TelePACE Studio training

Relay Ladder Logic for SCADAPack

Presented by: Jeroen Alewijnse

Table of Contents

Detailed Table of Contents

Detailed Table of Contents

Detailed Table of Contents

Detailed Table of Contents

Table of Contents

Table of Contents

Table of Contents

Course Objectives

- Understanding of SCADAPack Hardware
- Introduction to Relay Ladder Logic
- Basic knowledge of TelePACE Studio software package
- Create, Modify and Monitor a complete TelePACE program

About Control Microsystems
(now part of Schneider Electric)

Customer Support

- A world-class group of technicians, product and customer service representatives provide the best in pre-sales and post-sales support.
- Access to pre-sales and post-sales technical support through web, phone, e-mail, fax and in person.
- Factory and on-site training. Standard and custom courses.

Customer Support

- **Technical Support:**
  Available: Monday to Friday 8:00am - 6:30pm (Eastern)
  Direct Worldwide: (613) 591-1943
  Fax: (613) 591-1022
  Toll free within North America**: 1-888-226-6876**
  Email: supportTRSS@schneider-electric.com

- **Customer Service Representatives:**
  Available: Monday to Friday: 8:30am - 5:00pm (Eastern)
  Direct Worldwide: (613) 591-1943
  Fax: (613) 591-1022
  Toll Free within North America: **1-888-267-2232**
  Email: ordersTRSS@schneider-electric.com

- **Product Training:**
  Available: Monday to Friday 8:30am - 5:00pm (Eastern)
  Direct Worldwide: (613) 591-1943
  Fax: (613) 591-1022
  Toll Free within North America: **1-888-267-2232**
  Email: scada.training@schneider-electric.com

Customer Support

Best Web Resources

http://www.schneider-electric.com/products/ww/en/6000-telemetry-remote-scada-systems/

OVERVIEW OF REQUIREMENTS

Overview of Requirements

Application Requirements

- Determines which controller and I/O modules needs to be selected
- Application requirements can be divided into:
  - Programming Requirements
  - Communication Requirements
  - I/O Requirements
  - Power Supply Requirements

Application Requirements

Programming Requirements

- A programmable controller, such as the SCADAPack, is required when:
    - The controller must perform local, or remote, process control.
    - The controller is a master station for more than one slave station.
    - Data must be saved, manipulated or logged in the controller.

Application Requirements

Communication Requirements

- Includes the number of Communication Ports that a Controller must have

Application Requirements

Communication Requirements

- The Controller may typically communicate with one or more of the following:
  - remote slave stations
  - remote master stations
  - a local HMI (Human Machine Interface)
  - local or remote programming software
  - smart measurement devices
  - other controllers at the site (radio sharing, slave devices)

Application Requirements

Communication Requirements Cont…

Controller Type	RS-232	RS-232/RS-485	Ethernet	USB	Total
SmartWIRE 5201	0	1	0	0	1
SmartWIRE 5202	1	0	0	0	1
SCADAPack 330/334	1	2	1	1	5
SCADAPack 350/357	1	2	1	2	6
SCADAPack 32	3	1	1	0	5
SCADAPack 32P	2	1	1	0	4
SCADAPack 100	1	1	0	0	2

Application Requirements

I/O Requirements

- The controller is normally required to monitor and control digital and analog inputs and outputs (I/O).
- These inputs and outputs may include transmitters indicating levels or flow, HOA switches, solenoids and VFD motor controls.
- The controller uses on board I/O, 5000 Series I/O modules, or a combination of both to monitor and control these devices. The size of the application, i.e. the number of I/O points, will determine what I/O the controller uses.

Application Requirements

I/O Requirements cont…

	Digital Inputs[1]	Digital Outputs[2]	Analog Inputs	Analog Outputs
SCADAPack 330/334	16	10	8	2
SCADAPack 350/357	8[4]	8[4]	8	2[3]
SCADAPack 32	20	15	13	2[3]
SCADAPack 32P	4	1	0	0
SCADAPack 100	6[4]	6[4]	4	0

[1] Digital inputs include the interrupt input.
[2] Digital outputs include the status output.
[3] Optional, if ordered at time of purchase.
[4] Digital I/O is both input and output

Product Overview

- Select a controller based on the application requirements
- The controller is selected using the following considerations:
  - Is a programmable or non-programmable controller required for the application?
  - What type and number of serial communication ports are required for the application?
  - How many I/O points does application require?
  - Larger programs require faster controllers to keep scan time down
- Each type of controller available is described in the following slides

For more detailed information on these controllers refer to the hardware manual for the controller.

Product Overview

SCADAPack 330/334

- 2 RS-232/485 Ports, 1 RS-232 Port
- 1 Ethernet Port, 10/100 Mb/sec.
- USB Peripheral and Host Ports
- 3 Pulse Counter Inputs
  - 1, 0-10Hz or 0-5kHz (dry contact)
  - 2, 0-10kHz (turbine or dry contact

SCADAPack 334 adds:

- ***8 Analog Inputs***
- ***16 Digital Inputs***
- ***10 Relay Outputs***
- ***2 Analog Outputs***

Product Overview

SCADAPack 350/357

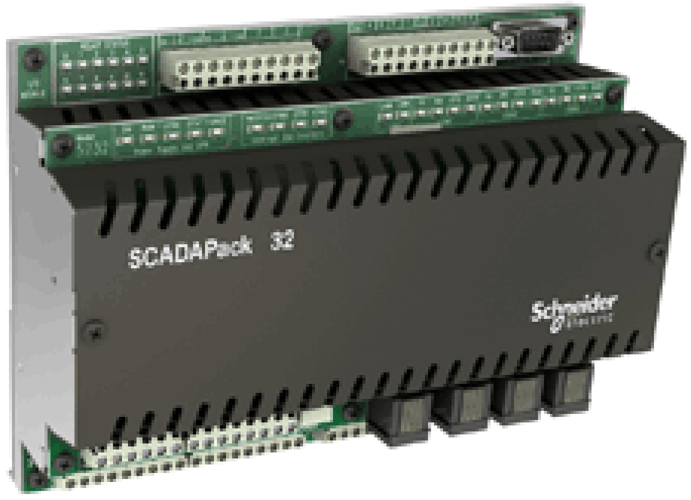
- 3 Serial Ports
- 1 RS-232, 1 RS-485,1 RS-232/RS-485
- 1 10/100MB Ethernet
- USB2.0 Host & Programming ports
- 6 Analog Inputs
- 8 Discrete I/O
- 2 Turbine Meter Inputs
- 1 Counter

SCADAPack 357 adds: 32 DI / 16DO / 8AI

Product Overview

SCADAPack 32

- PII Class 32bit 120MHz
- 8MB RAM
- 2MB SRAM
- 2MB Flash
- 8 Analog In
- 20 Digital In
- 12 Digital Out
- 4 RS-232 or 3 RS-232 and 1 RS-485
- Ethernet Port



Product Overview

SCADAPack E Series - DNP3 Centric Controllers

SCADAPack 535E

- AMD 100MHz Processor
- Five Serial Ports
- Two 100MB Ethernet Ports
- Compact Flash Card Support
- 32 DIs
- 16 DPDT 2amp Relay Outputs
- 12 Single ended 12 bit AIs
- 4 Unipolar 12 bit AOs

Product Overview

SCADAPack Transmitter

- The SCADAPack Transmitter product line includes Gas Flow Computers with highly integrated PLCs, Modbus multivariable transmitters with integrated sensor technology, Modbus pressure transmitters, and products specifically designed for solar-powered installations, all based on rugged, proven platforms

- 0.05% Accuracy
- LCD Display option
- 3-year warranty on parts and labor

Product Overview

SCADAPack Transmitter 4102

- Modbus Multivariable Transmitter
  - Static Pressure
  - Differential Pressure
  - Process Temperature

- Three Output Variations
  - Modbus RS232/RS485
  - Modbus RS232/RS485 w/ PID Control & AO
  - Modbus RS232/RS485 w/ PID Control & AO & 10baseT Ethernet

Product Overview

SCADAPack Transmitter 4012 / 4032

- 4012 Static Pressure Transmitter
- 4032 Differential Pressure Transmitter
- Three Output Variations
  - Modbus RS232/RS485
  - Modbus RS232/RS485 w/ PID Control & AO
  - Modbus RS232/RS485 w/ PID Control, AO & 10baseT Ethernet

Product Overview

SCADAPack Transmitter 4203 GFC

- High-performance 32-bit Arm7-based processor
- Support for up to 32 independent C++ applications
- Orifice, V-Cone and Linear Type Meters
- API 21.1 Compliant Audit Trail
- Integrated DP/P/T Measurements
- Configurable Display
- Runs TelePACE or ISaGRAF in addition to RealFLO

SCADAPack 350 inside

Product Overview

Trio Radios

Professional Fixed Data System Products

- SCADA and Telemetry
  - Point-to-Point
  - Point-to-Multi Point (MARS)

- **Ultra-Series**
  - License-Free Spread Spectrum Data Radios Industry leading performance with unique practical features

- **E-Series**
  - Licensed Data Radios the most advanced digital data radio

- **M-Series**
  - Licensed Low Cost Data Radios the new benchmark for low cost data radio

Product Overview

Accutech Wireless Sensors

- The **Accutech** series is comprised of self-contained, self-powered Field Units providing process data to a centralized Base Radio through a 900MHz or 2.4GHz spread-spectrum, frequency hopping wireless connection.
- Networks of up to 50 field units can be created and polled by a single base radio, with a typical range between field unit and base radio of 500 to 1000ft (152 to 305m).

Product Overview

Accutech Series cont…

BR10 Base Radio

- Supports up to 50 field units
- Multiple base radio support
- Serial Modbus RTU via RS-485
- Explosion-proof housing
- Integrated or remote antenna options
- 24VDC input power
- Stand alone server using SCADARange Manager
- Temperature Range:
  -40 to +85°C (-40 to +185°F)

AI10 (Current)
& AV10 (Voltage) Multi-Input

- Dual current (4-20mA) or voltage (0-10V) based analog inputs
- Dual discrete contact-closure inputs
- Explosion-proof junction box option
- High-gain antenna option
- Temperature Range:
  -40 to +85°C (-40 to +185°F) electronics

GL10 Gauge Level

- Extended gauge level sensor
- 30PSIG (2.068 BAR)
- Supports specific-gravity correction and multiple units of level measurement
- Temperature Range:
  -40 to +85°C (-40 to +185°F) electronics
  -40 to +121°C (-40 to +250°F) process

Product Overview

Accutech Series cont…

GP10 Gauge Pressure

- Highly accurate gauge pressure sensor: Accuracy: ± 0.1 % of sensor URL
- 250, 1000, 2500 and 5000PSIG (17, 35, 70, 350 BAR)
- Temperature Range:
  -40 to +85°C (-40 to +185°F) electronics
  -40 to +121°C (-40 to +250°F) process
- Remote pressure sensor option
- High-gain antenna option

RT10 RTD Temperature Sensor

- RTD Input :
- 100 Ohm DIN, US Curves
- 1000 Ohm DIN
- Basic accuracy ± 0.1% of reading
- Temperature Range:
  -40 to +85°C (-40 to +185°F)
- Probes available with either spring-loaded or direct-insertion fitting with probe lengths of 2.5", 4.5"or 6"
- Remote pressure sensor option
- High-gain antenna option

TC10 Thermocouple Temperature Sensor

- Thermocouple Input type: K
- Basic accuracy ± 0.1% of reading
- Temperature Range
  -40 to +85°C (-40 to +185°F)
- Probes are available with either spring-loaded or direct-insertion fitting with probe lengths of 4.5", 6" or 9"
- Remote pressure sensor option
- High-gain antenna option

Product Overview

Accutech Series cont…

SL10 Submersible Level

- Sensor type – FM Approved KPSI
- Pressure Ranges
  - 5 PSI, 11.55ft. Cable 15ft
  - 10 PSI, 23.11ft. Cable 30ft
  - 15 PSI, 46.21ft. Cable 40ft
  - 30 PSI, 69.32ft. Cable 50ft
- Temperature Range
  -20°C to 60°C (-4°F to 140°F)
- Vent options
  - Desiccant
  - Hydrophobic
  - Bellows

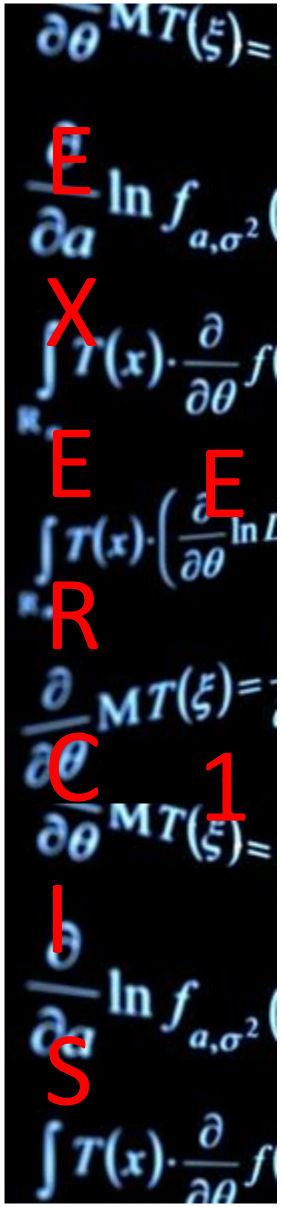
SI10 Switch Input

- Dual Contact -Closure Switch-Input
- Contacts sampled 11 times per second
- Temperature Range
  -40 to +85°C (-40 to +185°F)
- Flying leads option

Exercise One

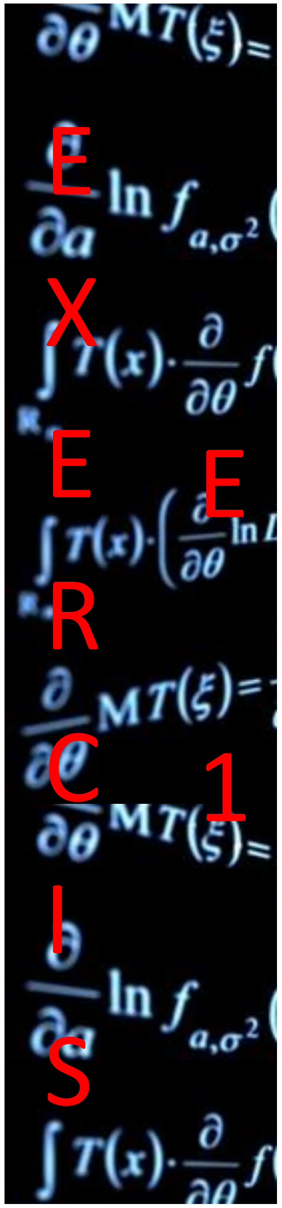
Exercise 1

Install TelePACE Studio Software and SCADAPack Hardware Manual



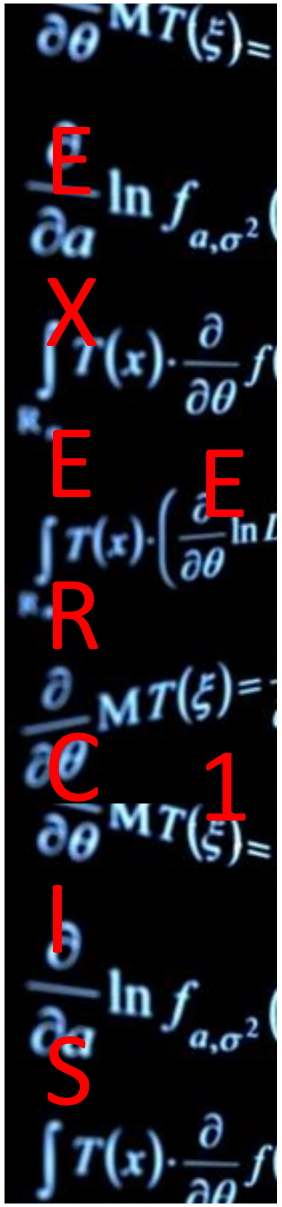
Exercise Objectives

- Obtain a copy of the TelePACE Studio software
- Learn proper steps to install software
- Obtain a copy of the SCADAPack Hardware Manual
- Install Manual on your PC
- Review of key points and navigation in manual



Materials Needed

- PC or laptop with Microsoft Windows XP or higher
- Soft copy of Install files



Procedure

To install TelePACE Studio:

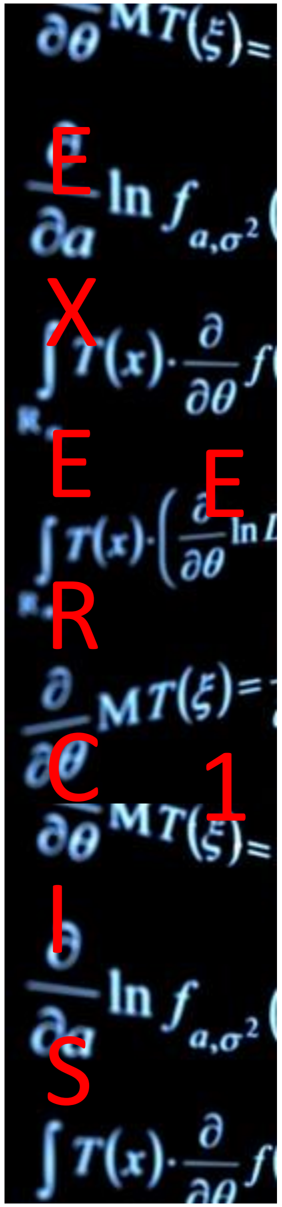
1. OpenTelePACE Studio folder.
2. Find the install wizard.
3. Follow the setup directions on the screen.

To install the Complete Hardware Documentation:

1. Locate the Complete Hardware Documentation folder

- May require unzipping of archived file

2. Run the ‘setup.exe’ file
3. Follow the setup directions on the screen
4. Instructor to review documentation



Overview of Requirements: Review Questions

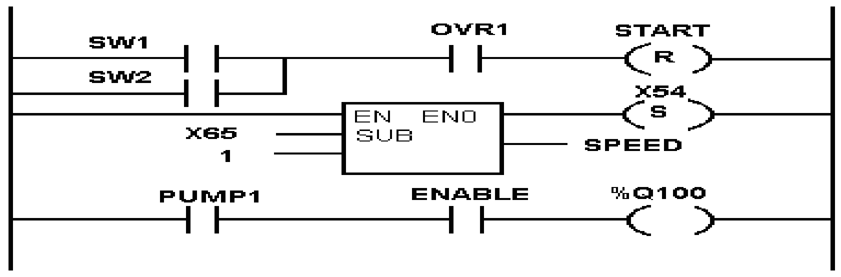
1. How many communication ports are available on the SCADAPack 350 (includes all RS-232, RS-485, Ethernet and USB ports) ?

LADDER LOGIC

Ladder Logic

Relay Ladder Logic

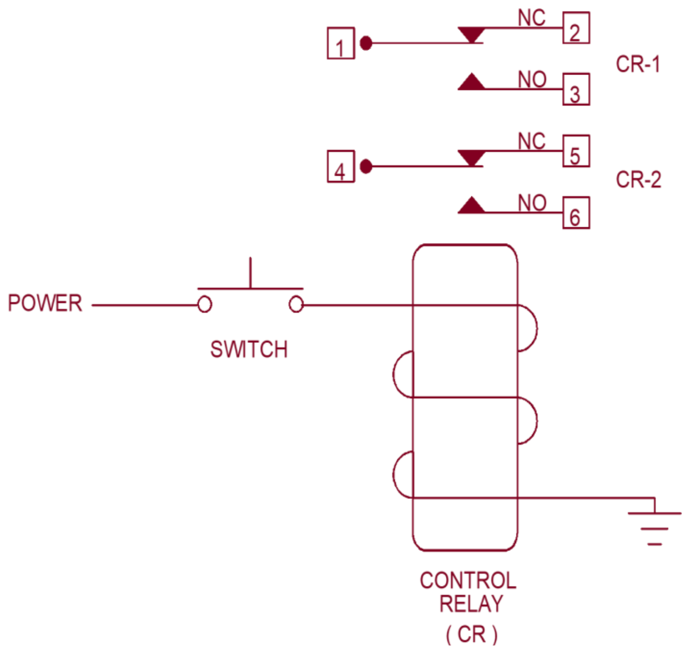
- Relay Ladder Logic has been used in the control of industrial processes for many years.
- Early control circuits were created by wiring input devices such as pushbuttons and limit switches to output devices such as motors.
- As the processes became increasingly complicated mechanical relays were added to the control circuits.



Ladder Logic

Control Relays

- A relay is a coil of wire wrapped around a core
- When the switch is closed current flows through the coil and an electromagnet is created

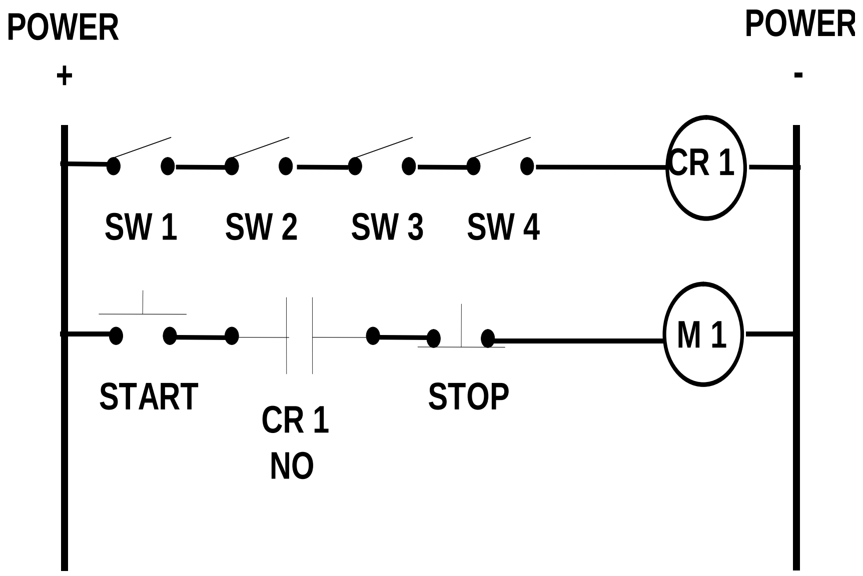


- When power is applied to the coil, the connection between contacts 1 and 2 of CR-1 is broken and the connection between contacts 1 and 3 is made
- The status of the contacts is directly controlled by the current flow through the coil

Ladder Logic

Simplified Relay Ladder Logic

- Early Relay Ladder Logic systems had some important limitations:
  - The physical size of the control and wiring panels became larger and larger as the complexity of the control systems increased.
  - Changes to the control system would involve large scale rewiring of the control and wiring panels.

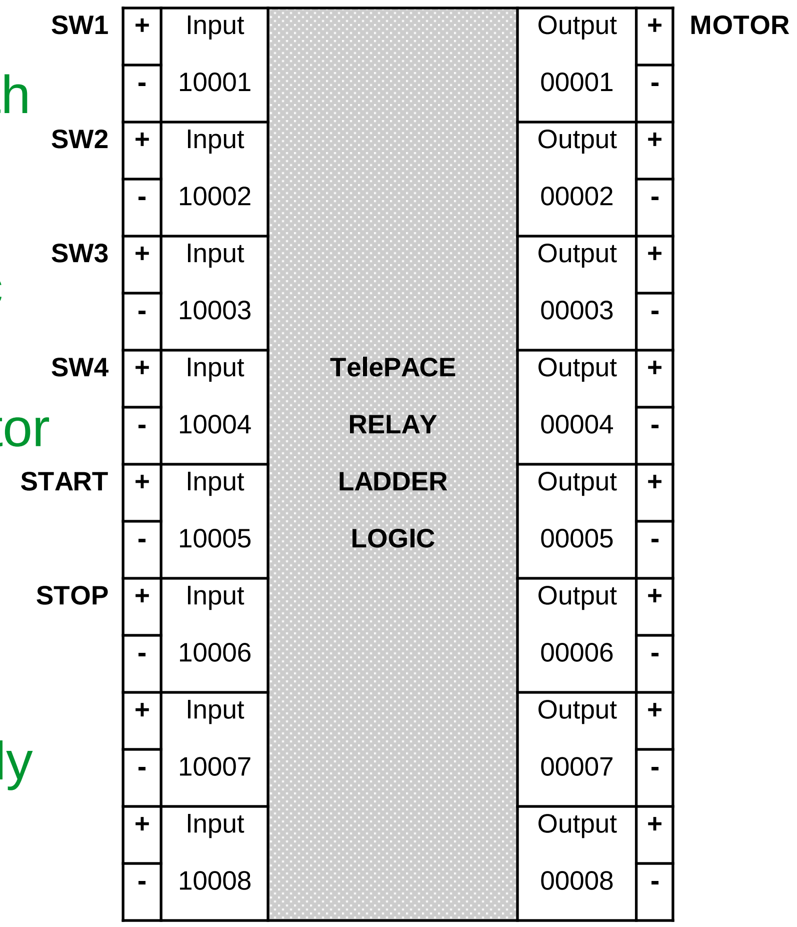


- SW 1 - SW 4 are switches.
- CR 1 is a control relay with a NO contact.
- START is a normally open pushbutton switch.
- STOP is a normally closed pushbutton switch.
- M1 is a motor with NO and NC contacts

Ladder Logic

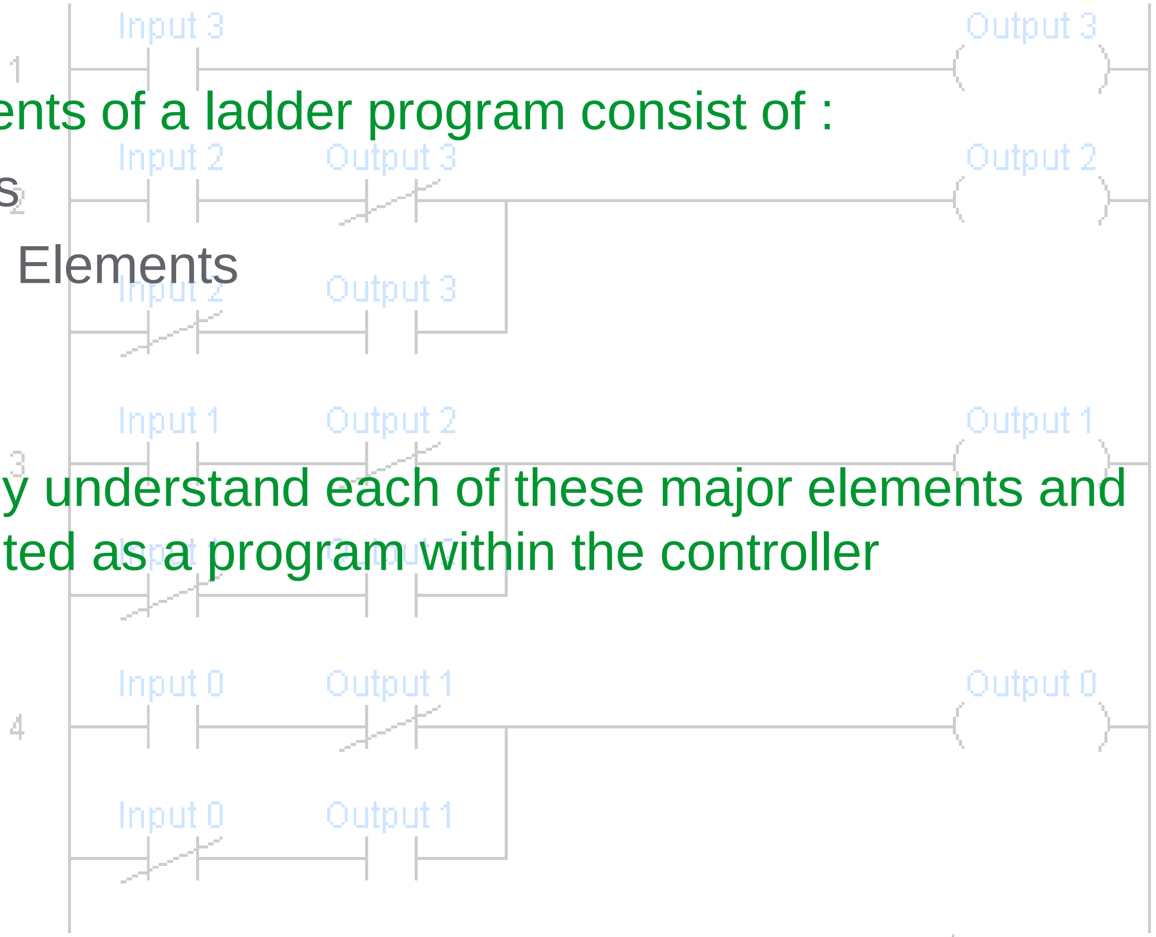
Programmable Relay Ladder Logic

- TelePACE Relay Ladder Logic, used with Control Microsystems controllers, overcomes the limitations of early relay ladder logic through programmable logic functions
- Logic functions such as Delay Start, Motor Runtime, Number of Starts, etc. can all be easily added to the control system, without additional wiring.
- SCADAPack 300 Series have universal I/O – Digital I/O points are simultaneously input and output.



TelePACE Studio Ladder Logic Program

TelePACE Studio Ladder Logic Program



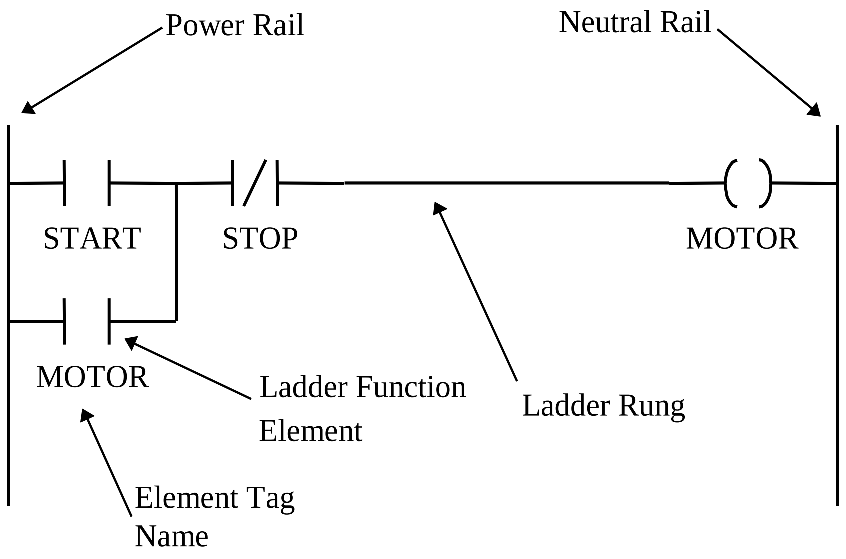
- The major components of a ladder program consist of :
  - Ladder Networks
  - Ladder Function Elements
  - Comments

- It is important to fully understand each of these major elements and how they are executed as a program within the controller

TelePACE Studio Ladder Logic Program

Networks

- A network is a diagrammatic representation of control logic similar to a wiring schematic, showing the interconnection of relays, timers, contacts and other control elements.
- The number of networks in a program is only limited by the memory available.



TelePACE Studio Ladder Logic Program

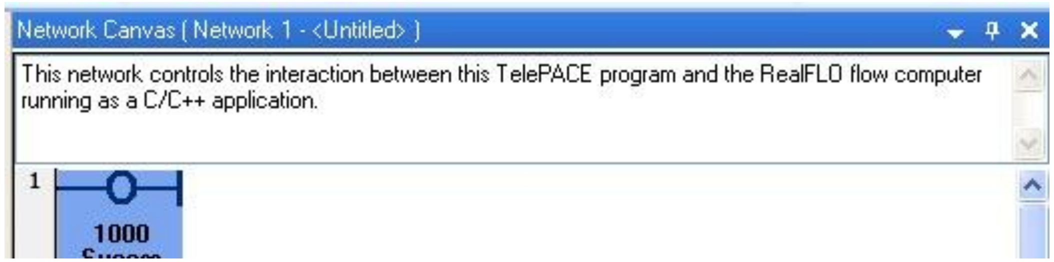
Network Elements

- *Network elements* are contacts, coils, and function blocks.
- Shunts are used to interconnect elements.
- Coils are always found connected to the neutral rail and represent either physical outputs in the controller or internal (memory only) outputs in the I/O database.
- Contacts represent either physical status inputs from the controller or internal (memory only) status inputs in the I/O database.
- Function blocks are used to perform specific functions, such as moving data, manipulating data or communicating data.

TelePACE Studio Ladder Logic Program

Network Comments

- The comment editor enables the programmer to fully document an application.
- Each network may have up to three pages of text documentation.
- The network title allows for quick access to networks when editing or monitoring programs.



This network controls the interaction between this TelePACE program and the RealFLO flow computer running as a C/C++ application.

TelePACE Studio Environment

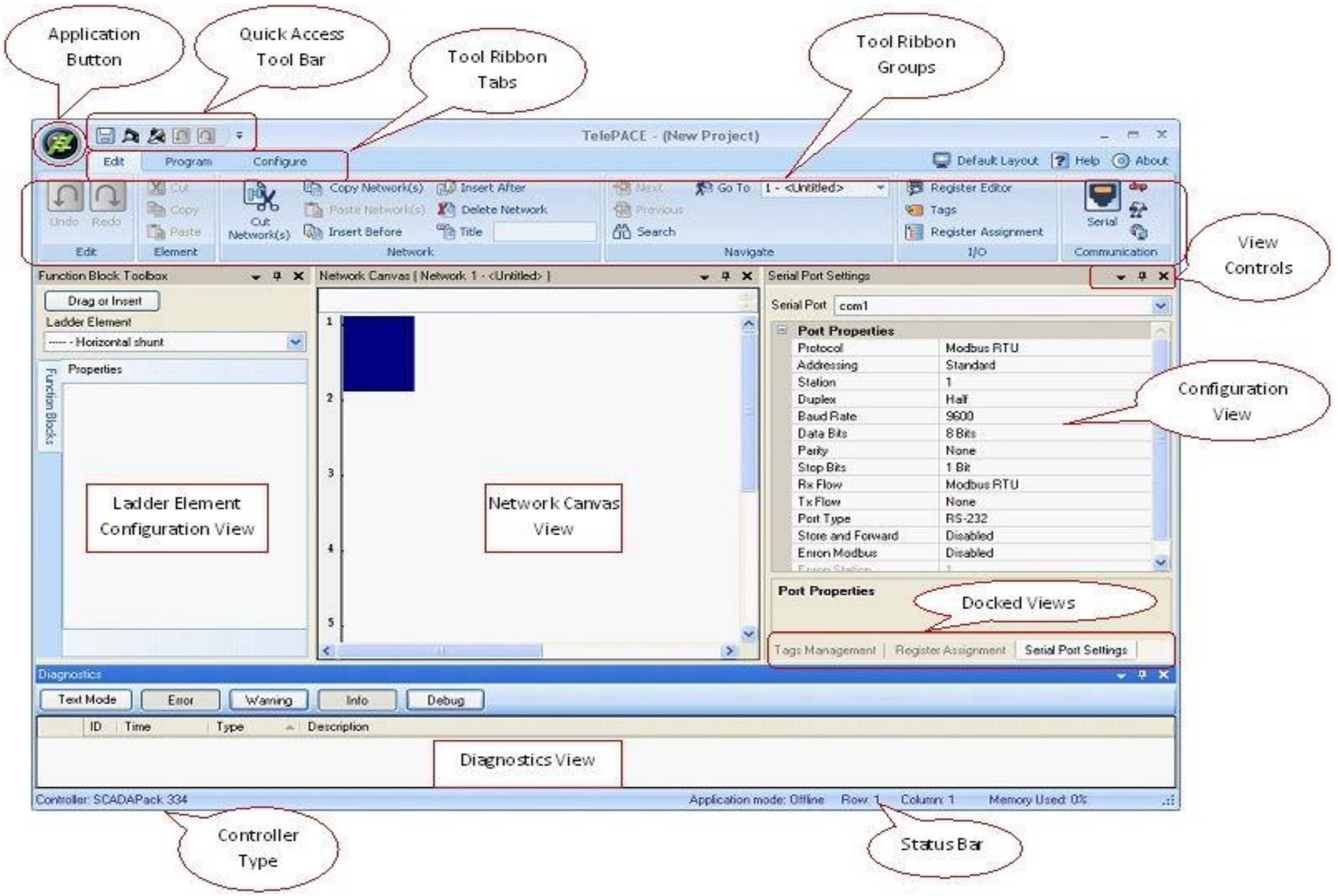
TelePACE Studio Layout [page 13]

- TelePACE Studio is a powerful programming and monitoring environment, which enables the user to:
  - Create ladder logic programs
  - Edit ladder logic programs On or Off line with a Controller
  - Read and write programs to a Controller
  - Configure the Controller serial communication ports

TelePACE Studio Layout

- If you have used earlier versions of TelePACE, you will notice that the TelePACE User Interface has changed.
- TelePACE Studio user interface is based on the Microsoft Fluent Interface. “Browse, pick, and click”.
- This type of interface allows users to focus on the creating, editing and monitoring of TelePACE projects rather than trying to remember where the menu commands are located in the menu bar.

TelePACE Studio Layout



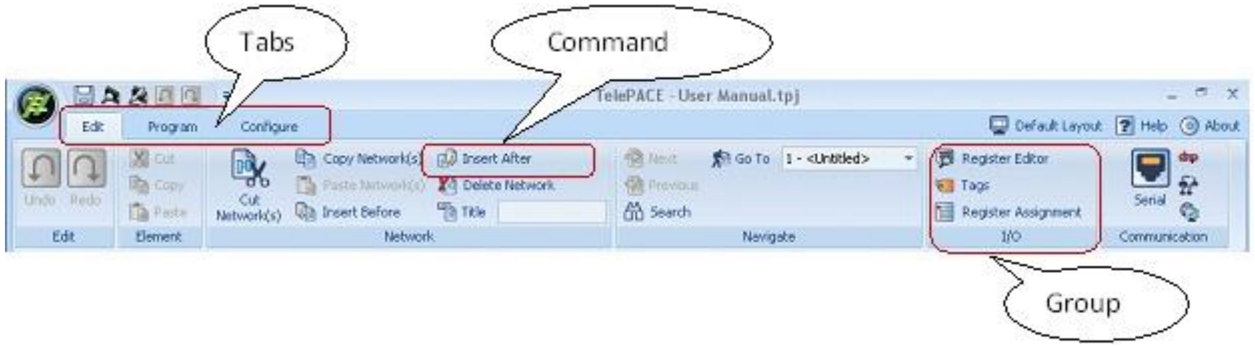
Serial Port: com1

Port Properties	
Protocol	Modbus RTU
Addressing	Standard
Station	1
Duplex	Half
Baud Rate	9600
Data Bits	8 Bits
Parity	None
Stop Bits	1 Bit
Rx Flow	Modbus RTU
Tx Flow	None
Port Type	RS-232
Store and Forward	Disabled
Enron Modbus	Disabled
Enron Station	1

Controller: SCADAPack 334 | Application mode: Offline | Row: 1 | Column: 1 | Memory Used: 0%

TelePACE Studio Layout

- The Menu and Tool Bars in previous versions of TelePACE have been replaced with **Tool Ribbons** that combine commonly used commands and tasks into meaningful groups
- There are three Tool Ribbons: Edit Ribbon, Program Ribbon and Configure Ribbon, which are selected using the tool Ribbon tabs at the top of each Tool Ribbon.



TelePACE Studio Layout

- The menu commands in previous versions of TelePACE would typically open a dialog to configure the parameters for the command
  - For example the Serial Port Settings command would open the Serial Port Settings dialog where port parameters would be set and then the dialog would be closed
- TelePACE Studio replaces most of these dialogs with **Views**
- Views may be left open in the TelePACE Studio Workspace to allow quick reference while programming or debugging projects
- Views can be moved, resized, hidden and pinned to any location in the Workspace

TelePACE Studio Layout

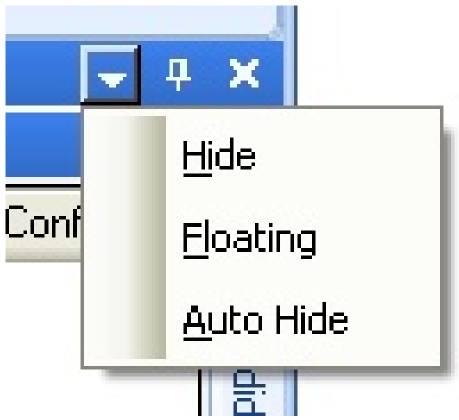
Managing Views

- The Default Layout is opened with six default views in the workspace, Function Block Toolbox, Network Canvas, Serial Port Settings, Register Assignment, Tag Management and Diagnostics
- TelePACE Studio is designed to be a flexible work environment and users are encouraged to modify the workspace to meet their needs
- When TelePACE is closed the workspace layout is saved and reused each time TelePACE is opened

TelePACE Studio Layout

Managing Views

- There are three view controls at the top of each view pane.
- These controls are used to manage the position of the view or to hide the view when it is not being used



TelePACE Studio Layout

Managing Views

Hide

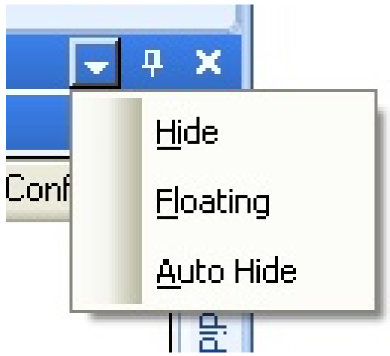
- Selecting the **Hide** control closes the current view. Re-select the command to re-open the view.

Floating

- Selecting the **Floating** control undocks the view. When the view is undocked it can be moved to any position in your workspace.

Auto Hide

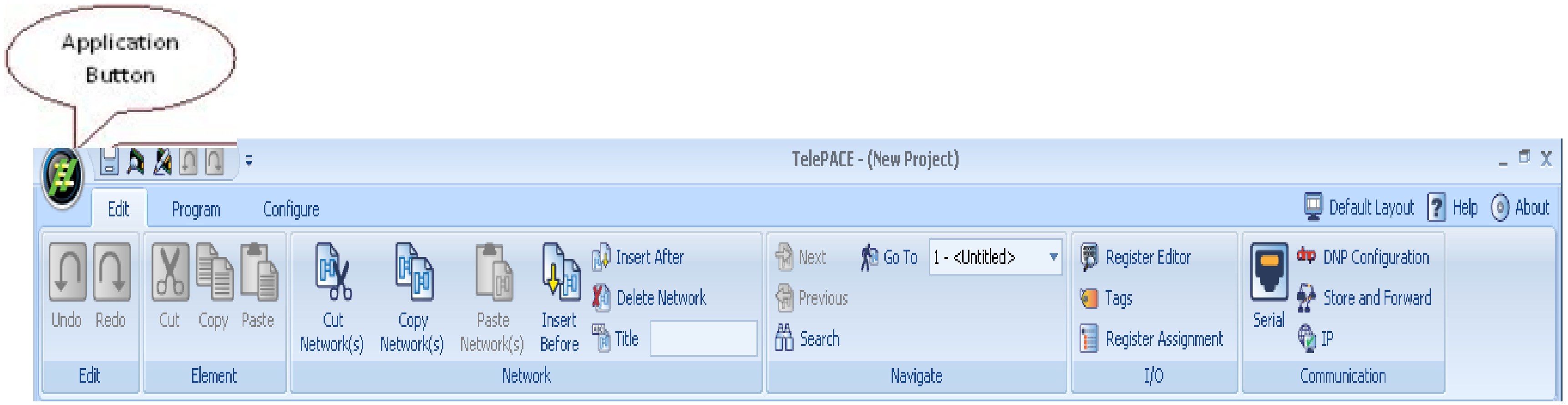
- The **Auto Hide** control sets the tab selection to the right hand side of the views. Moving the cursor over the tabs at the right hand side activates the selected view. The auto hide function is available using the **Auto Hide** control. This is the pin shaped icon to the right of the menu control



TelePACE Studio Layout

TelePACE Studio Application Button

- The place to start a new TelePACE project is the TelePACE Application button located in the upper left of the TelePACE user interface
- The **file** and **print** management functions are now accessible using the TelePACE Application Button

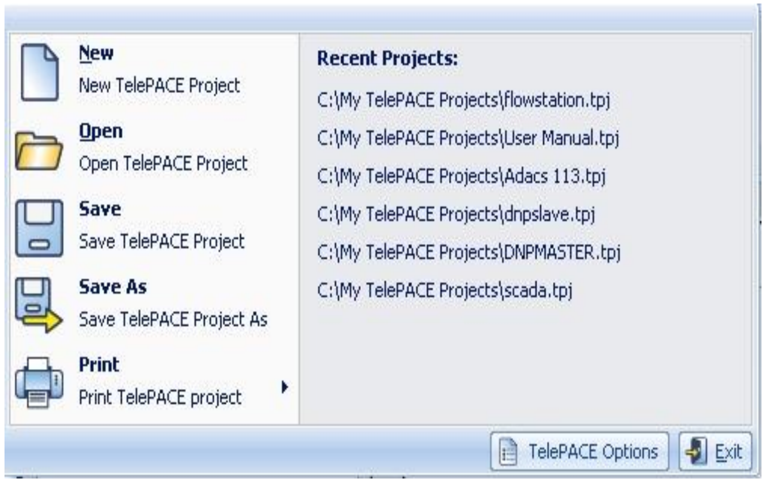


TelePACE Studio Layout

TelePACE Studio Application Button

- Once you click on the TelePACE button, the following menu appears:

File New: start a new TelePACE project file.
File Open: open an existing TelePACE project file.
File Save: save a TelePACE project.
File Save As: save a TelePACE project with a different name or location.
Print: set the print format and print
Recent Projects List: displays the last eight recently used TelePACE projects.
TelePACE Options: opens the TelePACE Options dialog to set the floating point format and Delete Network confirmation options.
Exit: closes the TelePACE application

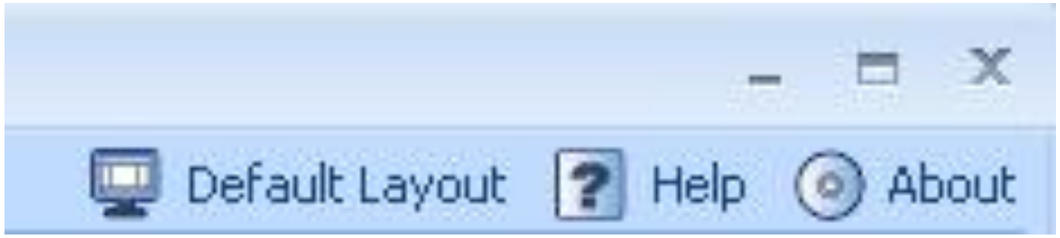

New
New TelePACE Project
Open
Open TelePACE Project
Save
Save TelePACE Project
Save As
Save TelePACE Project As
Print
Print TelePACE project
Recent Projects:
C:\My TelePACE Projects\flowstation.tpj
C:\My TelePACE Projects\User Manual.tpj
C:\My TelePACE Projects\Adacs 113.tpj
C:\My TelePACE Projects\dnpslave.tpj
C:\My TelePACE Projects\DNPMASTER.tpj
C:\My TelePACE Projects\scada.tpj
TelePACE Options
Exit

TelePACE Studio Layout

Ribbon Buttons

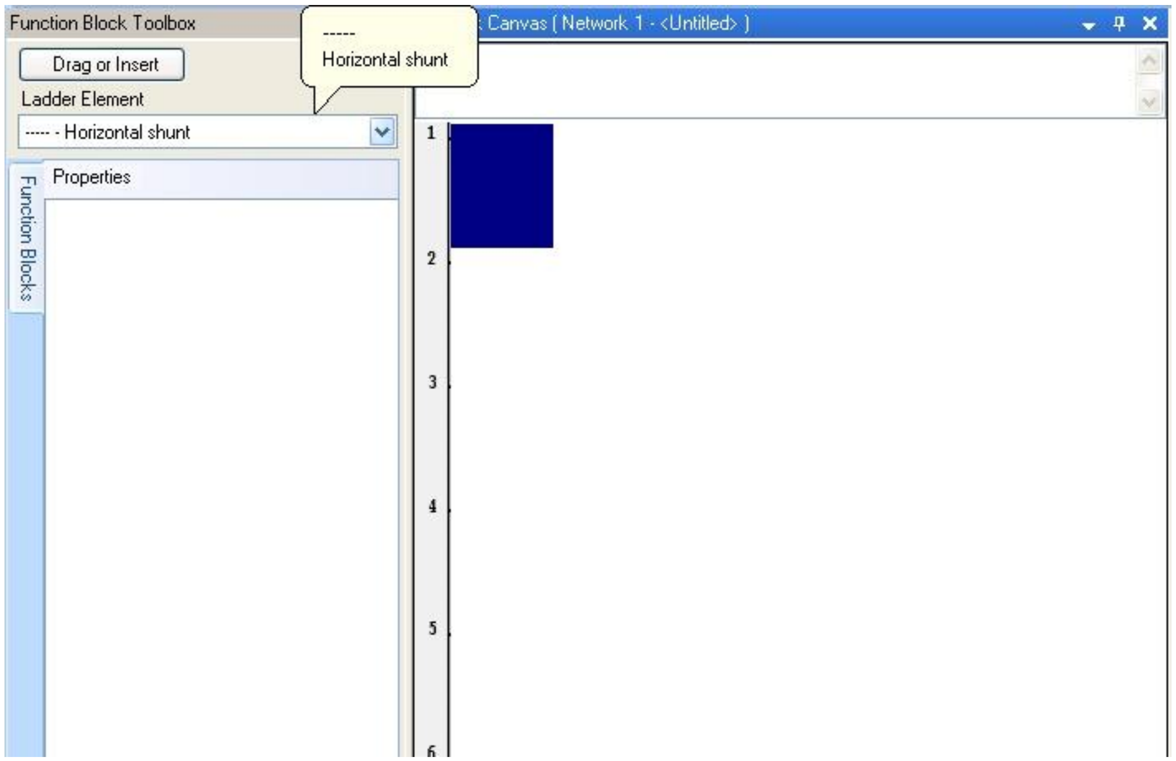
- The Ribbon Buttons are located at the top right corner of the TelePACE workspace

- There are three Ribbon buttons available:
  - The **Default Layout** button returns the TelePACE workspace to the default layout
  - The **Help** button opens the TelePACE help file
  - The **About** button opens the About TelePACE dialog



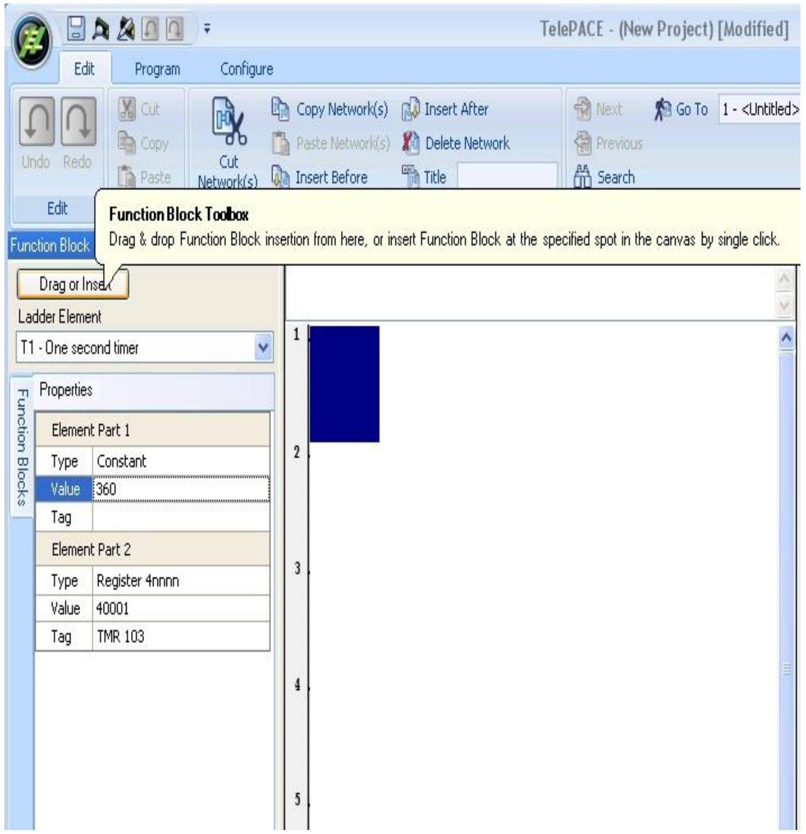
TelePACE Studio Layout

- The main views used to create and edit ladder logic in a TelePACE project are the Function Block Toolbox and the Network Canvas



TelePACE Studio Layout

- The Function Block Toolbox view contains an Edit Control button, a Ladder Element selection control and a Properties grid
  - The **Edit Control** button is initially labeled Drag or Insert and will change to End Edit Mode when a Ladder Element has been placed in the Network Canvas.
  - The **Ladder Element** selection control is a drop down menu that contains a complete selection of available Ladder Logic Elements.
  - The **Properties** grid contains the necessary configuration parameters for any ladder elements selected in the Ladder Element drop down menu.


TelePACE - (New Project) [Modified]

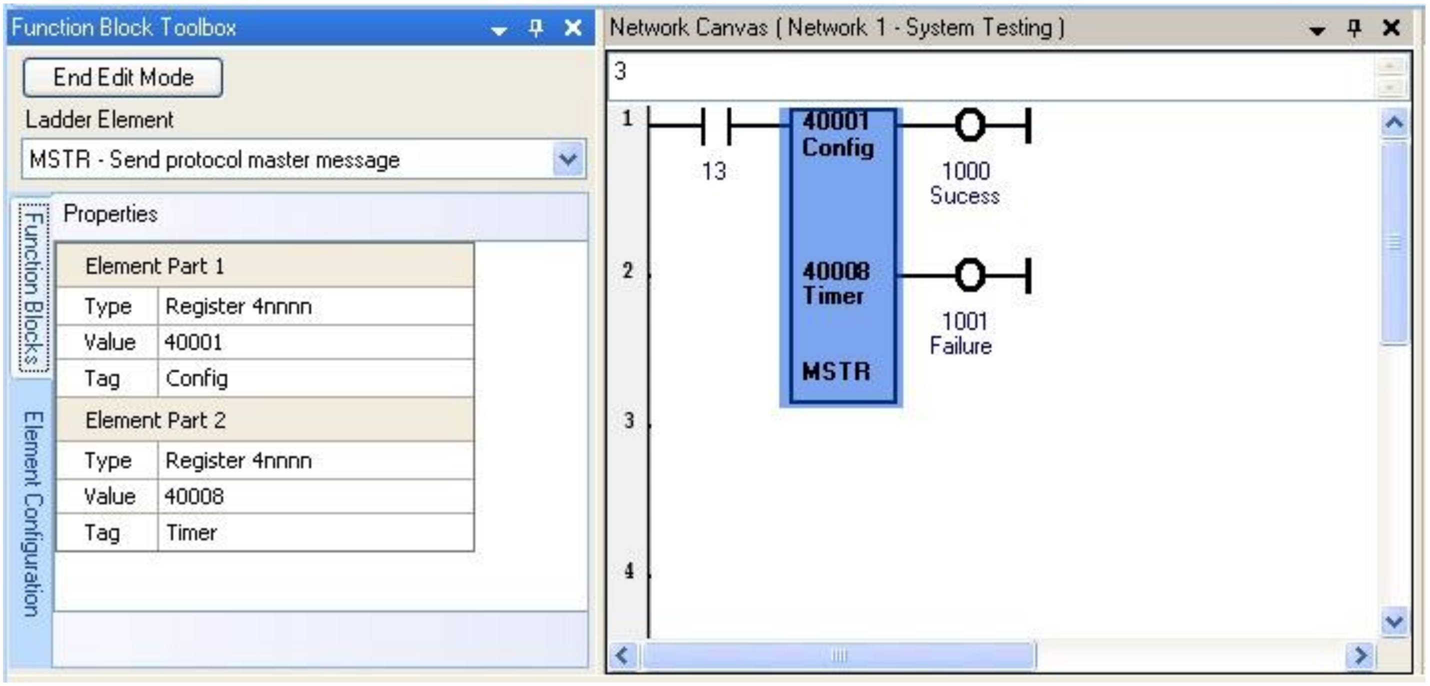
Function Block Toolbox
Drag & drop Function Block insertion from here, or insert Function Block at the specified spot in the canvas by single click.

Ladder Element: T1 - One second timer

Properties	
Element Part 1	
Type	Constant
Value	360
Tag	
Element Part 2	
Type	Register 4nnnn
Value	40001
Tag	TMR 103

TelePACE Studio Layout

- The Network Canvas displays the Ladder Logic program one network at a time



Ladder Element

MSTR - Send protocol master message

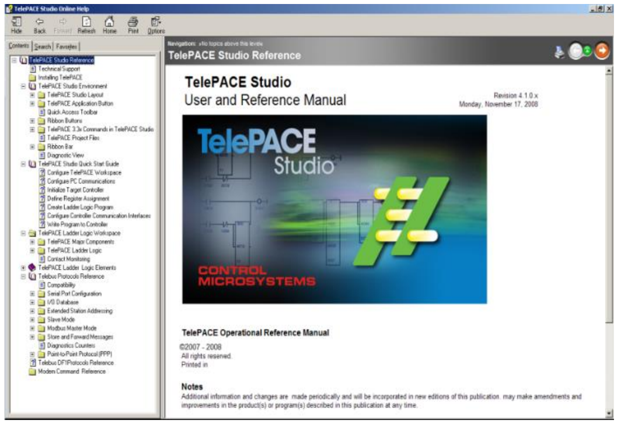
Properties

Element Part 1	
Type	Register 4nnnn
Value	40001
Tag	Config
Element Part 2	
Type	Register 4nnnn
Value	40008
Tag	Timer

TelePACE Studio Layout

TelePACE Studio Help File

- The **Help** file in previous version of TelePACE used PDF
- This format did not easily allow navigation through the file
- TelePACE Studio now uses a Compiled Help file (CHM) format
- This format provides for easy navigation forward and backward through the entire file as well as enhanced search capabilities



- TelePACE Studio Reference
  - Technical Support
  - Installing TelePACE
  - TelePACE Studio Environment
    - TelePACE Studio Layout
    - TelePACE Application Button
    - Quick Access Toolbar
    - Ribbon Buttons
    - TelePACE 3.3x Commands in TelePACE Studio
    - TelePACE Project Files
    - Ribbon Bar
    - Diagnostic View
  - TelePACE Studio Quick Start Guide
    - Configure TelePACE Workspace
    - Configure PC Communications
    - Initialize Target Controller
    - Define Register Assignment
    - Create Ladder Logic Program
    - Configure Controller Communication Interfaces
    - Write Program to Controller
  - TelePACE Ladder Logic Workspace
    - TelePACE Major Components
    - TelePACE Ladder Logic
    - Contact Monitoring
  - TelePACE Ladder Logic Elements
  - Telebus Protocols Reference
    - Compatibility
    - Serial Port Configuration
    - I/O Database
    - Extended Station Addressing
    - Slave Mode
    - Modbus Master Mode
    - Store and Forward Messages
    - Diagnostics Counters
    - Point-to-Point Protocol (PPP)
  - Telebus DF1Protocols Reference
  - Modem Command Reference

TelePACE Studio Reference

TelePACE Studio
User and Reference Manual

Revision 4.1.0.x
Monday, November 17, 2008

TelePACE Operational Reference Manual

©2007 - 2008
All rights reserved.
Printed in

Notes
Additional information and changes are made periodically and will be incorporated in new editions of this publication. may make amendments and improvements in the product(s) or program(s) described in this publication at any time.

TelePACE Studio Layout: Review Questions

1. Where would you find the **File** and **Print** options?

2. In the **Network Canvas** display, how many networks are displayed at the same time?

3. How do you hide and restore the function block palette?

Program Parameters

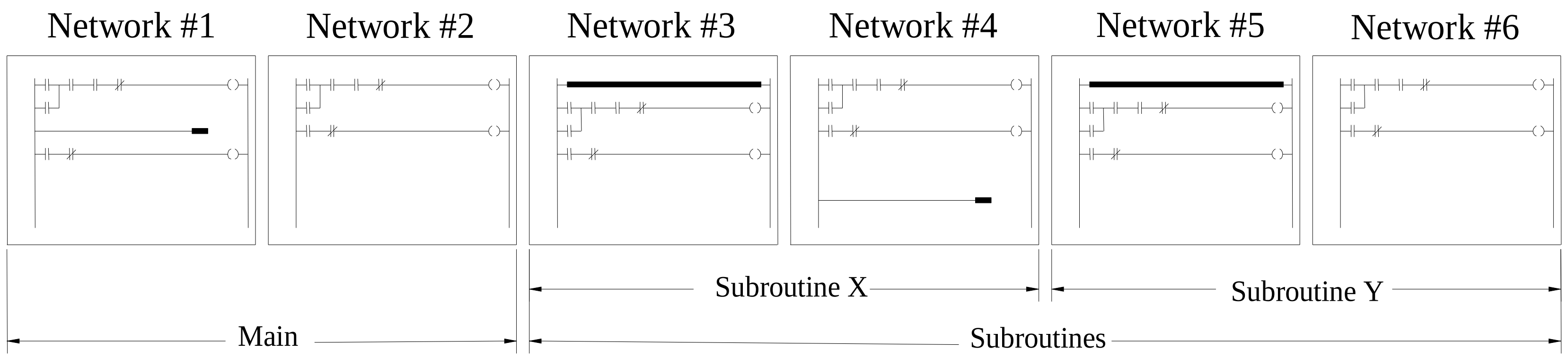
Program Parameters

Program Network Layout

- The Ladder Logic program consists of the main program followed by a number of subroutines
- The main program is defined as all the logic networks up to the start of the first subroutine, or until the end of the program if no subroutines exist
- A subroutine is defined as all the logic networks from a subroutine element until the next subroutine element, or the end of the program if there are no more subroutines

Program Parameters

Program Network Layout



- ***The main program consists of networks 1 and 2***
- ***The next four networks make up the subroutines, with networks 3 and 4 comprising subroutine X and networks 5 and 6 comprising subroutine Y***
- ***If subroutines were not needed, the entire program would consist of a main section only***

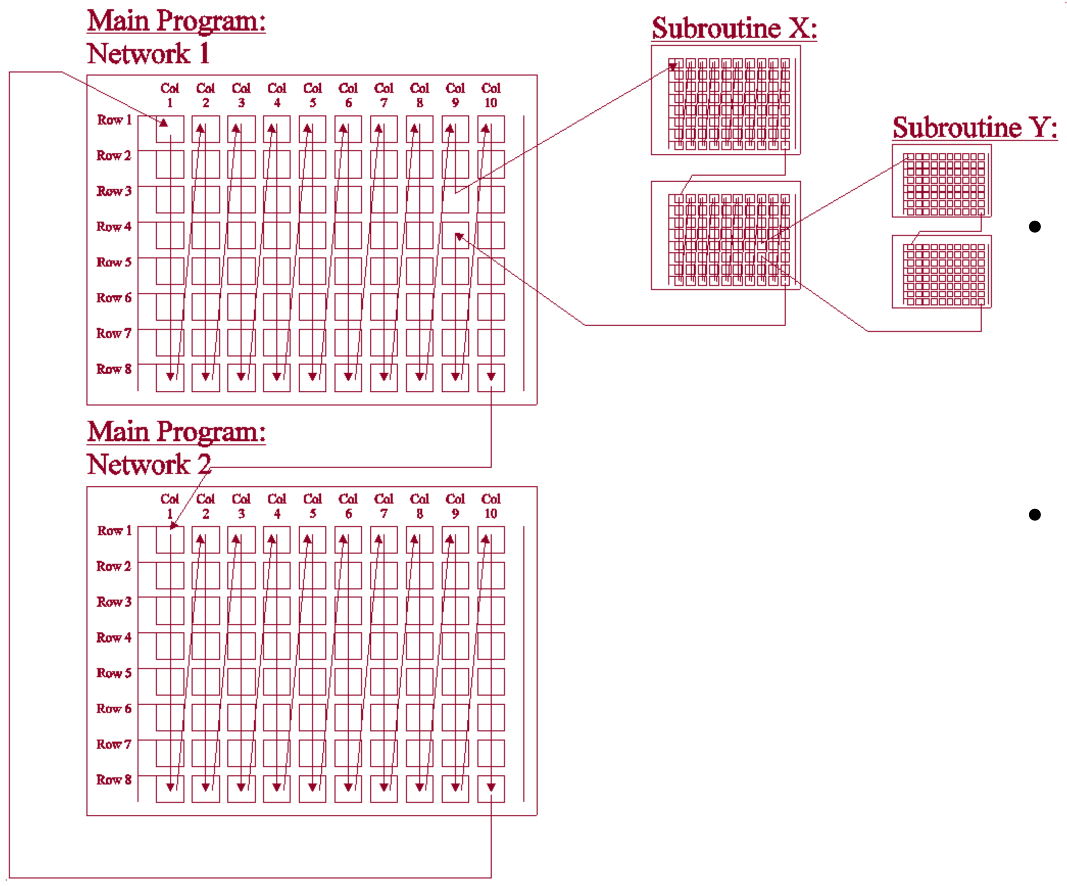
Program Parameters

Subroutines

- The program is executed from the start of the main program to the end of the main program
- If a subroutine call function block is encountered, execution transfers to the start of the subroutine and continues until the end of the subroutine
- Execution then returns to the element after the subroutine call element
- Subroutine execution can be nested.
  - This allows subroutines to call other subroutines
- Subroutine execution cannot be recursive. This prevents potential infinite loops in the ladder logic program

Program Parameters

Program Execution Order



- The controller evaluates each element, or function block, in a sequence that starts at the top left hand corner; or ROW 1, COLUMN 1.
- The ladder evaluation moves down column 1 until it reaches row 8. The evaluation then continues at the top of column 2 and moves down to row 8 again.
- This process continues until the entire network has been evaluated. If there is more than one network, evaluation continues to the next sequential network in the program until the entire program has been evaluated.

Program Parameters

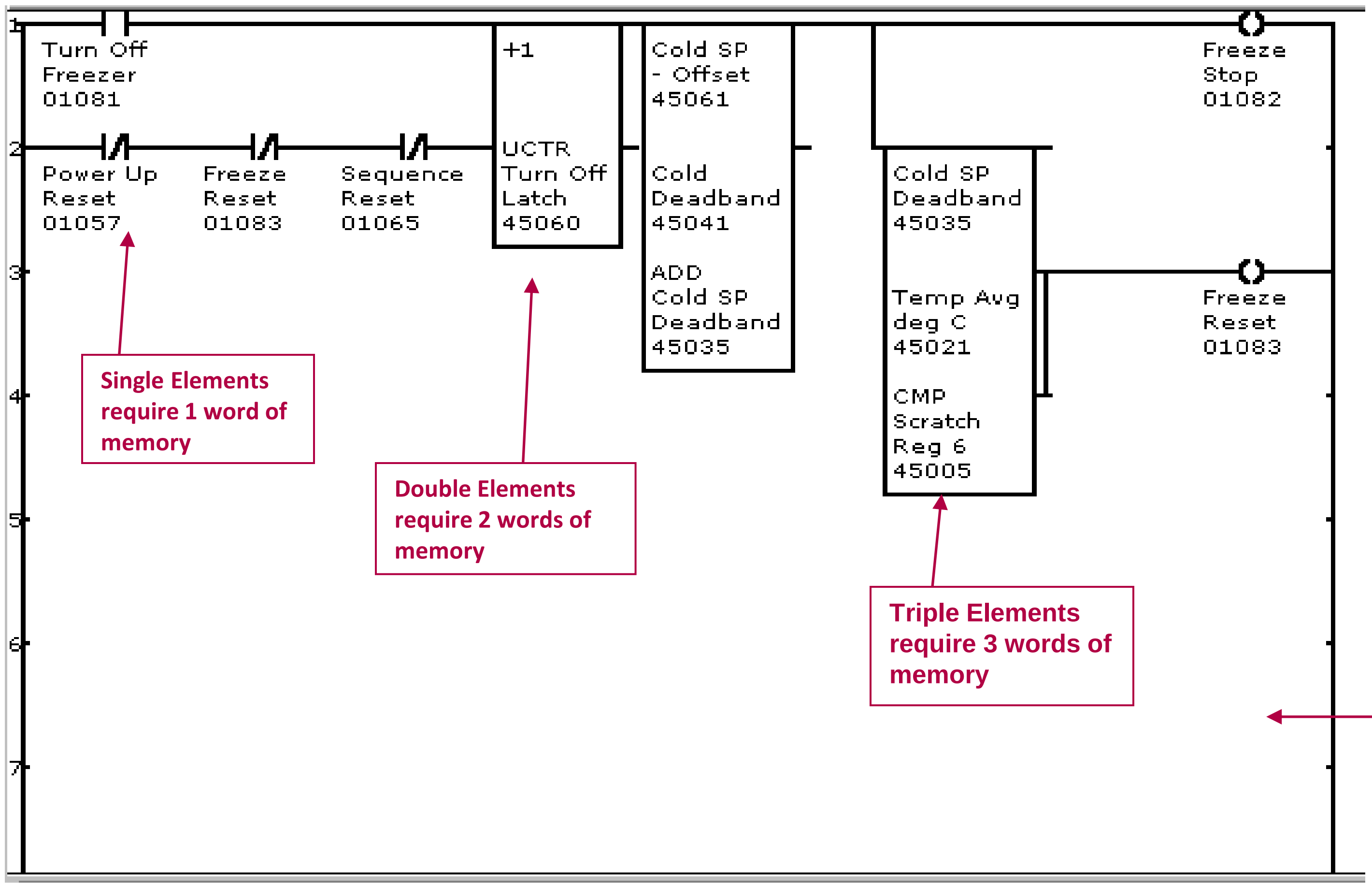
Ladder Logic Memory Usage

- Memory usage in a Ladder Logic application program is based on the number of networks and number of elements used by a program

Networks	Each network in an application requires one word of memory, whether the network is used or not.
Columns	Each column that is occupied in a network requires one word of memory.
Single Elements	Each single element requires one word of memory.
Double Elements	Each double element requires two words of memory.
Triple Elements	Each triple element requires three words of memory.

Program Parameters

Ladder Logic Memory Usage



Each Network requires 1 word of memory

Single Elements require 1 word of memory

Double Elements require 2 words of memory

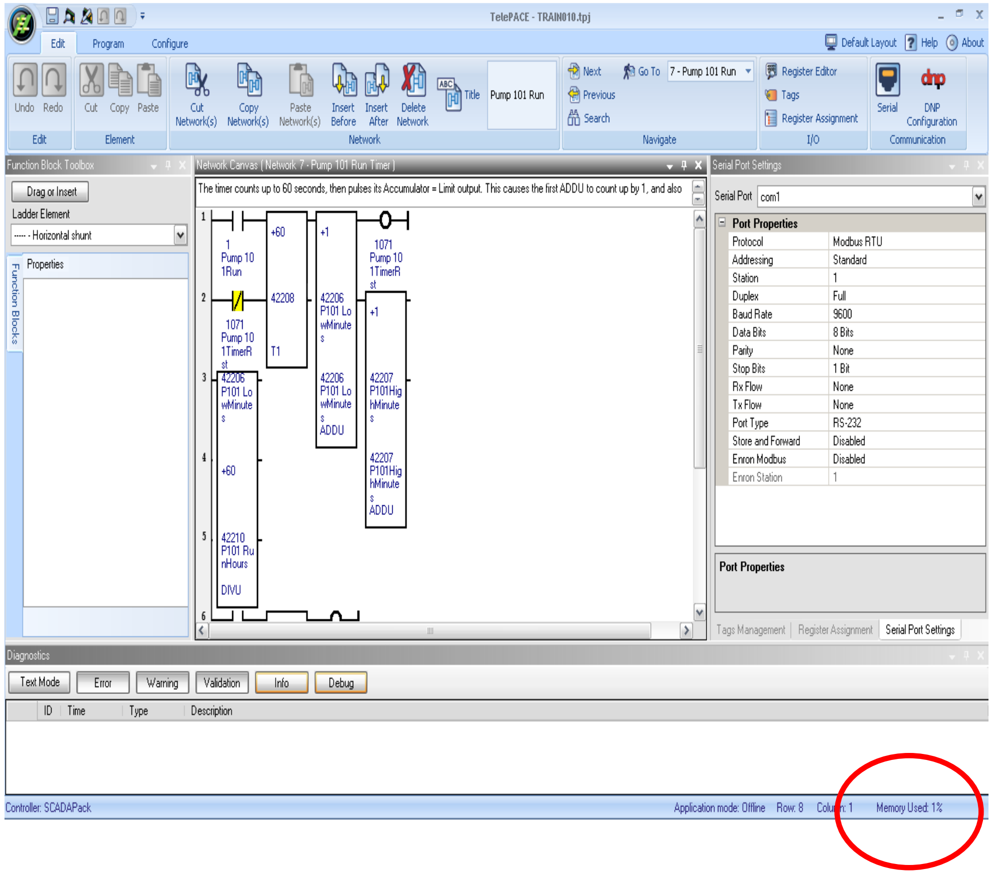
Triple Elements require 3 words of memory

Each Column requires 1 word of memory

Program Parameters

Ladder Logic Memory Usage

- There are 12k words of program space available in all SCADAPack models
- Limiting the number of unnecessary horizontal shunts can reduce the program size
- A “Memory Used” percentage is located in the bottom right corner of the screen



The timer counts up to 60 seconds, then pulses its Accumulator = Limit output. This causes the first ADDU to count up by 1, and also

Serial Port: com1

Port Properties	
Protocol	Modbus RTU
Addressing	Standard
Station	1
Duplex	Full
Baud Rate	9600
Data Bits	8 Bits
Parity	None
Stop Bits	1 Bit
Rx Flow	None
Tx Flow	None
Port Type	RS-232
Store and Forward	Disabled
Enron Modbus	Disabled
Enron Station	1

Controller: SCADAPack   Application mode: Offline   Row: 8   Column: 1   Memory Used: 1%

I/O Database

I/O Database

- The TeleBUS protocols read and write information from the I/O database.
- The I/O database contains User-Assigned Registers and General Purpose Registers:
  - **User-assigned registers** map directly to the I/O hardware or system parameter in the controller. Assigned registers are initialized to the default hardware state or system parameter when the controller's power is cycled. Assigned output registers do not maintain their values during power failures. However, output registers do retain their values during application program loading.
  - **General purpose registers** are used by ladder logic and C application programs to store processed information and to receive information from remote devices. General purpose registers retain their values during power failures and application program loading. The values change only when written by an application program or a communication protocol or on a "Cold-Boot".

I/O Database

The I/O database is divided into four sections:

- **Coil registers** are single bits which the protocols can read and write. Coil registers are located in the digital output section of the I/O database. The number of registers depends on the controller. Coil registers are numbered from 00001 to the maximum for the controller

- **Status registers** are single bits which the protocol can read. Status registers are located in the digital input section of the I/O database. The number of registers depends on the controller. Status registers are numbered from 10001 to the maximum for the controller.

I/O Database

- **Input registers** are 16-bit registers which the protocol can read. Input registers are located in the analog input section of the I/O database. The number of registers depends on the controller. Input registers are numbered from 30001 to the maximum for the controller

- **Holding registers** are 16 bit registers that the protocol can read and write. Holding registers are located in the analog output section of the I/O database. The number of registers depends on the controller. Holding registers are numbered from 40001 to the maximum for the controller.

I/O Database

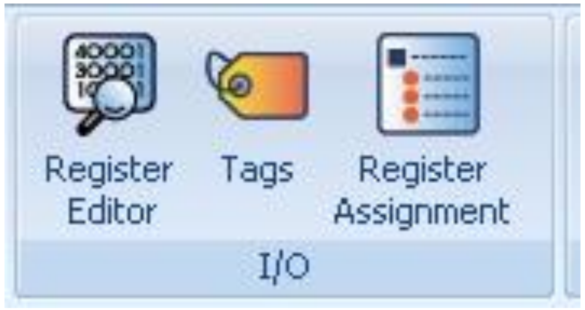
Register Type	Typical Usage *	Address Range	No. of Registers	Register Size	Value Range	Access
Coil Registers	Digital Outputs	00001 to 04096	4096	1 bit	1 or 0 (ON or OFF)	read and write
Status Registers	Digital Inputs	10001 to 14096	4096	1 bit	1 or 0 (ON or OFF)	read only
Input Registers	Analog Inputs	30001 to 39999 **	9999	16 bits	0 to 65535 in Unsigned format -32768 to 32767 in Signed format	read only
Holding Registers	Analog Outputs	40001 to 49999	9999	16 bits	0 to 65535 in Unsigned format -32768 to 32767 in Signed format	read and write

* *All registers may also be used as general purpose registers to save data in memory (e.g. memory for program data, internal variables).*

** *30001 to 39999 for SCADAPack 32*

30001 to 31024 for SCADAPack controllers

I/O Database



Register Assignment

- The Register Assignment command is selected from the I/O group in the Edit Ribbon
- All I/O hardware that is used by the controller must be assigned to I/O database registers in order for the I/O data to be used by the ladder program.
- Ladder logic programs may read data from, or write data to the I/O hardware through user-assigned registers in the I/O database

I/O Database

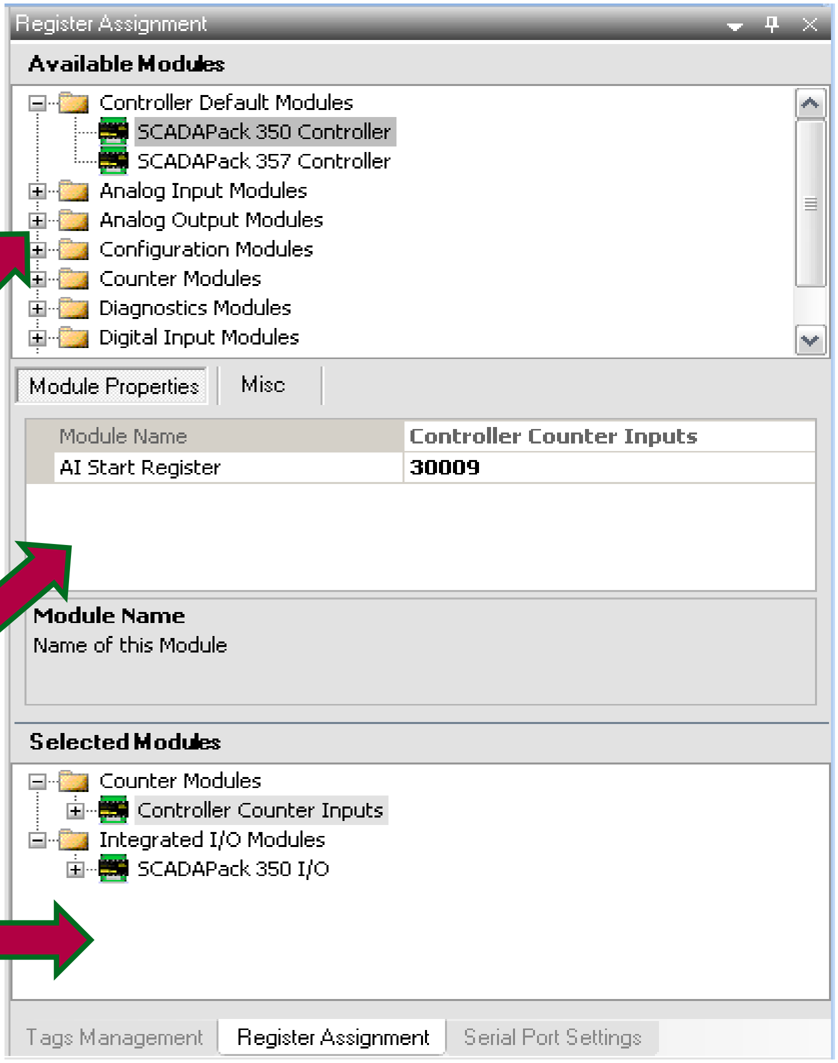
Register Assignment

- The Register Assignment workspace is presented as a single view, divided into 3 general sections

Available Modules view presents a list of the I/O modules supported by the selected PLC. A module that has to be added to the project is selected from this view

Module Properties data grid allows editing of all module properties

Selected Modules view displays the register assignment modules that have been added to the TelePACE project



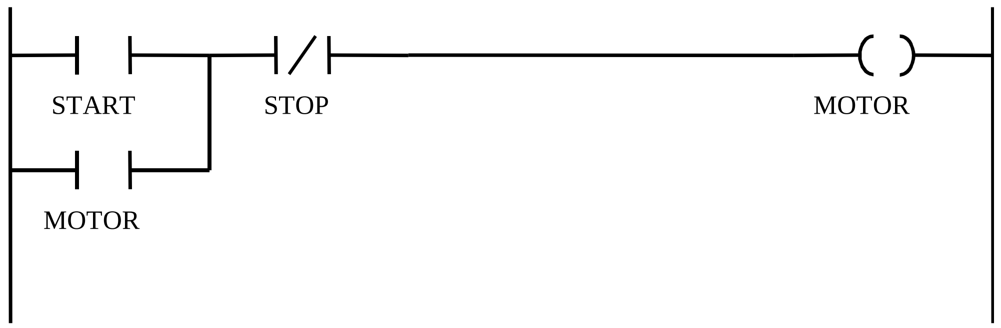
Module Name	Controller Counter Inputs
AI Start Register	30009

Module Name
Name of this Module

Program Development

Program Development

- To demonstrate the steps required developing a ladder logic program using TelePACE Studio we will develop a ladder logic application
- A simple Motor Control Circuit
  - Closing the START contact applies power to the Motor Control Coil
  - A contact on the motor coil holds on the power when the START contact is released
  - Pressing the STOP switch opens the normally closed STOP contact, and removes power from the motor control coil

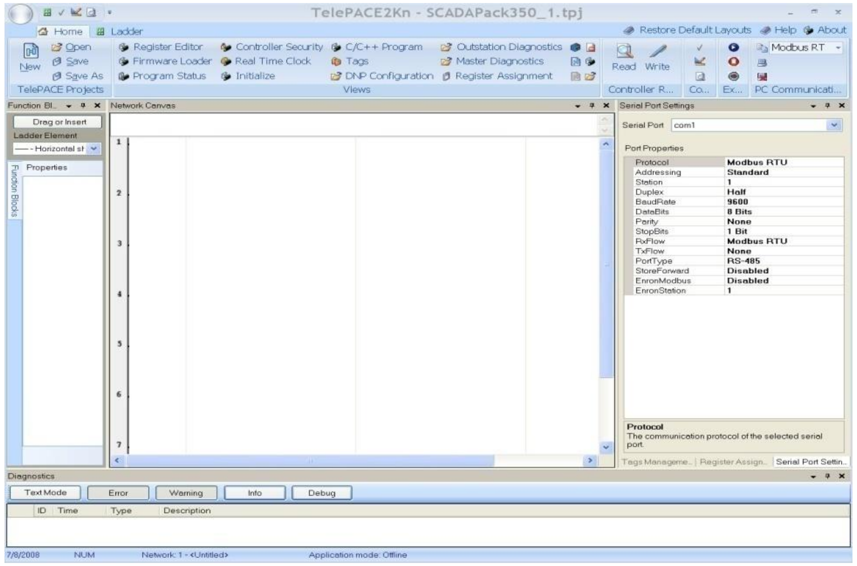


Program Development

Step 1:

Configuring the TelePACE Studio workspace

- The configuration of the TelePACE workspace involves setting views required to configure the needed parameters for an application.
- The Default Layout contains all the views needed to create a TelePACE project.
- The TelePACE workspace may be modified for personal use.



Serial Port Settings

Serial Port: com1

Port Properties

Protocol	Modbus RTU
Addressing	Standard
Station	1
Duplex	Half
BaudRate	9600
DataBits	8 Bits
Parity	None
StopBits	1 Bit
RxFlow	Modbus RTU
TxFlow	None
PortType	RS-485
StoreForward	Disabled
EnronModbus	Disabled
EnronStation	1

Protocol
The communication protocol of the selected serial port.

Program Development

Step 2:

Configuring the controller for SERVICE Mode

- Remove power from the controller.
- Press down on the LED POWER button.
- Apply power to the controller.
- Continue holding the LED POWER button until the STAT LED starts turns on.
- Release the LED POWER button.
- If the LED POWER button is released before the STAT LED turns on, the controller will start in RUN mode.

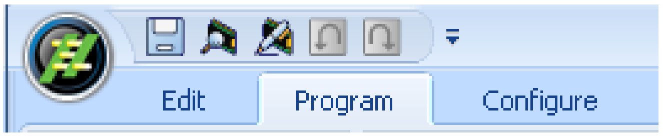
- When the controller starts in SERVICE mode:
  - default serial communication parameters of Modbus RTU, 9600 baud rate, no parity, 8 data bits, 1 stop bit and station address 1 are used for all communication ports;
  - the Ladder Logic program is stopped;
  - the C program is stopped;
  - all programs are retained in non-volatile memory

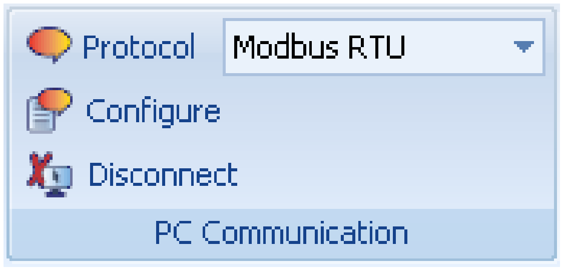
Program Development

Step 3:

Configure PC Communications

- The **PC Communication Group** is located in the Program Group of the Program Ribbon and is shown below.
- The PC Communication defines the communication protocol and communication link used for communication between a personal computer (PC) running TelePACE Studio and a target controller.





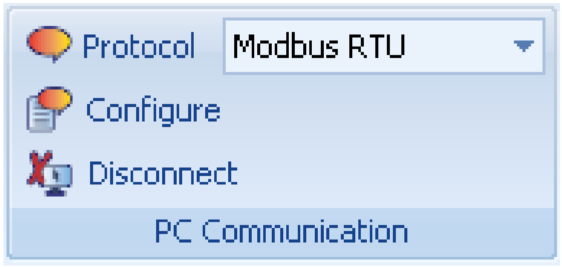
Protocol: Modbus RTU

- Use Modbus USB in class.

Program Development

Step 3 continued :

- To configure the Communication Link:
- Select a communication protocol using the **Protocol** drop down list.
- Select the **Configure** command to configure the communication properties for the Protocol.
- Select the **Disconnect** command to disconnect the communication interface to allow other applications to use the PC resource

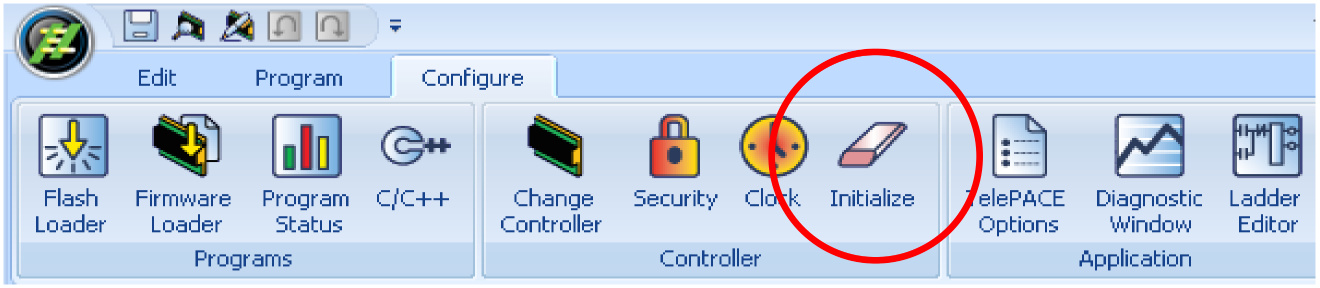


Program Development

Step 4:

Initialize Target Controller

- The **Initialize** command is used to restore a controller to default settings.
- This is typically done when starting a new project with a controller.
- The Initialize Controller dialog presented depends on the type of controller selected.

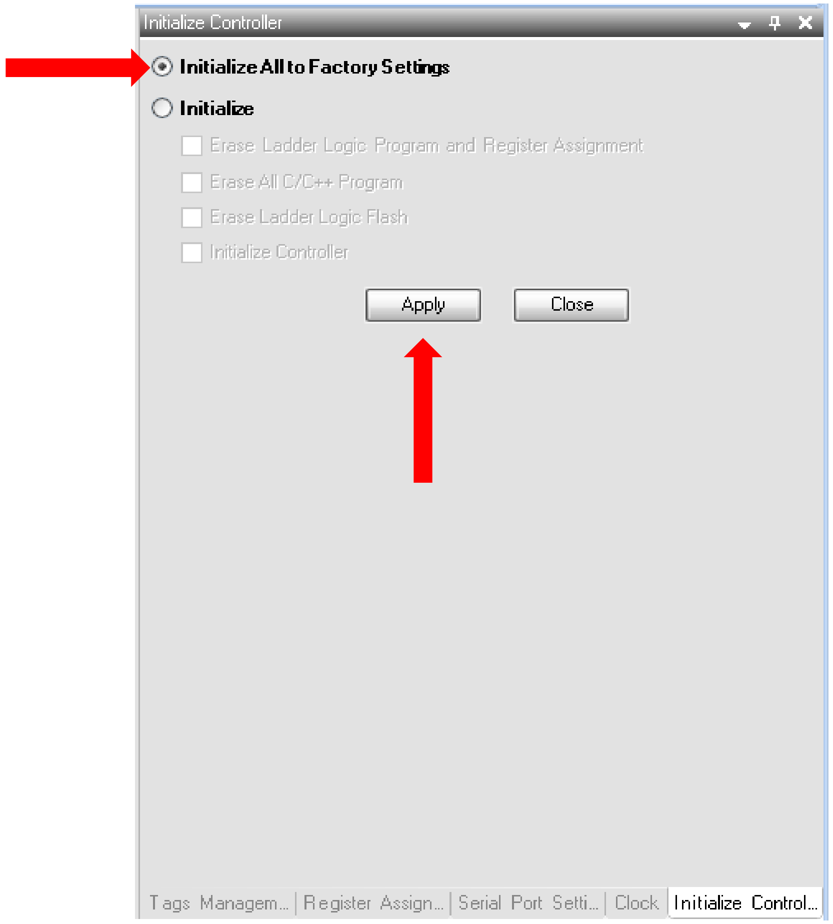


Program Development

Step 4 continued:

- Select “Initialize All to Factory Settings”
- Click “Apply”

NOTE: Initializing will erase all programs and data from the Controller


Initialize Controller

(● Initialize All to Factory Settings)

○ Initialize

☐ Erase Ladder Logic Program and Register Assignment

☐ Erase All C/C++ Program

☐ Erase Ladder Logic Flash

☐ Initialize Controller

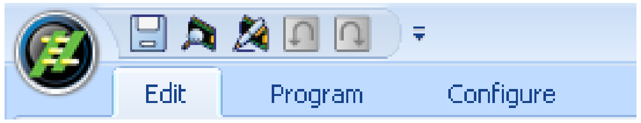
Apply
Close
Tags Managem... | Register Assign... | Serial Port Setti... | Clock | Initialize Control...

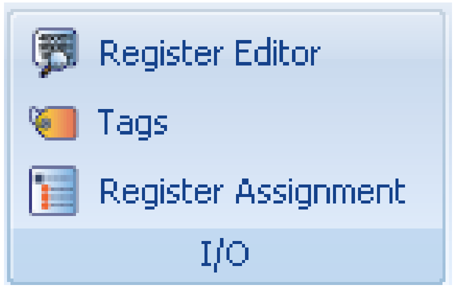
Program Development

Step 5:

Define Register Assignment

- The Register Assignment command is selected from the I/O group in the Edit Ribbon
  - All I/O hardware that is used by the controller must be assigned to I/O database registers in order for the I/O data to be used by the ladder program.
  - Ladder logic programs may read data from, or write data to the I/O hardware through user-assigned registers in the I/O database



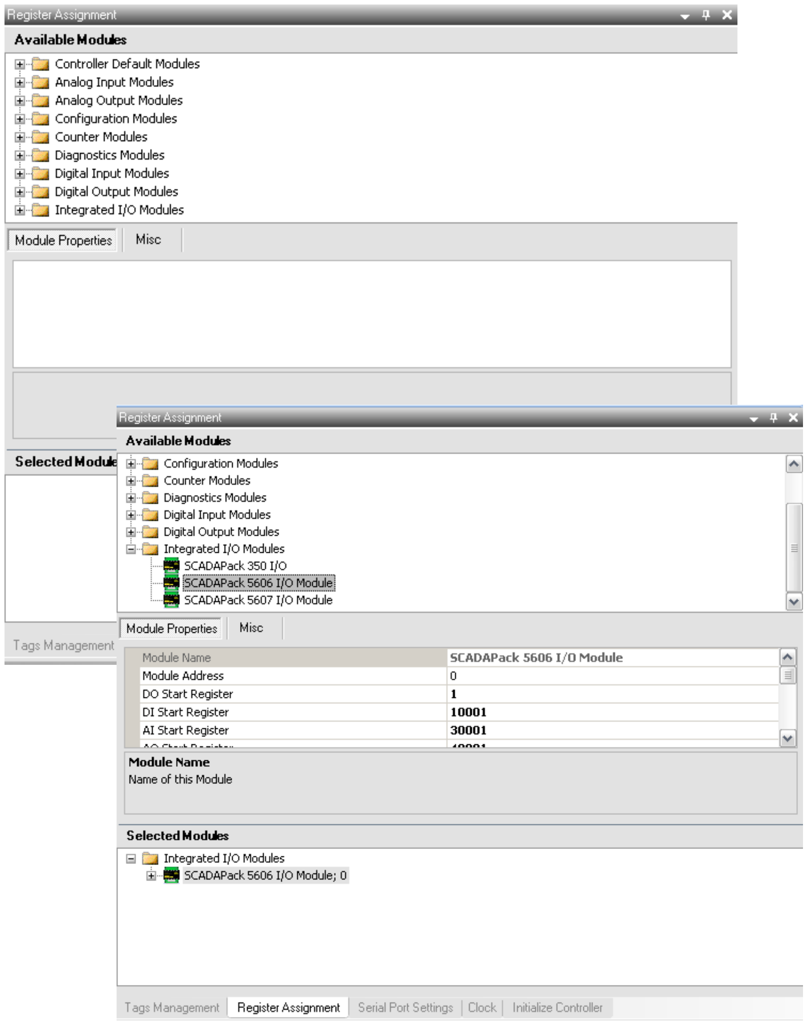


Program Development

Step 5 continued :

- To add a module from the **Available Modules** view to the **Selected Modules** view, use any of the following methods:

  1. Drag the module from the **Available Modules** view and drop it in the **Selected Modules** view
  2. Double click on the module in the **Available Modules** view to add it to the **Selected Modules** view
  3. Right click on the module in the **Available Modules** view, click **Add** from the context menu to add the module to the **Selected Modules** view



Register Assignment

Available Modules

- Controller Default Modules
- Analog Input Modules
- Analog Output Modules
- Configuration Modules
- Counter Modules
- Diagnostics Modules
- Digital Input Modules
- Digital Output Modules
- Integrated I/O Modules

Module Properties | Misc

Selected Modules

Tags Management

Register Assignment

Available Modules

- Configuration Modules
- Counter Modules
- Diagnostics Modules
- Digital Input Modules
- Digital Output Modules
- Integrated I/O Modules
  - SCADAPack 350 I/O
  - SCADAPack 5606 I/O Module
  - SCADAPack 5607 I/O Module

Module Properties | Misc

Module Name	SCADAPack 5606 I/O Module
Module Address	0
DO Start Register	1
DI Start Register	10001
AI Start Register	30001

Module Name
Name of this Module

Selected Modules

- Integrated I/O Modules
  - SCADAPack 5606 I/O Module; 0

Tags Management | Register Assignment | Serial Port Settings | Clock | Initialize Controller

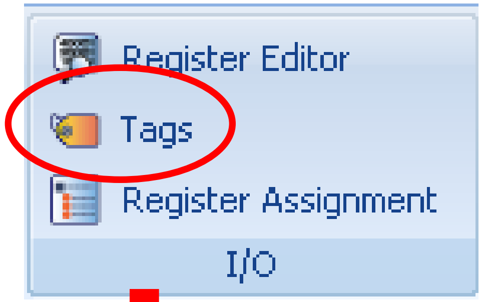
Program Development

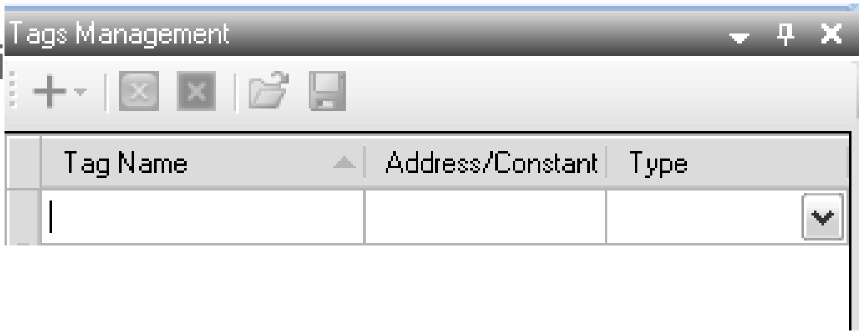
Step 6:

Adding Tag Names

- When the Tags command is selected from the I/O Group the Tag Management view is displayed

- To enter a tag name simply begin typing the name.
  - Tags names are limited to 16 alpha numeric characters and are case sensiti





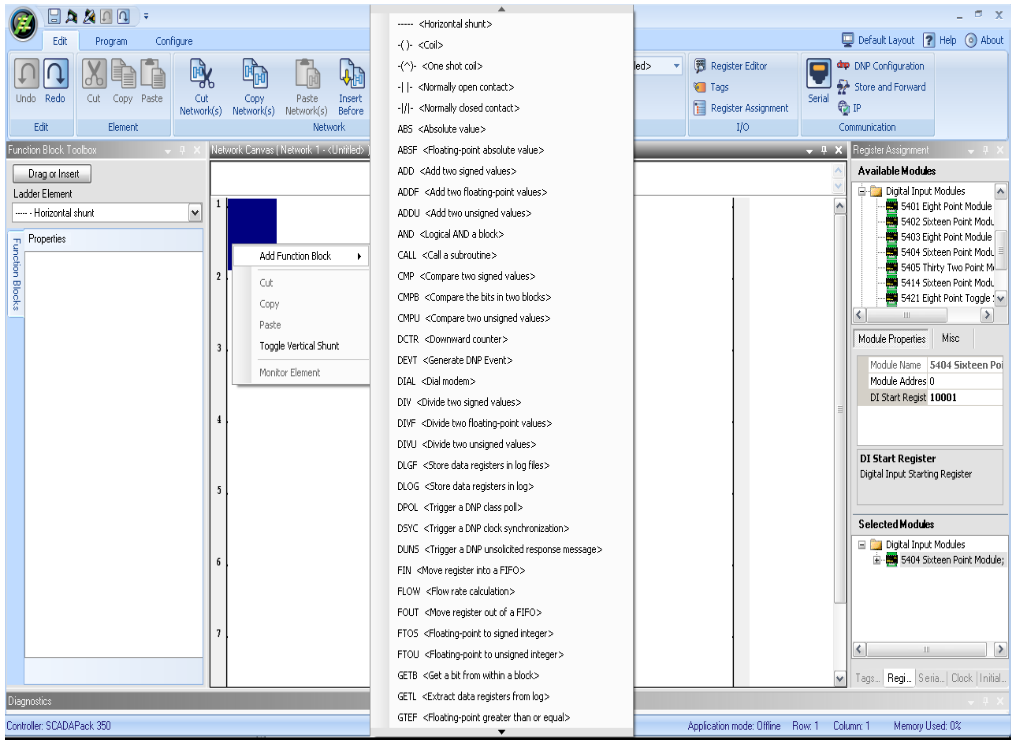
Tag Name	Address/Constant	Type

Program Development

Step 7:

Create Ladder Logic Program

- To insert an element in the Ladder network, position the cursor at the network position where the element is to be inserted Right-click the mouse button.
- A dialog box pops up and an element or function can be selected to insert



-----	<Horizontal shunt>
-( )-	<Coil>
-(^)-	<One shot coil>
-| |-	<Normally open contact>
-|/|-	<Normally closed contact>
ABS	<Absolute value>
ABSF	<Floating-point absolute value>
ADD	<Add two signed values>
ADDF	<Add two floating-point values>
ADDU	<Add two unsigned values>
AND	<Logical AND a block>
CALL	<Call a subroutine>
CMP	<Compare two signed values>
CMPB	<Compare the bits in two blocks>
CMPU	<Compare two unsigned values>
DCTR	<Downward counter>
DEVT	<Generate DNP Event>
DIAL	<Dial modem>
DIV	<Divide two signed values>
DIVF	<Divide two floating-point values>
DIVU	<Divide two unsigned values>
DLGF	<Store data registers in log files>
DLOG	<Store data registers in log>
DPOL	<Trigger a DNP class poll>
DSYC	<Trigger a DNP clock synchronization>
DUNS	<Trigger a DNP unsolicited response message>
FIN	<Move register into a FIFO>
FLOW	<Flow rate calculation>
FOUT	<Move register out of a FIFO>
FTOS	<Floating-point to signed integer>
FTOU	<Floating-point to unsigned integer>
GETB	<Get a bit from within a block>
GETL	<Extract data registers from log>
GTEF	<Floating-point greater than or equal>

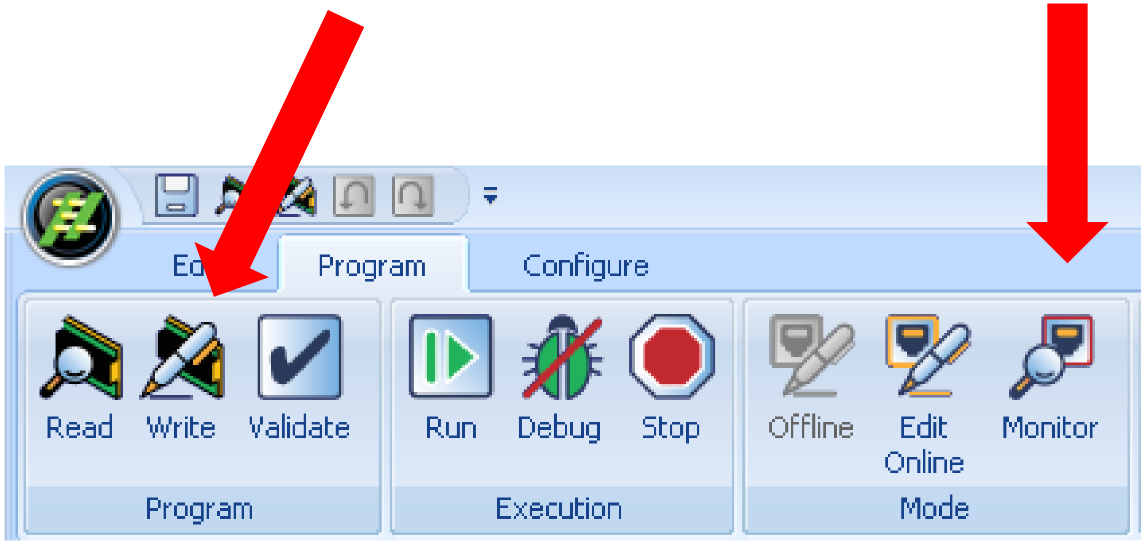
**** make sure to include comments***

Program Development

Step 8:

Write to Controller

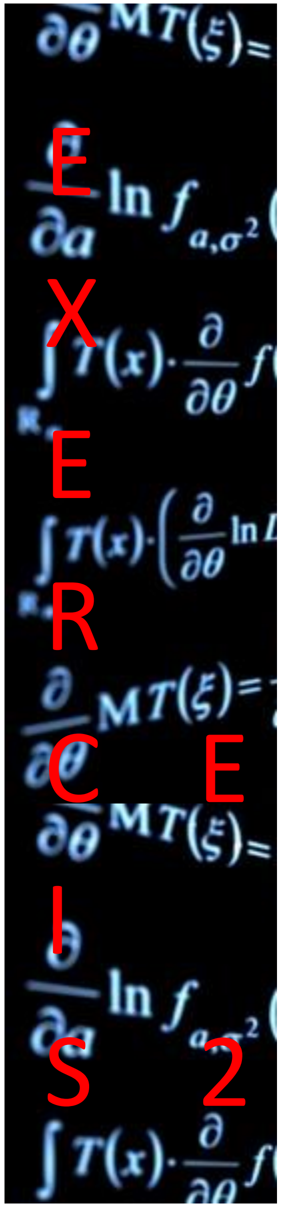
- Save your Project FIRST
- Select the Write Program to Controller icon
- Select Monitor



Exercise Two

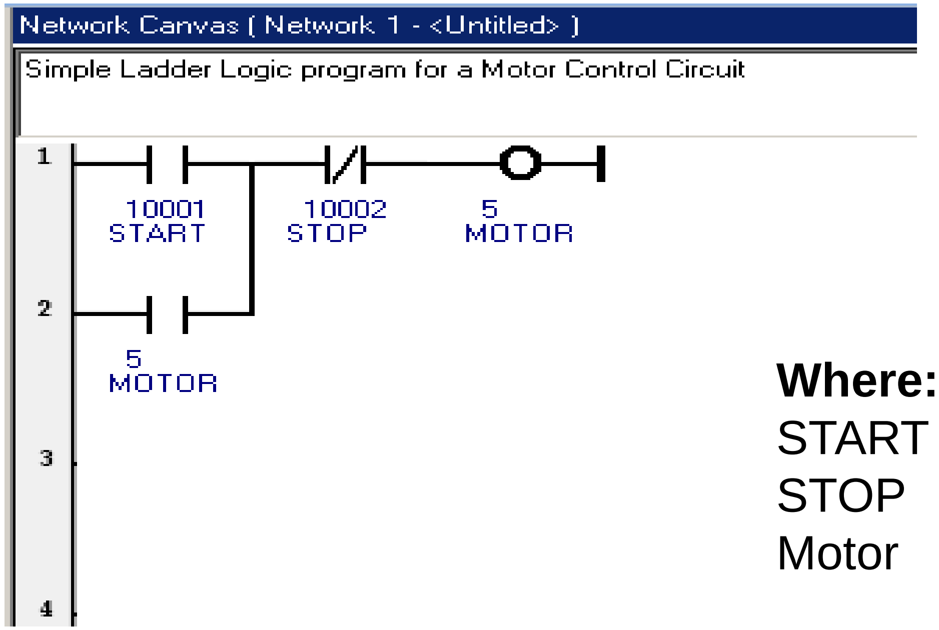
Exercise 2

- Build and Test a simple Motor Control Circuit program



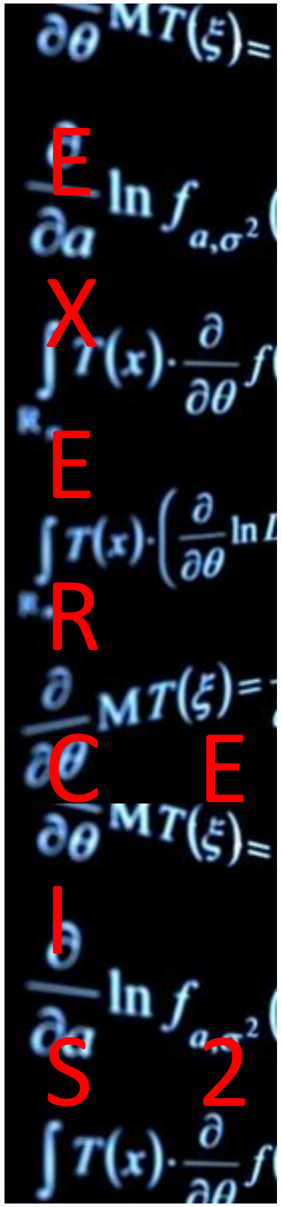
Exercise 2

- Follow the steps outlined in the Program Development section to build the following:


Network Canvas [Network 1 - <Untitled>]
Simple Ladder Logic program for a Motor Control Circuit
1
2
3
4
10001
START
10002
STOP
5
MOTOR
5
MOTOR

Where:

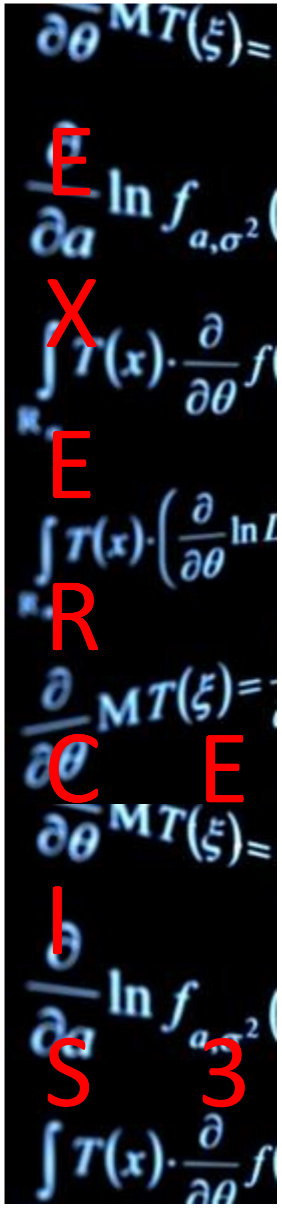
START	= Addr: 10001
STOP	= Addr: 10002
Motor	= Addr: 00005



Exercise Three

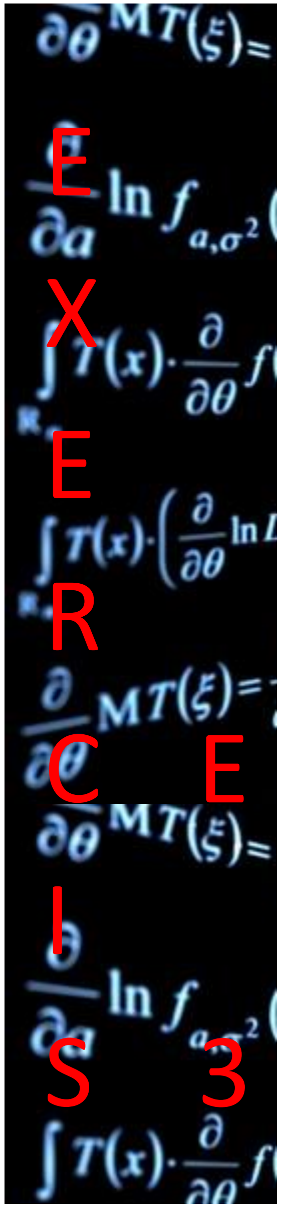
Exercise 3

- Lead / Lag Pump Control



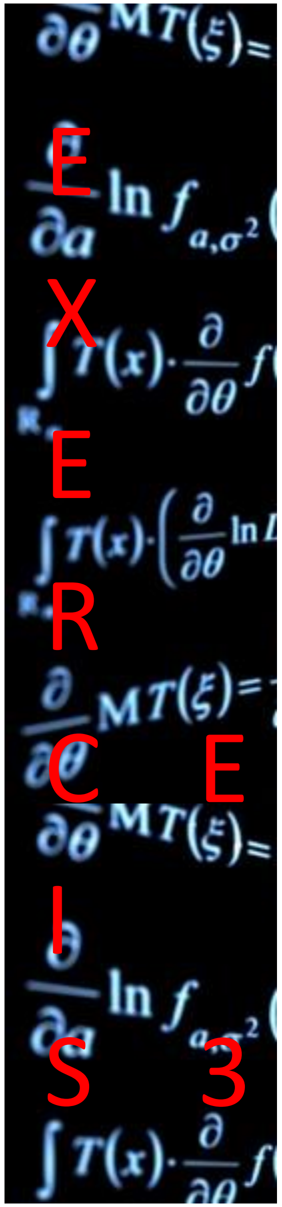
Exercise 3

- In this section of the TelePACE Ladder Logic training course a practical example of a simple lead/lag pump control program will be developed.
- A program will be developed from a very simple single pump start/stop switch control into a more complex two pump automatic lead/lag control system.



Exercise 3

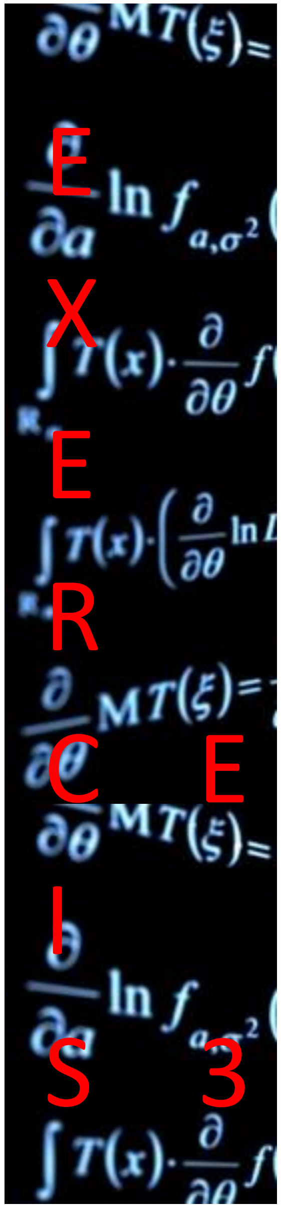
- The development of the lead / lag pump control is divided into five parts:
  - **Part One:** Controller is programmed to use the SCADAPack I/O as a means of controlling the turning on and off of a digital output. Digital inputs and then analog inputs are used to turn the output on and off. This process simulates the starting and stopping of a pump connected to the digital output.
  - **Part Two:** math functions are used to scale the analog input value used to control the pump operation. The types of numbers used in the TelePACE Ladder Logic are explained and the use of an initialization network is introduced.
  - **Part Three:** counter and timer functions are added to the program to store the accumulated pump run time and to store the number of pump starts.
  - **Part Four:** of this section adds a lag pump to the program and a lead / lag pump control program is developed.
  - **Part Five:** involves using an automatic switching sequence to control the lead/lag pump operation in the program.



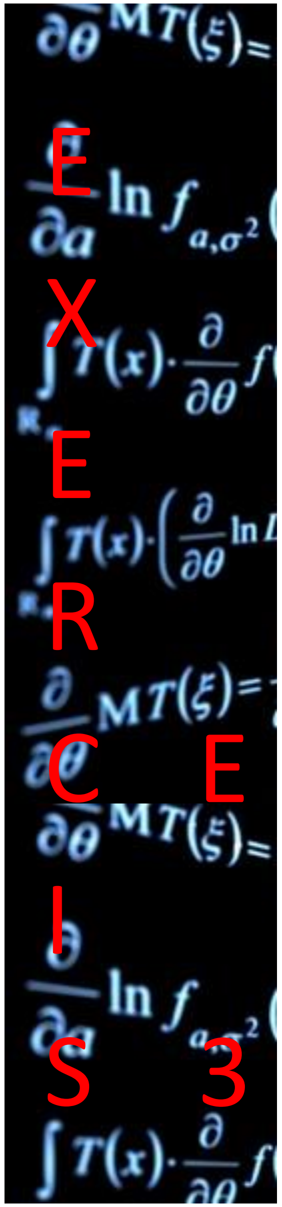
Exercise 3 – Part One

Create tag Names

- Open a new file in the TelePACE Ladder Network Editor
- Enter the Tag names from the next slide



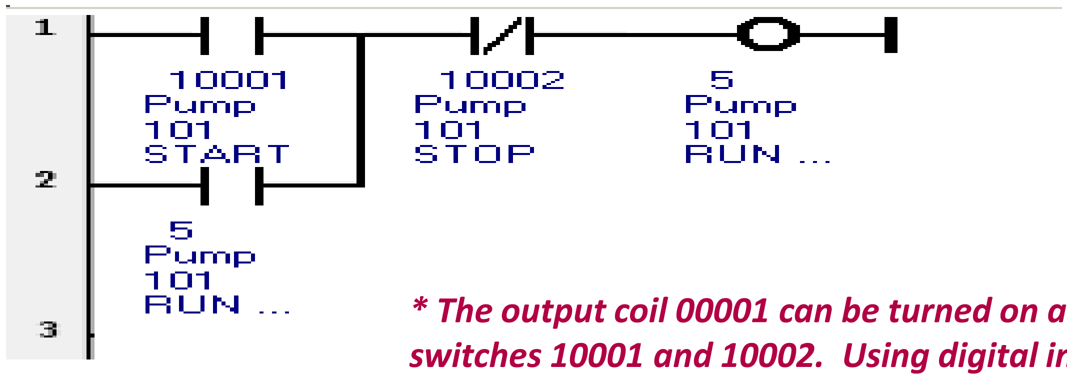
TAG NAME	Register Address
Pump 101 Start	10001
Pump 101 Stop	10002
Pump 101 Run Lead	00005
Power Up Reset	01057
LeadStop	01058
LeadStrt	01059
Pump 101 TimerRst	01071
Analog Input 00	30001
Multiply Low Word	42200
AIN 00 0_100.0%	42202
P101 Hi Minutes	42207
P101 Run Hours	42210
Pump 101 Starts	42224
LeadStrt Setpoint	43000
LeadStop Setpoint	43001



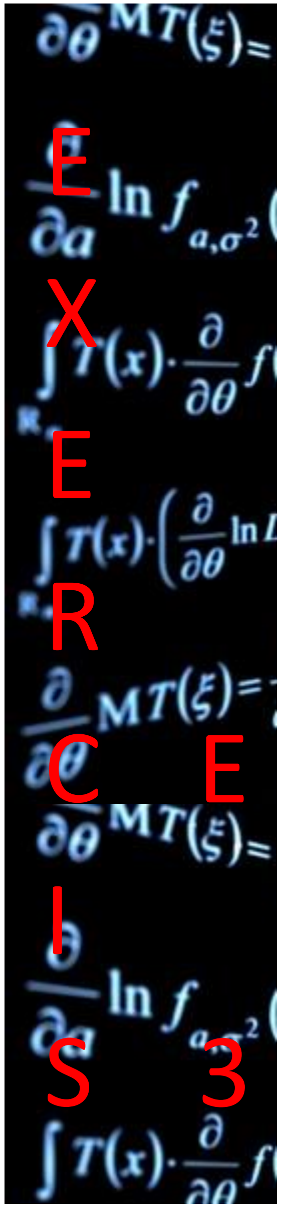
Exercise 3 – Part One

Create Program

- Build the following program
- Load program into controller
- Save the Ladder Logic program as TRAIN001.LAD



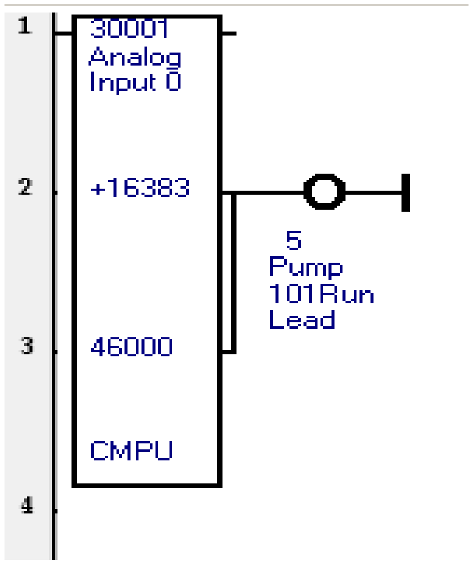
** The output coil 00001 can be turned on and off using the input switches 10001 and 10002. Using digital inputs to control digital outputs is a very common occurrence when using relay ladder logic. Pumps are usually started and stopped based on the level of the tank or reservoir they are being used to fill. In this case an analog input is used as a control for starting and stopping the pump*



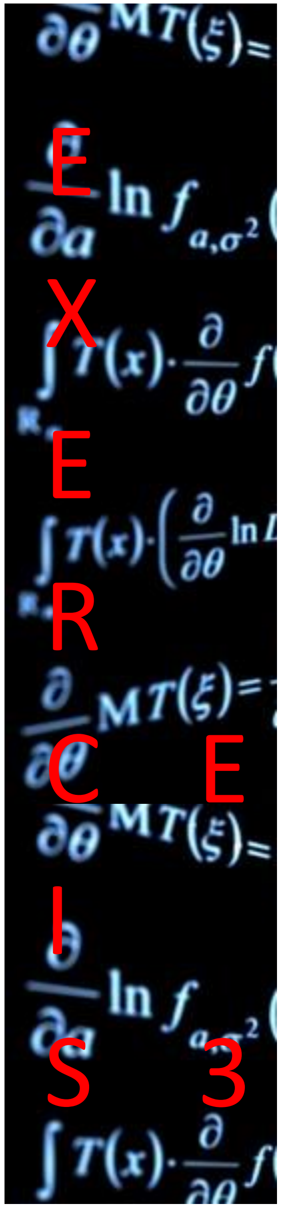
Exercise 3 – Part One

Analog Input to Start and Stop Pump

- Delete the elements that were inserted in the last step
- Select the Help menu and then search for the Ladder Logic Element **CMPU**



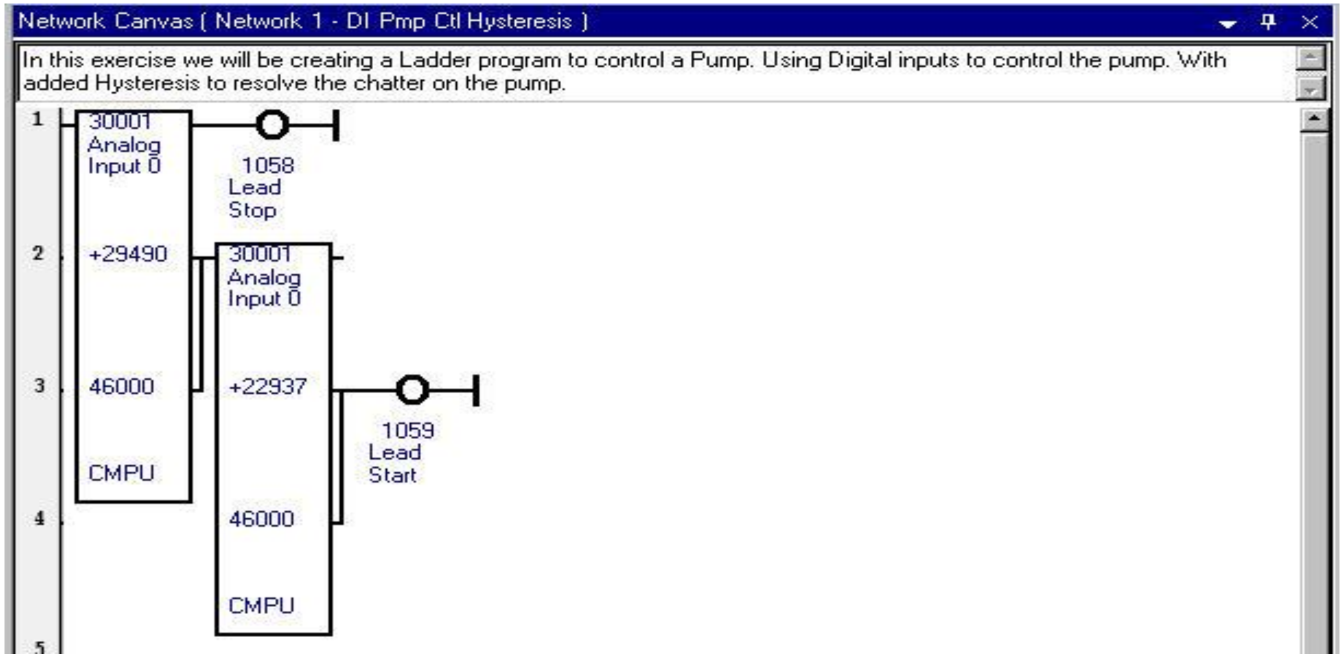
- Rebuild program as shown
- Load into controller
- Monitor



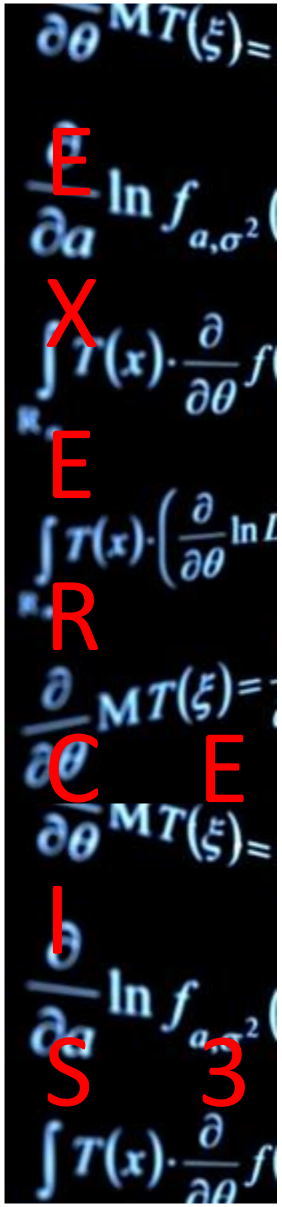
Exercise 3 – Part One

Adding Hysteresis to the Pump Control

- Modify the program as shown
- Load the program into the Controller and monitor the program execution



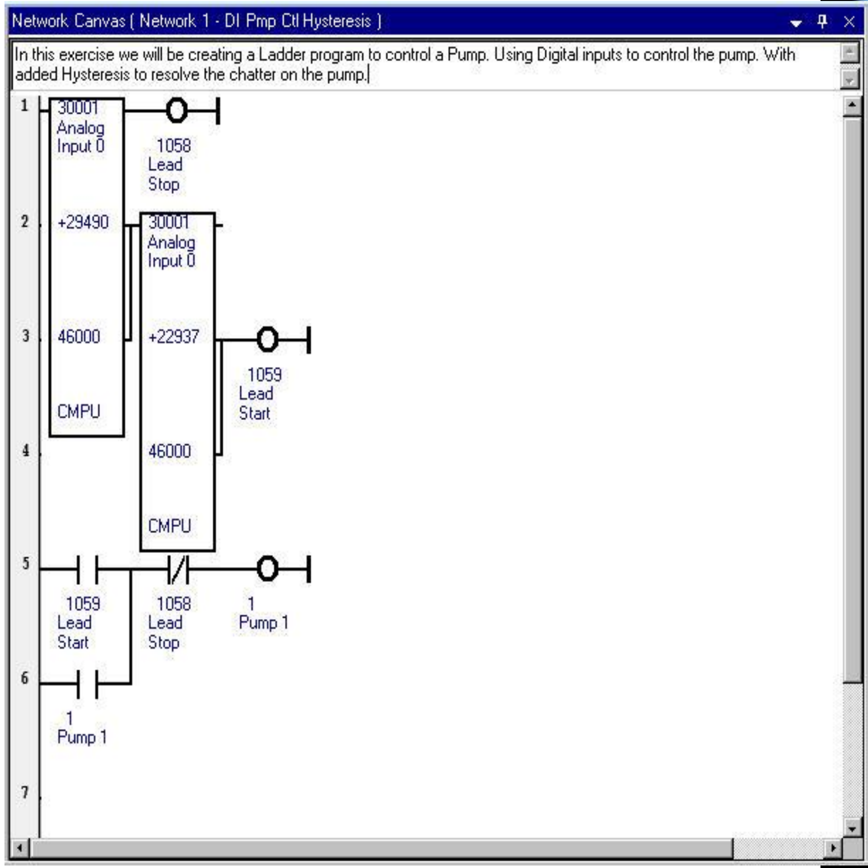
In this exercise we will be creating a Ladder program to control a Pump. Using Digital inputs to control the pump. With added Hysteresis to resolve the chatter on the pump.



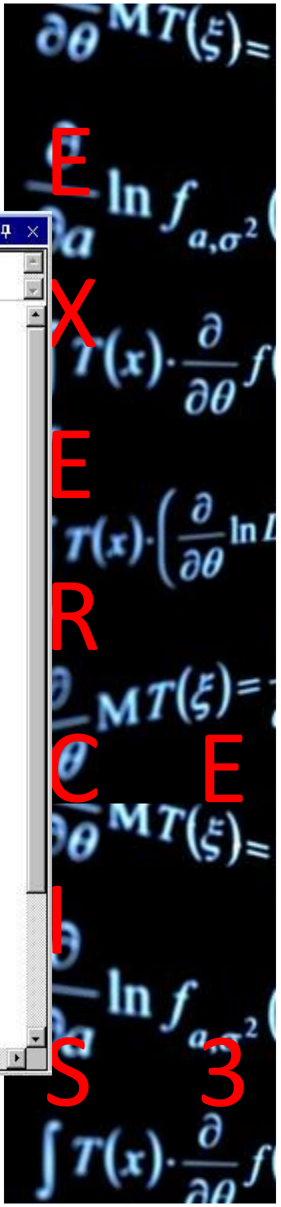
Exercise 3 – Part One

Pump Control using Hysteresis

- Modify the program as shown
- Load the program into the Controller and monitor the program execution



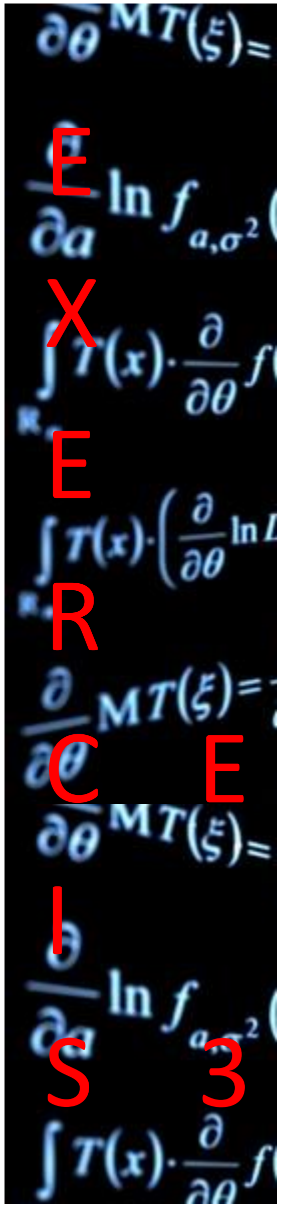
In this exercise we will be creating a Ladder program to control a Pump. Using Digital inputs to control the pump. With added Hysteresis to resolve the chatter on the pump.



Exercise 3 – Part Two

Scaling Analog Inputs

- The Ladder Logic program often must convert the raw I/O count back into the units of measurement of the process value, or to a percentage of the maximum input.
- The controller stores integer numbers in 16 bit words
- Each 16-bit word can contain an unsigned integer or a signed integer
  - Unsigned integers are positive and have the range of 0 to 65535.
  - Signed integers may be positive or negative.
  - Signed integers have a range of -32768 to +32767

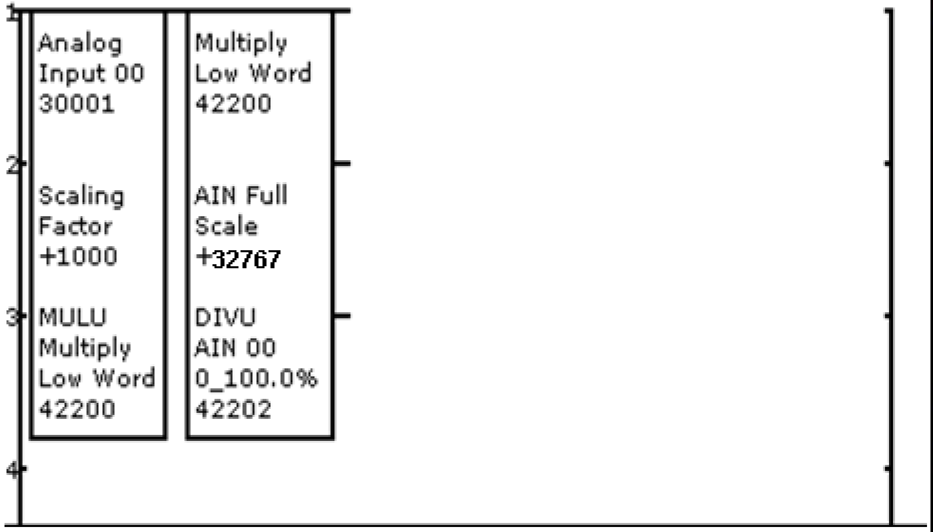


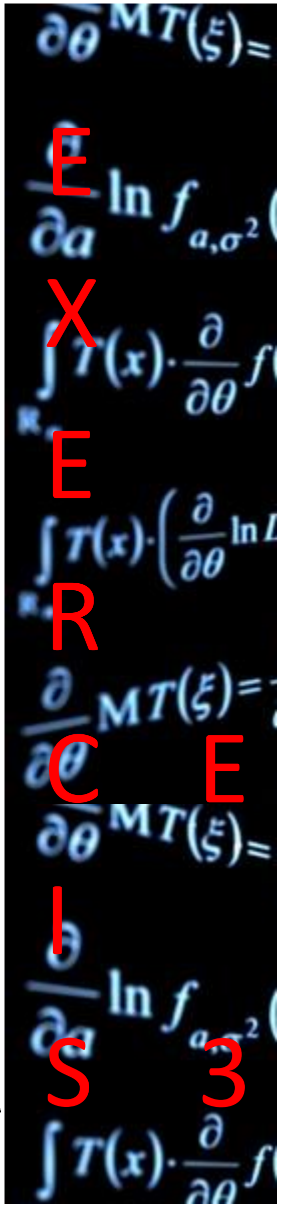
Exercise 3 – Part Two

Scaling Analog Inputs

$$\frac{[\text{ (Value of Input Register 30001)} \quad \text{x} \quad 100\% \text{]}}{\text{(Maximum Value of Input Register 30001)}}$$

- Insert a network **before** Network 1
- Modify the program as shown
- Load the program into the Controller and monitor the program execution

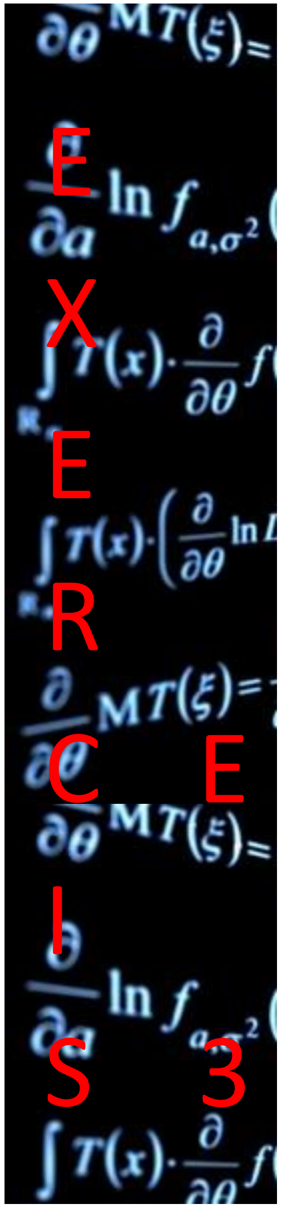




Exercise 3 – Part Two

Change Setpoints to Registers from Constants

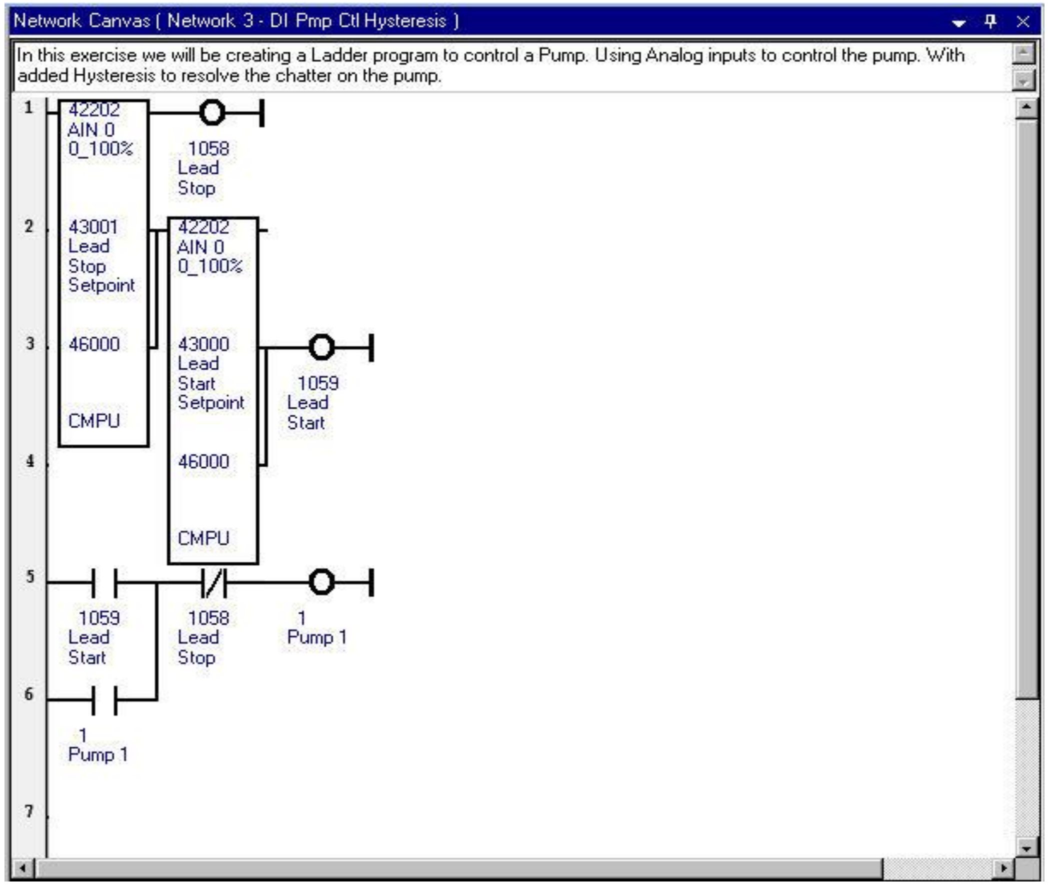
- The Start and Stop setpoint values in Network 2 will be changed from Constants to Analog Output Registers
- When a Constant is used in a Ladder Logic function, it can only be changed by editing the function using the TelePACE Studio Ladder Editor.
- To enable the Start and Stop setpoints to be changed through the I/O Database an Analog Output register must be used



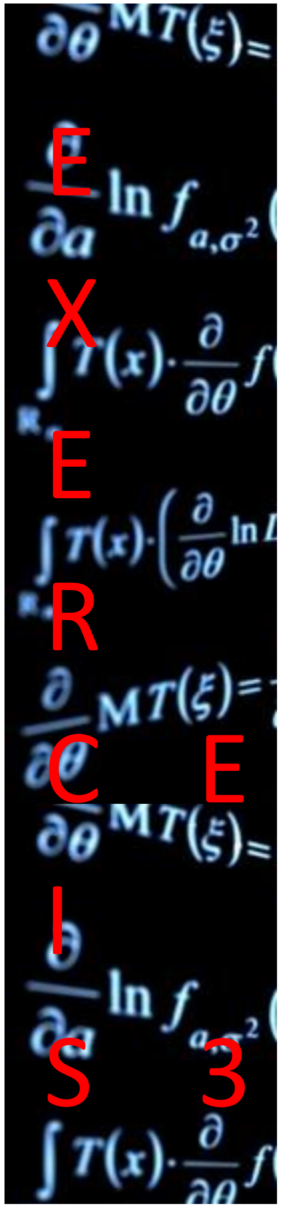
Exercise 3 – Part Two

Change Setpoints to Registers from Constants

- Modify the program as shown
- Load the program into the Controller and monitor the program execution


Network Canvas (Network 3 - DI Pmp Ctl Hysteresis)

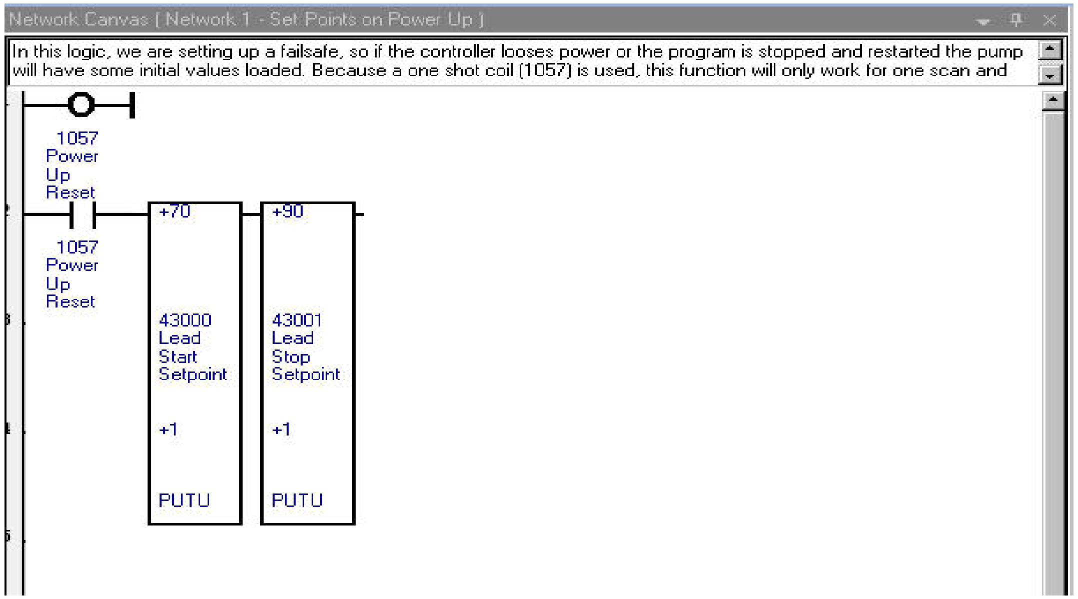
In this exercise we will be creating a Ladder program to control a Pump. Using Analog inputs to control the pump. With added Hysteresis to resolve the chatter on the pump.



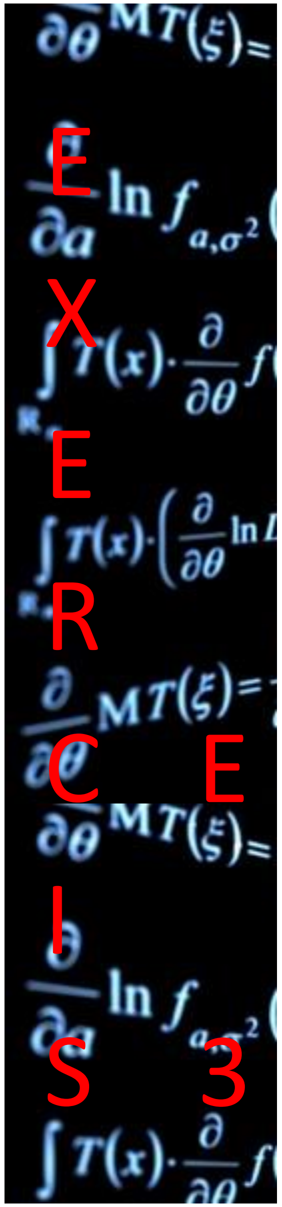
Exercise 3 – Part Two

Put start up values into the setpoint registers

- Go to Network 1 of the program and insert a network **before** Network 1
- Insert the elements as shown



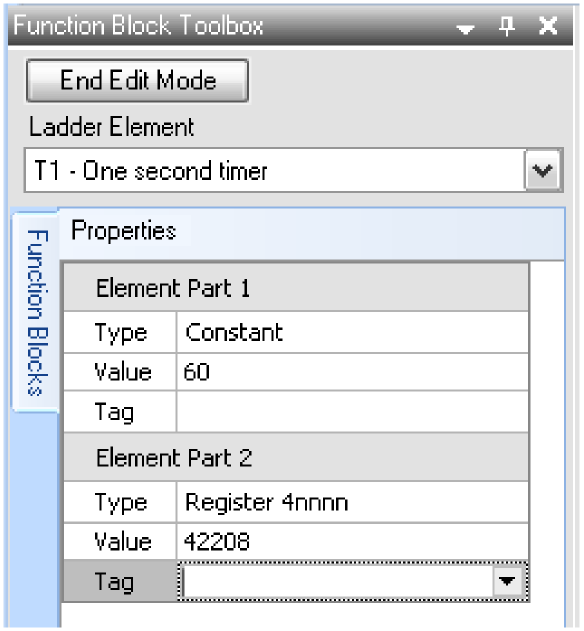
In this logic, we are setting up a failsafe, so if the controller looses power or the program is stopped and restarted the pump will have some initial values loaded. Because a one shot coil (1057) is used, this function will only work for one scan and



Exercise 3 – Part Three

Setup a One Minute Timer

- Insert a network **after** Network 3.
- In row 1 column 2 insert a T1 Timer function. Configure the timer as shown



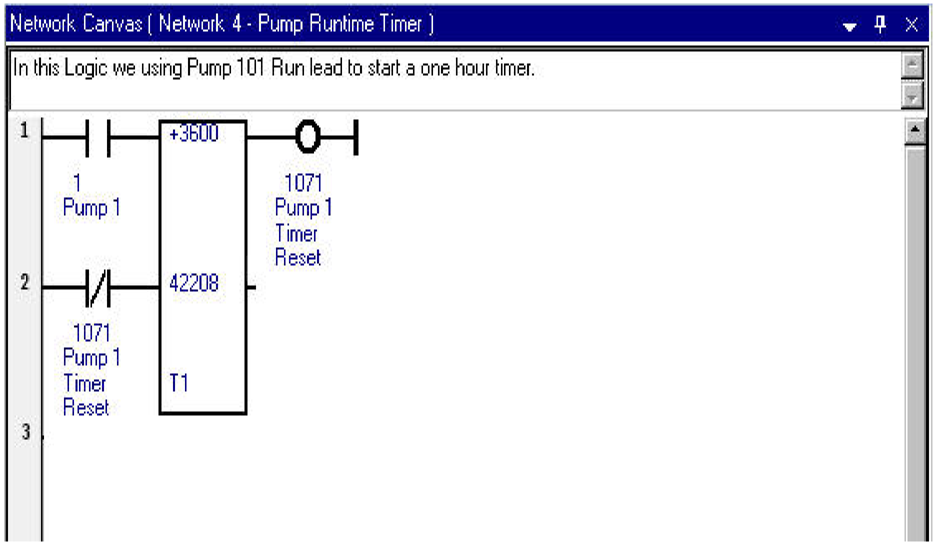
Function Block Toolbox

End Edit Mode

Ladder Element: T1 - One second timer

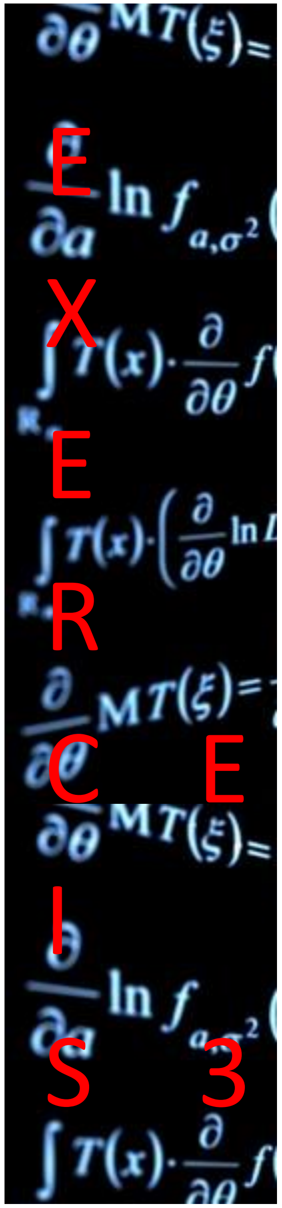
Properties

Element Part 1	
Type	Constant
Value	60
Tag	
Element Part 2	
Type	Register 4nnnn
Value	42208
Tag	



Network Canvas (Network 4 - Pump Runtime Timer)

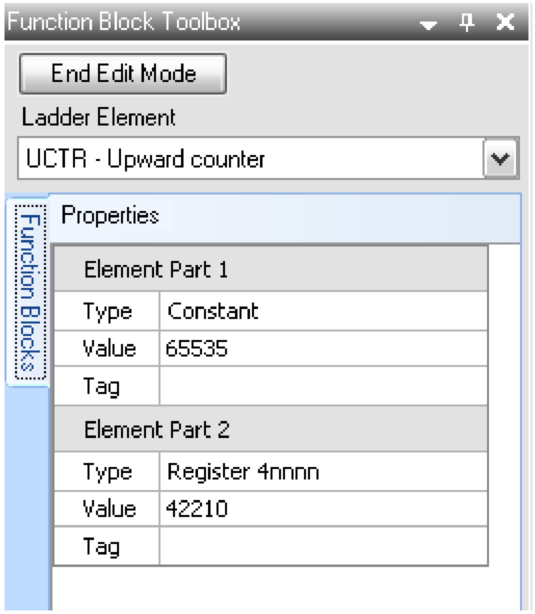
In this Logic we using Pump 101 Run lead to start a one hour timer.



Exercise 3 – Part Three

Accumulate the Pump On Time

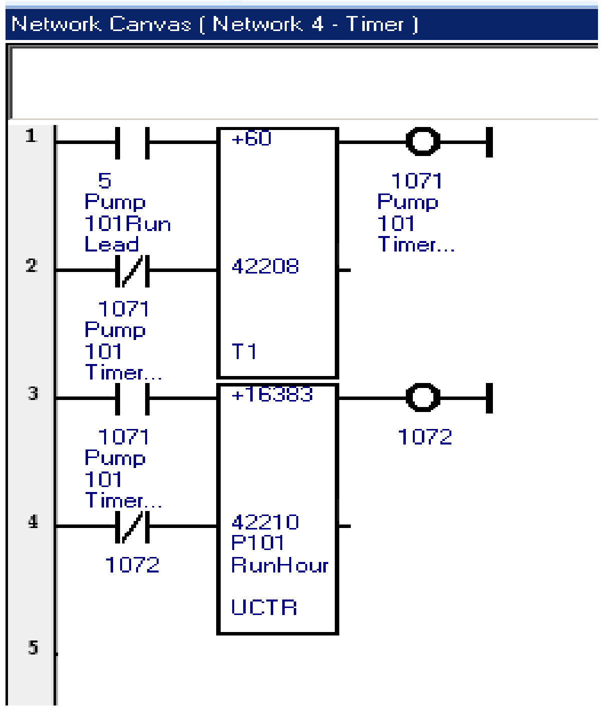
- Insert an UCTR function
- Configure the UCTR function as shown

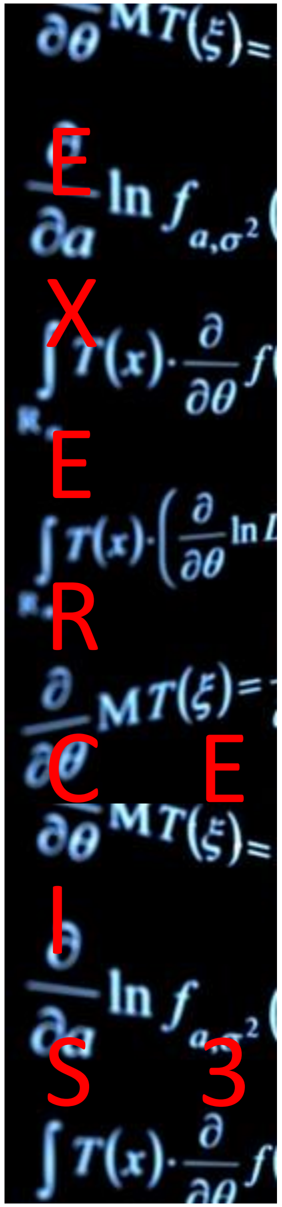


Ladder Element: UCTR - Upward counter

Element Part 1	
Type	Constant
Value	65535
Tag	

Element Part 2	
Type	Register 4nnnn
Value	42210
Tag	

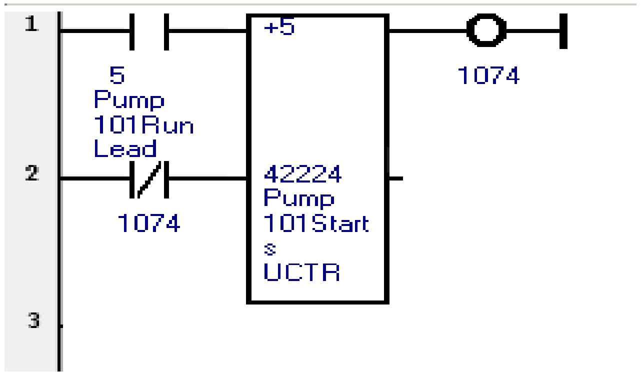


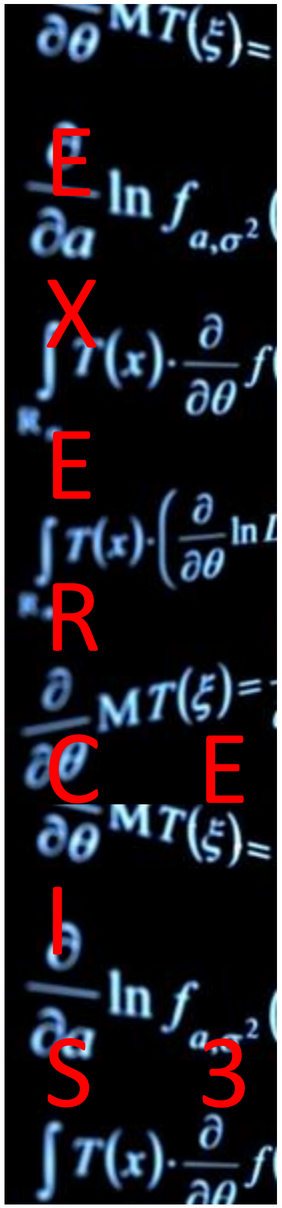


Exercise 3 – Part Three

Count the Number of Pump Starts

- Insert a network **after** Network 4.
- In row 1 Column 1 insert the N/O contact 00001.
- In row 1 Column 2 insert a UCTR function and configure it as shown
- In row 2 Column 1 insert a horizontal shunt.
- Load the program into the controller and monitor the operation of the program

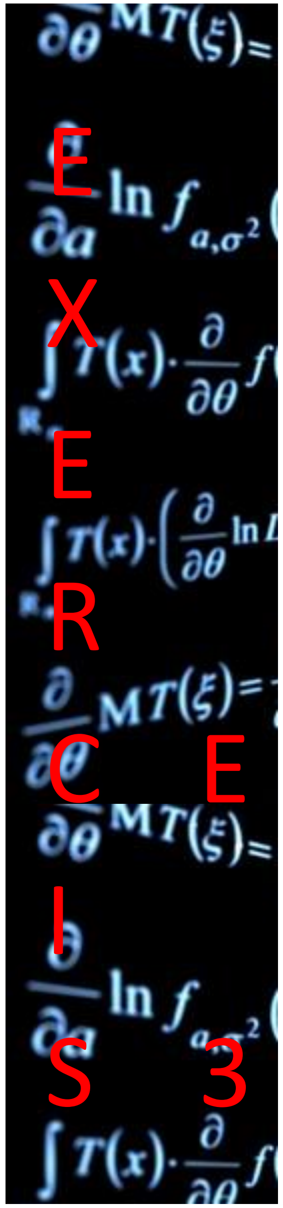




Exercise 3

In the program developed so far:

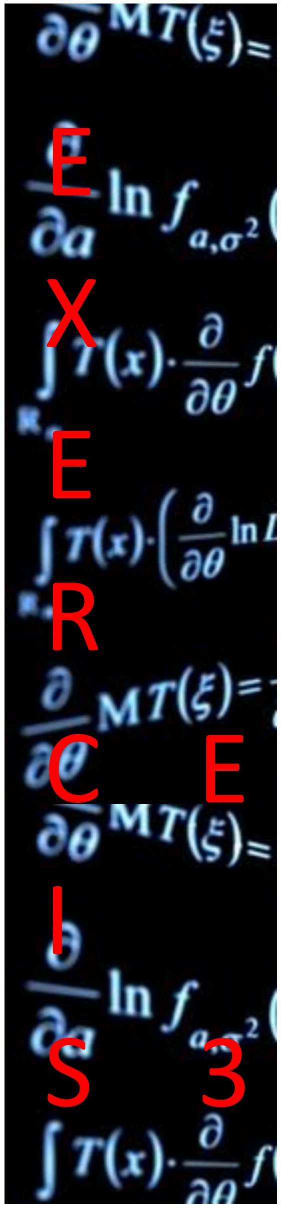
- The pump starts when the analog input 30001 value is 70.0% of maximum.
- The pump stops when the analog input 30001 value is 90.0% of maximum.
- The run time of the pump is measured and accumulated.
- The number of times the pump starts is saved.



Exercise 3 – Part Four

Add Lag Pump Control

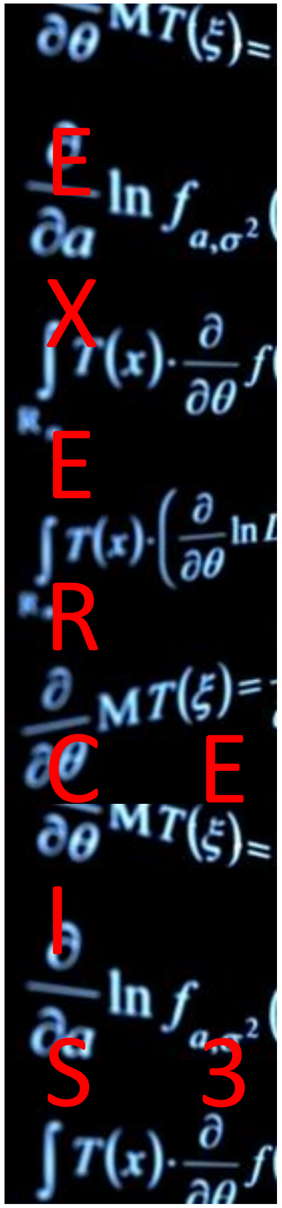
- **Stop Lead Pump:** When the value of register 30001 is equal to or greater than 90.0% of the maximum value
- **Stop Lag Pump:** When the value of register 30001 is equal to or greater than 80.0% of the maximum value
- **Start Lead Pump:** When the value of register 30001 is equal to or less than 70.0% of the maximum value
- **Start Lag Pump:** When the value of register 30001 is equal to or less than 60.0% of the maximum value



Exercise 3 – Part Four

Add Lag Pump Control

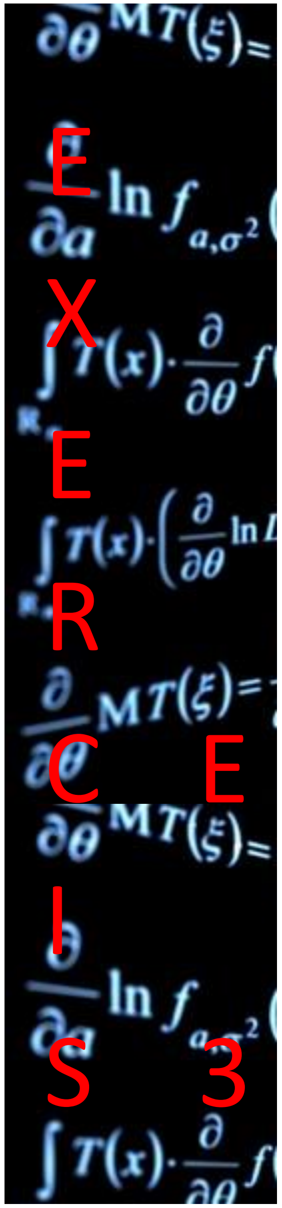
- Modifications to the program will include a Lag pump run timer and start counter.
- Use the Following steps to modify the program:
  1. Modify Network 1 to include the Lag pump start and stop setpoints.
  2. Add a network to control the Lag pump start and stop. Insert this network immediately after the Lead pump start and stop control network.
  3. Add a network to accumulate the Lag pump run time. Insert this network immediately after the Lead pump run timer network.
  4. Add logic to count the number of Lag pump starts into the timer network.
  5. Load the modified program into the controller.
  6. Use the Monitor function to test the program.



Exercise 3 – Part Five

Alternating Lead / Lag

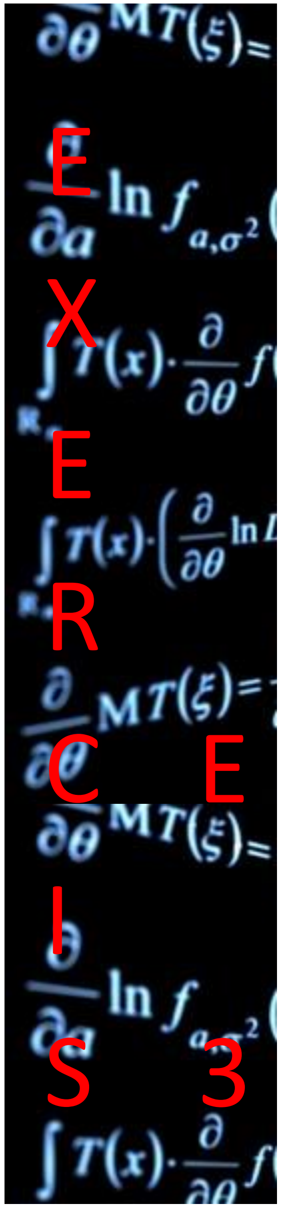
- In this last section of the TelePACE programming example, the program is modified to switch the lead and lag pumps every 50 hours.
- To begin this part of the programming example the Ladder Logic program TRAIN010.LAD has been created.
- In this program the lead and lag run timers have been changed to increment the accumulated minutes registers every 2 seconds.
- The lead and lag switching will occur every 10 seconds.



Exercise 3 – Part Five

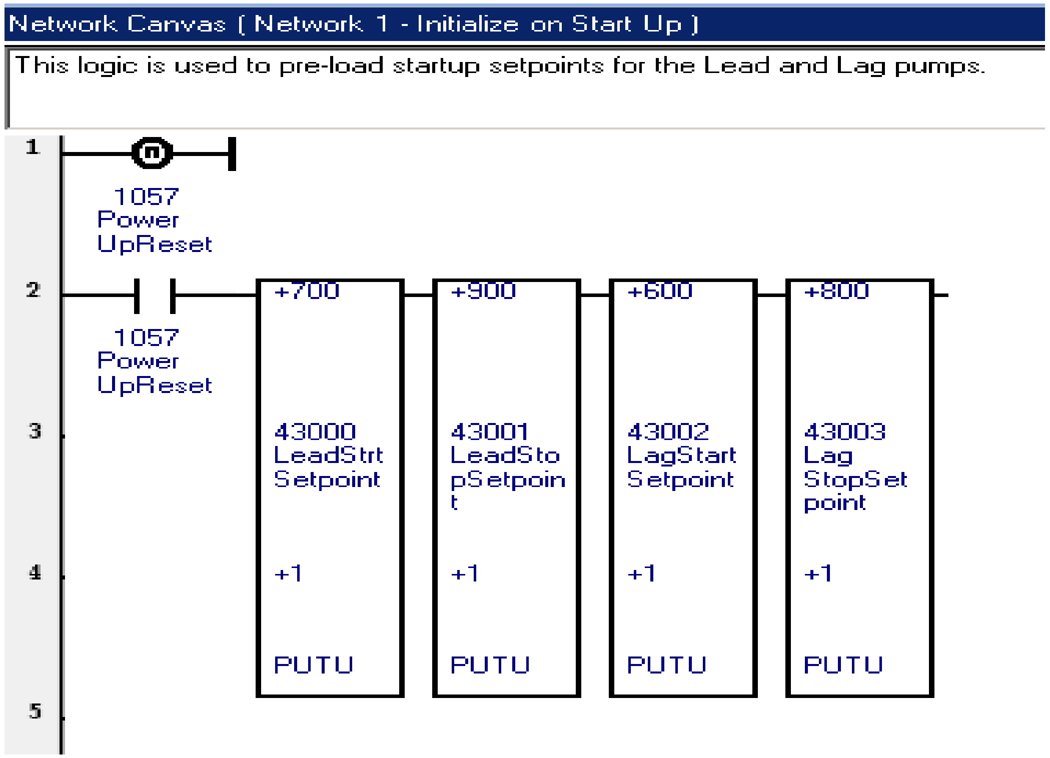
Alternating Lead / Lag

- Initialize the I/O Database in the controller.
- Load the program TRAIN010.LAD into the controller.
- Run the program and monitor the program execution.
- Adjust the analog input to 65% and monitor program execution.
- Adjust the analog input to 50% and monitor program execution.
- Adjust the analog input to 85% and monitor program execution.



Exercise 3 – Part Five

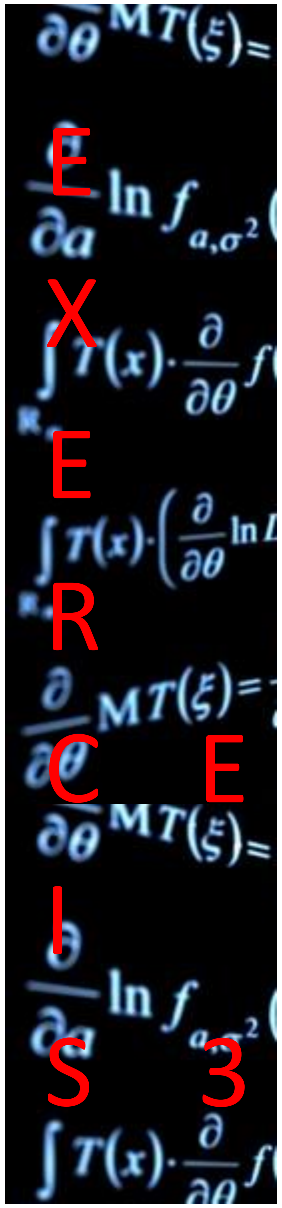
Alternating Lead / Lag



Network Canvas (Network 1 - Initialize on Start Up)

This logic is used to pre-load startup setpoints for the Lead and Lag pumps.

The instructor will discuss the program execution once the class has had the opportunity to examine the program



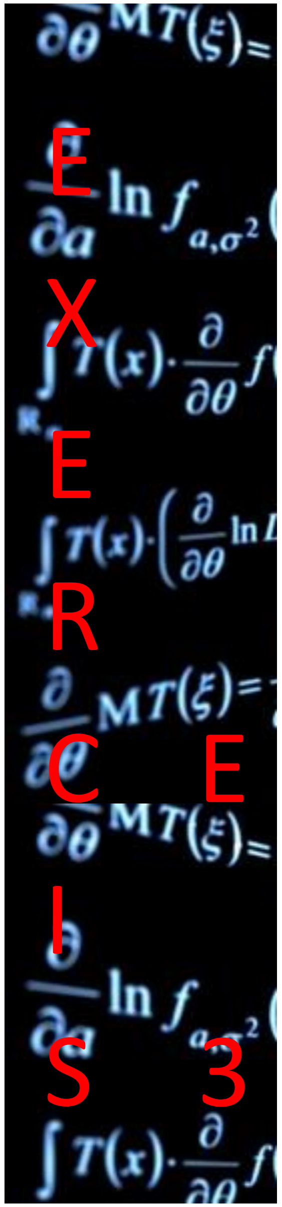
Exercise 3:
Review Questions

1. How do you view Tag Names while in the Default View?

2. What does the element PUTU do?

3. How does the Timer reset? (Exercise 3- Part 3)

4. What happens to the value in the accumulator when the program is stopped and then restarted?



Subroutines

Subroutines

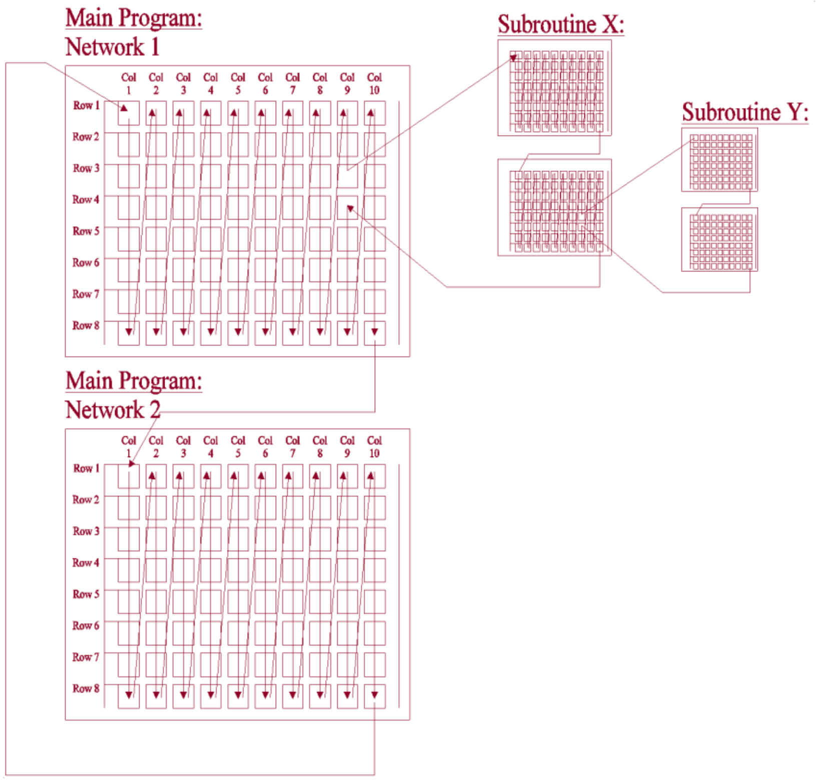
- A subroutine is a group of ladder logic networks that may be executed conditionally.
- Each subroutine begins with a network containing an **SUBR** element on its first rung.
- A subroutine ends when another subroutine element is encountered, or when the end of the ladder logic program is reached.
- In order to use a subroutine, a **CALL** element with the same subroutine number is enabled by the program.

Subroutines

- Ladder logic programs can grow to include many networks, causing the scan time to become longer than may be desired.
- The use of subroutines allows the programmer to decide when any part of the logic will be executed.
- One subroutine might need to be executed once per second, while another only once per hour.
- This can dramatically reduce the controller’s scan time

Subroutines

- Subroutines do not have to be programmed in any particular numerical order.
  - For example, subroutine 1 can follow subroutine 2, subroutine 200 can follow subroutine 400.
- A subroutine can call other subroutines. This is called nesting.
  - The maximum level of nesting is 20 calls. In practice, more than two or three levels of nesting can become quite confusing.



Exercise Four

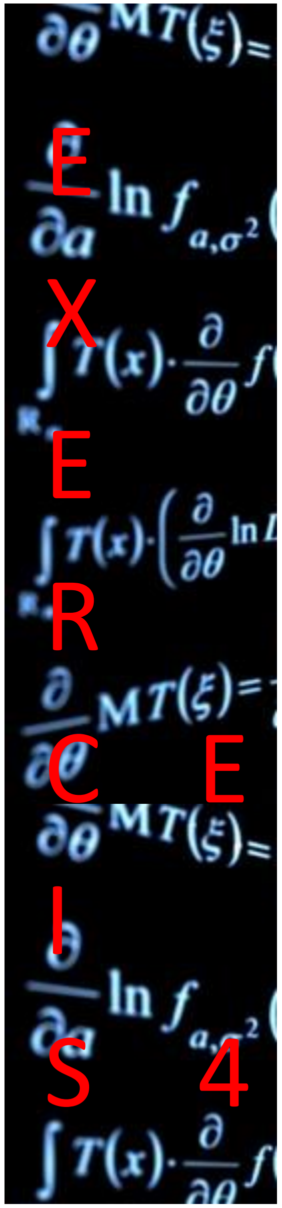
Exercise 4

- **Subroutines**

Exercise 4

Create Tag Names

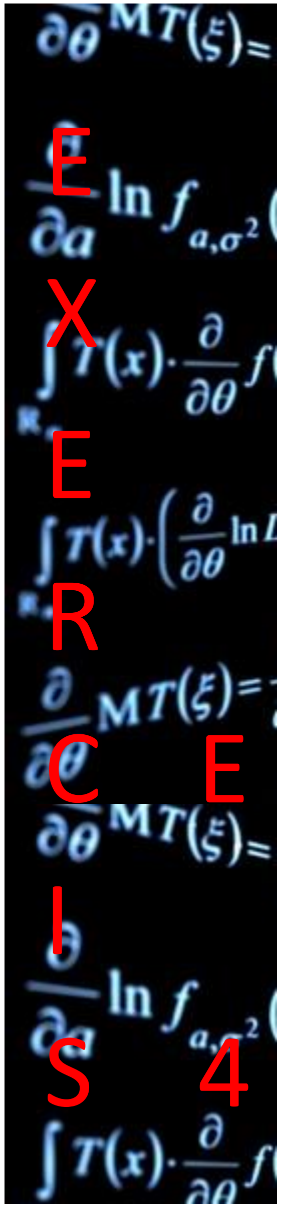
- Open a new file in the TelePACE Ladder Network Editor
- Enter the Tag names from the next slide



Exercise 4

Create Tag Names

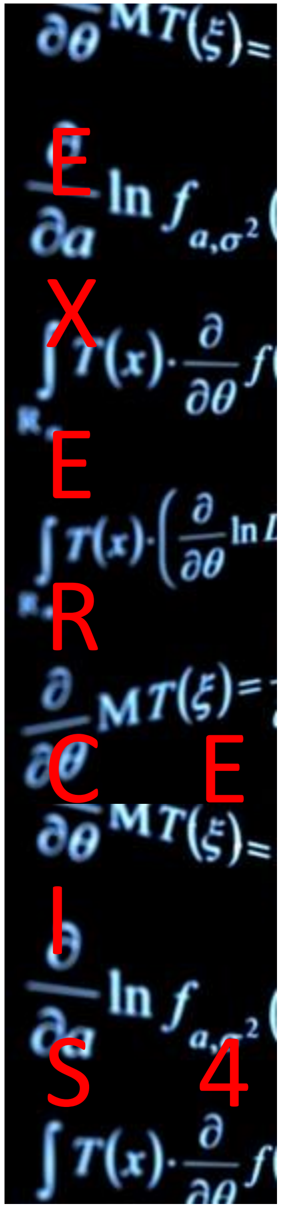
Tag Name	Register Address
Motor 1 Start	10001
Enable Sub 100	10002
Enable Sub 200	10003
Alarm Trigger	10004
Motor 1 Run	00005
Pulse Output	00006
Alarm Output	00007



Exercise 4

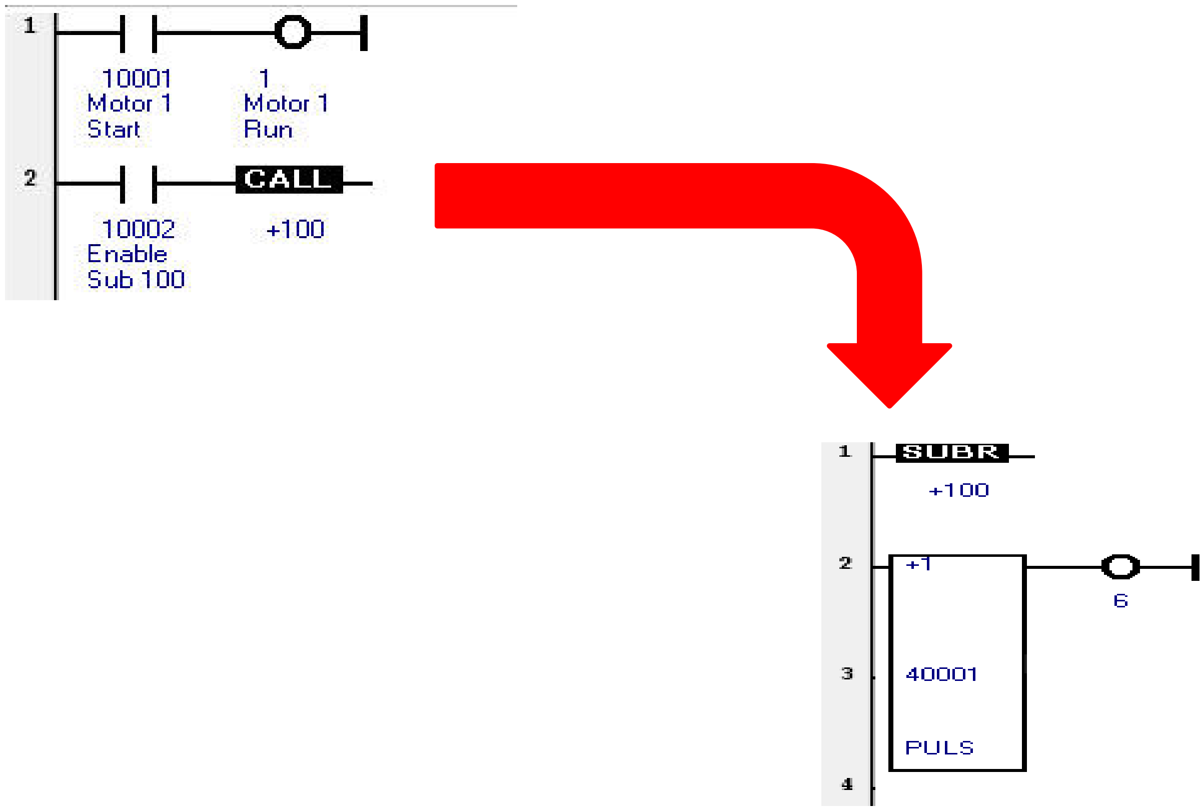
Create Main Program and Subroutine

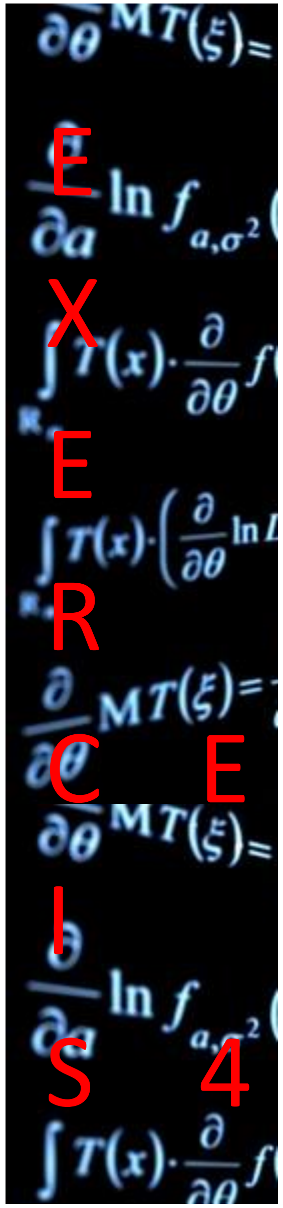
- Select the appropriate Controller Type
- Create a Default Register Assignment with the appropriate I/O board.
- Insert the single rung of main program logic shown, along with the CALL in Rung 2.
  - Note that a warning is generated when the CALL is inserted before the matching SUBR.
  - In an offline edit this is acceptable. In an online edit, however, it is not allowed to insert the CALL first as a jump to a non-existent subroutine could cause the controller to lock up.



Exercise 4

Create Main Program and Subroutine

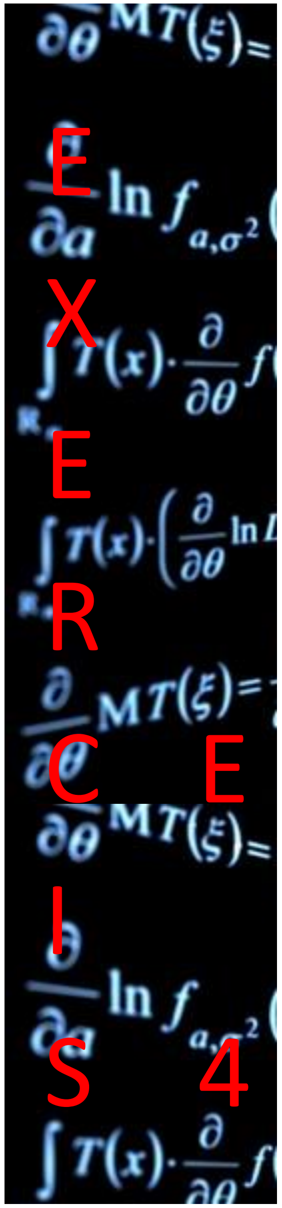




Exercise 4

Create Main Program and Subroutine

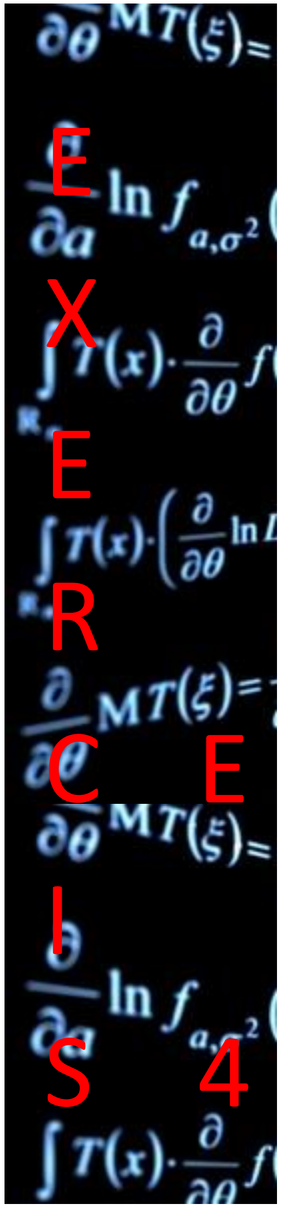
- Note that the main program logic will execute at all times. (i.e. switch 1 should always be able to control the first LED)
- Note that the second LED will only flash when Subroutine 100 is enabled with switch 2.
- See that the power rail and SUBR label only turn red (active) when the subroutine is enabled.
  - Also note that if the subroutine is disabled (switch 2 turned off) while the second LED is on, that LED will remain on until the subroutine is again enabled and the PULS is activated.



Exercise 4

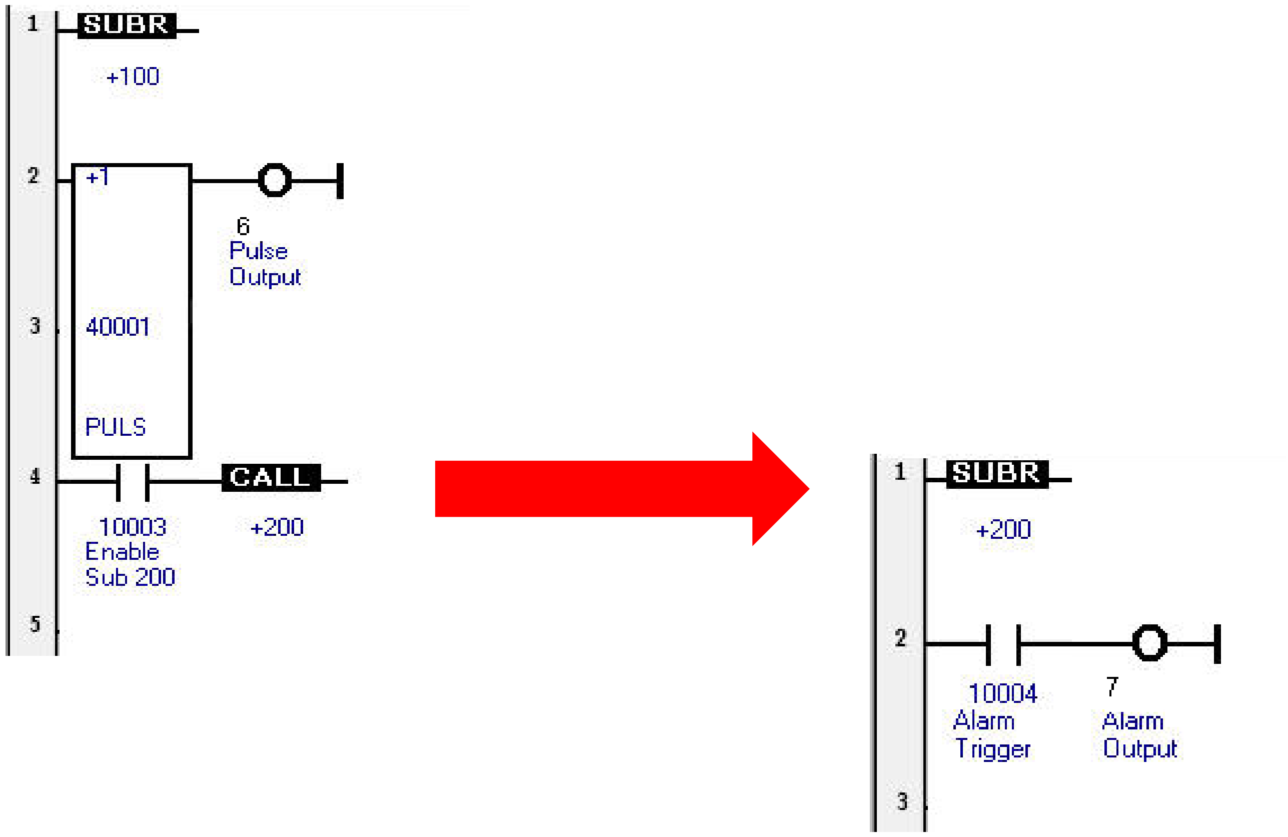
Nested Subroutine

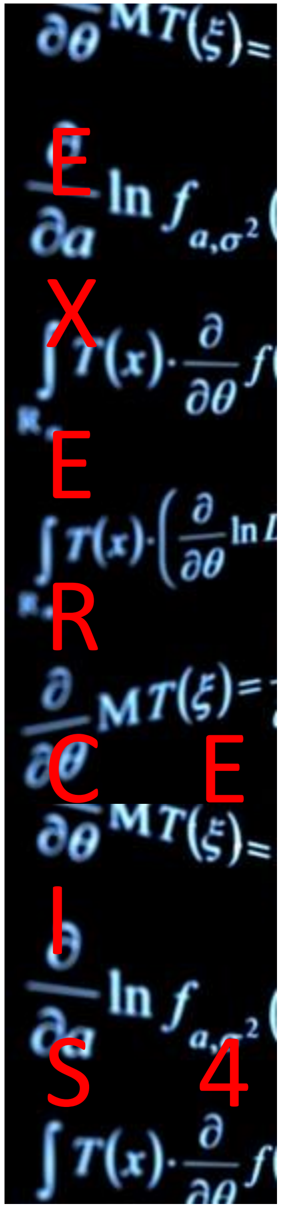
- Create a nested subroutine that will:
- With Subroutine 1 disabled (switch 2 turned off), turning on the third switch does not cause Subroutine 2 to be executed.
- Turning on switch 3, should enable Subroutine 2.
- The power rail and SUBR label in the subroutine become red after turning on the switch. (active)
- Switch 4 should be able to turn on the third LED, enabling both subroutines.



Exercise 4

Nested Subroutine

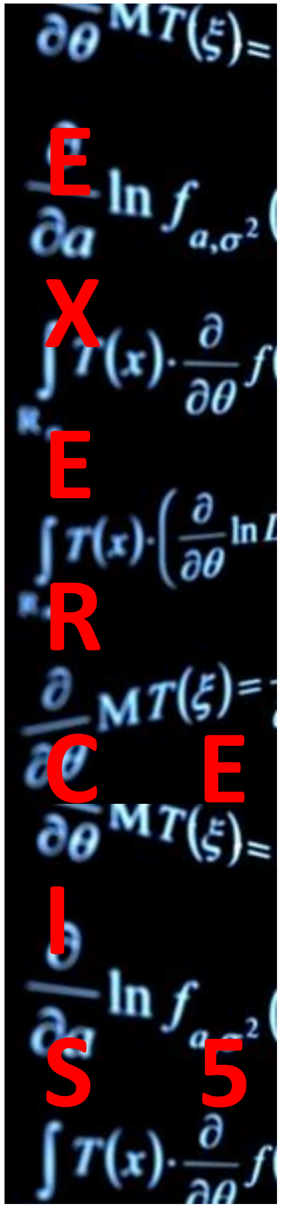




Exercise Five

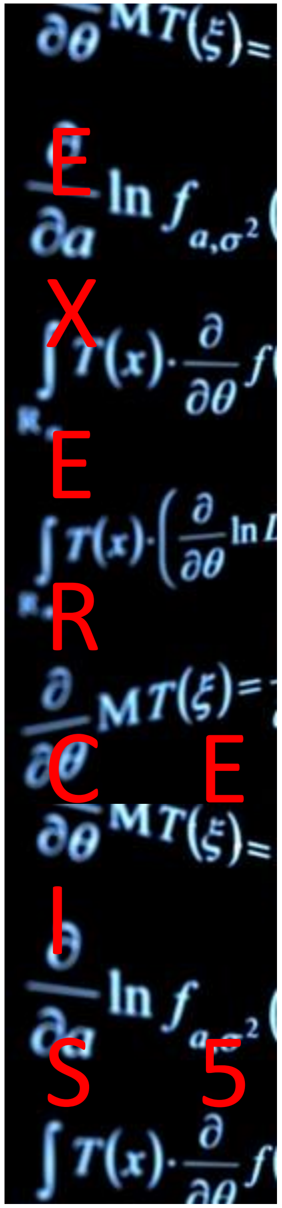
Exercise 5

- **Flow Accumulation**



Exercise 5

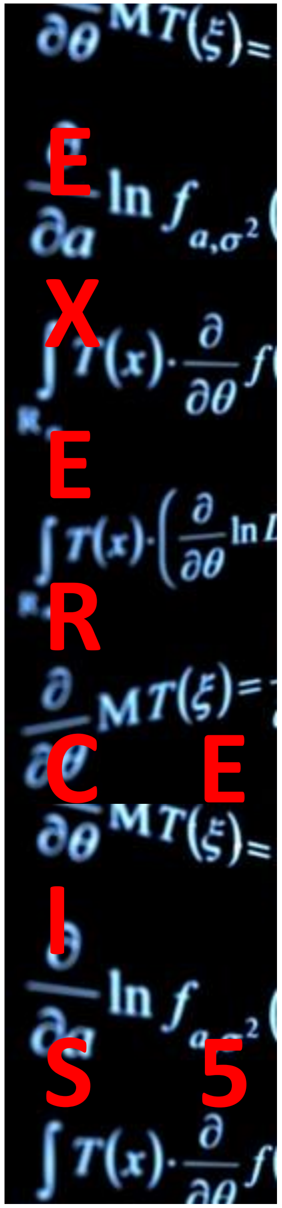
- The **FLOW** function accumulates flow from pulse-type devices such as turbine flow meters.
  - This function is designed to be used in the measurement of products that do not require a complex flow calculation.
  - If any correction factors must be applied, such as for temperature or pressure, that must be done in separate logic in the controller.
- In the following exercise, the turbine meter will be simulated by a pulse generator on a 5699 demonstrator I/O board.



Exercise 5

- **Create Tag Names**

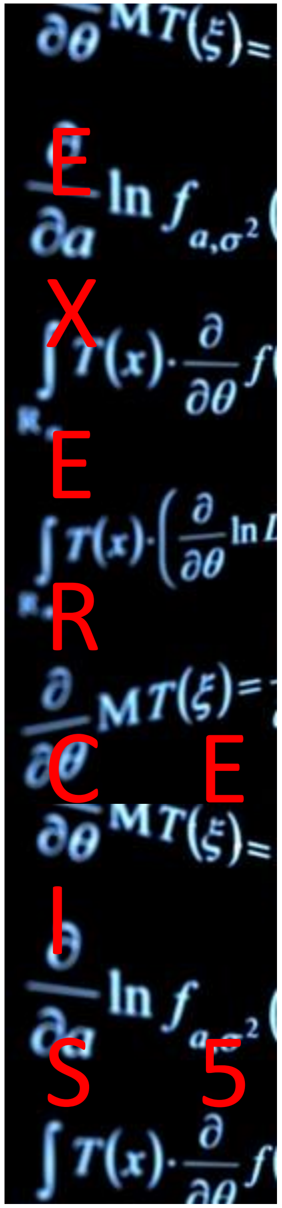
Error	01001
Accum Flow	10001
Log Data	10002
Enable Accum	10003
Counter Input	30011
K Factor	41001
Input Register	41003
Input Type	41004
Rate Period	41005
Status	41006
Flow Rate	41007
Number Records	41017
Volume Period 1	41018
End Time Period1	41020
FlowTime Period1	41022



Exercise 5

Build Ladder Logic

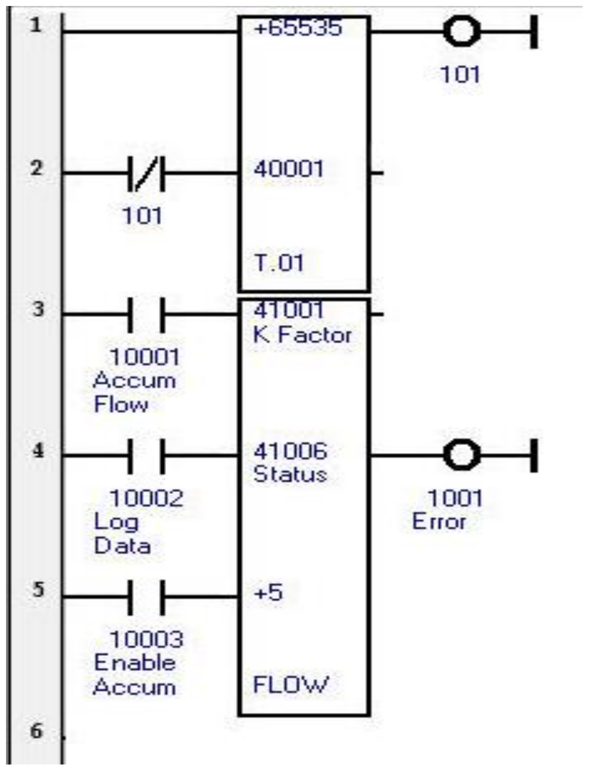
- Create a Default Register Assignment with the appropriate lower I/O board.
- Enter the ladder logic
- Ensure that switch 3 is turned on, to Enable the Flow function, and that switches 1 and 2 are both off before running the program.
- Download the program to the controller, monitor execution.

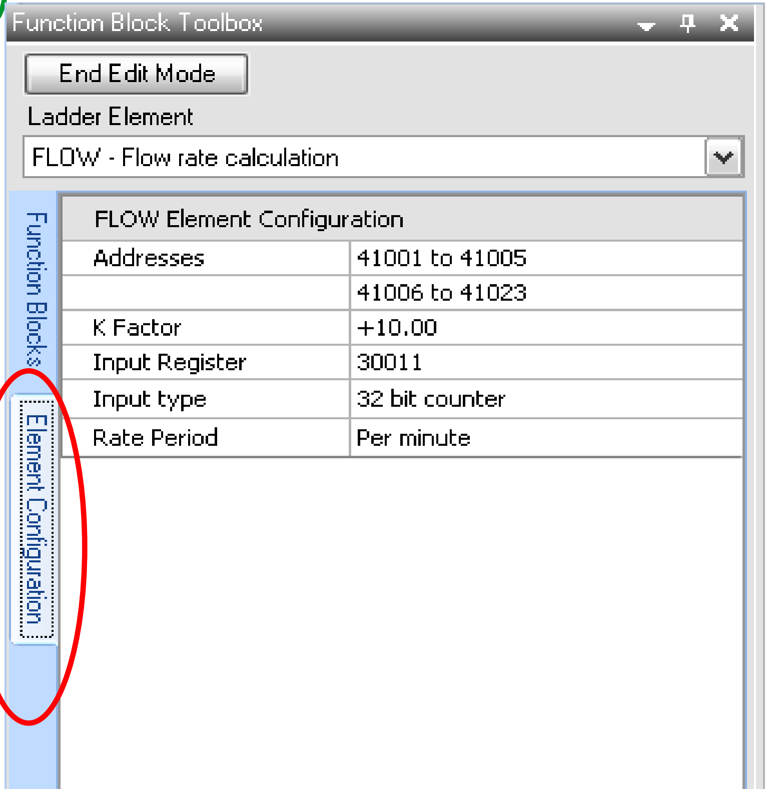


Exercise 5

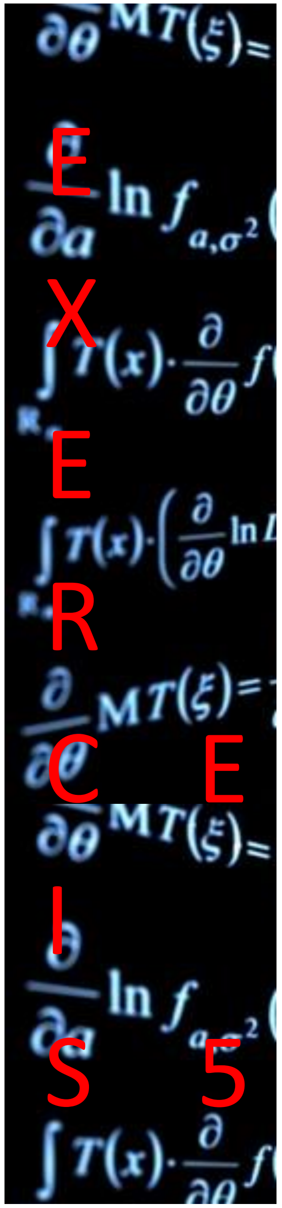
Build Ladder Logic

- Complete the FLOW function Element Configuration as shown (Input register 30009)





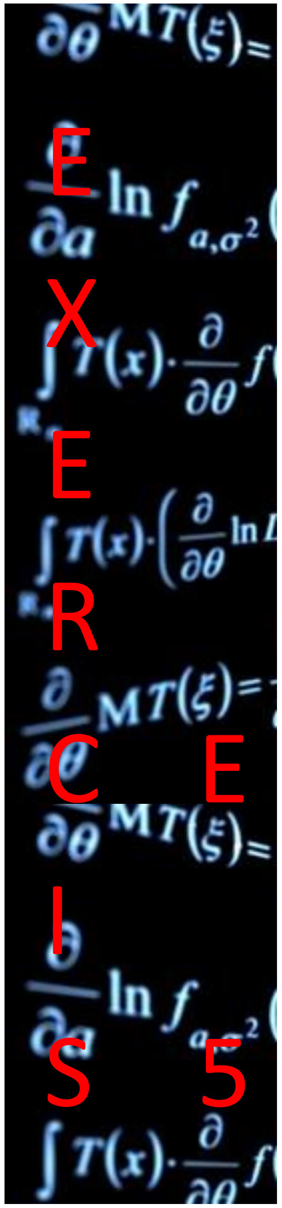
FLOW Element Configuration	
Addresses	41001 to 41005
	41006 to 41023
K Factor	+10.00
Input Register	30011
Input type	32 bit counter
Rate Period	Per minute



Exercise 5

Monitor Program Operation

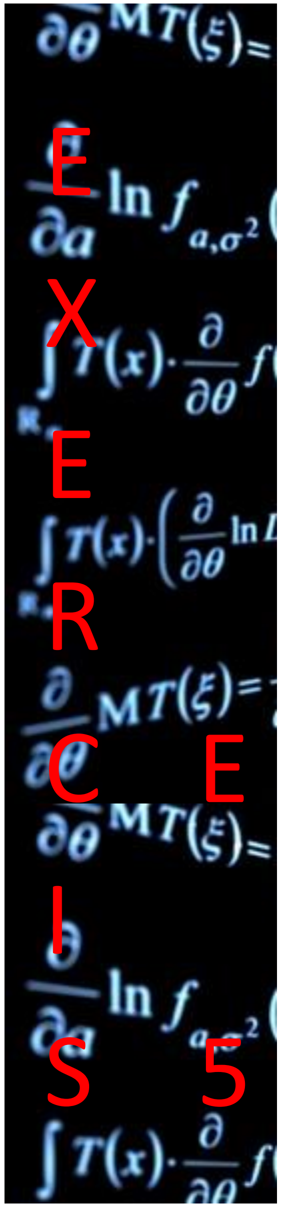
- Once online with the controller, create a new Group in the Register Editor.
- Add register 30009, the pulse accumulator, to the group using the Unsigned Double format.
- Select the FLOW function, then right click on it. Select Monitor Element. This will add all of the configuration registers and the output registers to the Register Editor group, with each register in its correct data format.
- Registers 41009, 41011, 41013 and 41015 are not required for this demonstration. Select them and hit the Delete button to remove them



Exercise 5

Monitor Program Operation

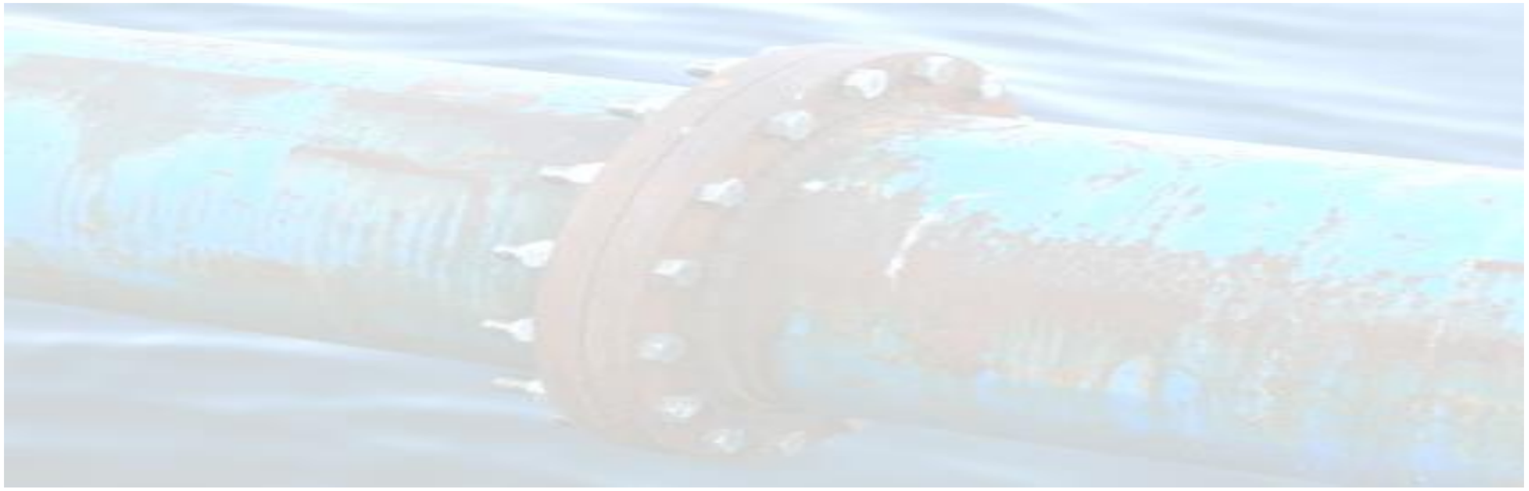
- Check register 30009. Its value should be incrementing at a rate set by the Counter Freq potentiometer on the 5699 I/O board. Adjust this pot and verify that the pulse accumulator rate also varies. It should vary between approximately 5 and 150 pulses per second (Hz).
- Note that registers 41001 through 41005 are the configuration registers, as entered in the FLOW function's Element Configuration dialog.
- Note that the Flow Rate, in register 41007, should be 0 at this point. The Accumulate Flow switch is still off, so the FLOW function is not operational.

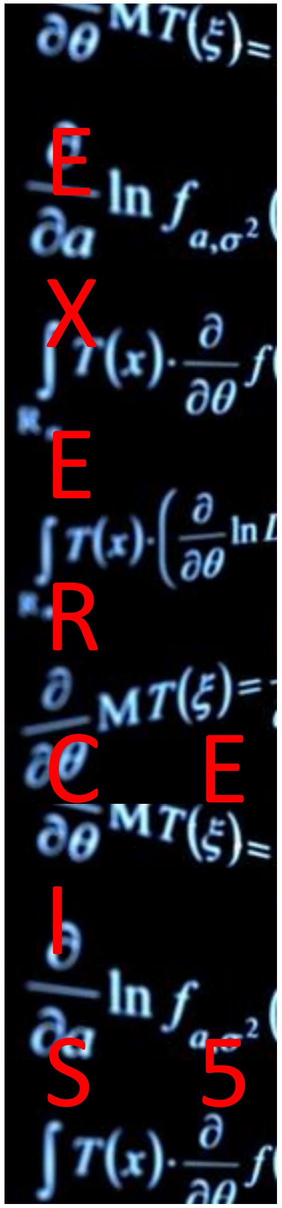


Exercise 5

Monitor Program Operation

- Set the Counter Freq potentiometer fully clockwise. This will set the input pulse rate to approximately 150 Hz.
- Turn on switch 1. This causes the FLOW function to begin accumulating flow.

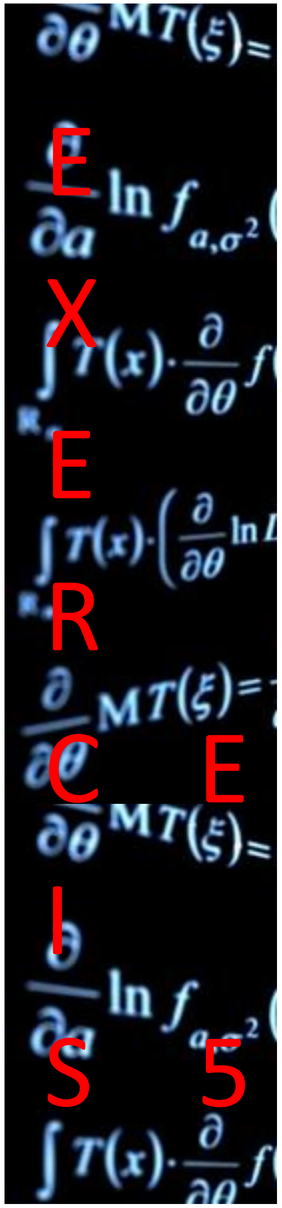




Exercise 5

Monitor Program Operation

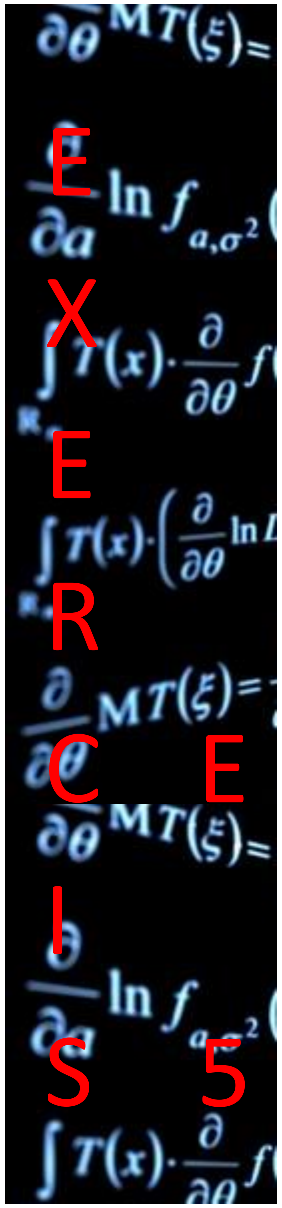
- Monitor register 41007, the Flow Rate.
  - Note that the flow should be approximately 900, if the pulse rate is 150 Hz and the K Factor is 10. The Rate Period is set to Per Minute. Thus the flow rate is calculated as: (150Hz / 10) * 60 seconds = 900. This may mean 900 gpm, 900 m3/min, or perhaps 900 barrels/min.
- Set the Counter Freq potentiometer fully counter-clockwise. This will be about 5 Hz.
- If the pulse rate is too slow, the FLOW function will generate Status code 5, “Pulse rate is too low for accurate flow rate calculation.” The Flow Rate should read approximately 30.
  - Note that at this low pulse rate the flow calculation will only update once every 10 seconds.



Exercise 5

Monitor Program Operation

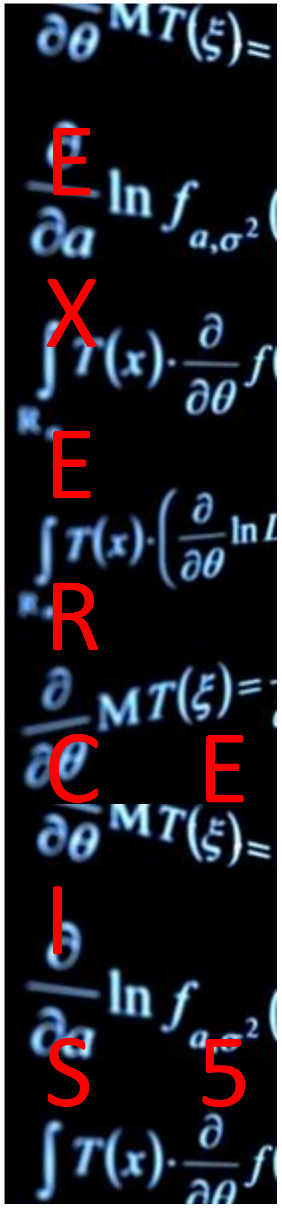
- Increase the flow rate until the Status register returns to 0. (No error)
  - Note that register 41018, (Volume Period1) is incrementing. This is the current period, and so the value is live. It only stops if the flow stops or the Accumulate Flow input is turned off.
  - Note that the End Time Period1 register is showing the current time in Unix time format.
  - Note that the FlowTime Period1 register counts the number of seconds of flow in the period.
- Simulate the end of a day by toggling the Log Data input on and off with switch 2.
  - Note that the three Period 1 values are frozen at this point, then pushed down to period 2.



Exercise 5

Monitor Program Operation

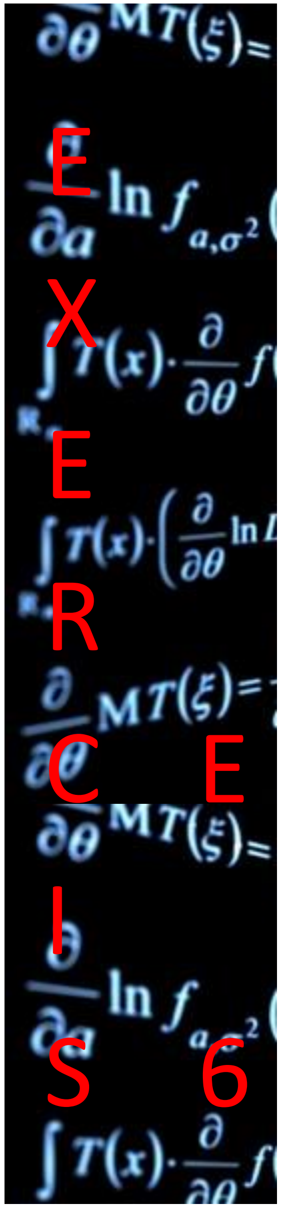
- New Period 1 values begin to accumulate as the new day begins.
- Once the Log Data input has toggled 5 times the original day's data will be pushed out the bottom of the history. This data is now lost.
- Experiment with new values for the K Factor and the Rate Period.
  - Note that the K Factor should not be changed while flow is being accumulated. First turn the Accumulate Flow input off to avoid flow calculation errors.



Exercise Six

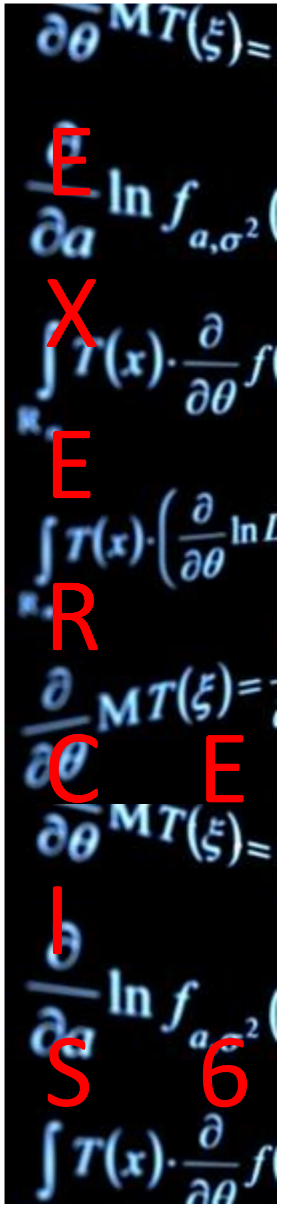
Exercise 6

- **PID Feedback Loop Control**



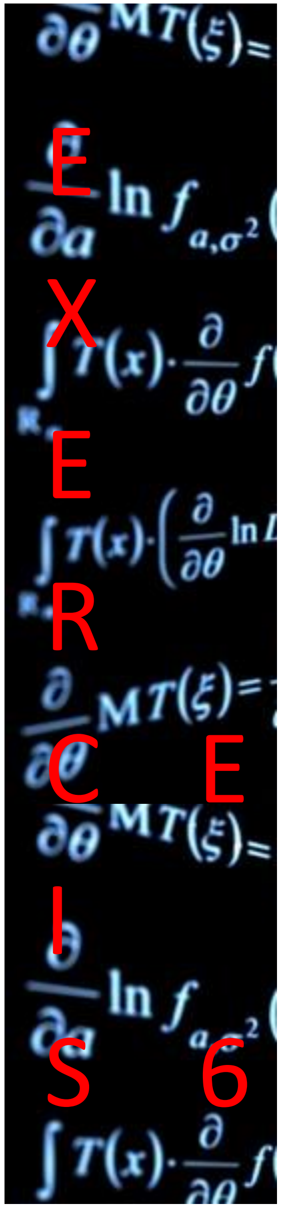
Exercise 6

- A PID controller compares a desired value (Setpoint) against a real-world measured value (Process Value).
- Any error detected will cause a corrective output from the controller to occur, in an attempt to reduce the error. This is called a feedback loop.
- The PID controller uses three tuning parameters called Proportional, Integral and Derivative to adjust how the controller reacts. These are also known as Gain, Reset Time and Rate.



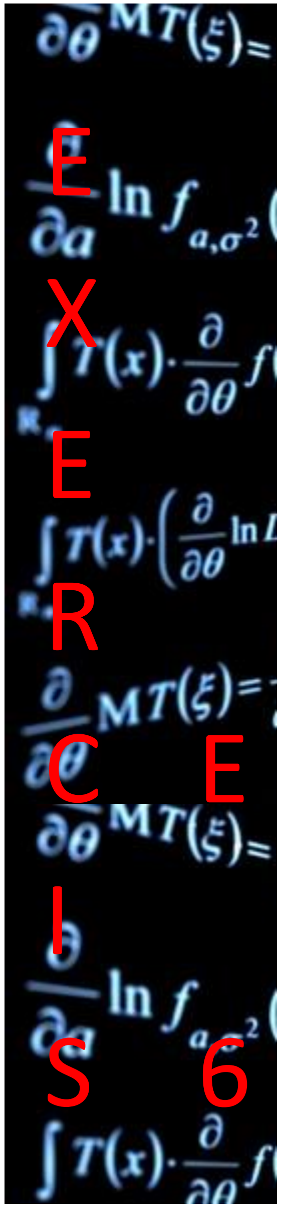
Exercise 6

- The PID controller to be used in the following exercise is the PIDA function.
- This controller has an analog-type floating point output, which can be sent to one of a SCADAPack’s analog out channels. This will send a 4-20 mA or 1-5 V signal to a control device such as a flow control valve.



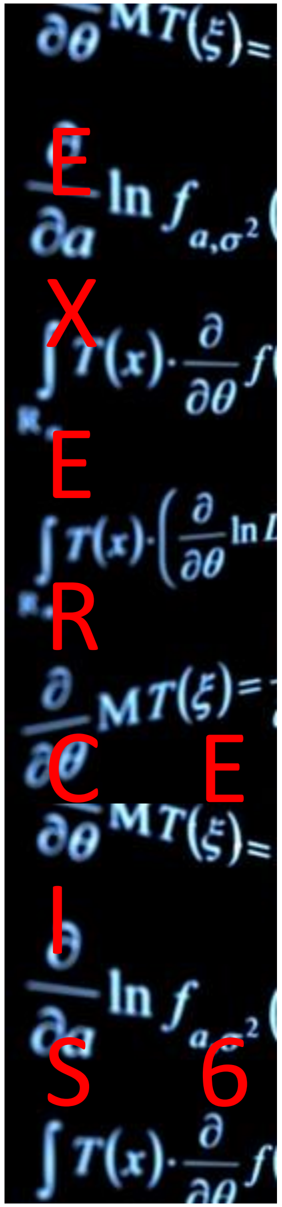
Exercise 6

- A PID controller typically also has a manual mode, in which the operator can take control of the output and set it to any desired value. In the PIDA function a digital input is available to allow switching between Auto and Manual modes, and a Manual mode value may be entered from an HMI.
- In the following demonstration, no actual feedback is possible. It will be up to the student to simulate feedback by adjusting the analog input potentiometer on their demonstrator I/O board.



- **Tuning of a PID loop** can be a complex and involved process. However, included here is a brief discussion of some of the key parameters.
- **Proportional Gain** – Increasing Gain tends to reduce the Error. If Gain is increased too far, however, the loop will begin to cycle or oscillate. Gain alone can never reduce the Error to zero.
- **Integral or Reset Time** – Reset is added to eliminate the Error. As Reset time is decreased the controller output becomes more likely to oscillate. The oscillation period also tends to increase. In the PIDA function it is measured in Seconds per Repeat.
- **Derivative or Rate Time** – This parameter is typically not used in applications such as flow control. If not used, its value must be set to 0 (zero) to disable it. Rate allows a degree of predictive action, in order to reduce dead time. This is the time between a PID output change and a resulting change in the process.
- **Introductory Tuning** – The tuning of a PID loop can be very complex, particularly if Rate is involved. Below is a simple introductory technique to set the tuning. This brief discussion will not include Rate.
  - Initially set the Reset value to a very high setting, for example 1000 seconds.
  - While the PIDA is controlling the process in Auto mode, increase the Gain slowly until the PID Output begins to oscillate.
  - Measure the oscillation time. This is the Natural Period of the loop.
  - Divide the Gain used at this point by 3, and use this value for Gain.
  - Use the Natural Period of oscillation as the Reset Time.

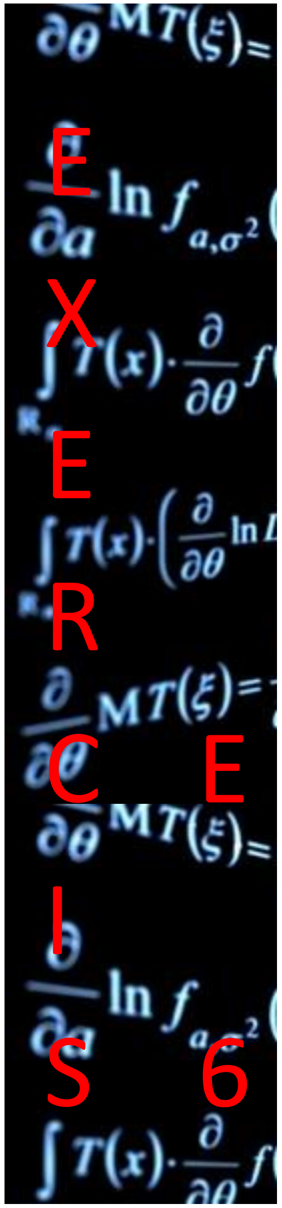
- ***CAUTION: The procedure above is a very basic guide for training purposes ONLY. Anyone performing a PID tuning procedure under actual field conditions must understand the process and follow all appropriate safety precautions*.**



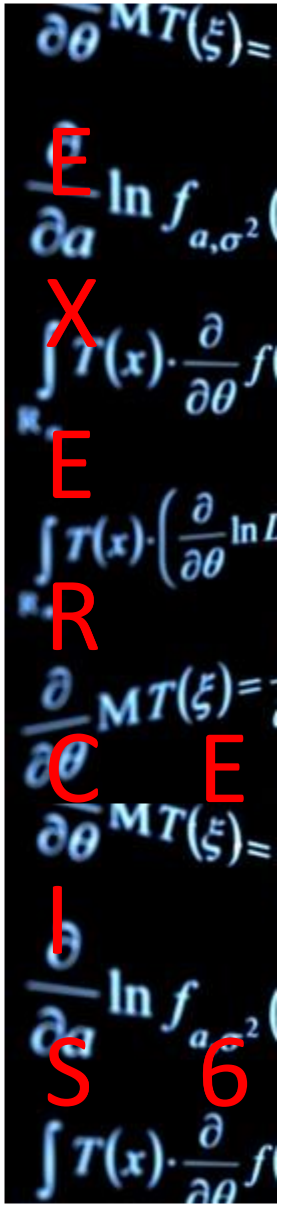
Exercise 6

- **Create Tag Names**

Tag Name	Register Address
Auto/Man Mode	10001
FlowRate Raw	30001
PID Out Raw	40001
SCAL Config	41001
SCAL Output	41010
Raw Out Range	41101
Raw Out Float	41103
Process Value	42001
Setpoint	42003
Gain	42005
Reset Seconds	42007
Rate Seconds	42009
Deadband	42011
Full	42013
Zero	42015
Cycle Time	42017
Manual Output	42019
PIDA Output	42021



Exercise 6



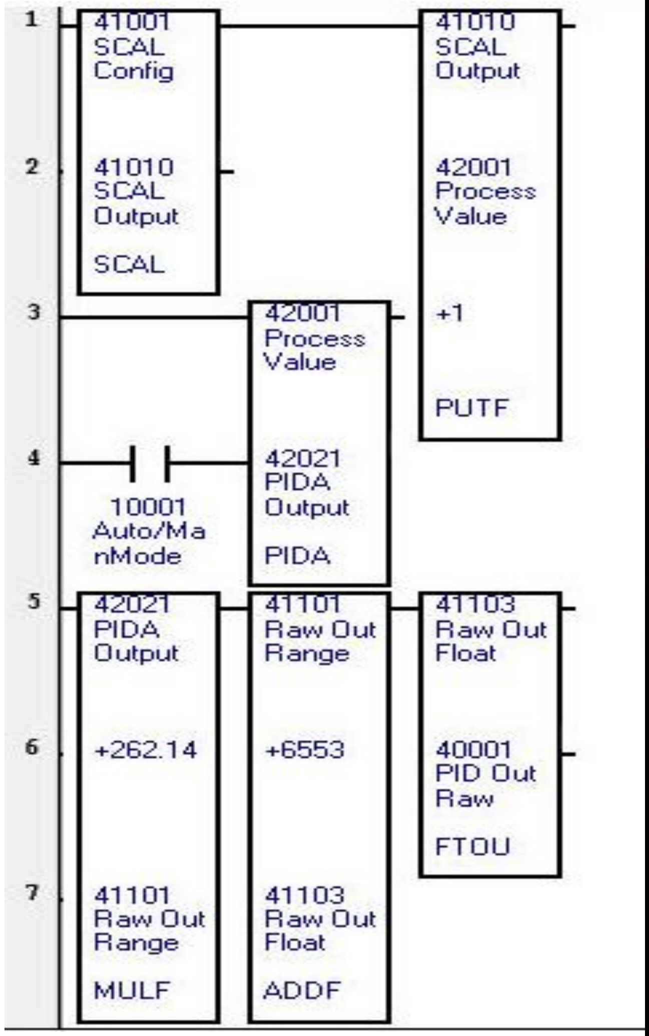
Build Ladder Logic

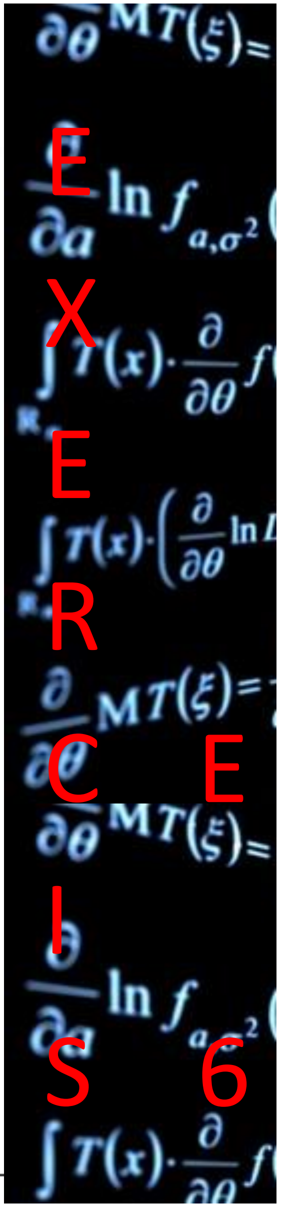
- This program will take an analog input from the demo I/O board and convert it to engineering units.
- The input will represent flow through a pipeline, with a minimum scaled value of 0 at 6552 and a maximum of 300 at 32760. The PIDA function will calculate a floating point output, with a range of 0 – 100%.
- This will be used to drive a control valve.
- The floating point number will then be converted back to a raw value and sent to one of the controller's analog outputs.

Exercise 6

Build Ladder Logic

- Select the appropriate Controller Type in the Controller menu.
- Create a Default Register Assignment with the appropriate lower I/O board.
- Click Add, then insert the SCADAPack AOUT module. Assign a Start register of 40001.
- Enter the ladder logic as shown

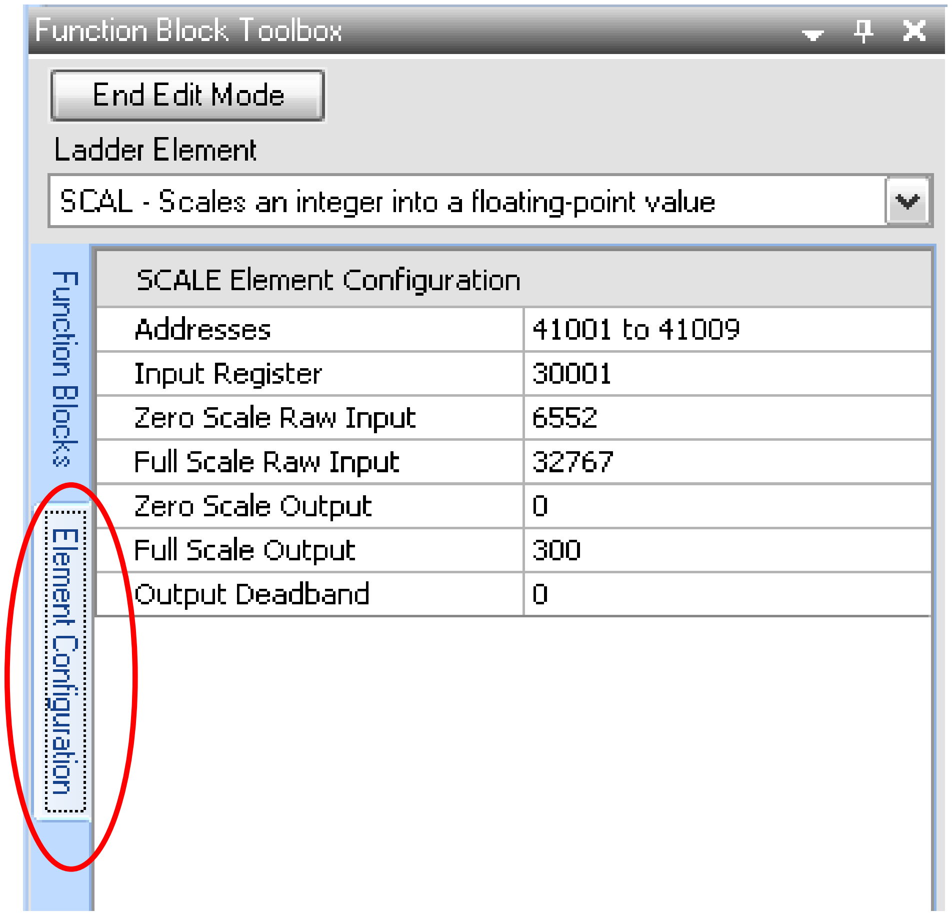




Exercise 6

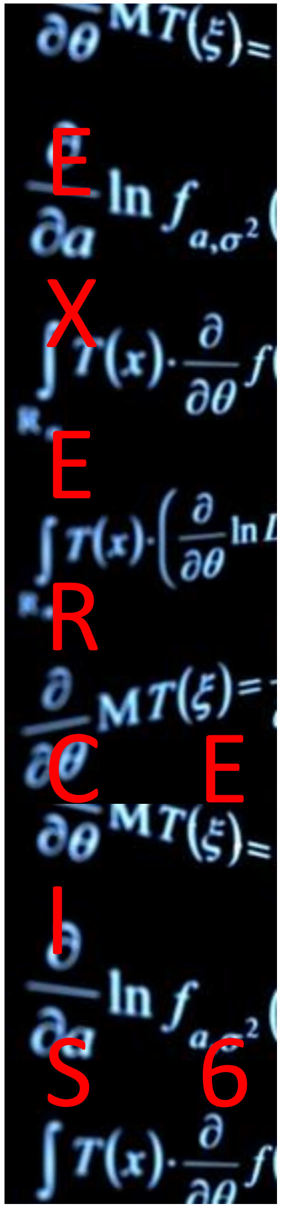
Build Ladder Logic

- Complete the SCAL function Element Configuration as shown



SCALE Element Configuration

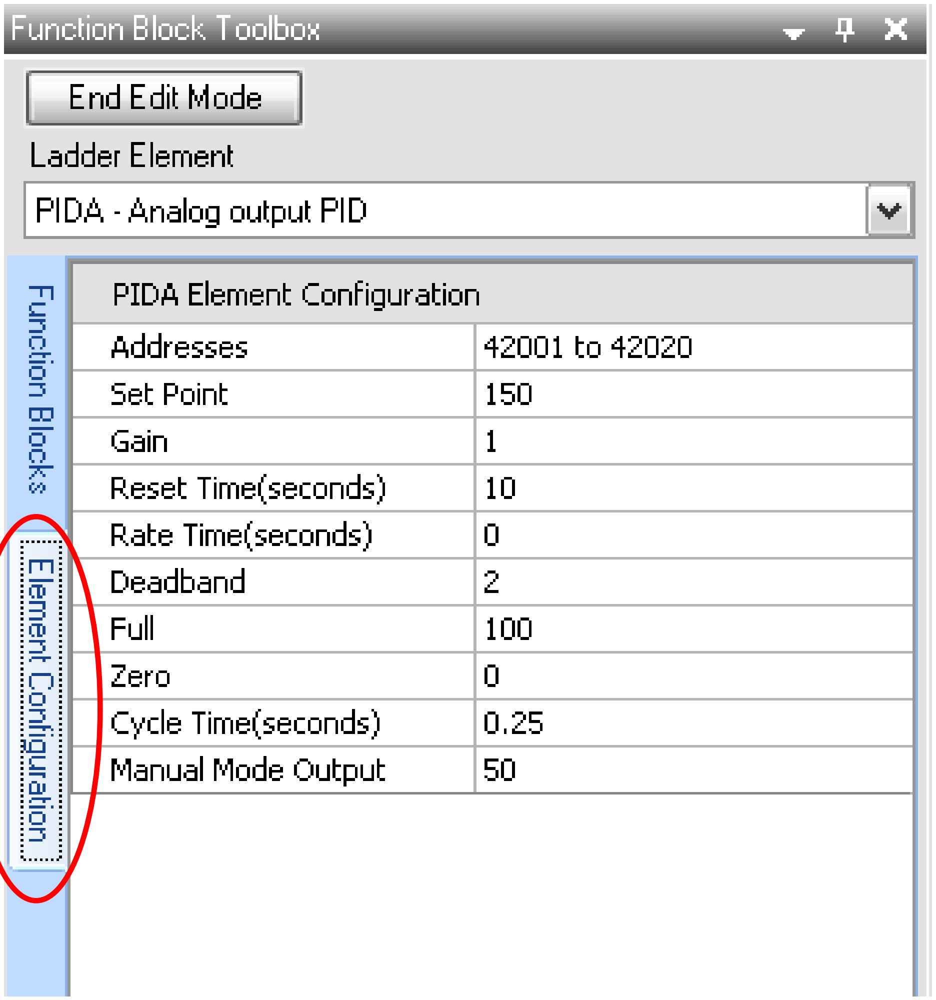
Addresses	41001 to 41009
Input Register	30001
Zero Scale Raw Input	6552
Full Scale Raw Input	32767
Zero Scale Output	0
Full Scale Output	300
Output Deadband	0



Exercise 6

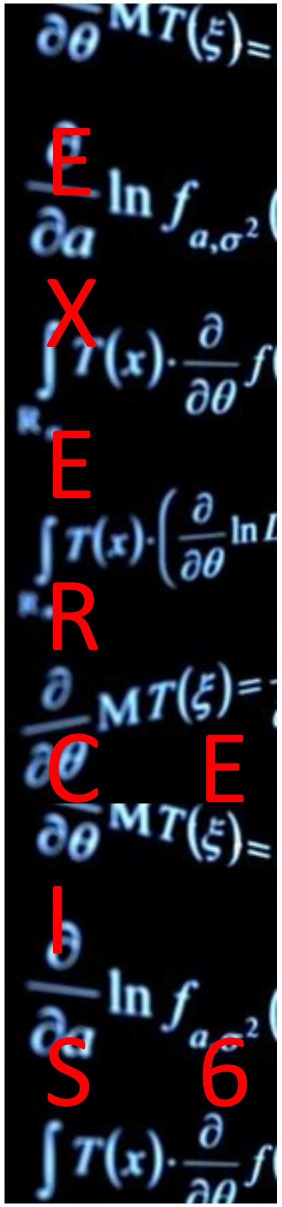
Build Ladder Logic

- Complete the PIDA function Element Configuration as shown



Ladder Element: PIDA - Analog output PID

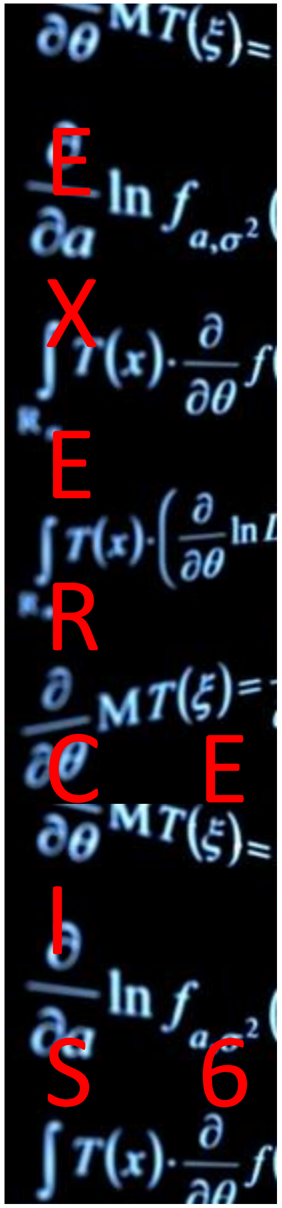
PIDA Element Configuration	
Addresses	42001 to 42020
Set Point	150
Gain	1
Reset Time(seconds)	10
Rate Time(seconds)	0
Deadband	2
Full	100
Zero	0
Cycle Time(seconds)	0.25
Manual Mode Output	50



Exercise 6

Monitor Program Operation

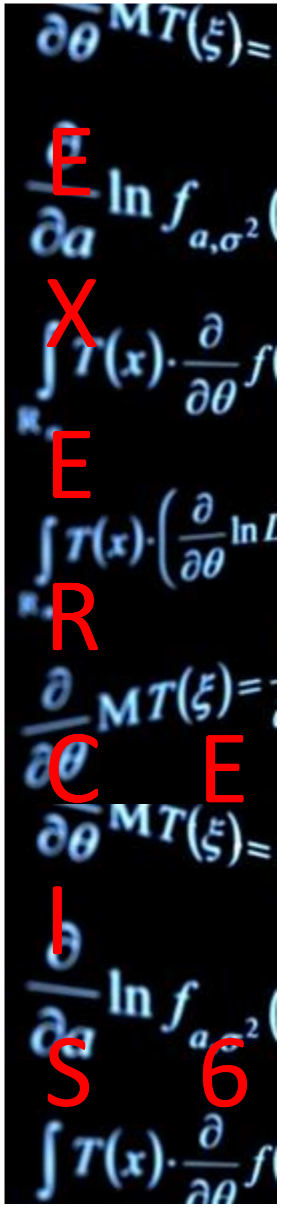
- Once online with the controller, create a new Group in the Register Editor.
- Press Add to manually add register 30001, the raw analog input representing flow rate, and 40001, the raw analog output to the control valve.
- Select the PIDA function, then right click on it. Select Monitor Element. This will add all of the configuration registers and the output registers to the Register Editor group, with each register in its correct data format.



Exercise 6

Monitor Program Operation

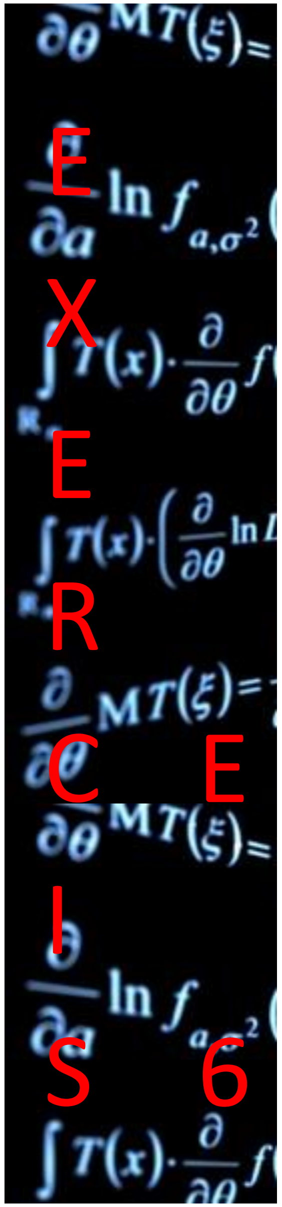
- Registers 42023 through 42029 are not required for this demonstration. Select them and hit the Delete button to remove them from the Register Editor group.
- Monitor register 30001, the raw flow rate, and 42001, the scaled Process Input values at the same time as the analog input potentiometer is adjusted. Note that at a raw value of 6552 the scaled value should be 0, and at a raw value of 32760 the scaled value should be 300. Also note that if the raw value is reduced to zero, the scaled value is -75. Why is this?



Exercise 6

Monitor Program Operation

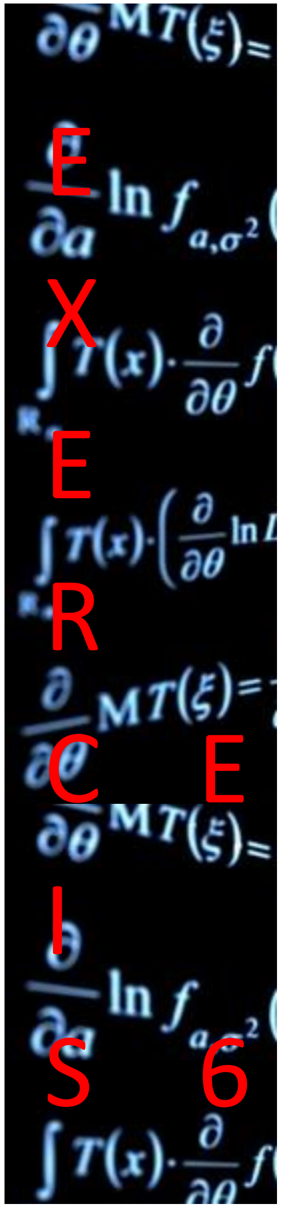
- With the Auto/Man Mode switch in Manual mode, the PID Output will immediately follow the value entered into the Manual Mode Output register. If it is desired to have the value sent to the valve change more slowly, additional logic will be required to ramp the value up or down.
  - Note that when a Manual Mode Output value of 0 is entered the PID Out Raw value, register 40001, goes to 6552. When 100 is entered the raw output value goes to 32767.



Exercise 6

Monitor Program Operation

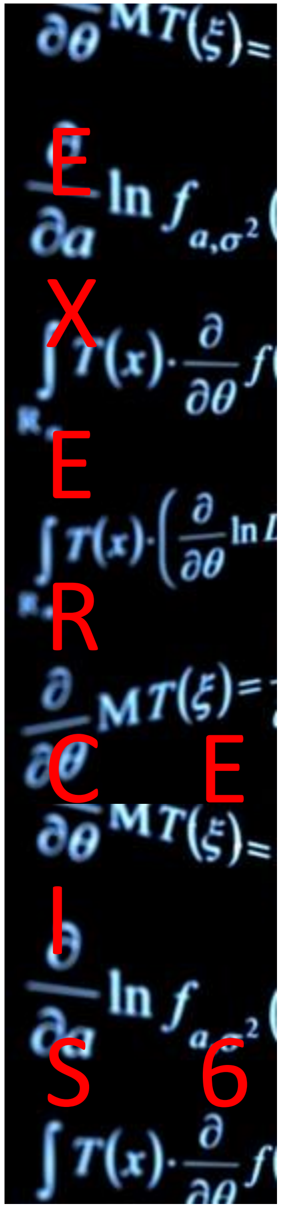
- Enter a Manual Mode Output value of 110. Note that the PID Output goes to 100. Try a value of -10. The PID Output will go to 0. It is clamped at these points by the Full and Zero settings.
- Adjust the analog input pot so that the difference between the Process Input (flow rate) and the Setpoint (the Error) is less than the Deadband.
- Now switch the PID controller into Automatic mode by turning on switch 1.
  - Note that the PID Output does not change. This is because the error is less than the Deadband, thus no new calculation is done.



Exercise 6

Monitor Program Operation

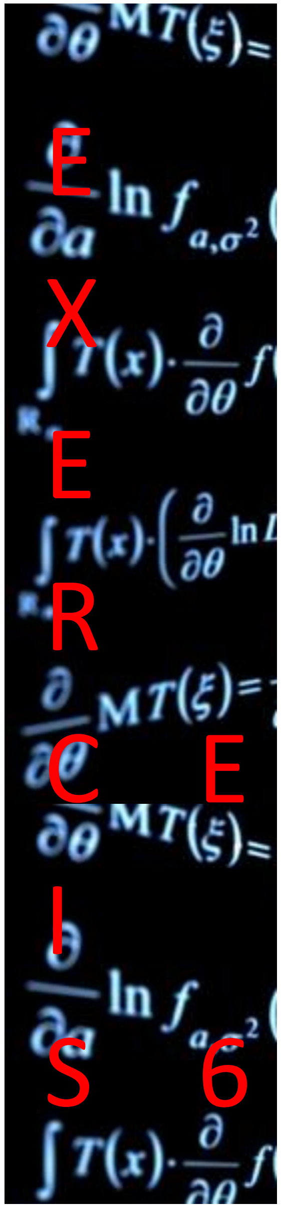
- Turn the potentiometer counter-clockwise to reduce the Process Value to a value slightly outside the Deadband range.
- Once the Error exceeds the Deadband, note that the PID Output begins to slowly rise.
- If the Process Value is flow rate, and the flow drops below the Setpoint, then the flow control valve must begin moving open.
- This allows more product through the valve, thus increasing the flow rate.



Exercise 6

Monitor Program Operation

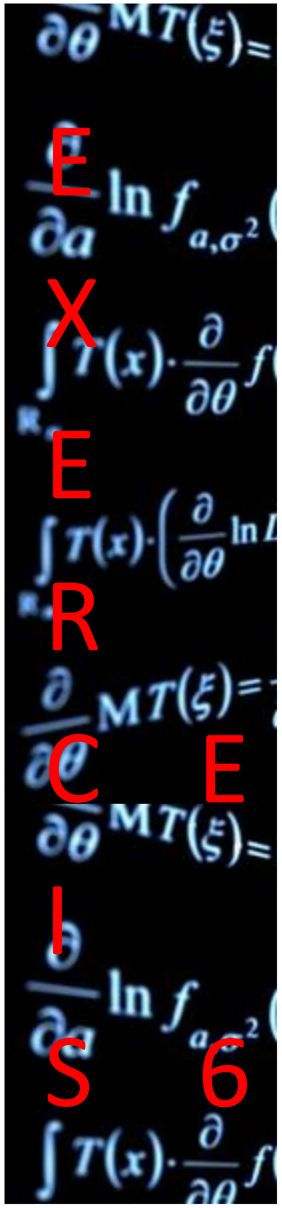
- Turn the pot farther counter-clockwise, and note that the PID Output begins to rise more quickly.
- If the student does not reduce the error by turning the pot clockwise, the PID Output will continue to rise until it is clamped by the Full value (100%).
- This would cause the control valve to rotate to its full open position.



Exercise 6

Monitor Program Operation

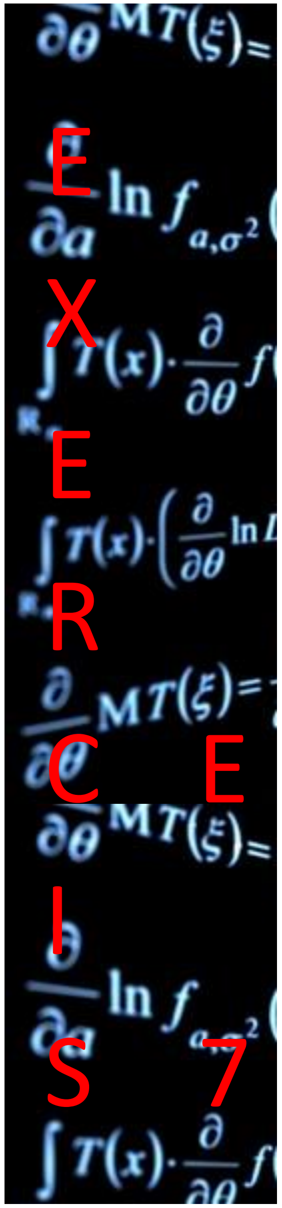
- Turn the pot back clockwise until the Process Value rises above the Setpoint.
- Now the flow rate on the pipeline is too high, and as a result the valve needs to begin closing to allow less product through. As a result the PID Output should begin dropping.
- At this point, try changing the values of Gain and Reset. Only change one at a time.
  - Note that increasing Gain or reducing Reset will both cause the PID loop to react more vigorously.



Exercise Seven

Exercise 7

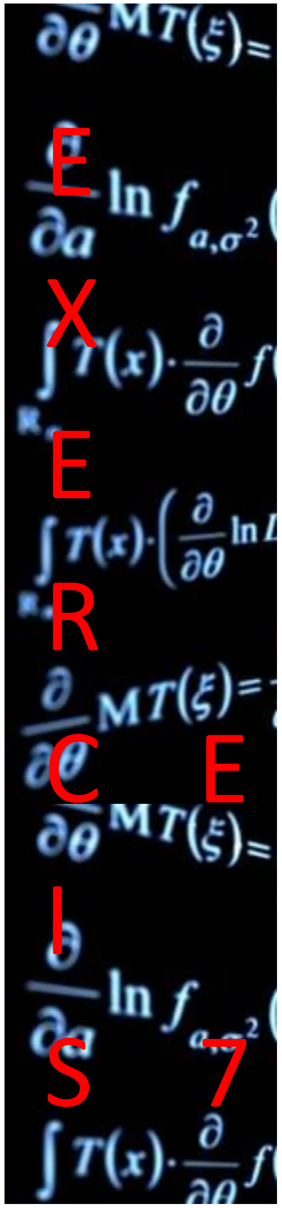
- **Data Logging and SCADALog**



Exercise 7

Data Logging and SCADALog

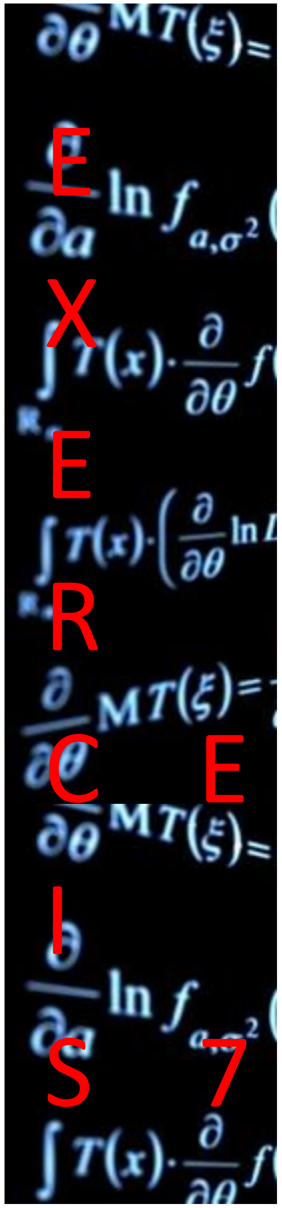
- The DLOG function block is used to populate a data log in a SCADAPack controller.
- Each time a low to high transition occurs on its Grab Data input a record is generated.
  - A record consists of a number of data fields.
- As many as 16 data logs may exist in a controller at any time, and each data log may contain anywhere from one to eight fields.
- This will allow for as many as 128 data points to be captured in a single SCADAPack.



Exercise 7

Data Logging and SCADALog

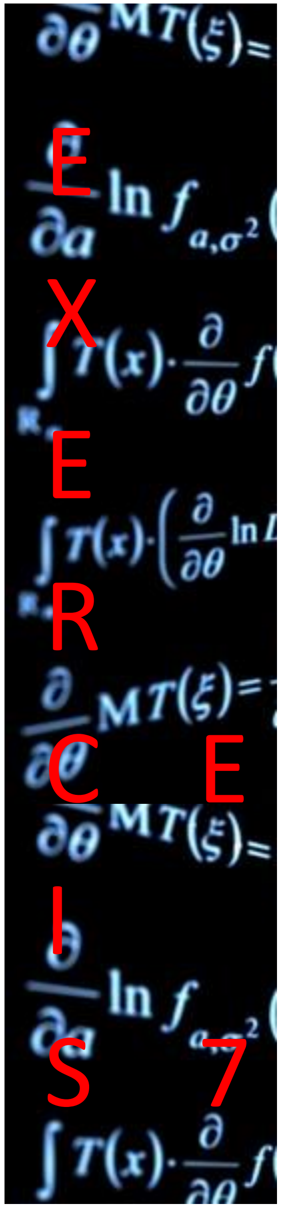
- The log contents may be deleted at any time by toggling the Delete Log input low then high again.
- This may be done after data is read out of the log using the GETL function if desired.
- Each data log field may be assigned to one of six data types.
  - These include 16 bit unsigned and signed integer, 32 bit unsigned and signed integer, 32 bit floating point and Time & Date
- There is no option to log boolean values but, if this is desired, a block of booleans may be transferred into a holding register using the MOVE function.



Exercise 7

Data Logging and SCADALog

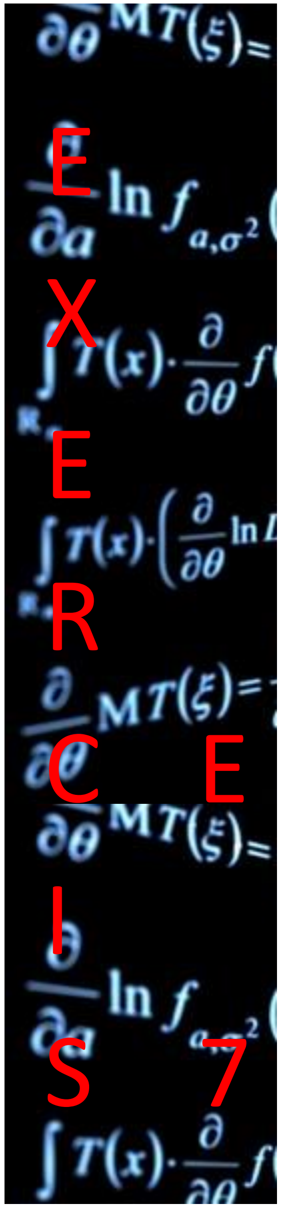
- The Time & Date data type is stored in two 32 bit unsigned integer registers.
- If the Time & Date is read using the GETL function the user will need to interpret the data manually.
- SCADALog will convert it to actual time and date information automatically.
  - The first 32 bit register contains the number of complete days since 01/01/97. The second 32 bit register contains the number of hundredths of a second since the start of the current day



Exercise 7

Data Logging and SCADALog

- The programmer must be aware of the maximum available memory for data logs in the controller they are working with.
- This is documented in the DLOG function under the TelePACE Studio Help section.
  - For example, a data log record containing Time & Date (4 words), two floats (2 words each) and two integers (1 word each) would require 10 words. Each time the Grab Data input is toggled this amount of memory is used.

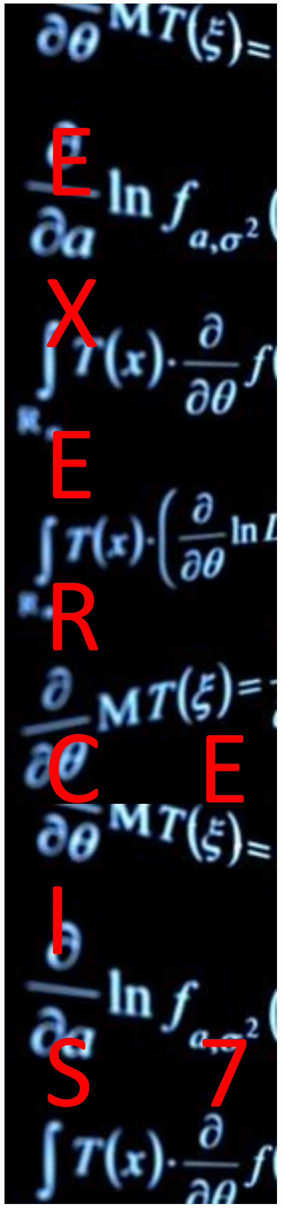


Exercise 7

DLOG

- **Create Tag Names**

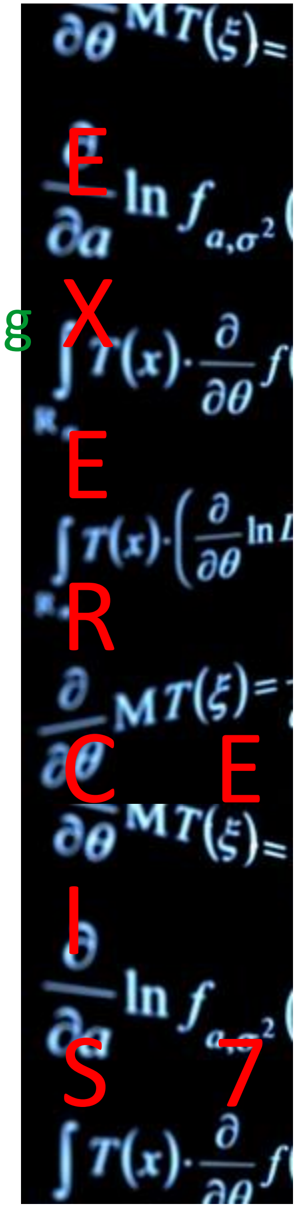
Tag Name	Register Address
Enable Get Log	10001
Read Log Data	10002
Analog Input 0	30001
Status Inputs	41001
Data Log Config	41201
Data Log Status	41219
Get Log Sequence	41301
Get Log Error	41303
Sequence Echo	41304
Record Length	41306
DaysFrom 01/01/97	41307
100ths Of Secs	41309
AIN 0 Field	41311
DINs Field	41312



Exercise 7

Build Ladder Logic

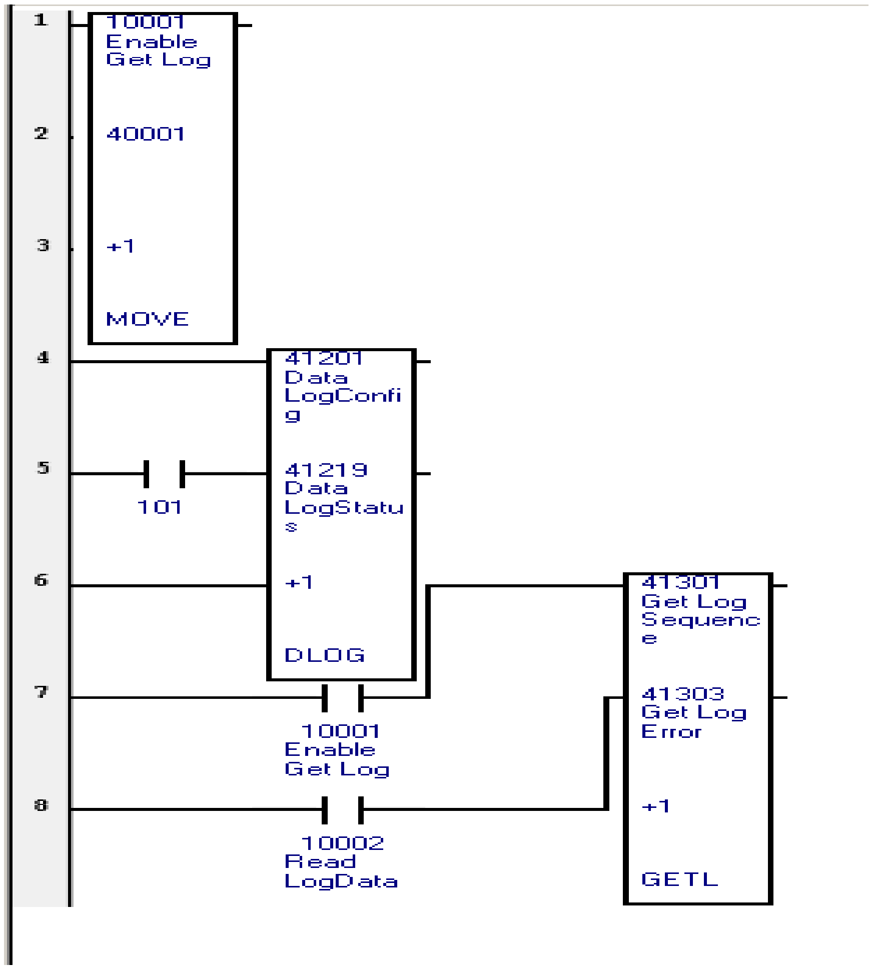
- This program will gather data every two seconds, triggering the DLOG's Grab Data input with a PULS function.
- The DLOG has three fields: Time & Date, Analog Input 0, and Status Inputs.
- As there is no boolean data type available, a MOVE function is used to copy the first 16 digital inputs into a holding register.
- The GETL function is included to demonstrate its use in retrieving log records and placing them into modbus registers.

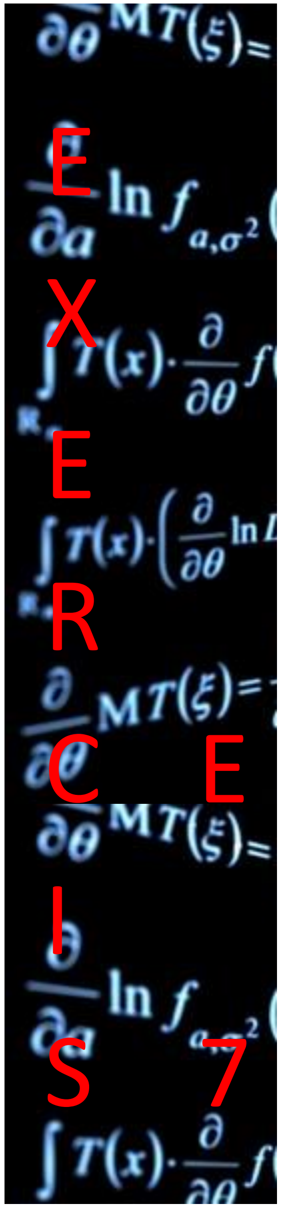


Exercise 7

Build Ladder Logic

- Select the appropriate Controller Type in the Controller menu.
- Create a Default Register Assignment with the appropriate lower I/O board.
- Enter the ladder logic as shown

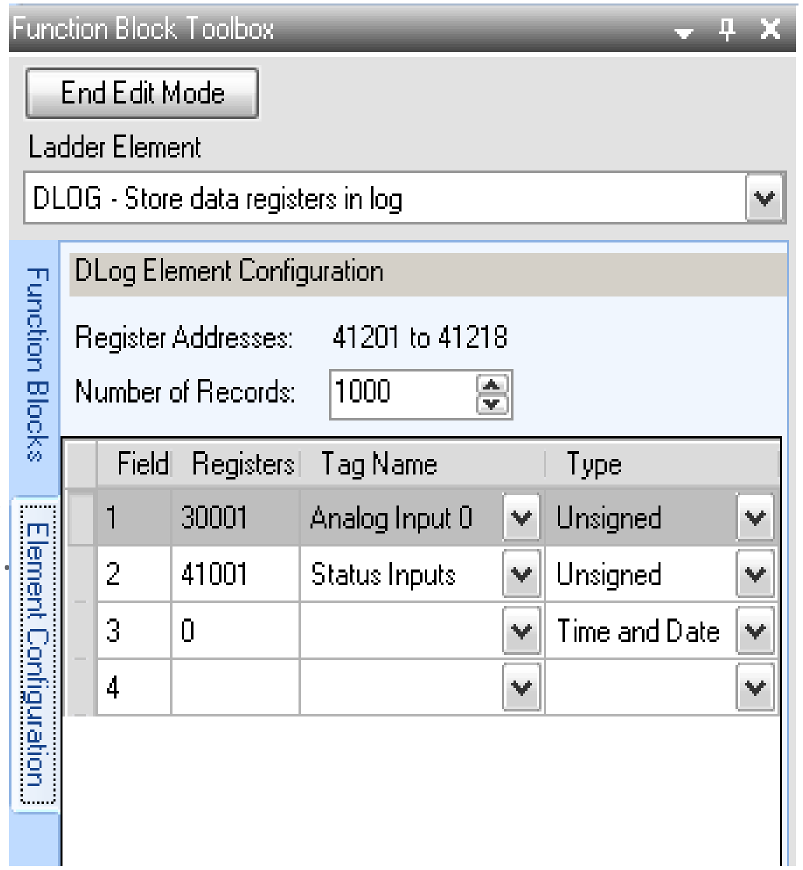




Exercise 7

DLOG

- Complete the DLOG function Element Configuration as shown
  - The DLOG will gather a total of 1000 records in a first in – first out buffer.
  - Three fields will be gathered in each record. The first is Time & Date. This does not require modbus registers – just select the appropriate data type. Second is register 30001, which is Analog Input 0. Third is register 41001, which contains the first 16 digital inputs



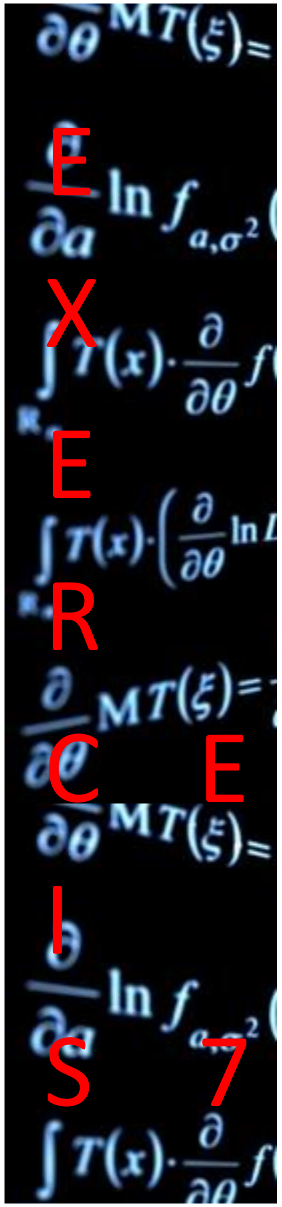
Ladder Element: DLOG - Store data registers in log

DLog Element Configuration

Register Addresses: 41201 to 41218

Number of Records: 1000

Field	Registers	Tag Name	Type
1	30001	Analog Input 0	Unsigned
2	41001	Status Inputs	Unsigned
3	0		Time and Date
4			

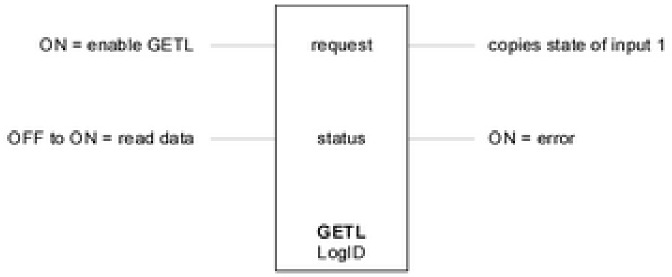


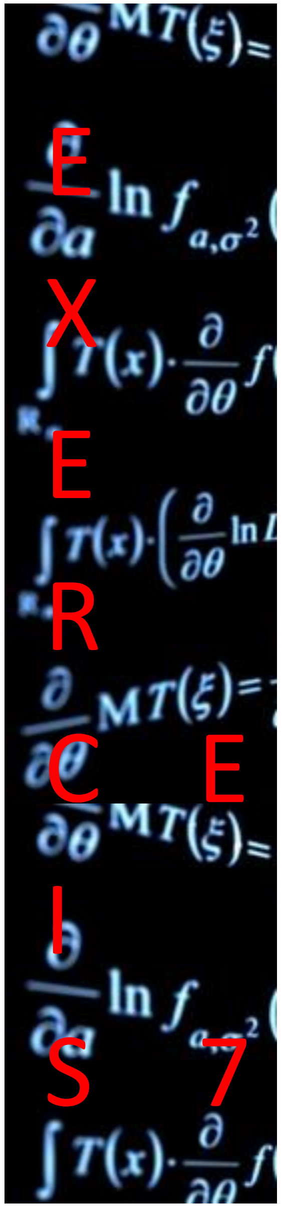
Exercise 7

Monitor Program Execution

- Once online with the controller, create a new Group in the Register Editor.
- Add the following registers in order to test the GETL function:

• 41303 – Unsigned	41301 – Unsigned Double
• 41306 – Unsigned	41304 – Unsigned Double
• 41311 – Unsigned	41307 – Unsigned Double
• 41312 – Unsigned	41309 – Unsigned Double

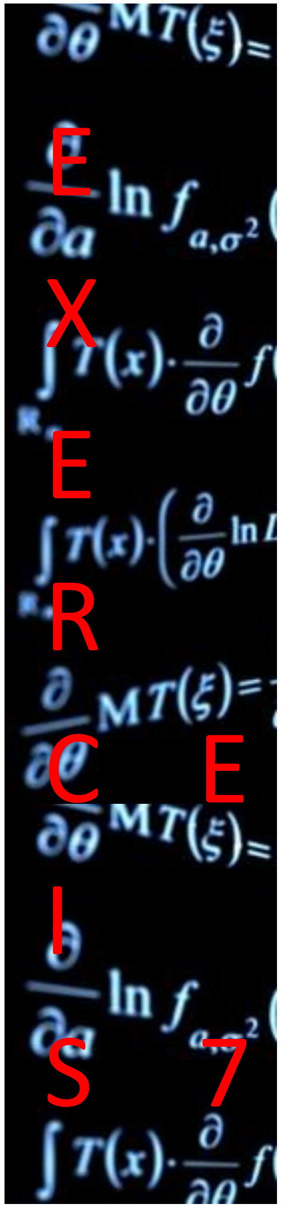




Exercise 7

Monitor Program Execution

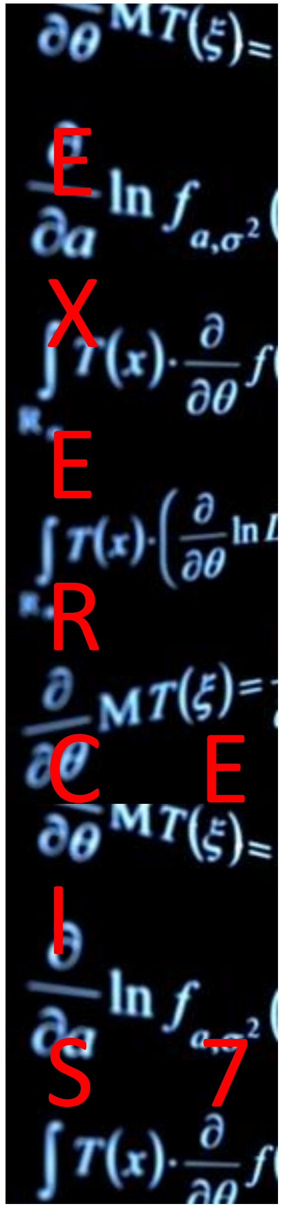
- The DLOG has been gathering data every two seconds, starting with sequence # 0. Enter a desired sequence # to view into register 41301.
- With switch 1 on to enable the GETL function, toggle switch 2 (Read Data) on and off again.
- If a 23 error code is generated, the sequence # does not yet exist. Enter another value, then toggle the Enable input (switch 1) off and on again to clear the error.
- If no error is generated the log will return the field values for Time & Date, AIN 0 and DINs.



Exercise 7

Monitor Program Execution

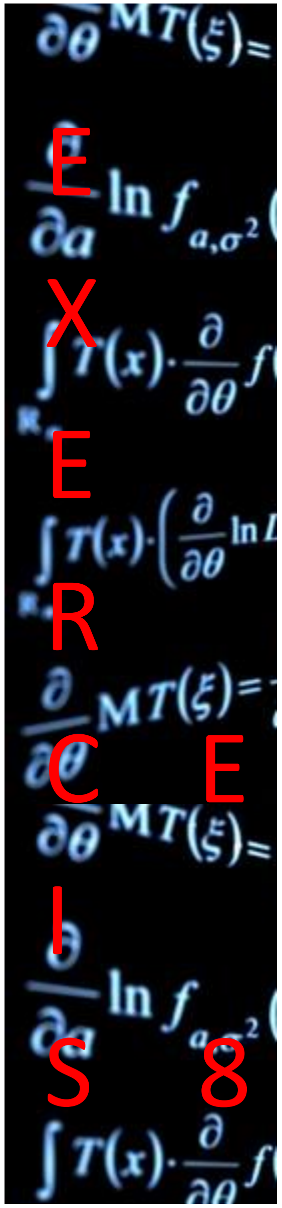
- Once you are satisfied with the operation of the DLOG and GETL functions, close TelePACE Studio.
- Open SCADALog, and start a new configuration by clicking on File→New.
- Set up comms to the controller by clicking on Communication→PC Communication Settings. The Modbus RTU protocol is used as with TelePACE Studio.
- Read the controller's data log configuration by clicking on Data Log→Configuration, then clicking the Read button. This will read the setup of all logs in the controller.



Exercise Eight

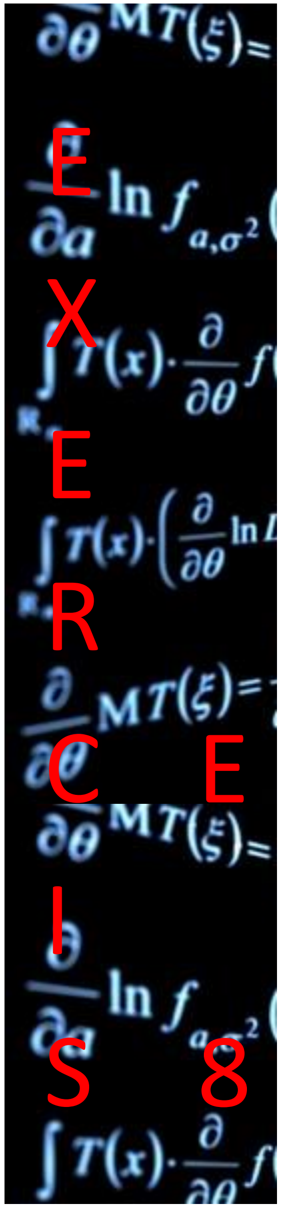
Exercise 8

- **Modbus Master Communications**



Exercise 8

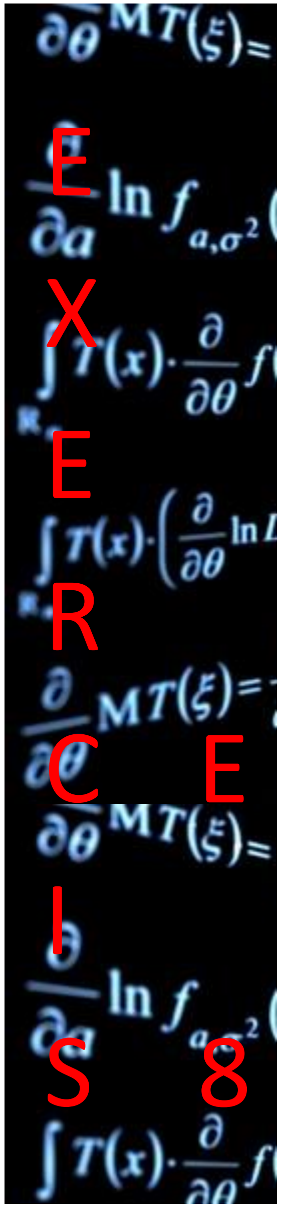
- The **MSTR** function block is used to initiate a master message to exchange data with another device via the serial port of a SCADAPack controller.
- The message may use either the Modbus or DF1 protocols.
- Modbus is an open-source, non-proprietary protocol which is implemented by many equipment manufacturers in order to maximize inter-operability.
- DF1 is a protocol primarily used by Allen-Bradley and anyone wishing to communicate with their equipment.
- The following exercise will demonstrate only the Modbus protocol.



Exercise 8

Modbus Master Communications

- Each time a low to high transition occurs on the master’s Enable input a message is sent. A timer starts at the same time.
- If a valid reply arrives before the timer’s Time Out delay is reached, the Message Complete output goes true and the message succeeds.
- If no reply has arrived after the Time Out delay, or if the reply is corrupted, the Message Error output goes true.

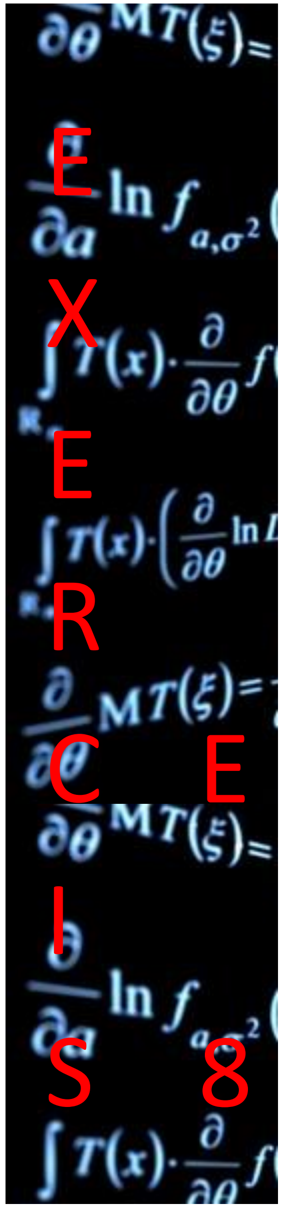


Exercise 8

Modbus Master Communications

- Only one master message may be active on each serial port at any one time.
- The modbus protocol must receive a reply to each message before the next message is sent.
- A separate message may be sent on a different port at the same time, as it will be on a different network

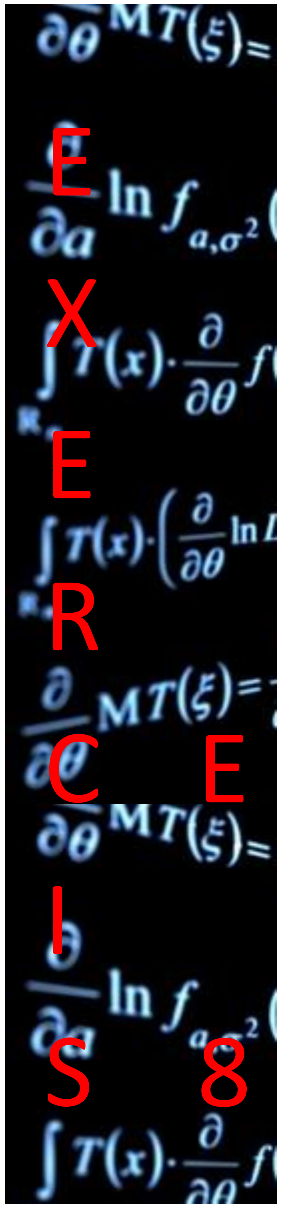
```
0E  1F  BA  0E  00  B4  09  CD  21  B8  01  4C  CD  21  54  68
69  73  20  70  72  6F  67  72  61  6D  20  63  61  6E  6E  6F
74  20  62  65  20  72  75  6E  20  69  6E  20  44  4F  53  20
6D  6F  64  65  2E  0D  0D  0A  24  00  00  00  00  00  00  00
CB  64  06  45  8F  05  68  16  8F  05  68  16  8F  05  68  16
4C  0A  37  16  87  05  68  16  A8  C3  15  16  AE  05  68  16
70  25  6C  16  8D  05  68  16  8F  05  68  16  8E  05  68  16
4C  0A  35  16  94  05  68  16  8F  05  69  16  4A  07  68  16
A8  C3  05  16  38  05  68  16  A8  C3  06  16  EF  04  68  16
A8  C3  14  16  8E  05  68  16  A8  C3  10  16  8E  05  68  16
52  69  63  68  8F  05  68  16  00  00  00  00  00  00  00  00
```



Exercise 8

Modbus Master Communications

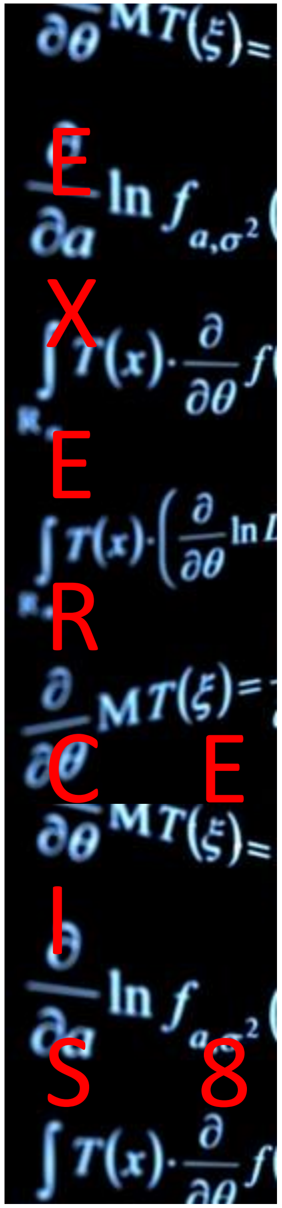
- Modbus is a low-level protocol which does not include such refinements as Retries and Report By Exception.
- If these features are desired then additional ladder logic code must be written to implement them
- It is possible to add this functionality and more by instead using the DNP3 protocol.



Exercise 8

Modbus Master Communications

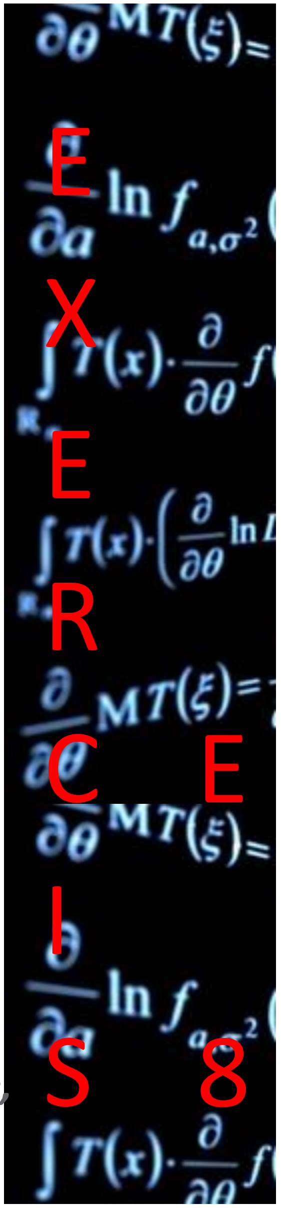
- Within the Modbus protocol there are eight primary functions.
- The programmer may choose to send a Read or a Write message.
- It is possible to read any of the four Modbus data types: coil, input status, analog input or holding register.
- When writing, it is only possible to write to the output registers – the coil and holding register types.



Exercise 8

Modbus Master Communications

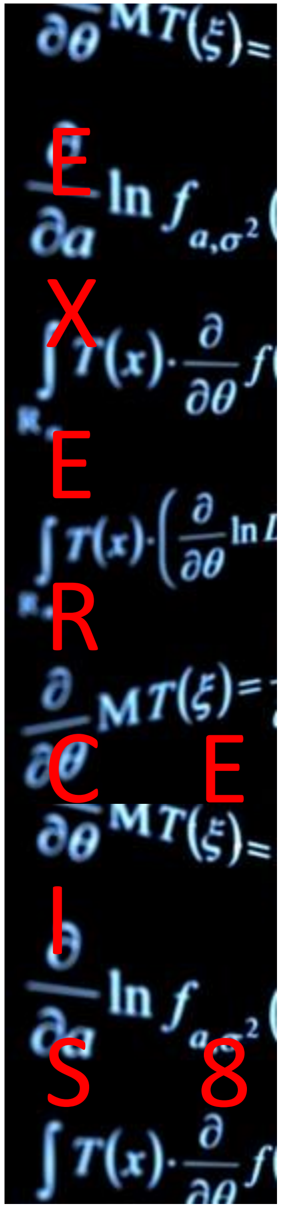
- Within the Modbus protocol there are eight primary functions.
- The programmer may choose to send a Read or a Write message.
- It is possible to read any of the four Modbus data types: coil, input status, analog input or holding register.
- When writing, it is only possible to write to the output registers – the coil and holding register types.
  - ***The MSTR function section of the TelePACE Studio Helpfile also discusses such issues as the maximum number of registers that can be read or written in a single message, and how to use the DF1 protocol in a MSTR message.***



Exercise 8

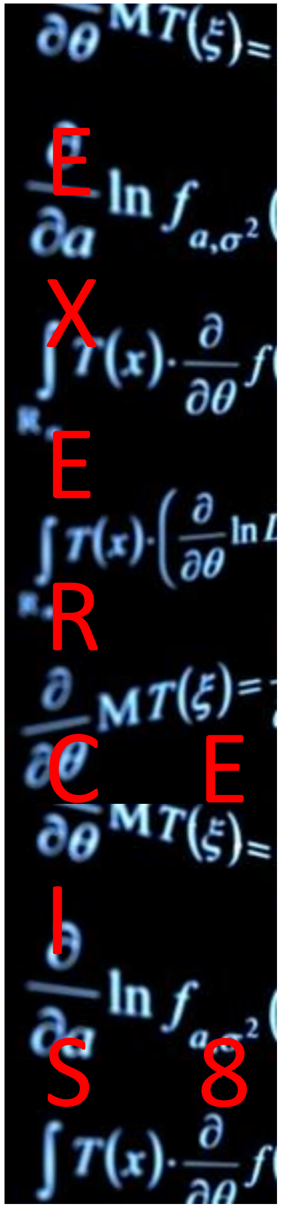
- **Create Tag Names**

Tag Name	Register Address
Complete	00101
Error	00102
Master Enable	10001
Analog Input 0	30001
Slave AIN 0	41001
Master Config	42010



Exercise 8

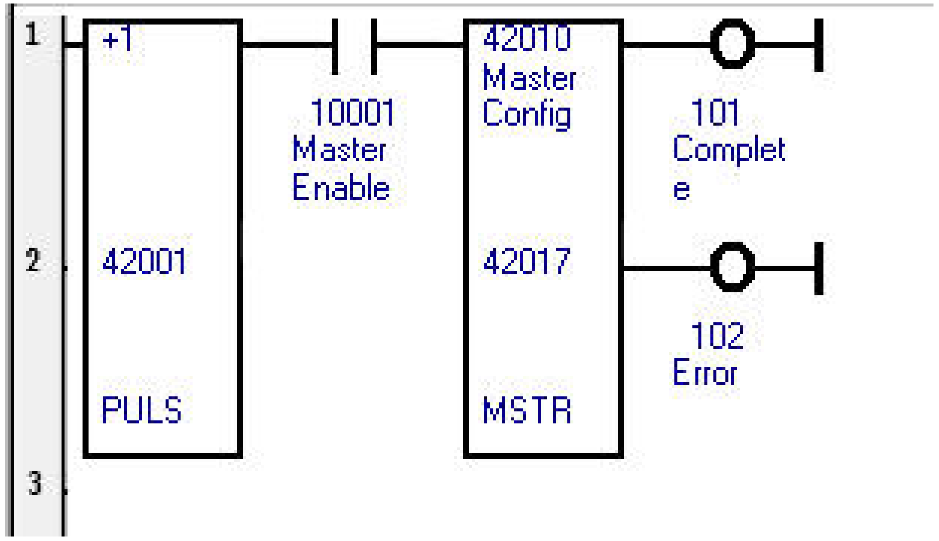
- For this demonstration two student’s SCADAPacks will be connected together to create a simple network.
- As only one controller may be the Master at any time
  - The Master Enable switch is used to select this.
- The PULS function in this program will turn on once every ten seconds.
- With the Master Enable switch On, the Off–On transition generated by the PULS will trigger the MSTR function to send a message.
- If the message succeeds the upper output will go true very quickly. If it fails, the lower output will go true after two seconds.

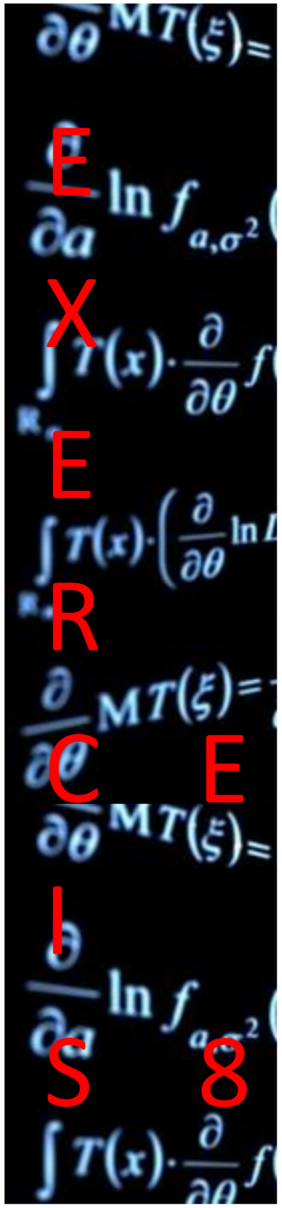


Exercise 8

Build Ladder Logic

- Enter the ladder logic as shown

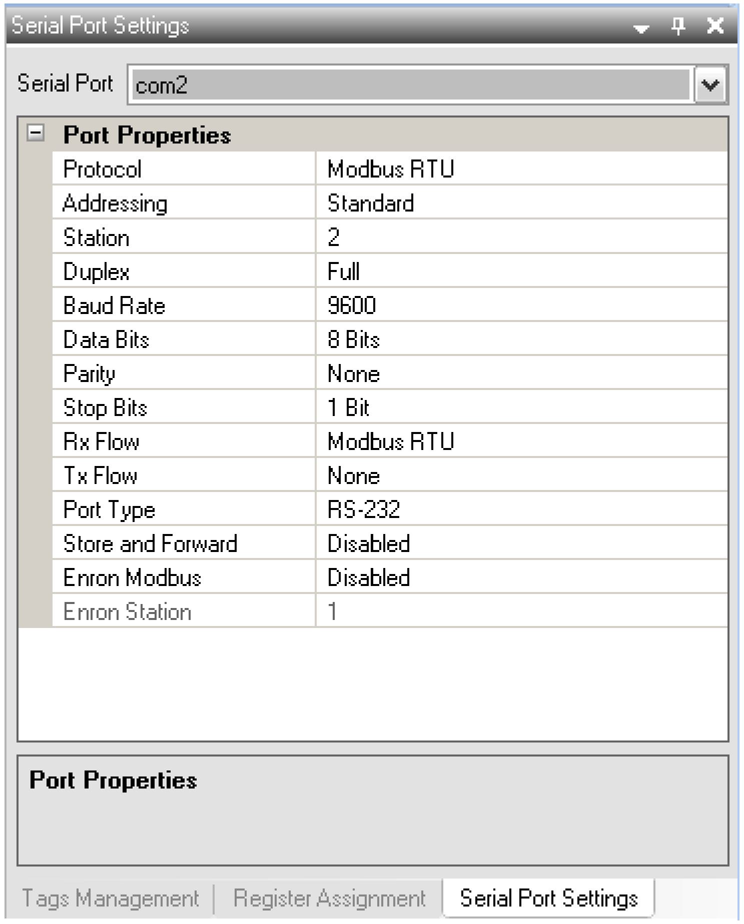




Exercise 8

Build Ladder Logic

- Select the Serial Port Settings tab
- Set the COM Port parameters as shown



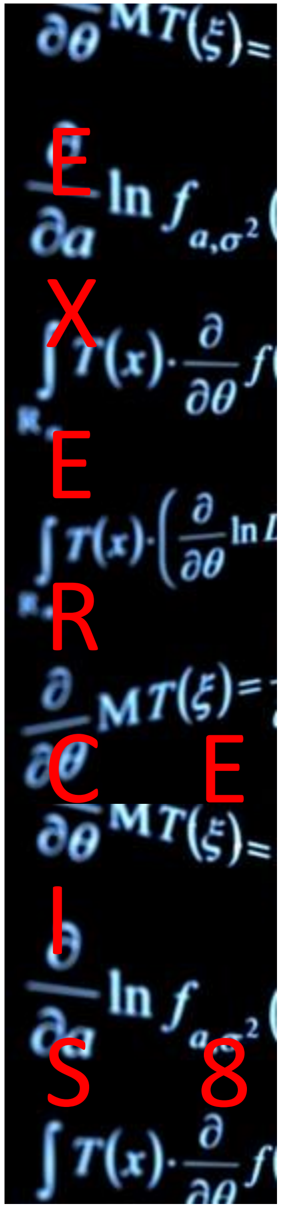
Serial Port Settings

Serial Port: com2

Port Properties

Protocol	Modbus RTU
Addressing	Standard
Station	2
Duplex	Full
Baud Rate	9600
Data Bits	8 Bits
Parity	None
Stop Bits	1 Bit
Rx Flow	Modbus RTU
Tx Flow	None
Port Type	RS-232
Store and Forward	Disabled
Enron Modbus	Disabled
Enron Station	1

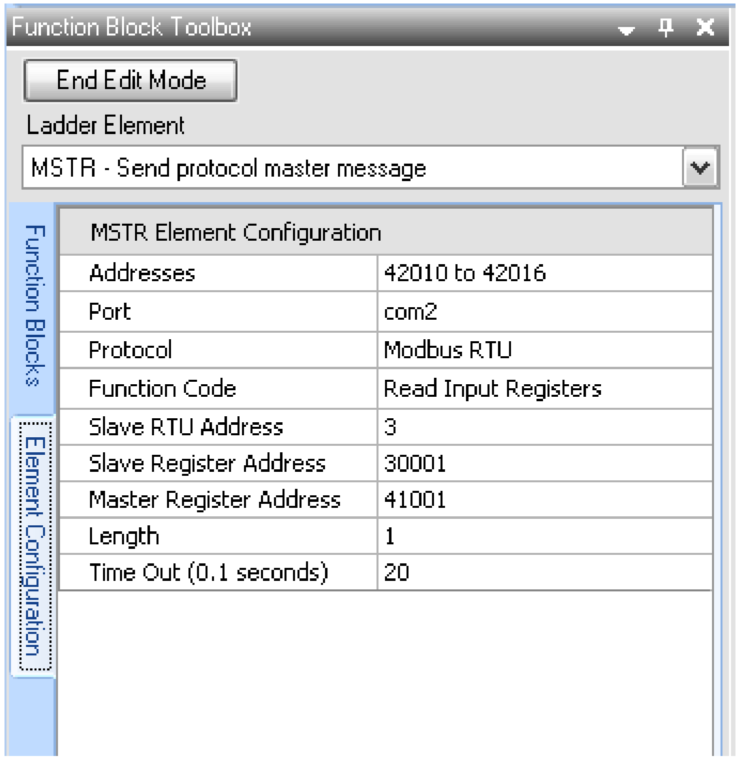
Port Properties



Exercise 8

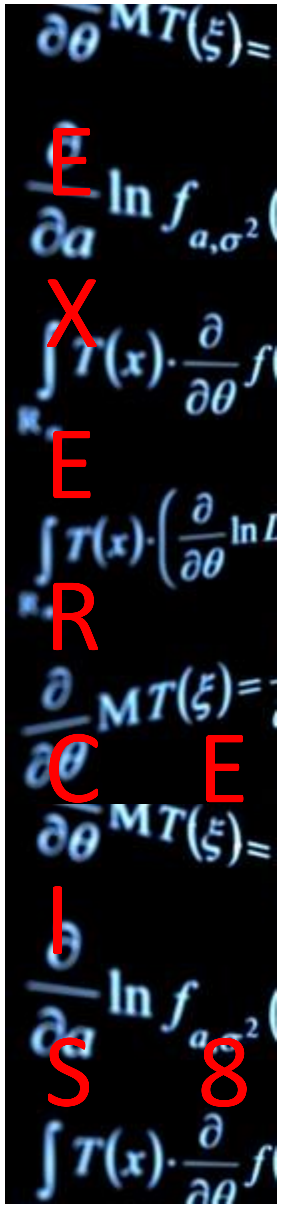
Build Ladder Logic

- Complete the MSTR function Element Configuration as shown
- The Slave RTU Address will be the modbus station number of the other student's controller



Ladder Element: MSTR - Send protocol master message

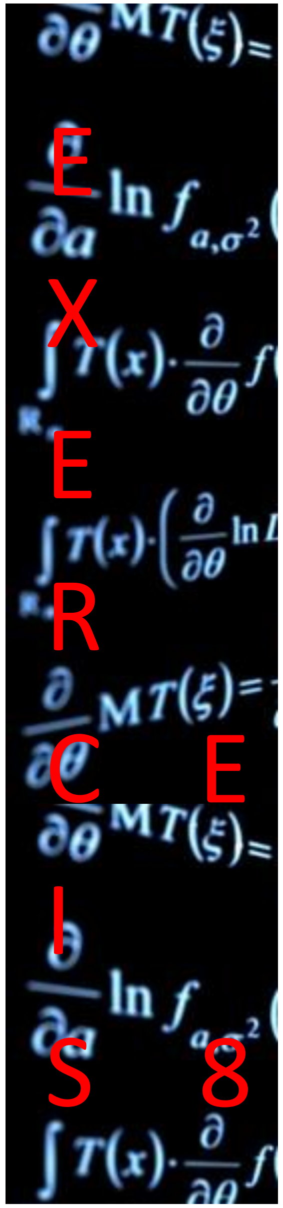
MSTR Element Configuration	
Addresses	42010 to 42016
Port	com2
Protocol	Modbus RTU
Function Code	Read Input Registers
Slave RTU Address	3
Slave Register Address	30001
Master Register Address	41001
Length	1
Time Out (0.1 seconds)	20



Exercise 8

Monitor Program Operation

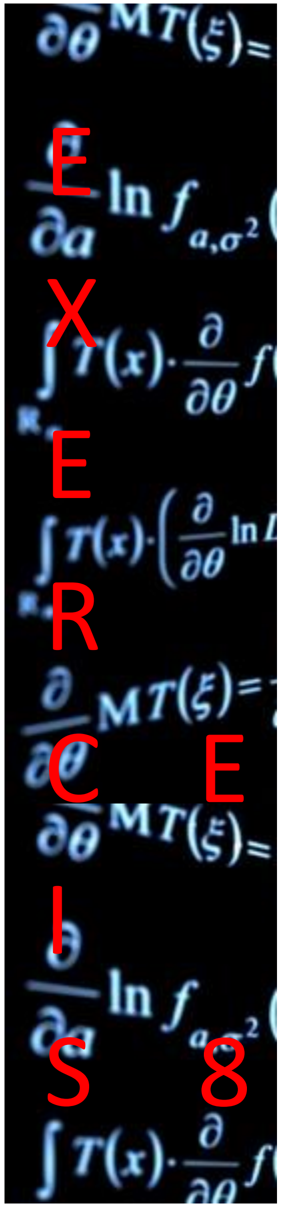
- Each student will take a turn being the Master station. The other student in each pair will be the Slave station
- When the PULS output goes true on the Master station it will send a message to the Slave, either requesting an analog input value or setting the state of a digital output.



Exercise 8

Monitor Program Operation

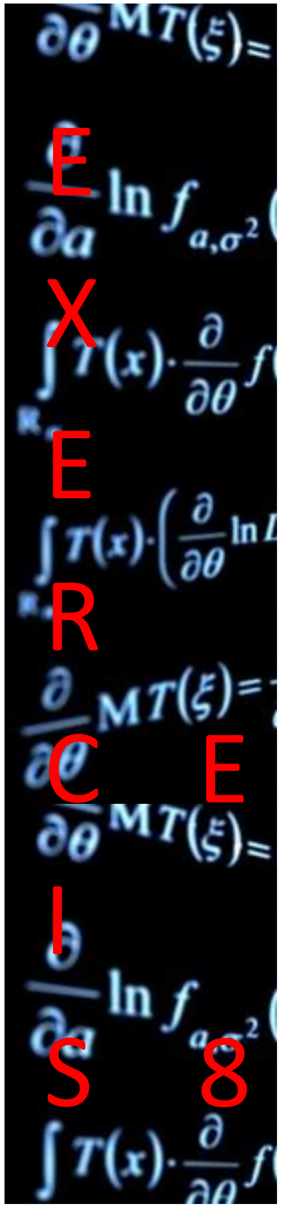
- Once online with the controller, create a new Group in the Register Editor.
- Add the following registers in order to test the MSTR function:
  - **30001 – Unsigned** **41001 – Unsigned**
- The Master should turn on their Master Enable switch.
- A master message will be sent the next time that controller's PULS function sets its output True.
  - The message will be re-sent every 10 seconds until the Master Enable switch is turned off.



Exercise 8

Monitor Program Operation

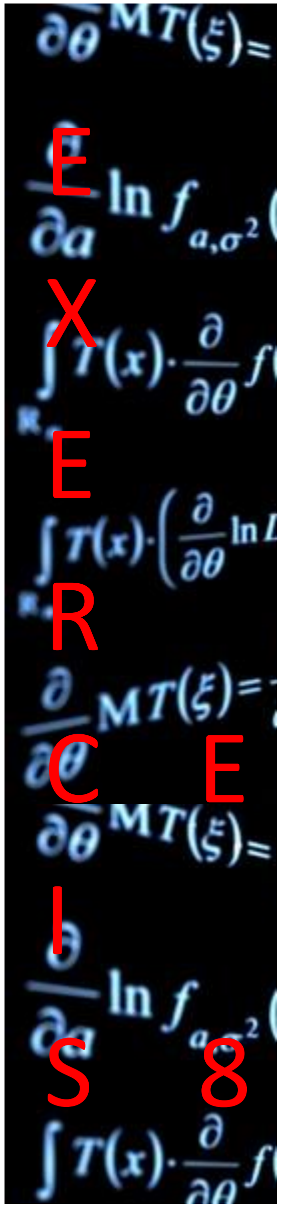
- The Master should watch their Complete and Error outputs.
  - If Complete goes true then the message succeeded.
  - If Error goes true then some configuration problem exists.
- If the message has succeeded, then the Master controller's register 41001 will now display the same value as the slave controller's register 30001. (Analog Input 0)
- If the message fails, the Error output will go on two seconds after the MSTR was enabled.



Exercise 8

Monitor Program Operation

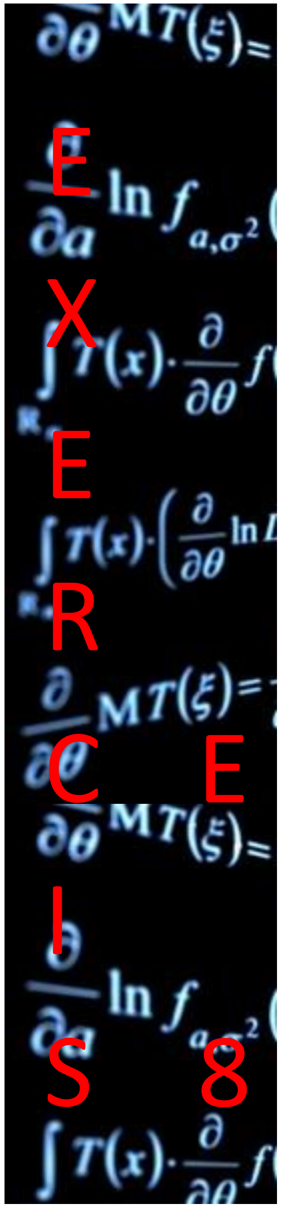
- The Slave controller should adjust the value of Analog Input 0, and then the Master should check to see if this new value is properly received.
- Once the first controller is working as Master, that person should turn Off their Master Enable switch and the other student should turn their switch On to reverse roles.
- Now ensure that the new Master station is receiving live data every 10 seconds into its register 41001 from the other station’s register 30001.



Exercise 8

Monitor Program Operation

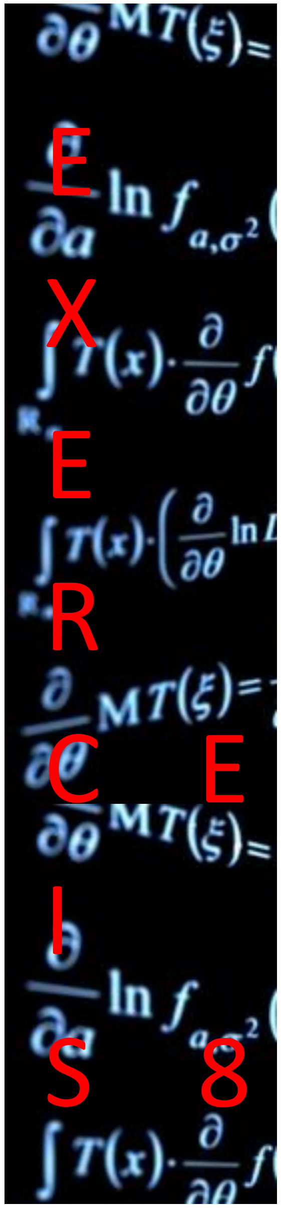
- We will now try to write data to the other controller.
- Go to Edit Online mode.
- Select the MSTR function and go to Element Configuration.
- Change the Function Code from Read Input Registers to Write Single Coil.
- Change the Slave Register Address from 30001 to 00006. (6th coil, tied to an LED)
- Change the Master Register Address from 41001 to 10004. (The 4th switch)



Exercise 8

Monitor Program Operation

- The 4th (right-most) switch on the controller whose Master Enable switch is on will be the data source.
- Whether that switch is On (1) or Off (0), it's state will be written to the Slave controller once every 10 seconds.
- Ensure the Slave's 6th LED goes on within 10 seconds of the Master's 4th switch is turned on, and goes off when the switch is turned off.
- If this fails, perform the same troubleshooting as before.



Review

- **Any Questions / Comments / Suggestions ?**

CONTROL MICROSYSTEMS

Demos

Training & Seminars