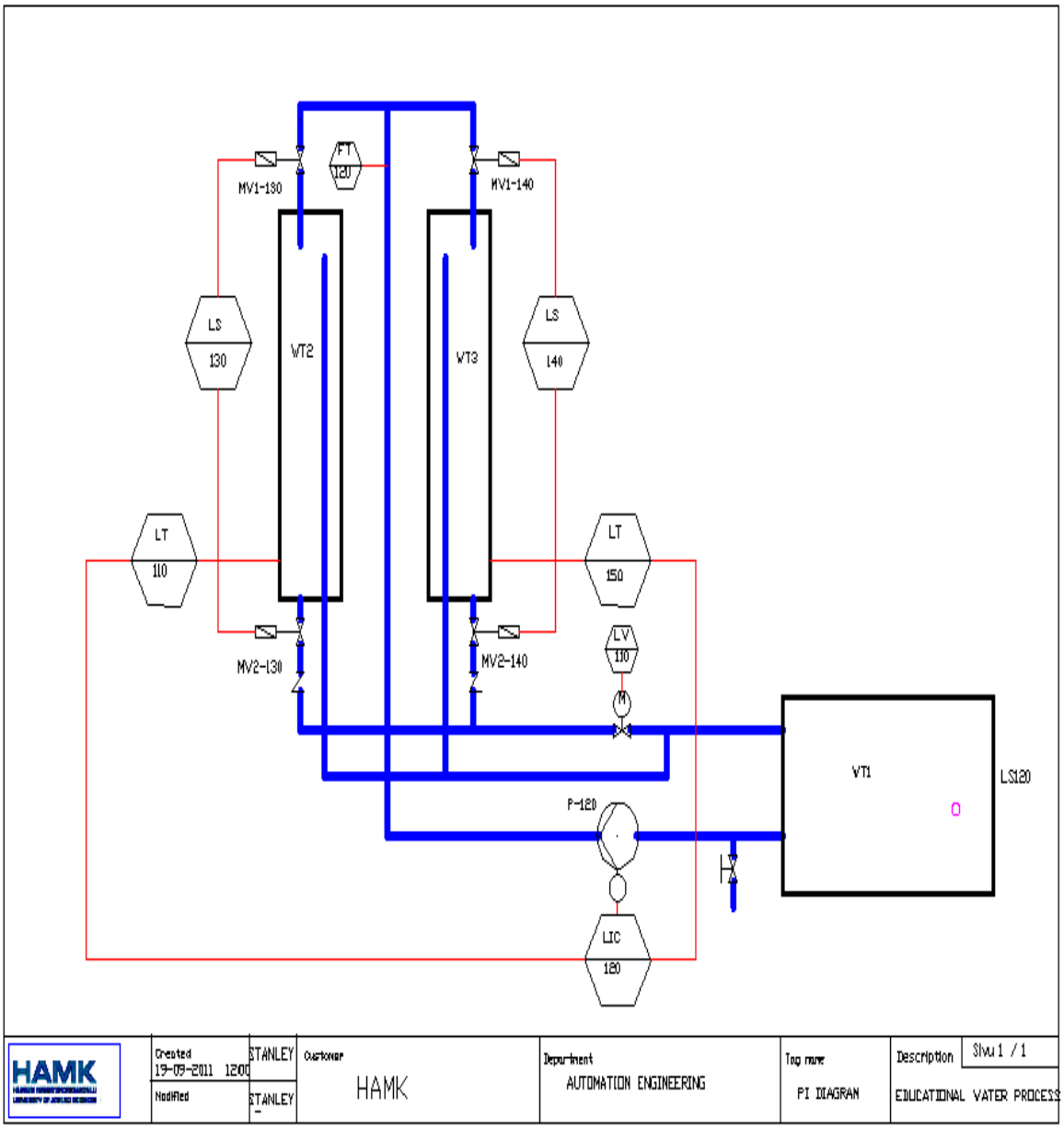


Figure 10 Process and instrumentation diagram

2.4 Loop function principle

The loop function principle describes the detailed arrangement of instrumentation components in a loop. (Table 1). (Battikha 2007, 268).

Table 1 Loop function table

LICA- 110		Tank level measurement and controller
	P-120	pump
	LT-150	Hydrostatic level measurement transmitter
	LT-120	Hydrostatic level measurement transmitter
LS-130		Switch for Tank WT2 level
	MV1-130	Tank WT2 inlet valve
	MV2-130	Tank WT2 outlet valve
LS-140		Switch for Tank WT3 level
	MV1-140	Tank WT3 inlet valve
	MV2-140	Tank WT3 outlet valve
LS-150		Tank WT2 level measurement
	LT-150	Tank WT2 hydrostatic level measurement transmitter.
LS-110		Tank WT3 level measurement
	LT-110	Tank WT3 hydrostatic level measurement transmitter.

2.5 Process circuit diagram

The process circuit diagram shows the detailed wiring arrangement. This helps to trace where a particular wire is going and coming from. This makes trouble shooting and maintenance activities less cumbersome (Figure 11).

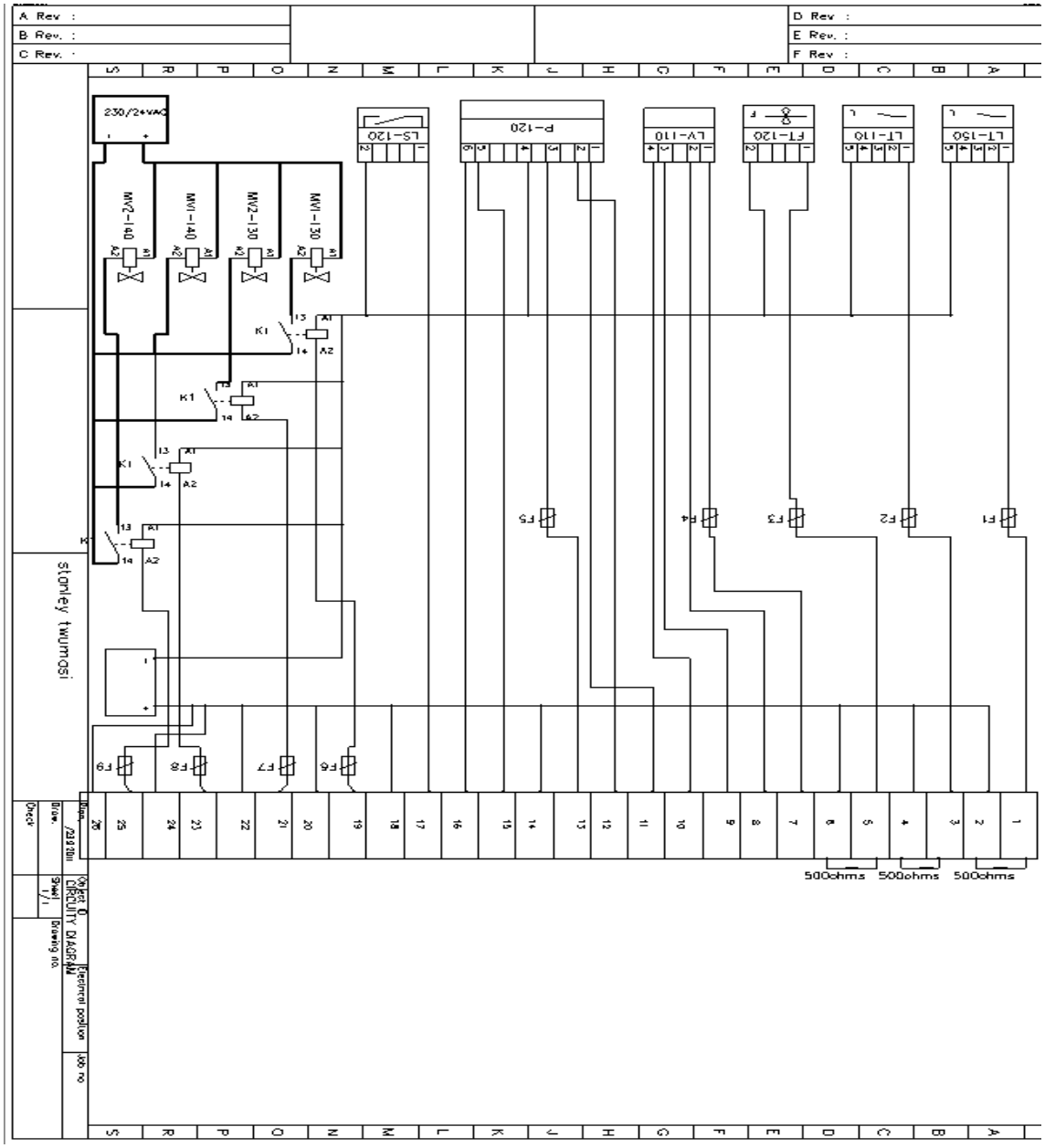


Figure 11 Circuit diagram

2.6 Instrumentation specification sheet

The objective of the instrument specification sheet is to list all the important details of the various instruments involved in the process (Table 2). (Battikha 2007, 266).

Table 2 Instrument specification sheet

Device		Amount	Manufact.	Supply	Signal	I/O	Note!
Pump CRE 4-20		1	Grundfors	230 VAC	0...10VDC RPM Control ON/OFF 24 VDC Fault Output	AO BO BI	max 2,2 l/s
Magnetic Valve L180B17-1(25) (filling)		2	Sirai	24 VAC	24 VDC	BO	
Magnetic Valve 6213/R1 (drainage)		2	Burkert	24 VAC	24 VDC	BO	
Level transmitter, (hydrostatic pressure) NAS 0,2A		2	Trafag	24 VDC	4...20 mA	AI	
Motor Valve	Valve VB7215-049 (DN32)	1	Regin			AO AI	0...2
	Actuator AQM24-05	1	Regin	24 VAC	Control 0...10VDC Position 4...20 mA		Run Time 108 s
MagneticFlowmeter MP150/8EVL		1	Valmet	230 VAC	3600puls/l		max 2,2 l/s
f/I-muunnin EM-M17b		1	Phoenix	24 VDC	4...20mA	AI	0...10V
Level Switch, Float Switch AKO 5315		1	AKO-elect.		Limit can be set	BI	Upper/Lower

2.7 I/O reservation table

The inputs and outputs reservation table shows how the channels of the bus terminals are assigned to the process sensors and actuators (Table 3).

Table 3 I/O reservation table

CHANNEL	KL3064(AIU)	KL4004(AOU)	KL1104(BIU)	KL2408(BOU)
1	LT-150	LV-110	P120(pump fault)	MV2-130
2	LT- 110	P -120	LS- 120	MV1-130
3	FT -120			MV2-140
4				MV1-140
5				PS -120
6				
7				
8				
BUS COUPLER	BK9000	BK9000	BK9000	BK9000
TERMINATOR	KL 9010	KL 9010	KL 9010	KL 9010

3 BUS COUPLER AND BUS TERMINALS

Bus Terminal Controller BK 9000 is a Bus Coupler with integrated PLC functionality which has a fieldbus interface for Ethernet. The BK 9000 is an intelligent slave that can be used as non-central intelligence in the Ethernet system. One unit consists of the Bus coupler (Figure 12), any number of bus terminals (Figures 13-16) between 1 and 64, and a bus end terminal (Figure 17). (Beckhoff Automation 2011).

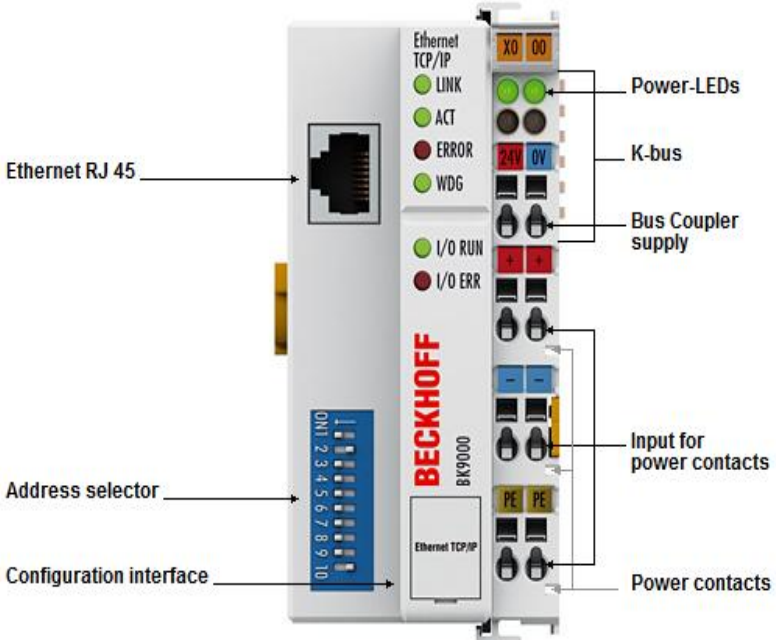


Figure 12 BK9000 Ethernet TCP/IP bus coupler (Beckhoff Automation 2011).

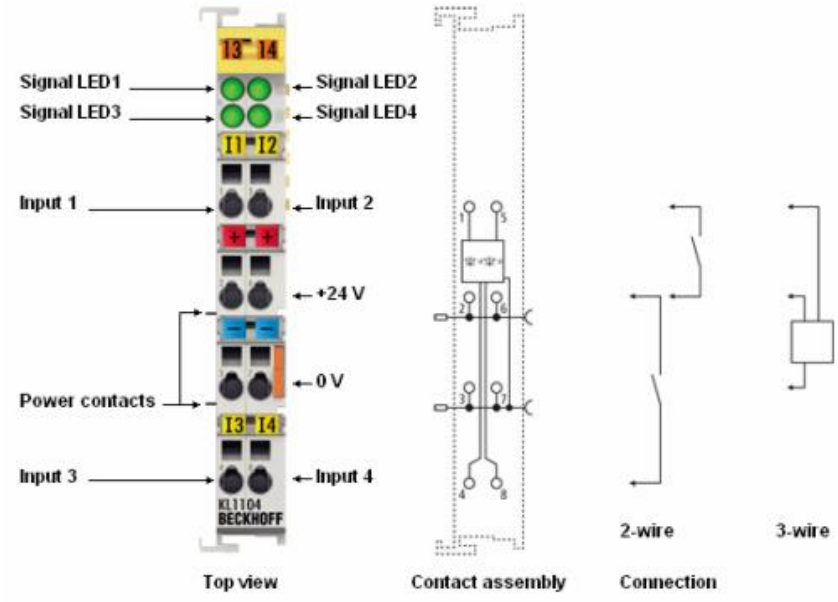


Figure 13 KL 1104 four channel binary input terminals 24VDC (Beckhoff Automation 2011).

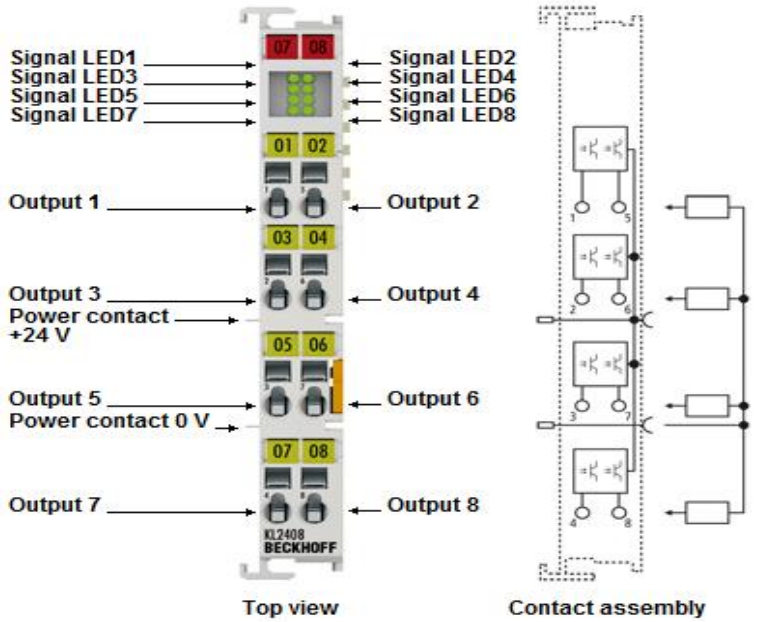


Figure 14 KL 2408 eight channel binary output terminals 24VDC (Beckhoff Automation 2011).

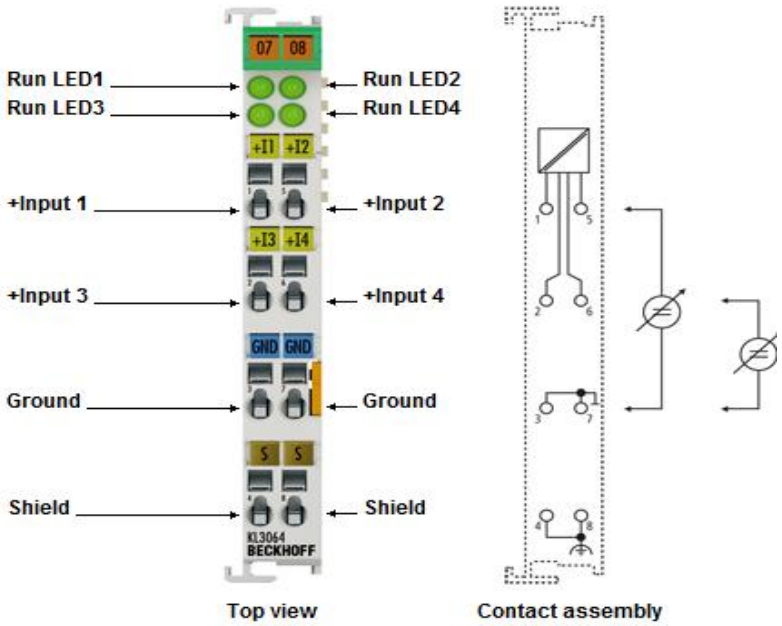


Figure 15 KL3064 four channel analog input terminals (0-10V). (Beckhoff Automation 2011).

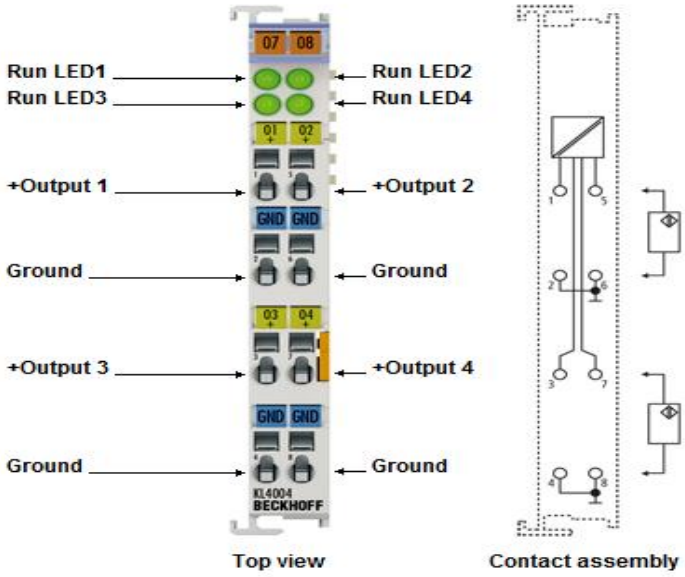


Figure 16 KL4004 four channel analog output terminals (0-10V). (Beckhoff Automation 2011).

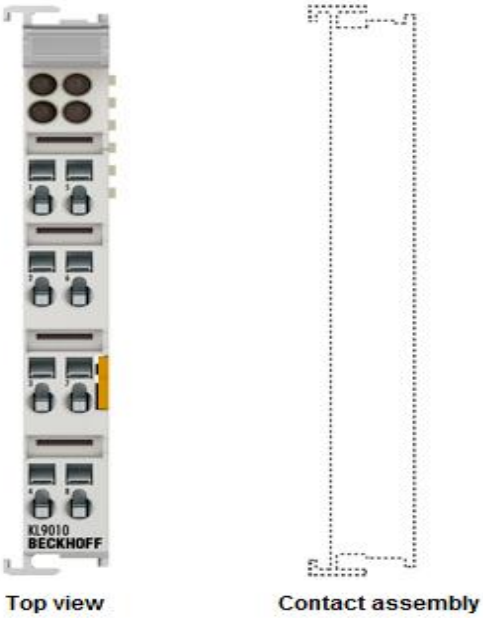


Figure 17 KL 9010 End terminal (Beckhoff Automation 2011).

3.1 Current to voltage signal conversion

The analog input terminal KL 3064 can read only voltages and the signals from the level transmitters and the flow transmitter are current signals (4-20mA). 500Ω resistors were placed across these devices and the voltages across the resistors measured and connected to the analog input terminal KL 3064 (Figure 18).

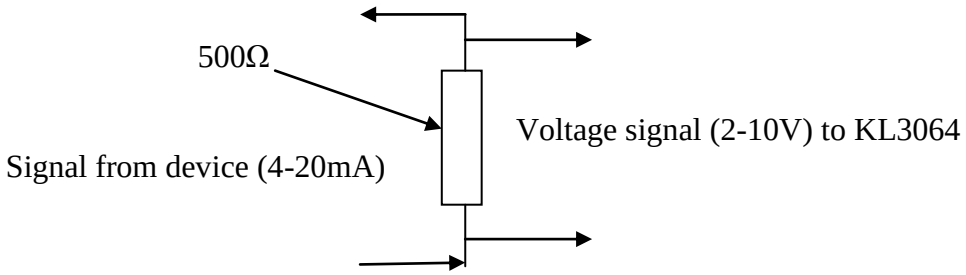


Figure 18 Current to voltage resistor converter

4 CONFIGURATION OF BECKHOFF I/O SYSTEM (RTU/PLC)

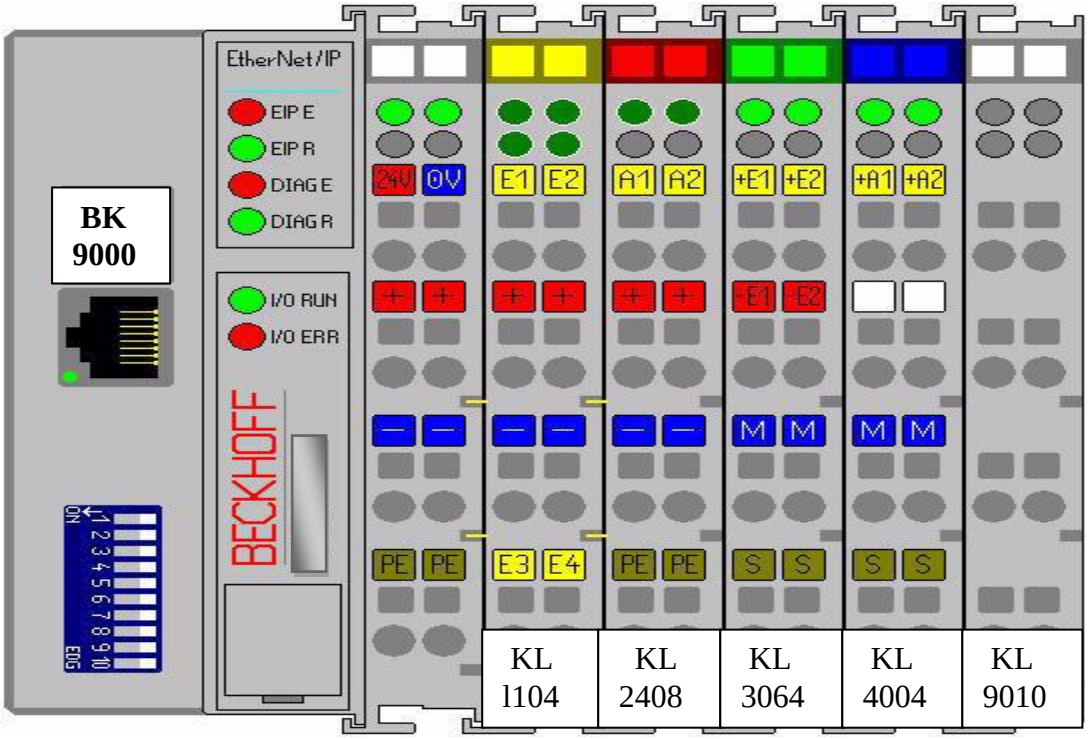


Figure 19 Bus coupler and bus terminals (Beckhoff Application note 2011)

The TwinCat (Beckhoff) PLC is used as the remote terminal unit to provide an interface to the instruments and control devices of the process (Figure 19).

The TwinCat system manager has a scan device function that enables it firstly to scan the PC for known devices and place them into the system manager under I/O devices.

Secondly the system manager scans the network connected to a device to locate any known boxes and placing them into the system manager under the appropriate device.

Finally it scans the terminals connected to a bus coupler and places them into the system manager also under the appropriate box.

This enables all inputs and outputs of the system to be made available for use by a higher level controller or a custom written external control program.

4.1 Scanning devices automatically

Devices can be scanned automatically or manually .To scan a PC automatically for known devices, firstly right clicking on I/O devices and then selecting “Scan Devices… (Figure 20).

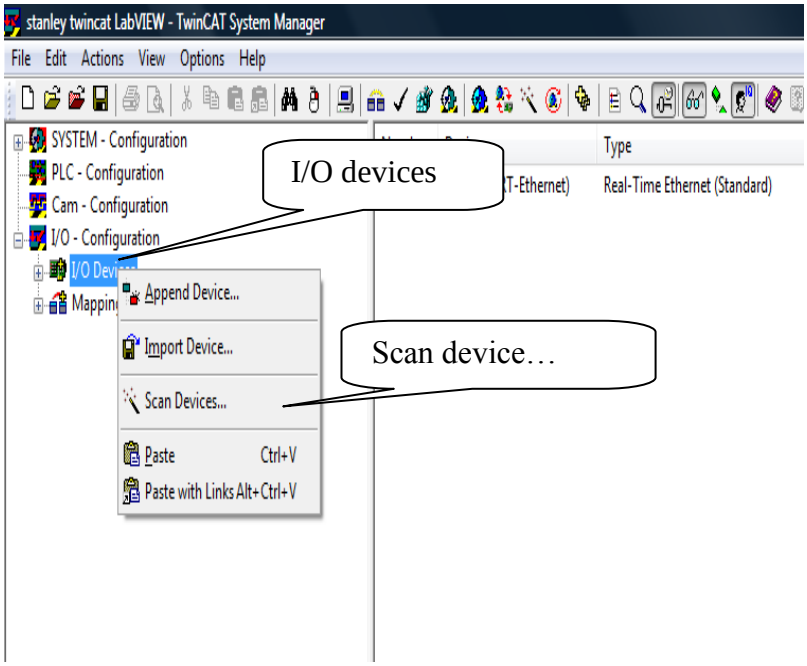


Figure 20 Scanning devices automatically

The system manager will scan the PC for any known devices, and place them under I/O device. In this case the system manager should find a real time Ethernet (standard) in the PC, with a BK9000 connected to the Ethernet port (Figure 21)

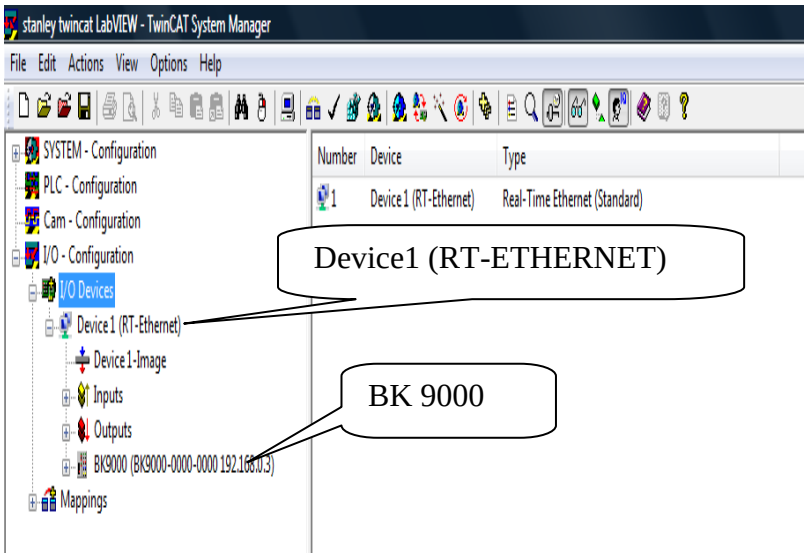


Figure 21 System manager detects device 1(RT-Ethernet)

4.2 Scanning boxes automatically

Right clicking on the device 1(RT-Ethernet) and then selecting "Scan Boxes… (Figure 22) the boxes can be automatically scanned.

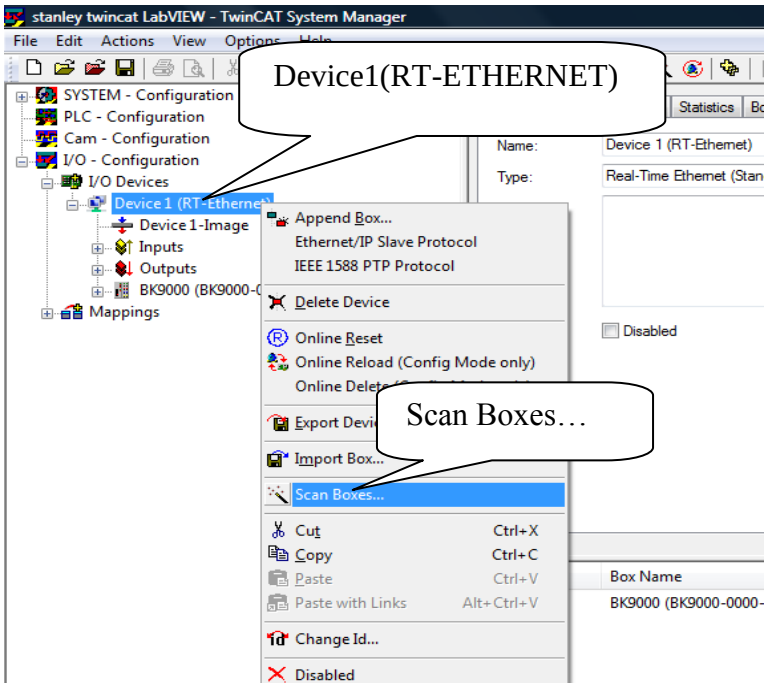


Figure 22 Scanning boxes automatically.

The system manager will scan the Ethernet network and display a list of detected bus couplers, by selecting BK 9000 and clicking OK. The system manager will place BK 9000 under device 1(RT-Ethernet) (Figure 24).

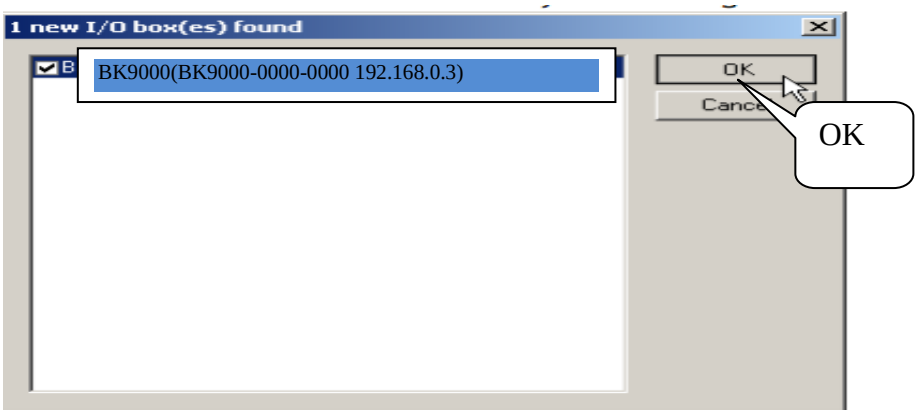


Figure 23 System manager detects bus coupler BK 9000 automatically.

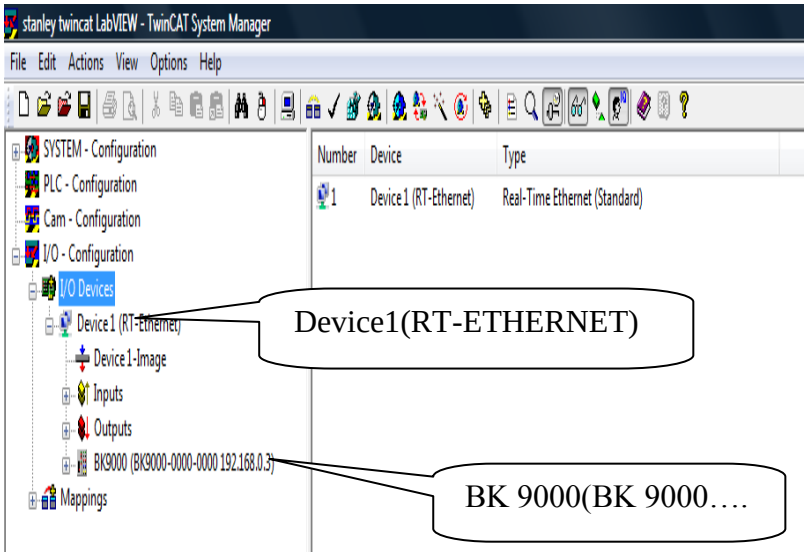


Figure 24 System manager places BK 9000 under device 1.

4.3 Scanning terminals automatically

To automatically scan the “terminals” connected to a bus coupler and to update the I/O terminal configuration, you have to first right click on the “box”. (BK9000-0000….) and then select “Scan Terminals…” (Figure 25).

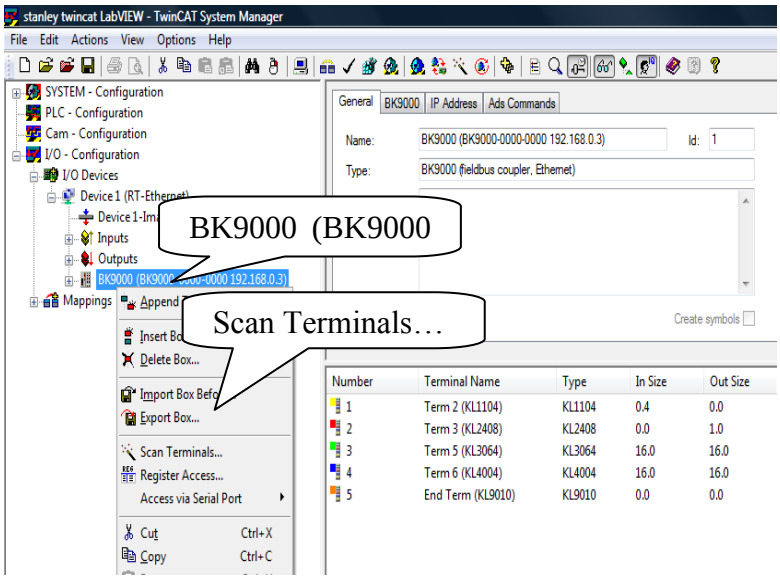


Figure 25 Scanning terminals automatically.

The system manager will scan the bus coupler (BK 9000) and update the I/O terminal configuration for the selected box (Figure 26).

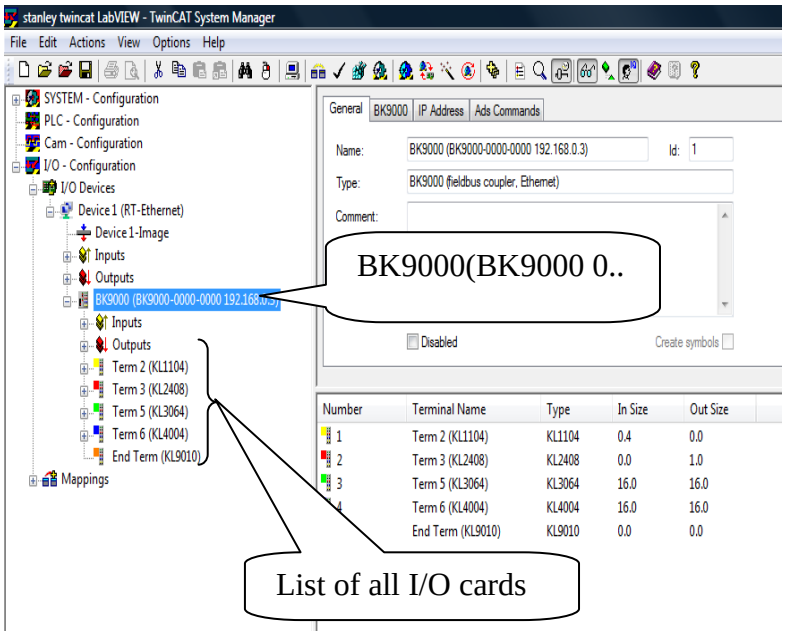


Figure 26 System manager detects all i/o cards automatically

4.4 Scanning devices manually

To add a “device” to system manager manually, you have to first right click on “I/O Devices” and then select “Append Device… (Figure 27).

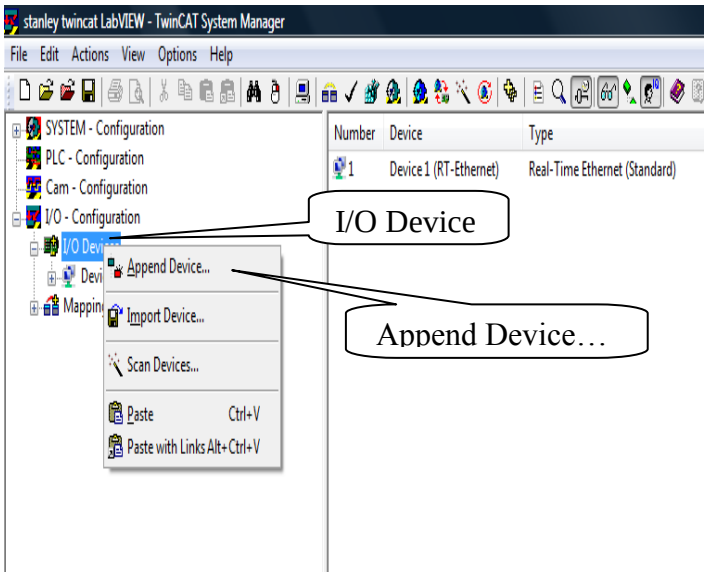


Figure 27 Adding device 1(RT-Ethernet) manually.

A list of all currently supported hardware devices appears in the "Insert Device" screen. You have to left click on Ethernet and select Real-time Ethernet (Standard) and click OK (Figure28).

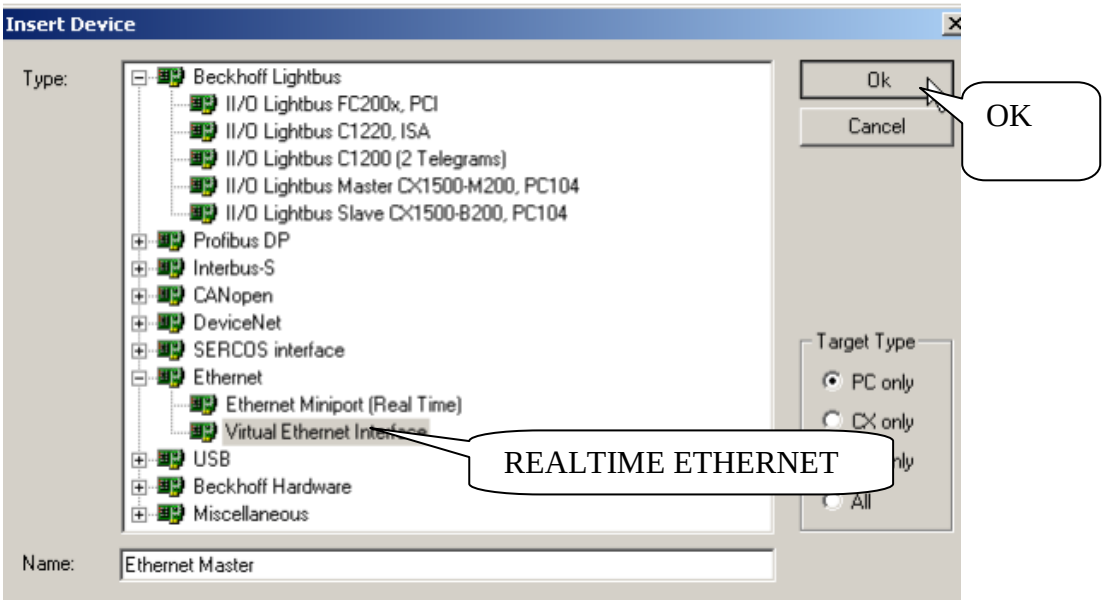


Figure 28 System manager detects device 1(RT-Ethernet).

Now the system manager will place Real-time Ethernet (Device1 RT-Ethernet) under I/O devices (Figure 29).

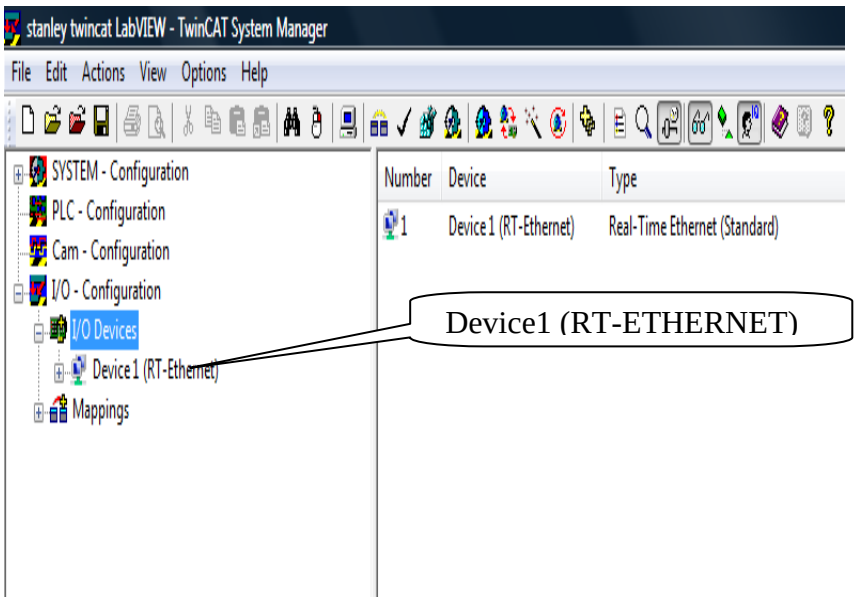


Figure 29 System manager places Device1 (RT-Ethernet) under i/o devices.

4.5 Scanning boxes manually

You have to first right click on the device 1(RT-Ethernet) and then select "Append Box…" (Figure 30).

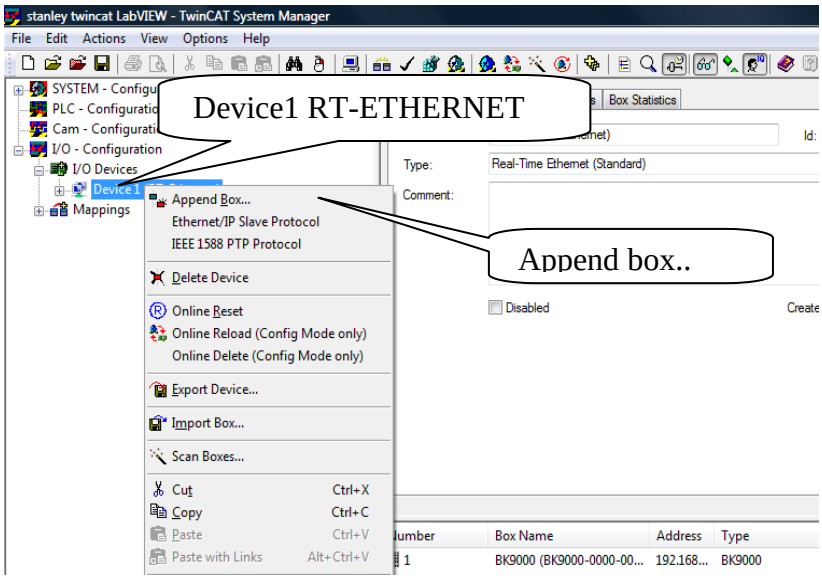


Figure 30 Adding the bus coupler manually.

The list of all currently supported hardware devices appears in the "Insert box" screen. Select BK 9000 and click OK (Figure 31).

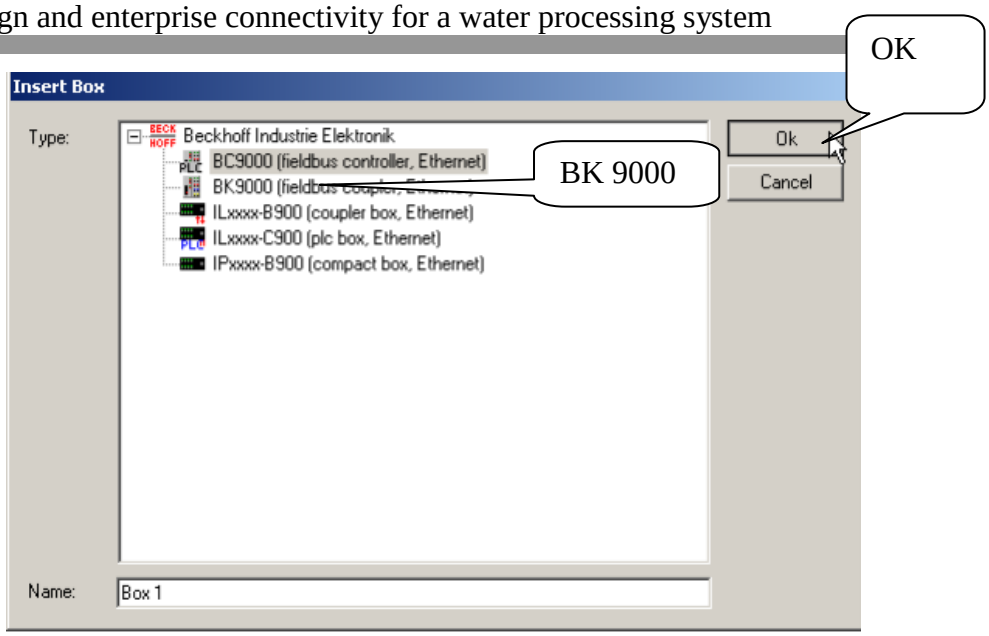


Figure 31 System manager detects bus coupler BK 9000.

The system manager shows device 1(RT-Ethernet) with one box (BK 9000) connected to it (Figure 32).

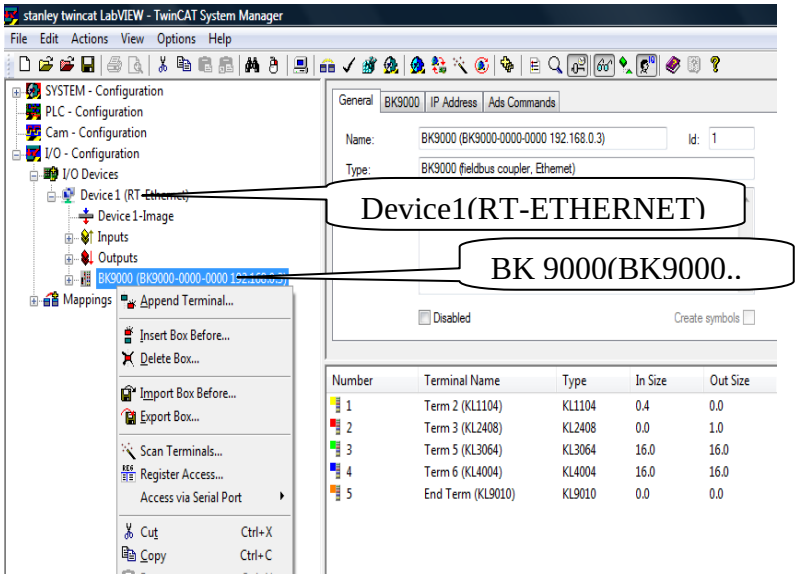


Figure 32 Bus coupler BK9000 is attached to device 1(RT-Ethernet).

4.6 Scanning terminals manually

To add terminals manually you first have to right click on the box BK 9000 then select the append terminal (Figure 33).

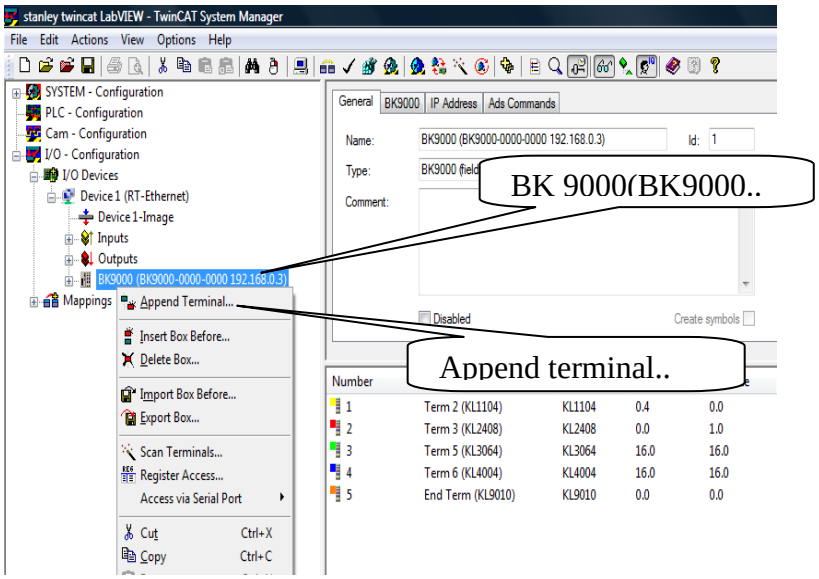


Figure 33 Appending terminals manually

Double click on the type of I/O required in the insert terminal pop-up menu .Name it and click ok. The system manager will add the terminal to the terminal configuration list for BK 9000 (Figure 34).

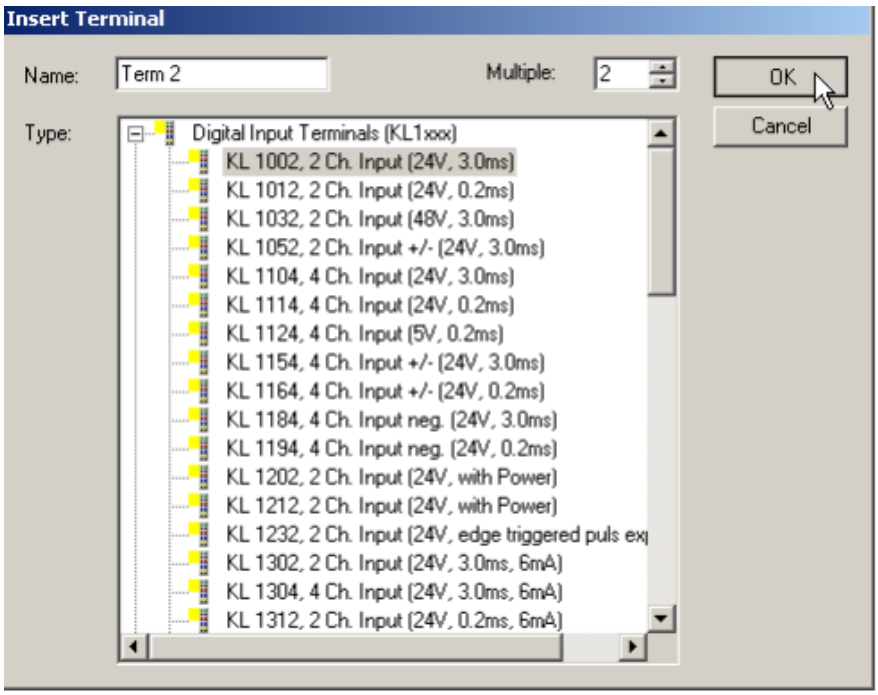


Figure 34 List of a bus terminals

5 LABVIEW AND TWINCAT DATA PATH IMPLEMENTATION

In order for live process variables to be seen and interpreted on the human machine interface created with LabVIEW a data path must be established between the two software.

Data exchange between LabVIEW and TwinCAT software's is implemented by ADS (Automation Device Specification) interface. An interface that permits communication between ADS devices (Figure35).

An object that is accessible via ADS and offers server services is known as an ADS device examples are TwinCAT and LabVIEW software's./4/

But in this case the TwinCAT is acting as a server while the LabVIEW acts as a client.

There is an ADS-OCX that provides methods, events and properties so that information can be exchange between TwinCAT and the LabVIEW software via the TwinCAT message router. (Beckhoff Automation 2011).

The ADS-OCX is executed as an Active X control element implemented with LabVIEW software. (Beckhoff Automation 2011)

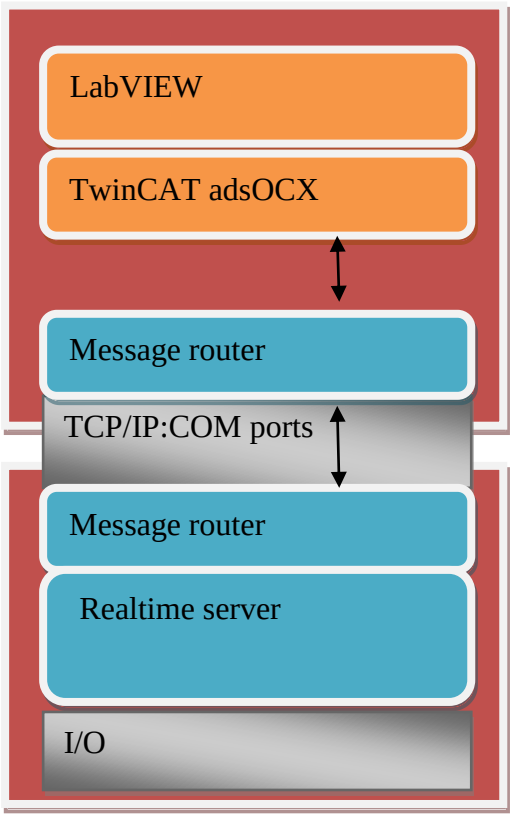


Figure 35 ADS device concept(Beckhoff Automation 2011).

6 GRAPHICAL PROGRAMMING.

LabVIEW has two main components to each virtual instrument (VI) the front panel **(**Figure 36) and the block diagram (Figure 37).

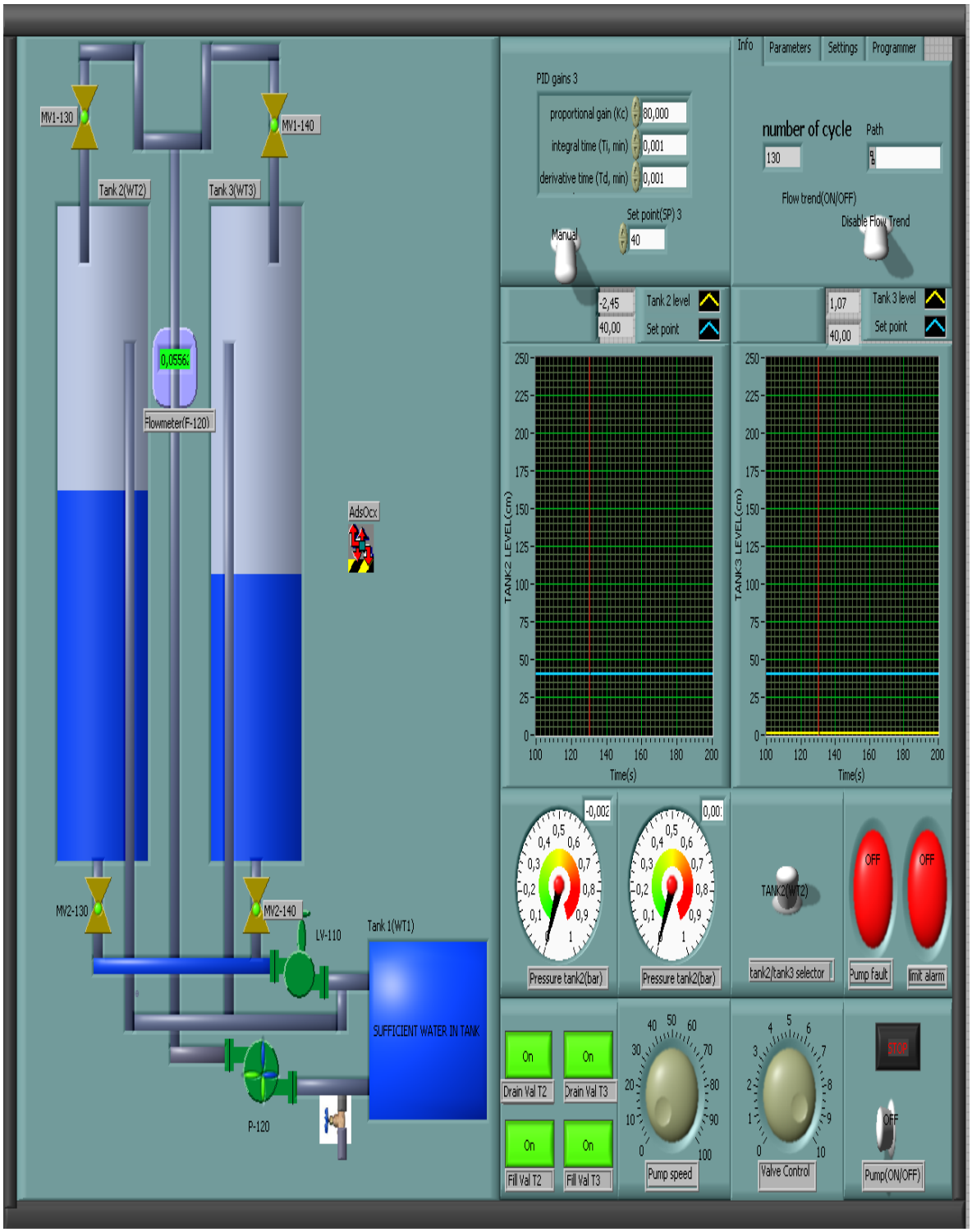


Figure 36 The water process human machine interface

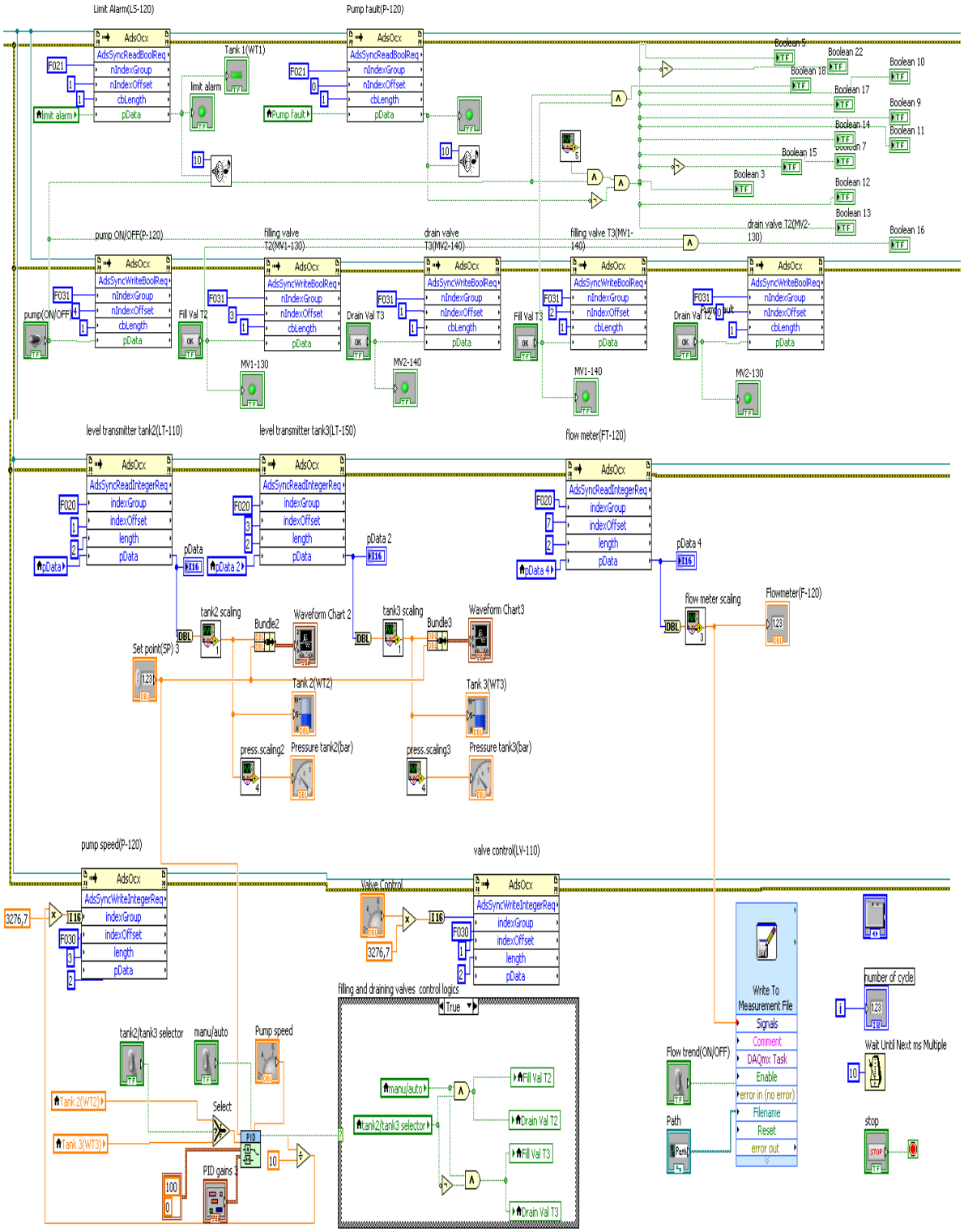


Figure 37 Graphical programming of the human machine interface

6.1 Inserting an a active X object

To start the programming, the first thing needed on the front panel is an ActiveX container (Figure 39). Right-clicking anywhere on the front panel brings up the controls pallet. Going to the .NET and active X pallet the active X container icon was chosen (Figure 38).

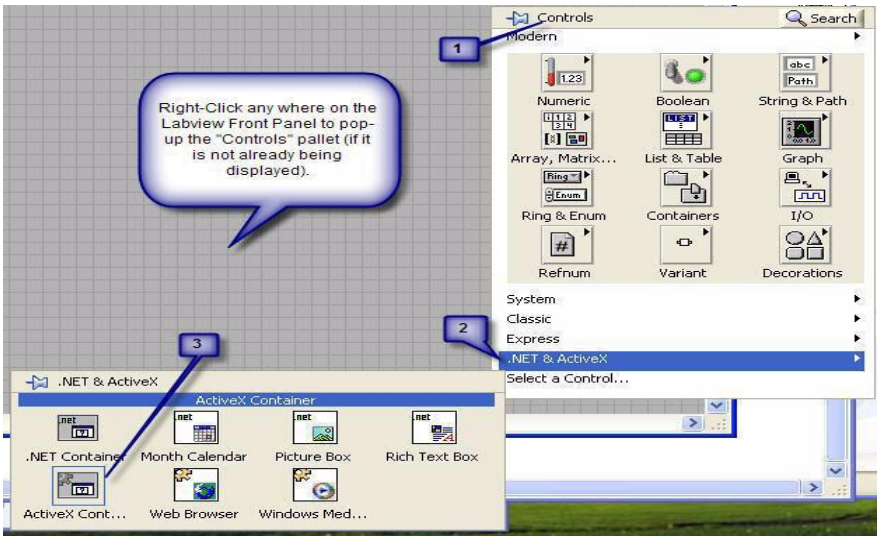


Figure 38 .Net and active x (Beckhoff application note 2011).

Moving the mouse over to the front panel and clicking **ActiveX Container** an empty ActiveX container should be on the screen (Figure 39).

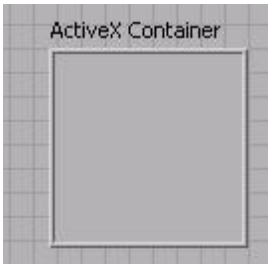


Figure 39 ActiveX container

An ActiveX Control needs to be placed inside the container. Right-clicking the mouse inside the active X container, a pop-up window appears and **Insert ActiveX Object...** was selected (Figure 40).

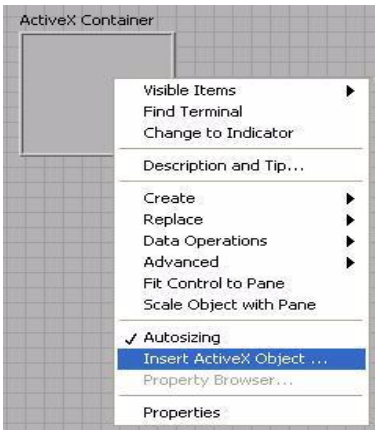


Figure 40 Inserting an active x object

A new pop-up window appears with a list of all available ActiveX controls. Click **AdsOcx** Control, then OK (Figure 41).

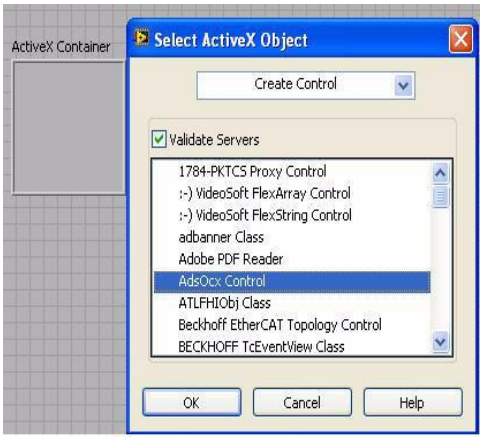


Figure 41 List of active x controls

Moving to the LabVIEW diagram window. The AdsOcx control should appear in the diagram .This is the graphical reference to the Beckhoff Ads Ocx control.

6.2 Adding a property

Next is adding blocks to expose the **Properties** and **Methods** the AdsOcx control allows to interface within LabVIEW. Right-clicking in an open area of the diagram window displays the **Functions** pallet (if it is not already being displayed). Moving the mouse over the **Connectivity** pallet item displays the **Connectivity** pallet. Hovering the mouse over the **ActiveX** pallet item shows all available **ActiveX** tools. The **Property Node** icon was chosen by clicking on it (Figure 42).

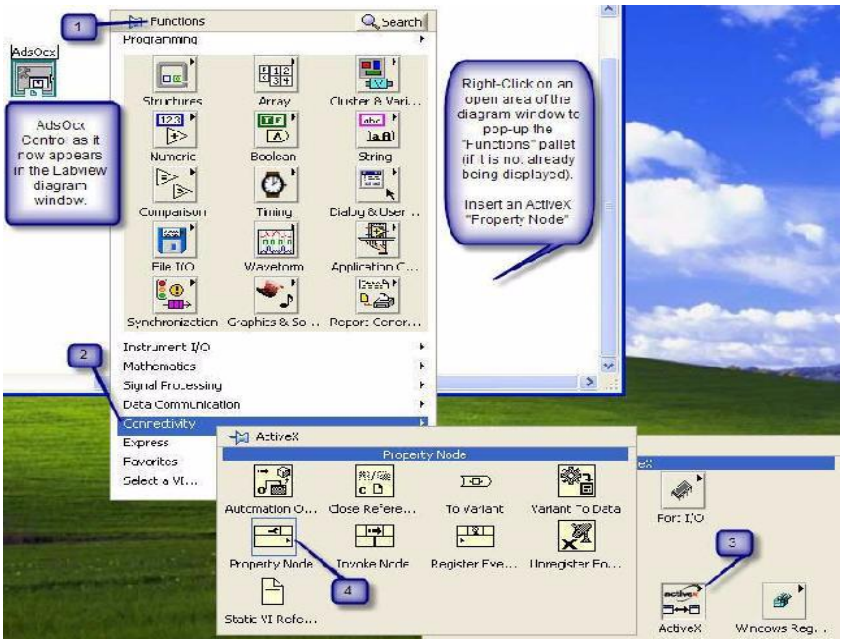


Figure 42 ActiveX pallet and active X tools (Beckhoff Application note 2011)

6.3 Adding AdsAms Server NetId and AdsAms Server Port Properties

Placing the mouse on the **Properties** item in the pop-up and Left-Clicking brings a list of all items that are available. Selecting **AdsAmsServerNetId** and left-clicking replaces the item that was previously shown when the Property Node box was first wired to the **AdsOcx Property Node** icon. A second Property item was added from the list. By right-clicking in the pink area again and choosing **Add Element** from the list of selections (Figure 43).

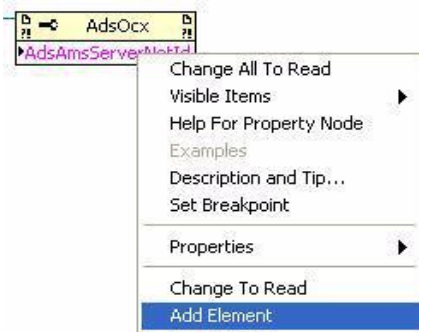


Figure 43 AdsAms Server NetId and element adding

The same item is now in the list twice. Right-clicking on the new, lower pink box, brings up the pop-up window again. The **Properties** item was selected. Scrolling down the list of available properties the **AdsAmsServerPort** was chosen (Figure 44).

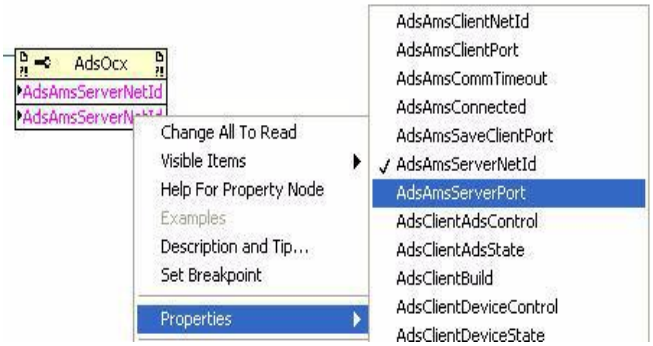


Figure 44 AdsAms Server Port

Now the property item can be written to, by right-clicking in the pink area and selecting **Create Constant** from the pop-up menu (Figure 45). Repeat the same steps for the blue property item.

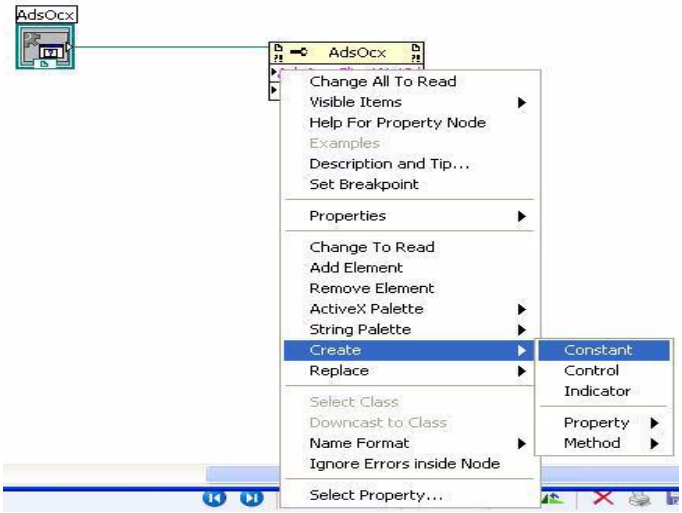


Figure 45 Creating constant

A box is now available for entry of data to both the **AdsAmsServerNetId** and **AdsAmsServerPort.**

6.4 Entering data for AdsAms Server NetId

To find out what to enter for the **AdsAmsServerNetId**, the TwinCAT System Manager was consulted. By right-clicking on the Beckhoff **Twin-CAT icon** in the computer's system tray area and clicking the properties item (Figure 46).

Figure 46 TwinCAT icon properties selecting

The TwinCAT system properties box will appears (Figure 47). The **AMS Router** tab was selected and the **AMS Net Id: address** copied out of this window and the value pasted in the pink **AdsAmsServerNetId** box in the Labview diagram (Figure 48).

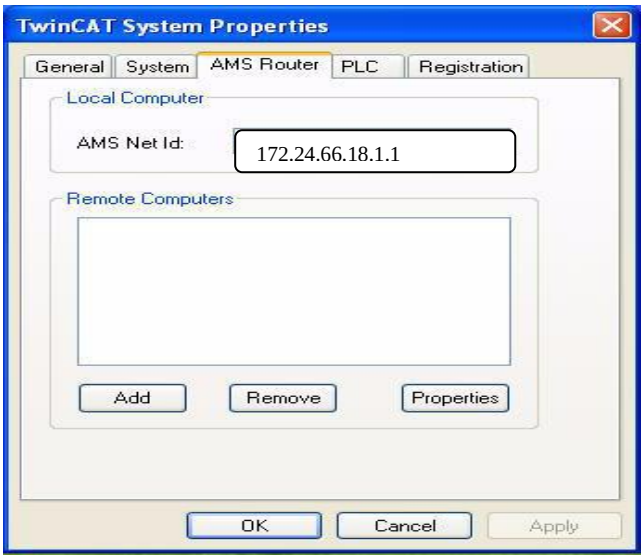


Figure 47 TwinCAT system properties and Ams router tab

6.5 Entering data for AdsAmsServer Port

The **Port** number needs to be 301 to get the variables in the TwinCAT system manager's **Additional Task**.

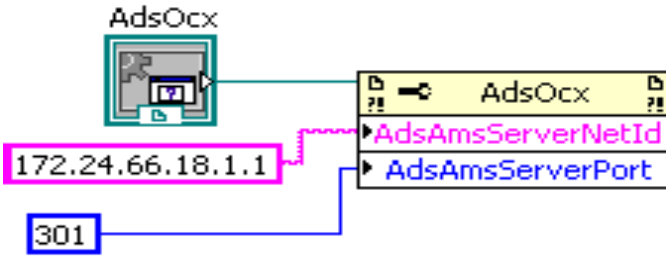


Figure 48 AdsAms Sever NetId and AdsAms Server Port numbers

6.6 Adding a flat sequence structure

The OCX control only needs to be referenced once for all the items to be exposed. A structure called a flat sequence was put around the code created so far in the LabVIEW diagram to make sure this section happen only once. Right-clicking in an open area of the LabVIEW diagram window displays the functions pallet. Moving the mouse over the **Structures** item, a new pop-up window appears. From that window the **Flat Sequence** icon was selected.

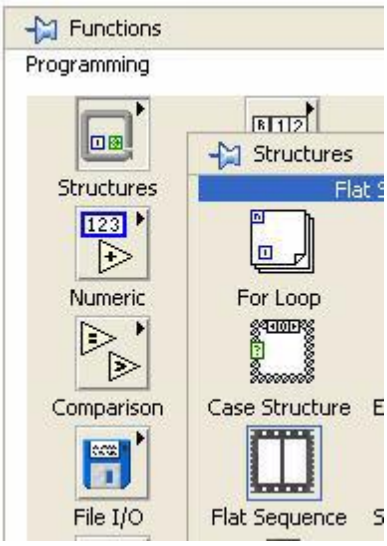


Figure 49 Function pallet and flat structure

The **Flat sequence structure** was rubber-band completely around the items created in the diagram .The LabVIEW connections diagram should now look like the picture in (Figure 50).

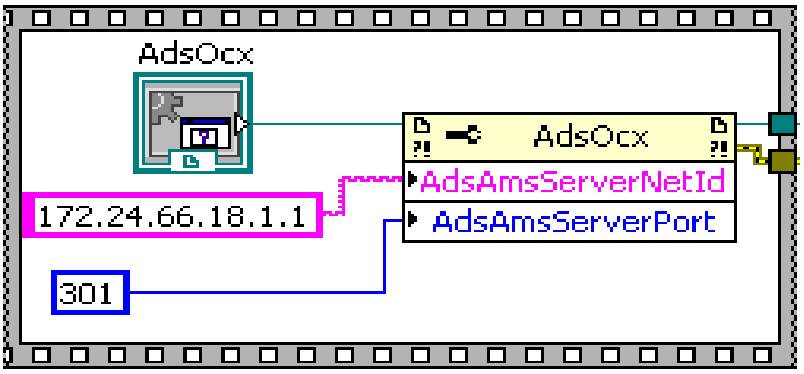


Figure 50 AdsOcx Connection diagram for TwinCAT referencing

The number written to the AdsAms Server Netld will be what is needed to access the server on the computer. By left-clicking on the flat sequence structure and selecting add frame after. The flat sequence structure is divided into two sections.

The first section contains the code created. This allows the code in the first section of the flat sequence to be executed first before the codes in the second section.

6.7 Creating a while loop

By right-clicking somewhere inside the second section of the flat sequence structure brings up the **Functions** pallet (Figure 51). Moving the mouse over the **Structures** item, the **Structures** pallet appears. By left-clicking and selecting the **While Loop** icon, the while loop was used to rubber banded a fairly large area inside the second section of the flat sequence structure. This space is where most of the coding will be done.

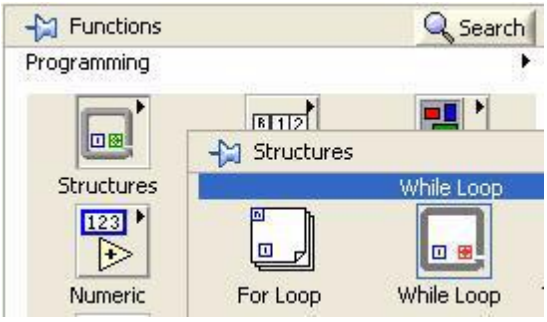


Figure 51 Function pallet, structure pallet and while loop icon

The outputs of the **AdsOcx** properties box was then wired to the edge of the **While Loop** structure (Figure 52).

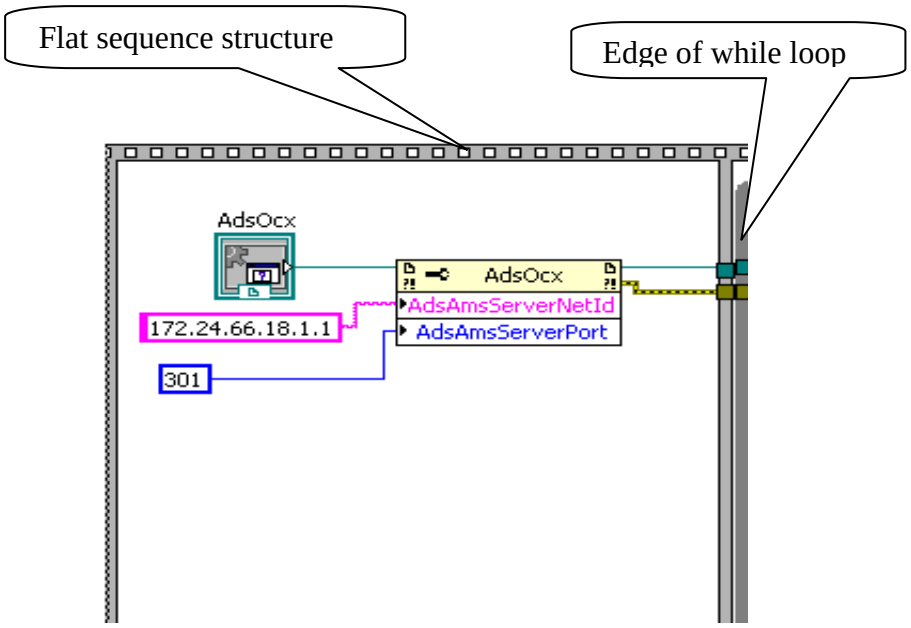


Figure 52 AdsOcx properties box

By this a "tunnel" is created to pass the data from one structure to the other. The Blue-Green line is the reference to the AdsOcx Control and the Pink line is what LabVIEW calls a "cluster" of data. A cluster is just an array that contains differing data types.

To expose the **Methods** the **AdsOcx** control uses to interface within LabVIEW an open area of the diagram window was right clicked to display the **Functions** pallet (Figure 53). Moving the mouse over the **Connectivity** pallet item displays the **Connectivity** pallet and moving the mouse over the **ACticeX** pallet item displays all available **ActiveX tools** and then the **Invoke Node** icon was selected.

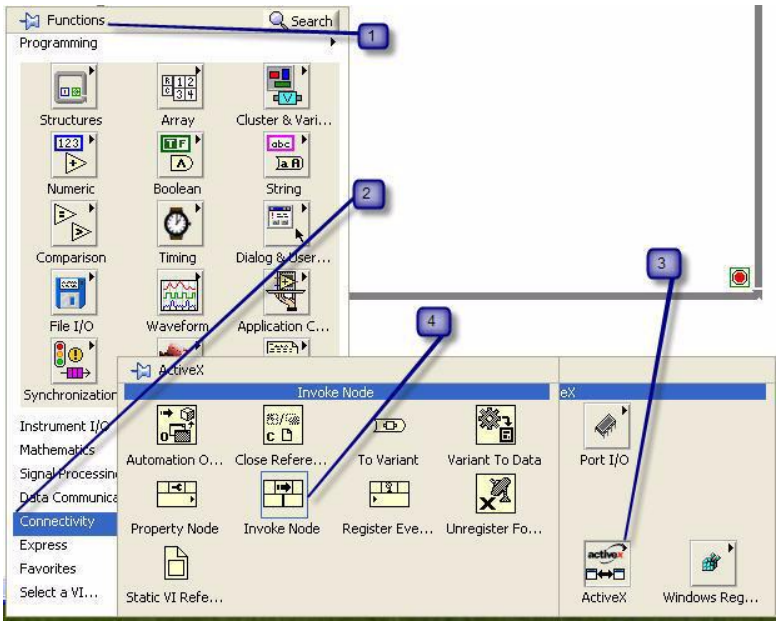


Figure 53 ActiveX pallet and active X tools

Changing to the wiring tool on the diagram, the boxes on the left of the **While Loop** structure was wired to the inputs of the new **Ads Ocx Invoke Node** block and the block named limit switch. Pressing and holding down the control key and left-clicking on the limit switch block diagram created, a copy was then dragged. The output of the limit switch block diagram was then wired to the input of the second diagram and named pump fault Figure 54).

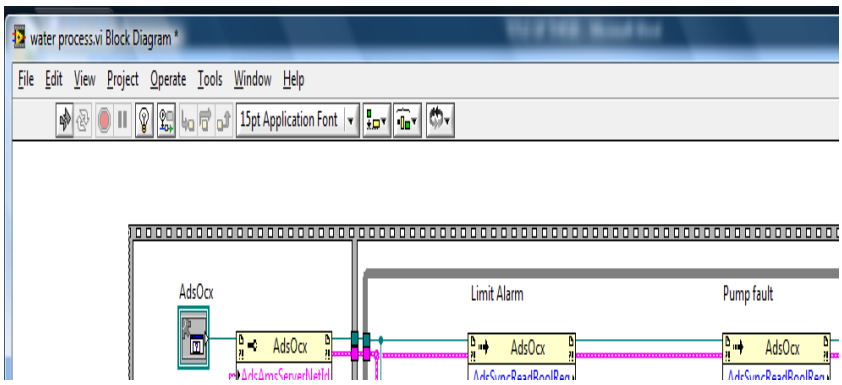


Figure 54 Creating AdsOcx invoke nodes for binary inputs block diagrams

6.8 Coding binary input terminals

There are only two binary inputs to be coded these are the limit switch and the pump fault signal. This was done by right-clicking in the **Method** text area of the limit switch block diagram by doing this a pop-up menu appears as shown in figure 55.

By picking the **Methods** item a second window appears with all the possible methods the **AdsOcx Control** can expose. Scrolling down the list of items the **AdsSyncReadBoolReq** item was found, clicked and selected (Figure 55).

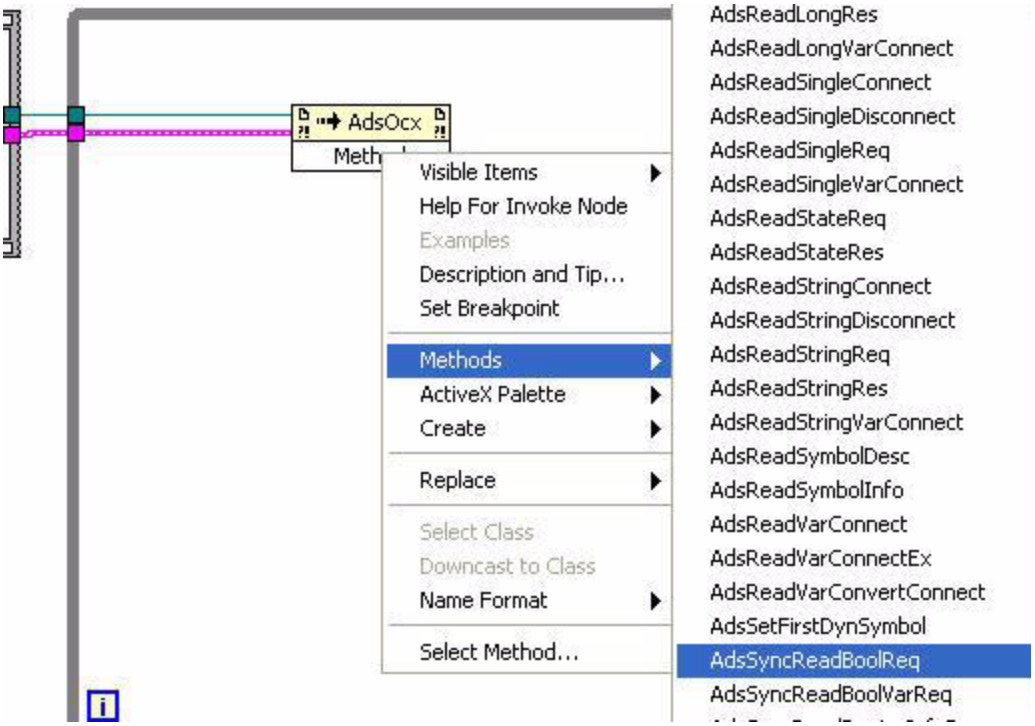


Figure 55 AdsOcx control method adsSyncReadBoolReq

The **AdsOcx Invoke Node** box now changes to show the items required to implement this method. The picture below (Figure 56) shows the coding of the limit switch. The same procedure was repeated on the pump fault block diagram (Figure 57).

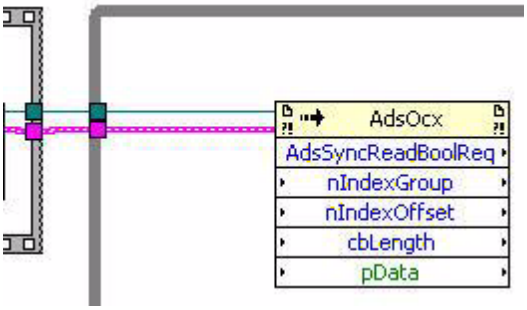


Figure 56 Limit alarm block diagram

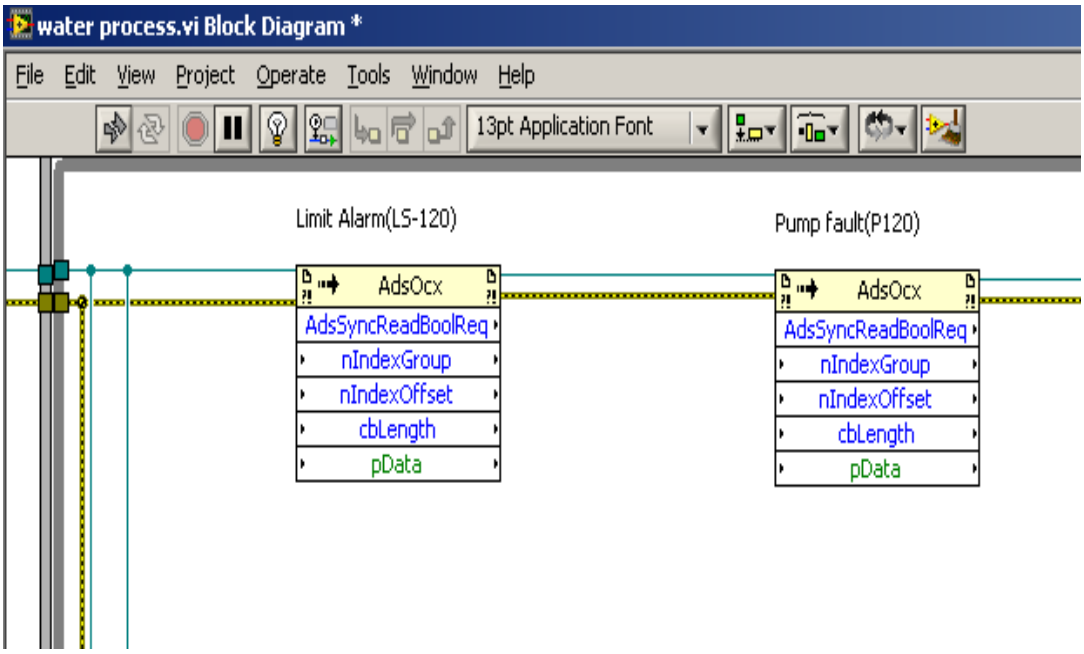


Figure 57 Limit alarm and pump fault block diagram

6.9 Coding binary output terminals

There are five binary outputs to be coded. These are the two filling and two draining valves for each tanks, and the binary output to turn the pump on and off.

These binary outputs were created by right-clicking anywhere inside the while loop to display the **Functions** pallet. Hovering the mouse over the **Connectivity** pallet item displays the **Connectivity** pallet.

Then hovering the mouse over the **ACticeX** pallet item displays all available **ActiveX tools**. The **Invoke Node** icon was then chosen as it was done when coding the binary inputs in figure 54. Four copies of this was dragged and wired together as shown in figure 58.

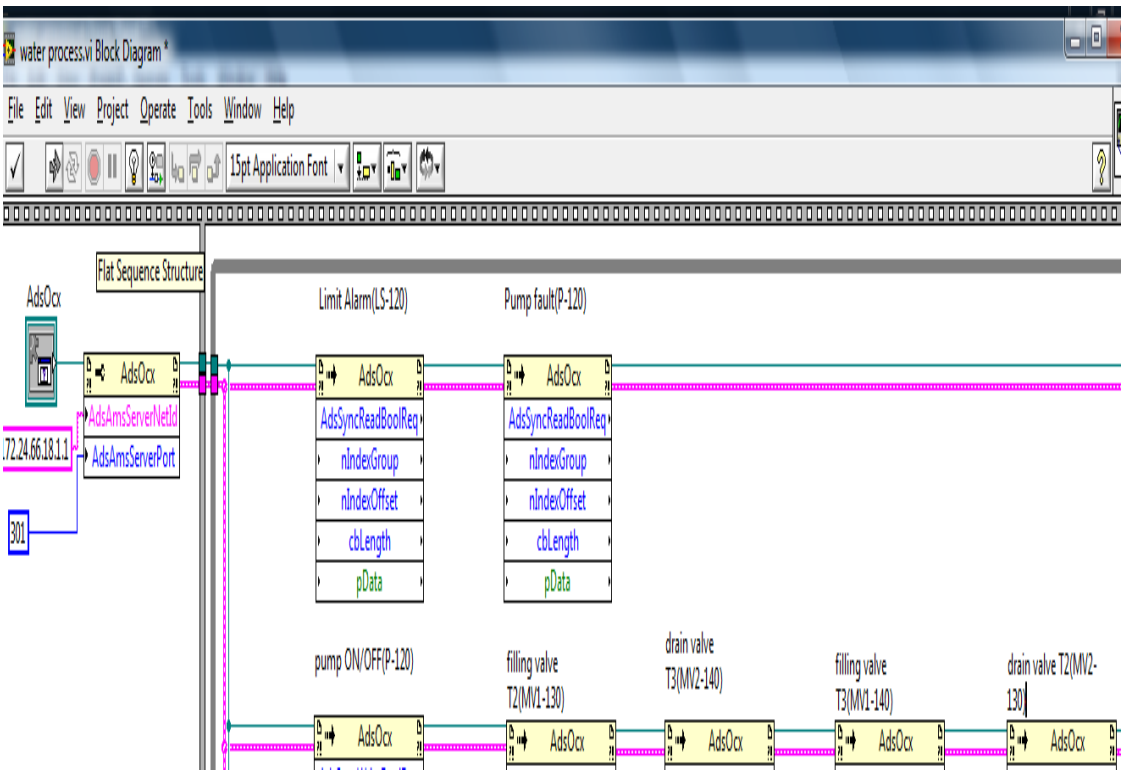


Figure 58 Creating AdsOcx invoke nodes for binary outputs block diagrams

Now right-clicking in the **Method** text area of the pump (ON/OFF) block diagram a pop-up menu appears.

Picking the **Methods** item and scrolling down the list of items .The method **AdsSyncWriteBoolReq** was chosen this time (Figure59).

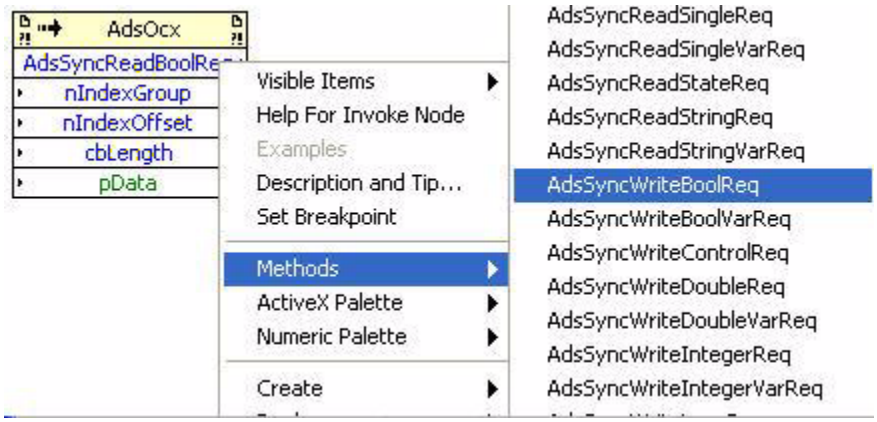


Figure 59 AdsOcx control method adsSyncWriteBoolReq

This procedure was repeated to all the coded binary outputs block diagrams.

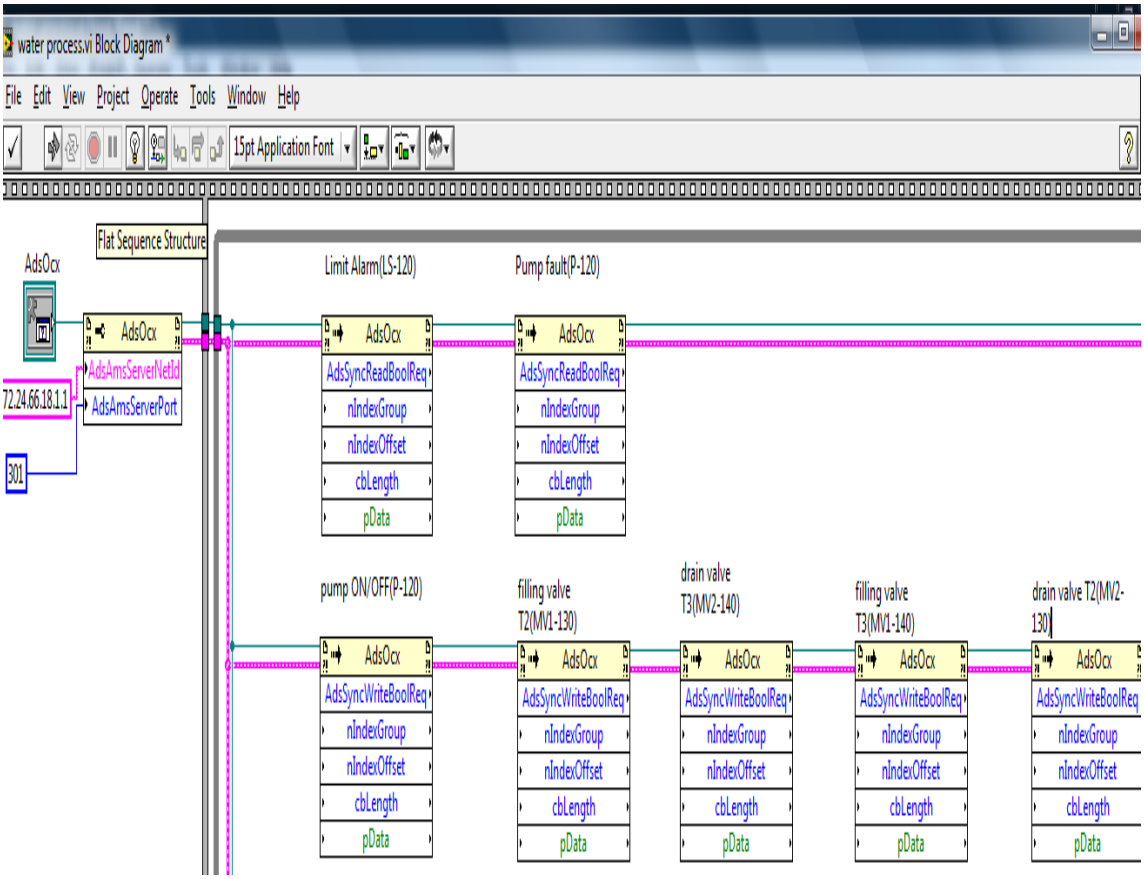


Figure 60 Binary inputs and outputs block diagrams

6.10 Coding analog input terminals

There are three analog inputs to be coded. These are the two level transmitters each for the two tanks, and the flow meter transmitter.

The same steps were repeated as in our previous coding, this time selecting the method **AdsSyncReadIntegerReq.**

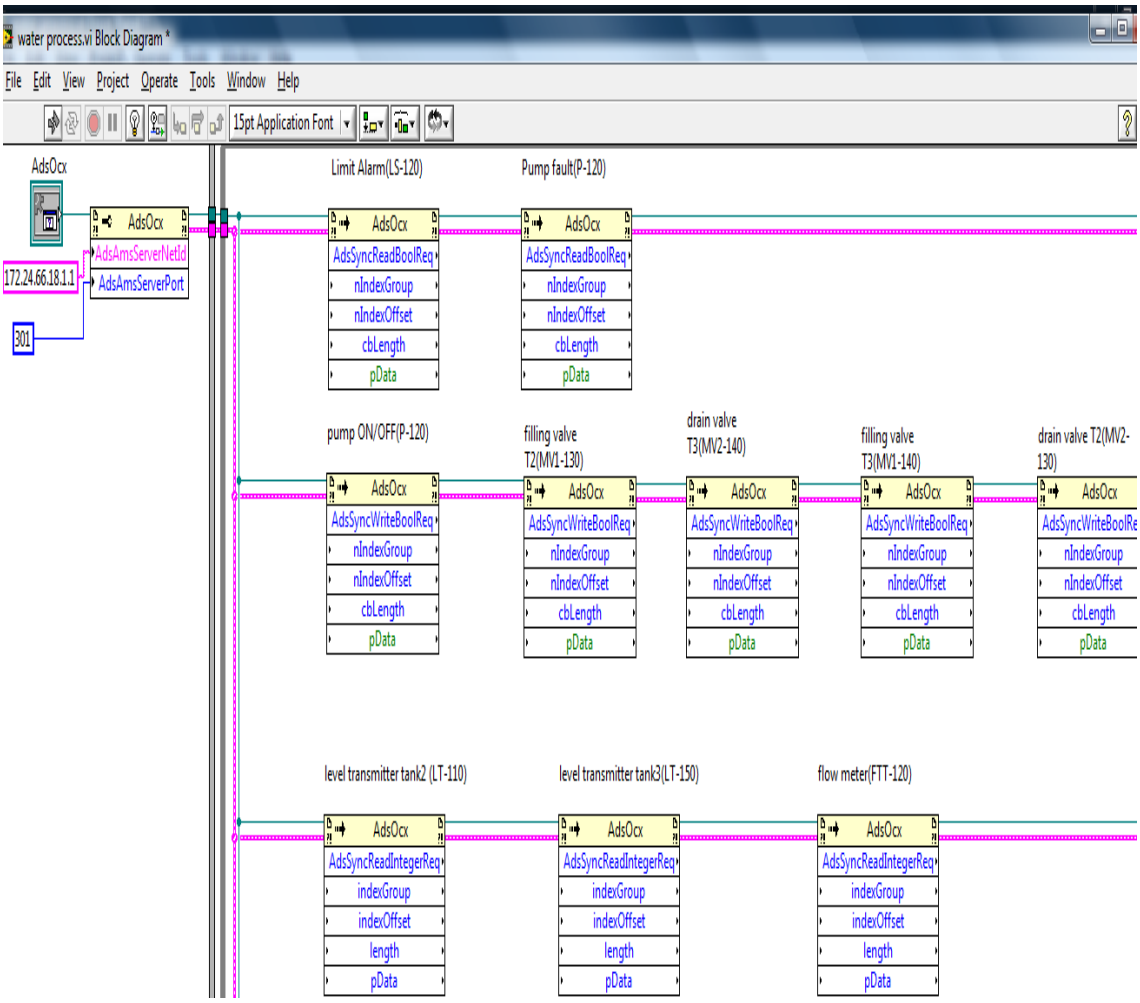


Figure 61 AdsOcx invoke nodes and AdsSyncReadIntegerReq for analogy inputs block diagrams

6.11 Coding analog output terminals

There are two analog outputs to be coded. The code controlling the speed of the pump and the code controlling the control valve. Two block diagrams one for the pump speed and the other for the control were created and wired together. By right-clicking in the **Method** text area of the pump speed block diagram a pop-up menu appears.

Picking the **Methods** item and scrolling down the list of items. The **AdsSyncWriteIntegerlReq** method was chosen.

The same steps were repeated on the valve control block diagram. This is how the coding should look like (Figure 62).

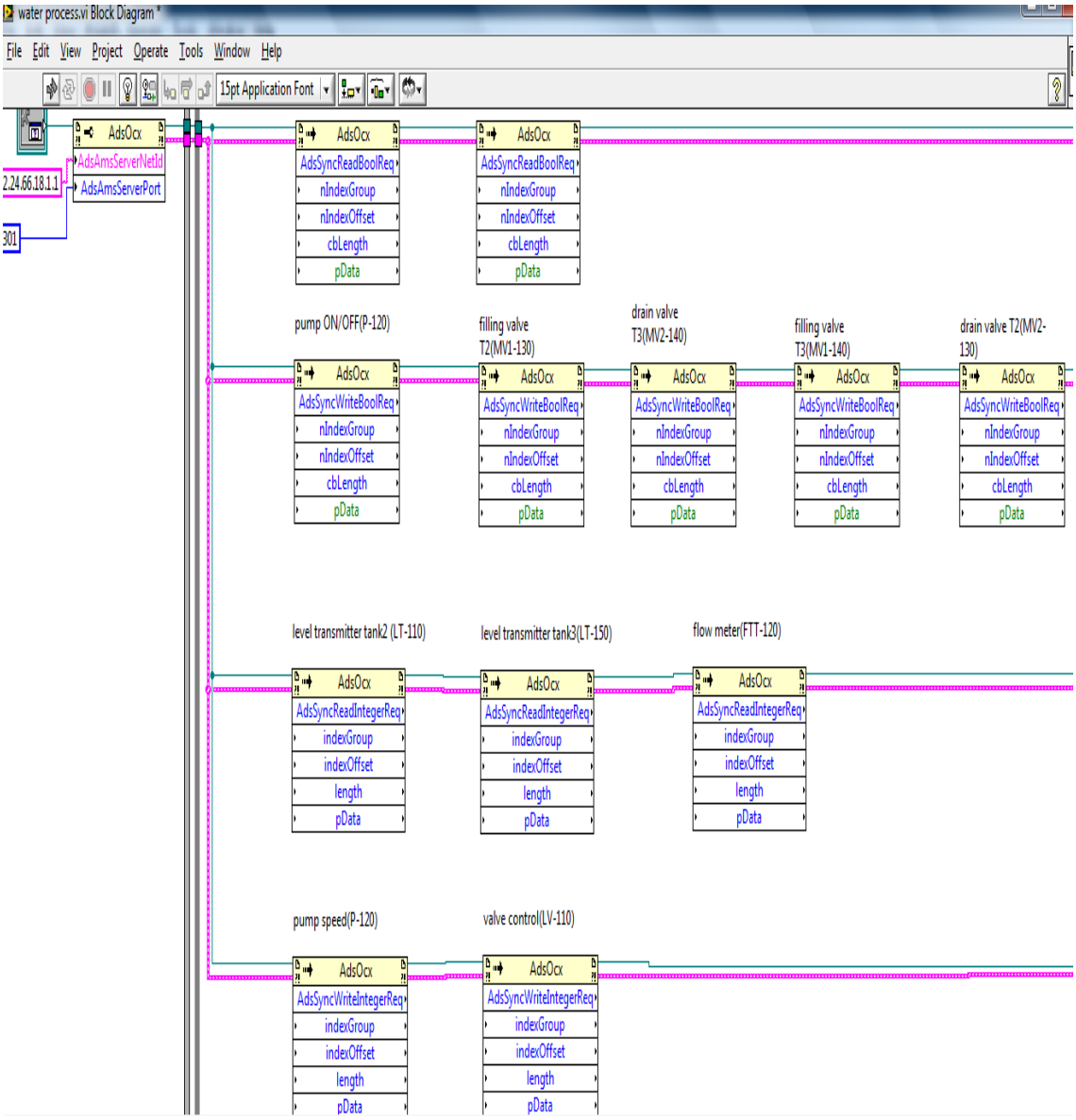


Figure 62 AdsOcx invoke nodes and AdsSyncWriteIntegerReq for analogy outputs block diagrams

6.12 Adding constants to coded terminals

All the block diagrams can be written to or read from. Constant values for **nIndexGroup**, **nIndexOffset**, and **cbLenght** are important. These constants were created by right clicking on the inputs to these constants, selecting create and choosing constant. The constants created are automatically wired to the appropriate input (Figure 63).

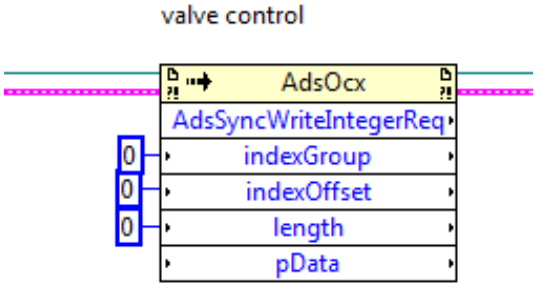


Figure 63 Creating index Group, index offset and length constants

It is necessary that the **nIndexGroup** should be change as a HEX value to match the TwinCAT System Manager item being referenced.

This was done by right clicking on the **indexGroup** letter box. A pop-up screen appears and format and precision was selected (Figure 64).

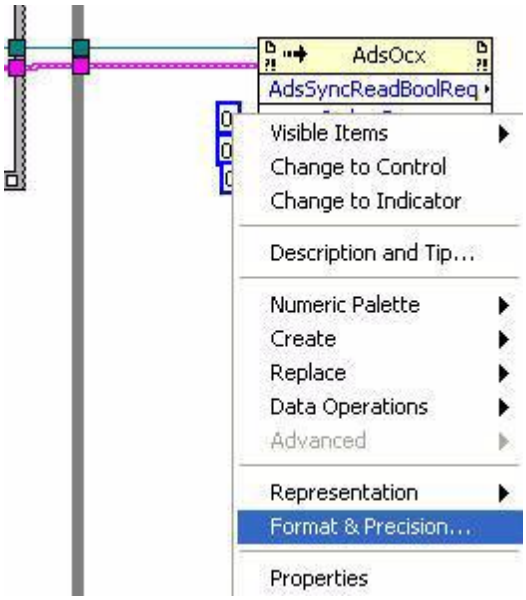


Figure 64 Changing to Hexadecimal value

A numerical constant properties window appears .Hexadecimal was selected and ok pressed (Figure 65). Repeat the same procedure to all the block diagrams.

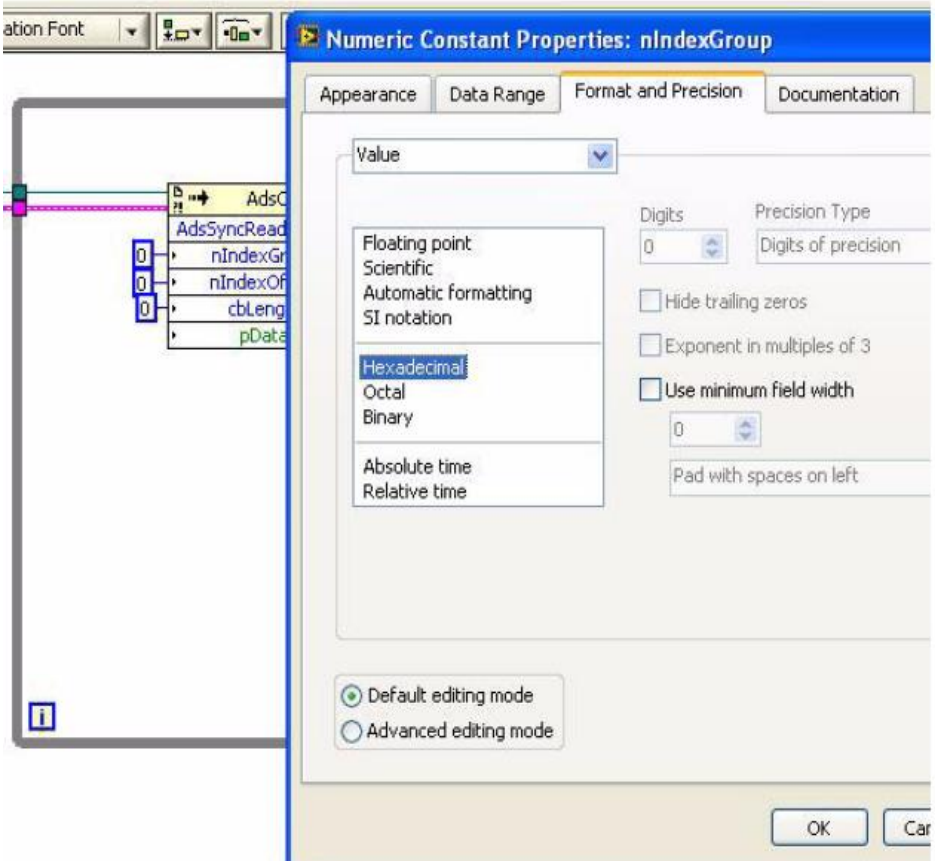


Figure 65 Numerical constant properties

In the TwinCat system manager expanding the additional task, task 1, and finally the inputs exposes all the configured input terminals.

To have the rest of the constants that needs to be entered into a block diagram the appropriate terminal must be selected to expose these constants.

With the Alarm limit (limit switch of tank1 WT1) highlighted the system manger exposes all the variables related to the alarm limit block diagram that needs to be entered as shown in figure 66.

The same procedure must be followed by the rest of the inputs to expose the constants that needs to be entered:

Likewise expanding the output exposes all output units. The same steps must be repeated as it was done to the input units.

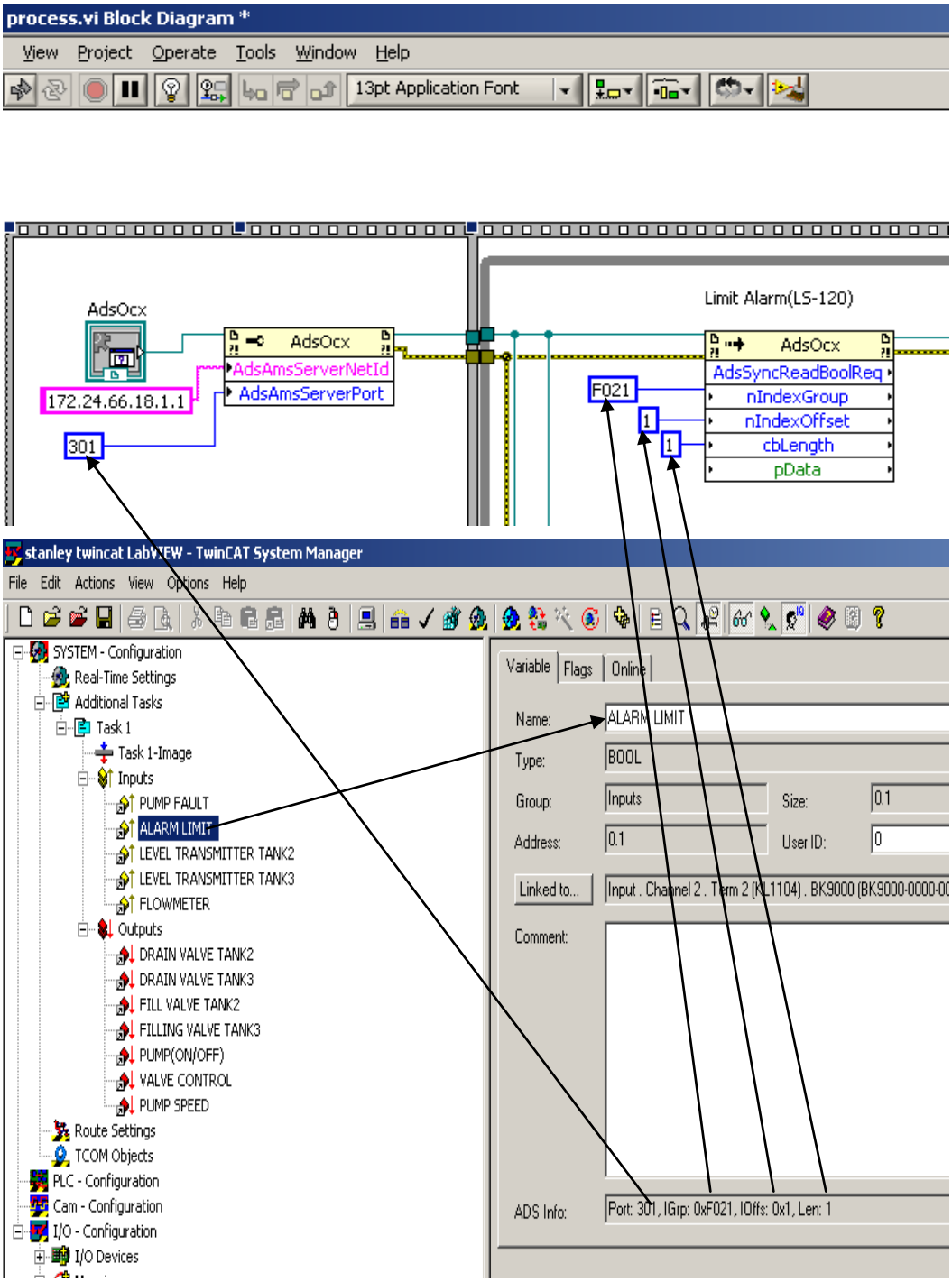


Figure 66 Filling in input terminals constants exposed by the TwinCAT system manager

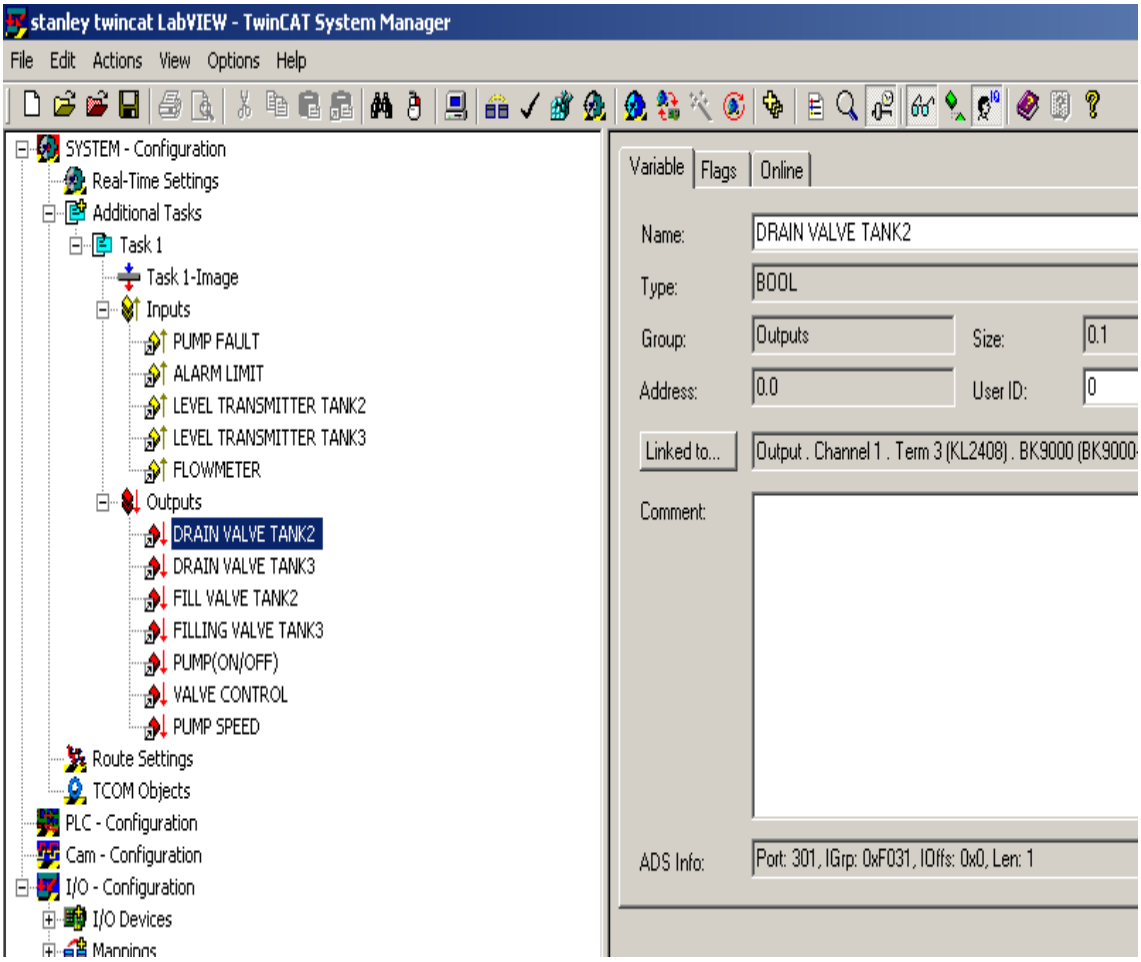


Figure 67 Filling in output terminals constants exposed by the TwinCAT system manager

7 INDICATORS, DATA DISPLAYERS AND ALARMS

All process variables must be displayed in an easily understood graphics form in the LabVIEW front panel and the right controls or indicators must be created. These indicators and controls are found inside the controls palette from the LabVIEW front panel.

7.1 Pump fault indicator and alarm

To indicate on the HMI When there is a pump failure a pump fault indicator was created. This was done by right clicking on the pump's fault block diagram pData letter box, selecting create and choosing indicator. The indicator is automatically wired to the output of the pData.

Right clicking on the indicator and selecting create and choosing local variable creates a local variable and then right clicking on the local variable and selecting change to read changes the read function of the local varia-

ble to a write function. This was then wired to the input of pData (Figure 68).

To create an alarm sound, right clicking inside the while loop. The function palette appears, graphics and sounds were selected and the beep.vi was chosen. This was also wired to the output of the pData as shown in figure 68.

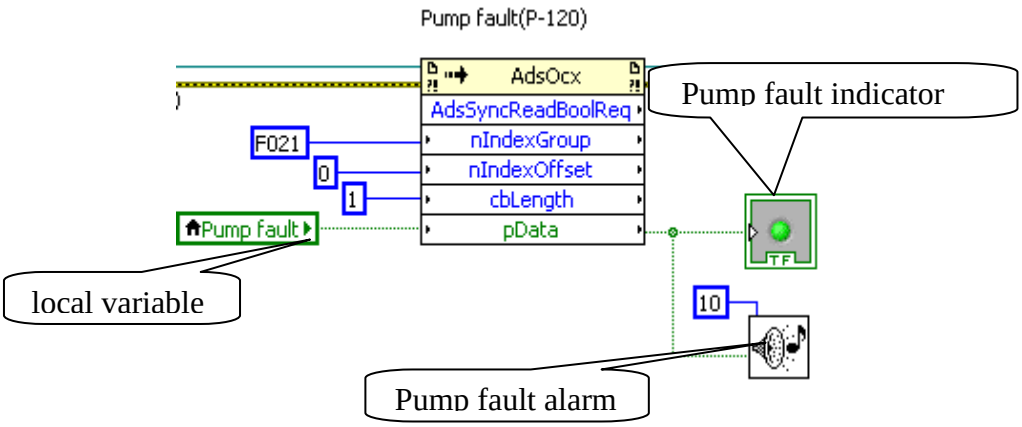


Figure 68 Creating local variable, pump fault alarm and indicators

7.2 Water Tank1(WT1) limit switch and alarm

This indicates and receives an alarm sound from the HMI when the water level is below the lower limit in Tank1 (WT1). This was done by repeating figure 68 on the limit alarm block diagram.

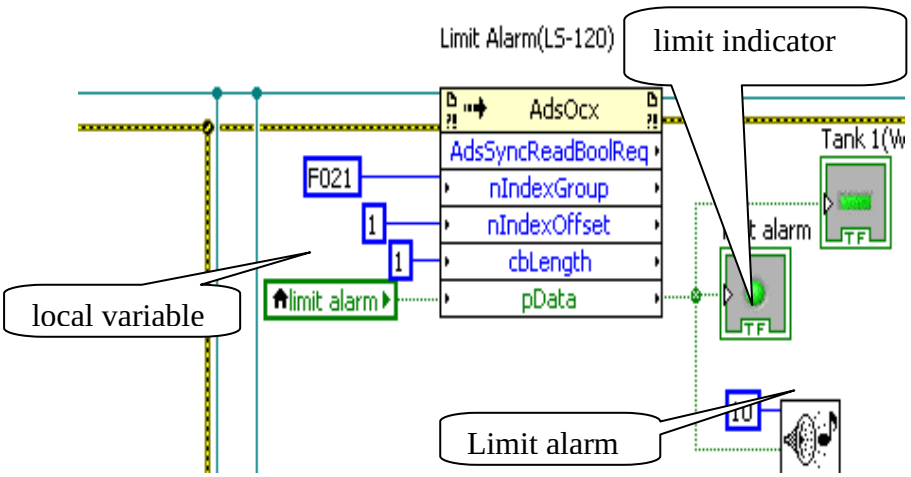


Figure 69 Creating local variable, limit alarm and limit indicator

7.3 Tank2 and Tank3 filling - draining valves switches, and pump switch

These switches are used to turn the filling valves, draining valves of Tank 2 and 3 and the pump either on or off. This is achieved by right clicking on the input of the pData of pump (ON/OFF) block diagram and selecting create and then choosing control. A switch is created which automatically get wired to the input of pData.

These steps were repeated on filling valve tank3 (MV1-140), draining valve tank3 (MV2-140), filling valve tank2 (MV1-130) and draining valve tank2 (MV2-130) block diagrams (Figure 70).

Indicators to indicate when the valves are on or off were also wired to the inputs of their pDatas.

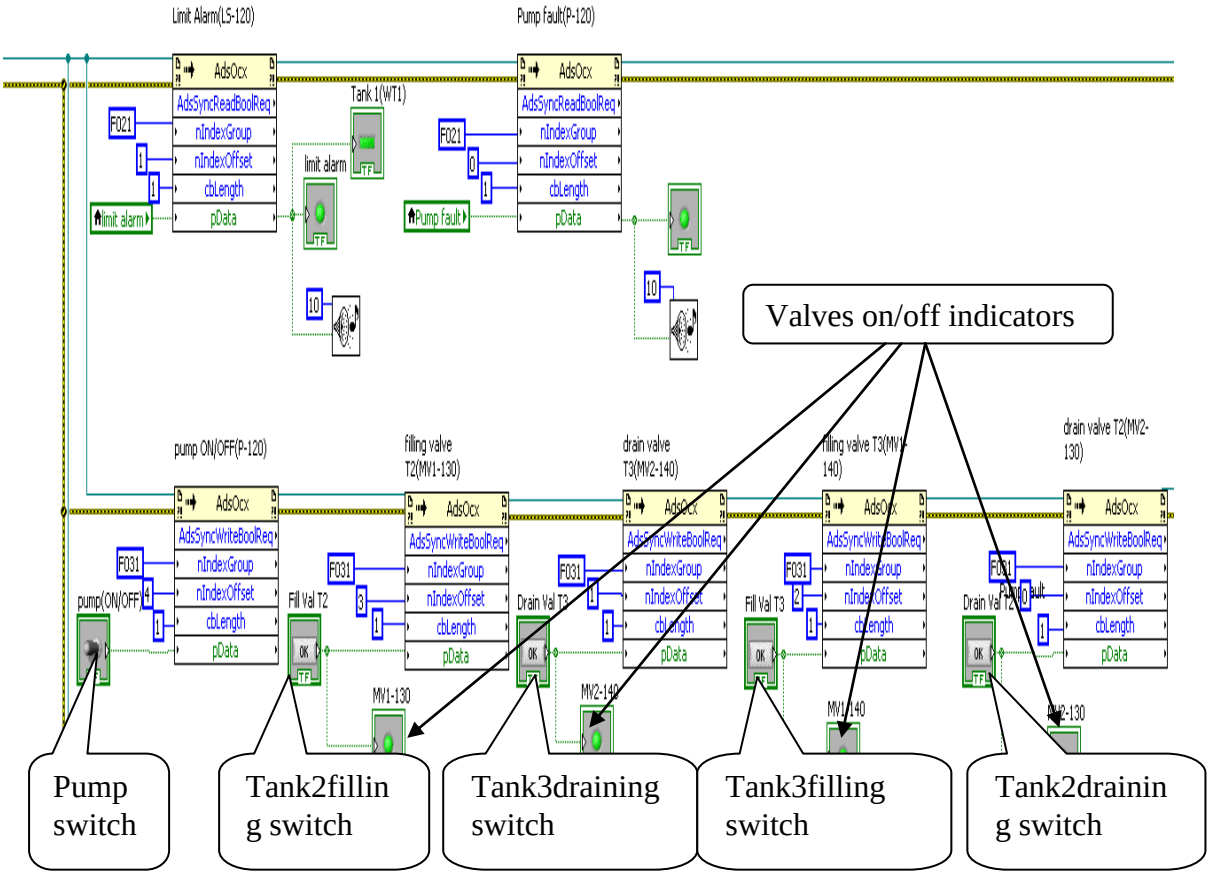


Figure 70 Creating switches and indicators for the pump and valves

7.4 Tank 2 and Tank3 level transmitters, and flow meter transmitter

There are two level transmitters each connected to Tank 2 and Tank 3 to measure the level of water inside the tanks, a flow meter measures the amount of water flowing into Tank 2 (WT2) and Tank 3 (WT3).

To indicate this information on the HMI. The output of the pData of level transmitter Tank2 block diagram was right clicked, create selected and indicator chosen. The indicator automatically gets wired to the pData output.

A local variable of the indicator was created, by right clicking on it and selecting change to write .This was then wired to the pData input.

The same steps were repeated on level transmitter Tank 3, control and flow meter block diagrams.

The data being read is an integer and must be converted to a floating point number. This is achieved by selecting numeric on the function palette, conversion and then choosing double precision float. This is wired to the output of the pData.

The output of the double precision float is wired to a sub vi. For scaling, the reason being that the -+10VDC analog bus terminals generate values based on counts. 0 count is equal to 0 VDC, 32767 counts is equal to +10VDC and -32767counts is equal to -10VDC. /5/

After scaling the values are wired to tank indicators and wave form charts to display the level of water in Tank2 and3 under level transmitter tank 2 and level transmitter tank 3 block diagram, and a numeric indicator to display the amount of water flowing into Tank2 and 3 under flow meter block diagram.

These indicators can be located inside the control palette of LabVIEW under numeric and graphs.

To indicate the set point and the levels of Tank2 and Tank3 respectively on the wave form charts during water level control inside the tanks.

Data from the set point knob and data from tank 2 level transmitter (LT-110) is wired into bundle 2 and then from bundle 2 to wave form chart 2. Similarly data from the set point knob and data from tank 3 level transmitter (LT-150) are also wired into bundle 3 and from bundle 3 into wave form chart 3.

The function of the bundles is similar to the function of a multiplexer and can be found in the function pallet under cluster and variant.

Wave form chart 2 displays the set point and the level of Tank 2 and chart3 displays the set point and the level of Tank 3.

The pressure inside Tank 2 and 3 is calculated using the formula P=ρgh inside sub vi's press. Scaling 2 and press. Scaling 3.
P=pressure in tank.
ρ= density of water (1000kg/m^3 at 4 ^{0}C).
g=acceleration due to gravity (9.81 m/s^2).
h=level of water in tank from transmitters

After which the values is wired to gauge indicators to display the values of pressures in Tank 2 and 3.

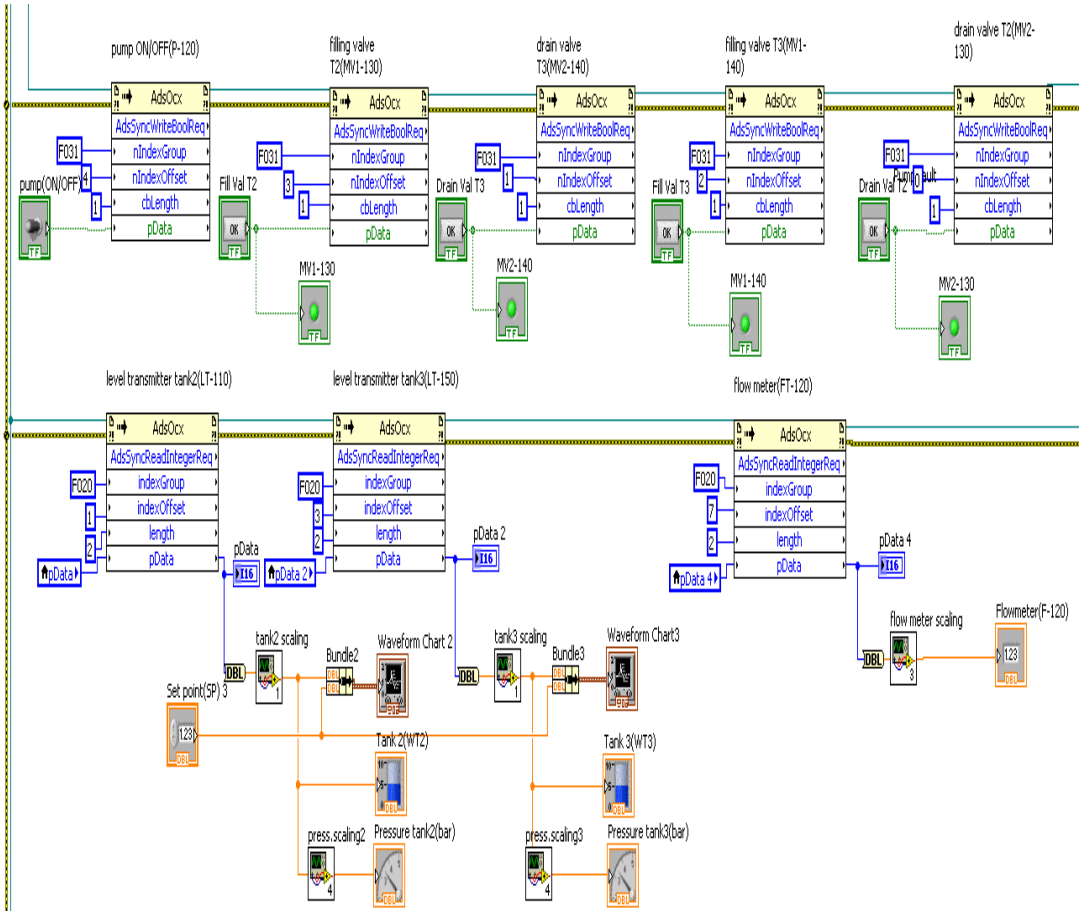


Figure 71 Creating indicators for level transmitters, pressure and flow meter

The output of the flow meter transmitter is converted from integer to double precision float or floating point number and then wired to a flow meter scaling sub vi for scaling.

The output from the flow meter scaling sub vi is then wired to a write measurement file function which generate live data on the total amount of water that has flown into Tank 2 or 3 from Tank1 into a specified folder on the master control unit on the control network.

These live data can be accessed by any workstation on the information network with either note pad, excel or spread sheet application.

7.5 Pump Speed and control valve

The pump speed can be controlled either manually or automatically by a PID controller. The pump is the final control element when the controller is in automatic state .The control valve controls the amount of water flowing back into Tank 1(WT1).

The output of the PID controller and the output from the control valve knob are converted into integers and scaled after which they are wired to the inputs of pData of pump speed and control valve block diagrams respectively.

The set point wired to bundles 2 and 3 is also wired to the set point of the PID controller (Figure 73).

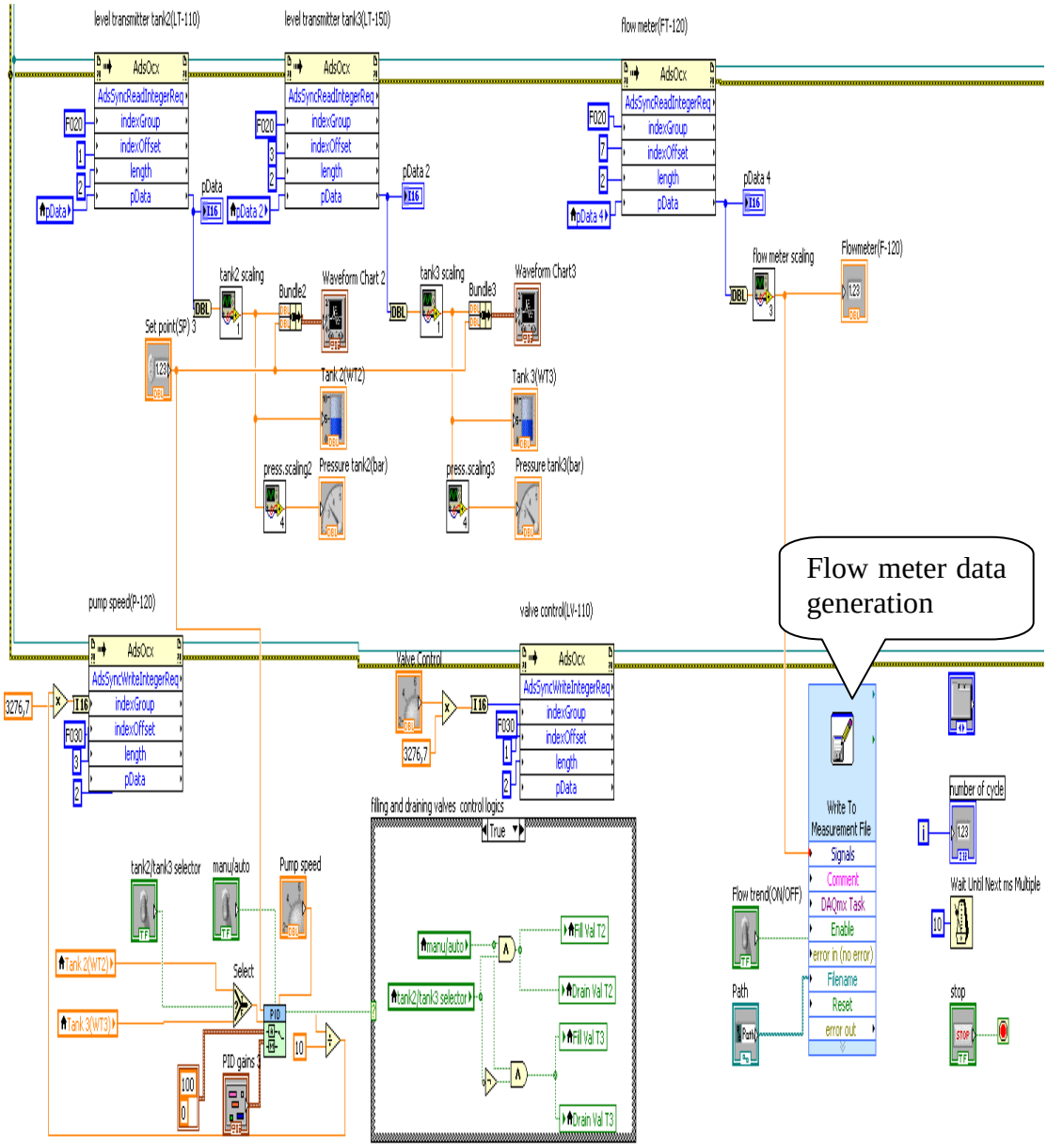

level transmitter tank2(LT-110)
level transmitter tank3(LT-150)
flow meter(FT-120)
AdsOcx
AdsSyncReadIntegerReq
indexGroup
indexOffset
length
pData
tank2 scaling
Waveform Chart 2
tank3 scaling
Waveform Chart3
flow meter scaling
Flowmeter(F-120)
Set point(SP)
Tank 2(WT2)
Tank 3(WT3)
press.scaling2
Pressure tank2(bar)
press.scaling3
Pressure tank3(bar)
Flow meter data generation
pump speed(P-120)
valve control(LV-110)
AdsSyncWriteIntegerReq
Valve Control
filling and draining valves control logics
tank2/tank3 selector
manu/auto
Pump speed
Select
PID
PID gains
Fill Val T2
Drain Val T2
Fill Val T3
Drain Val T3
Write To Measurement File
Signals
Comment
DAQmx Task
Enable
error in (no error)
Filename
Reset
error out
Flow trend(ON/OFF)
Path
number of cycle
Wait Until Next ms Multiple
stop

Figure 73

7.6 X-scale offset and multiplier

This allows the wave form charts to start with an empty data when the LabVIEW virtual instrument is started.

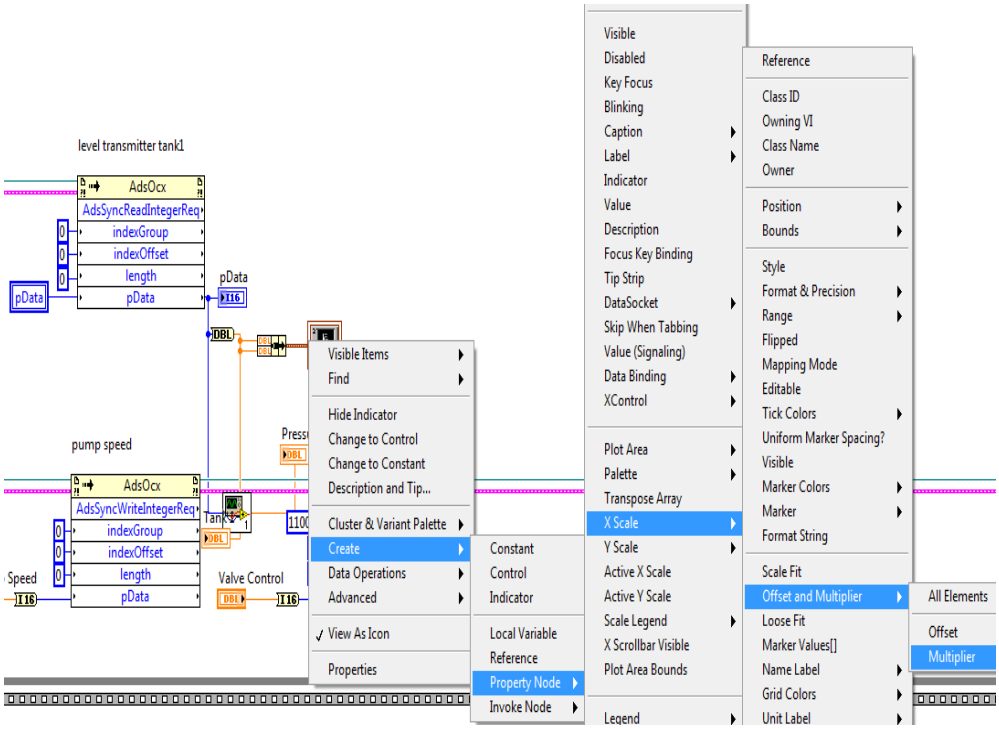


Figure 74 Creating the X scale multiplier

This was done by right clicking the input of the wave form chart2 and then selecting create, property nodes, X scale, offset and multiplier and final multiplier.

The write function of the X scale multiplier of the wave form chart is change to a read function by right clicking on the output of the X scale multiplier and selecting change to read.

Next a history data function needs to be added. This is achieved by right clicking on the X scale multiplier letter box and selecting history data (Figure 74).

An empty data was created and wired to the history data input by right clicking the input and selecting create and chosen constant.

The same procedure was repeated to create the X scale multiplier and history data for wave form chart 3.

A numeric controller was created and wired to the inputs of the X scale multipliers in order to be able to control the timing x axes of the wave form charts (Figure 75).

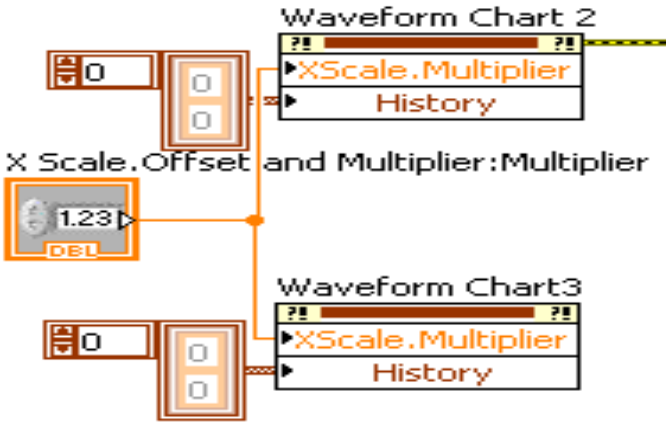


Figure 75 X scale multiplier and history data

7.7 Wait until millisecond multiplier and iteration

The function of the wait until next milliseconds in the loop is to control the while loop execution rate. The **millisecond multiple** is the input that specifies how many milliseconds lapse when the VI runs and the iteration indicates the number of cycles that has been executed (Figure 76).

Figure 76 Wait until next milliseconds and iteration

7.8 Process control principle

The control loop consists of a selector which allows either Tank 2 (WT2) or Tank 3 (WT3) to be controlled. When Tank 2 is selected the filling and draining valves of Tank 2 open and that of Tank 3 closes. The signal from Tank 2 level transmitter becomes the process variable.

Likewise when Tank 3 is selected the filling and draining valves of Tank 3 open and that of Tank 2 closes and the signal from Tank 3 level transmitter now becomes the process variable.

The control logics of the filling and draining valves are executed by the case structure only when the process is in the automatic stated.

In both cases the process variables (levels of water inside tanks) are compared to the set point. The errors generated are forward to the PID controller which performs the necessary algorithm and transmits a signal to the final control element (pump) to adjust it accordingly (Figure77).

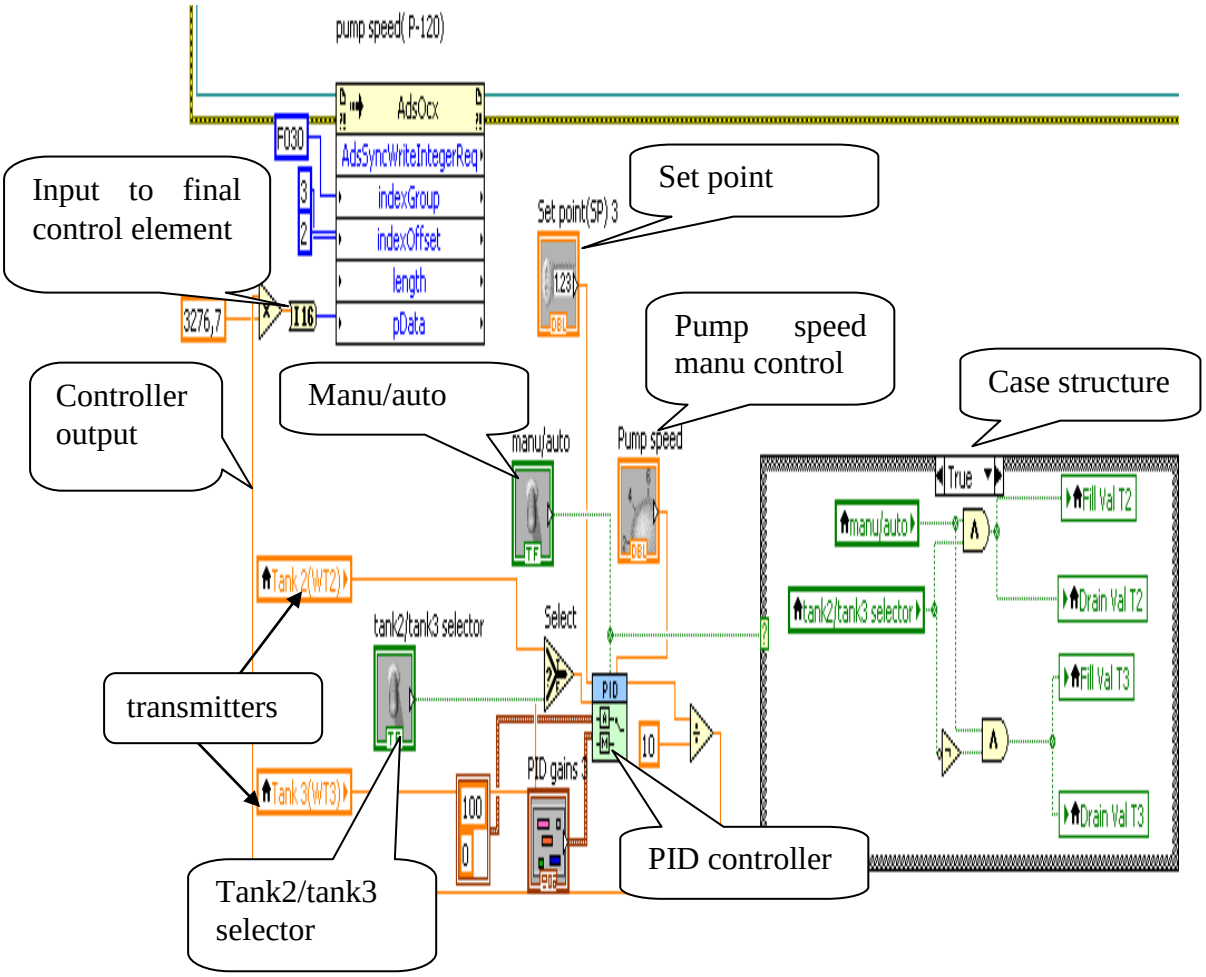


Figure 77 Control loop.

8 ENTERPRISE CONNECTIVITY

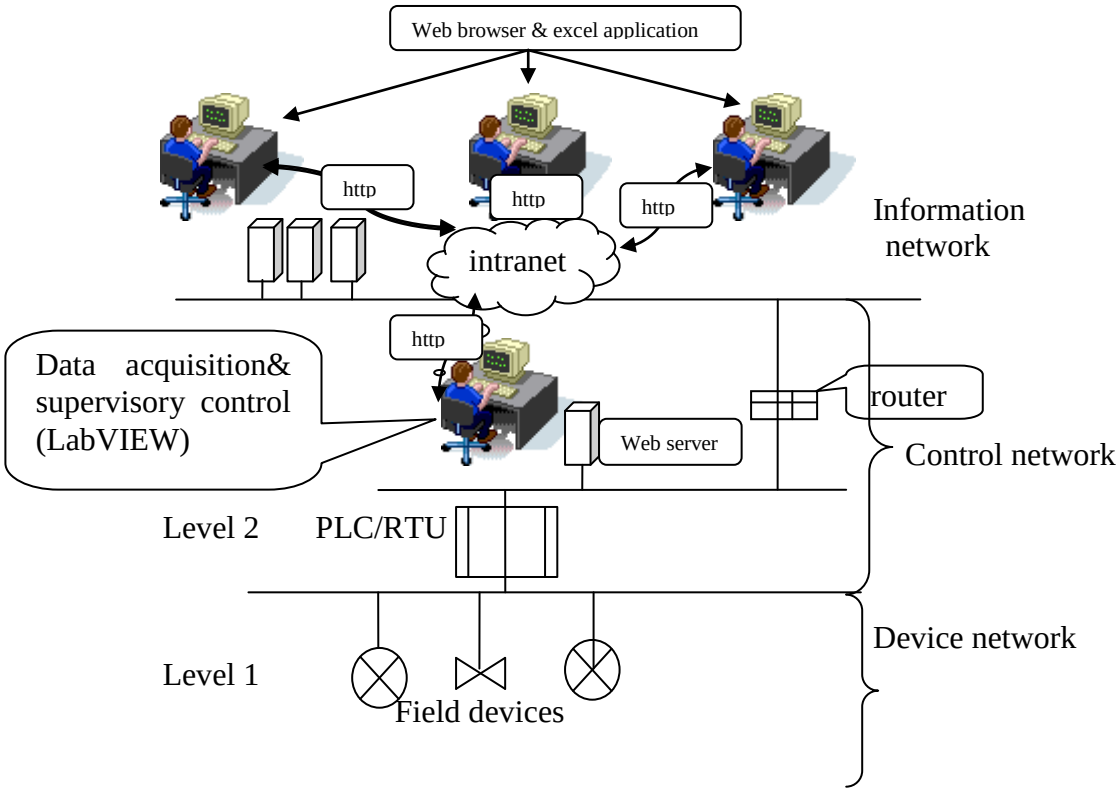


Figure 78 Water process integration in the local area network of Hamk

In most processing plants decisions are made based on up- to–date data acquired from the automation system. The process data must therefore be processed and presented in different and useful formats to all the available workstations on the network to access the information in a format they can understand and interpret.

8.1 Process data access

Data or information from the water process can be relayed to the information network in an HTML (Hyper Text Markup language) format by activating the LabVIEW web server on the workstation located on the control network (Figure78). The hyper text markup language is primarily a format in which web pages are delivered in. (Berge 2005, 24).

The hyper text transfer protocol (http) acts as a communication protocol. Microsoft internet explorer installed on the workstations located on the information network acts as the client application and the master control workstation on the control network as the host or server.

The client application submits an HTTP (http://172.24.66.18:8000/water % 20 process.html) message requesting information on the process from the server. The server which stores the contents of the process in an HTML format returns a response message to the client. The response normally contains completion status information about the request and the content requested by the client in its message body.

A request sent to monitor, control or have a snap shot of the process by the client computer can either be allowed or refused by the master control unit on which the server is located. It is advisable that the client applicant uses a Netscape browser.

Process data for analytically and business oriented issues can be acquired in an Excel format in a shared folder on the master control workstation on the control network by all workstations on the network.

9 CONCLUSIONS

The developed SCADA system proved to be very efficient and it exhibited all the functions of a typical industrial SCADA system.

The SCADA system and it enterprise connectivity were also a success .The system was able to generate and save all required or important process data and to disseminate these data into the local area network of Hamk University of Applied Sciences, Valkeakoski unit.

The process information is easily accessible by the most commonly known communication protocols and can easily be understood and interpreted also by most common software.

Interoperability was no longer a bottleneck. The system was open to accept third party system integration either through the developed software or hardware infrastructure of the design SCADA system of the process.

9.1 Recommendation

It is recommended based on the results of this project that an advanced web visualization that supports Active X to be used to access the process remotely. This allows the client computer on the network to remotely load, view and interact with the animated graphic display. Plain web visualization, however simply takes a static snap shot when it’s used to access the process and does not allow supervisory control to be executed by a remote work station on the network. Another means by which these two software can exchange data can be achieved by using OPC (Object linking and embedding for process control).

The OPC standards specify the communication of industrial process data, alarms and events, historical data and batch process data between sensors, instruments, controllers, software systems, and notification devices.

SOURCES

Berge, J. 2005.Software for automation: architecture, integration and security.Instrumentation, System and Automation Society: North Carolina.

Bailey, D & Wright, E. 2003. Practical SCADA for industry.
Newnes: London.

Beckhoff application note .Accessed 16.8.2011
ftp://ftp.beckhoffautomation.com/ApplicationNote/web/BK-AppNote-013/html/index.html.

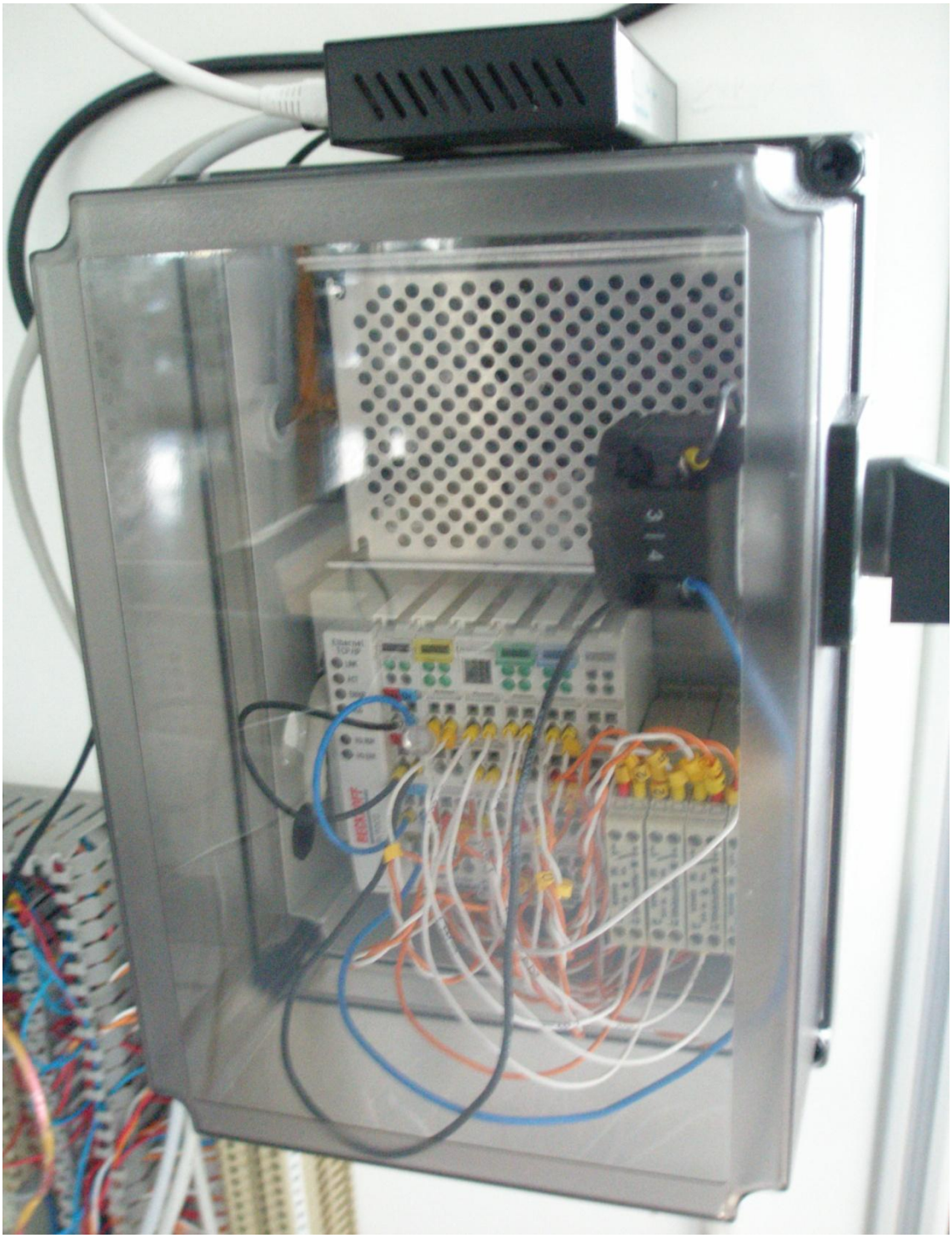
Battikha, N.E.2007.Condensed handbook of measurement and control.
Instrumentation, System and Automation Society: North Carolina.

Beckhoff Automation 2011. Accessed 16.8.2011
http://www.beckhoff.com/english.asp?bus_terminal/bk9000_bk9050.htm

Nation Communication System.2004. Supervisory control and data acquisition technical bulletin. Accessed 4.6.2011
http://www.ncs.gov/library/tech_bulletins/2004/tib_04-1.pdf.

Appendix 1

CONTROL CABINET BOX AND FIELD LAYOUT



Appendix2

CONTROLLER AND POWER SUPPLY