FOUNDATION™ fieldbus Application Guide

Function Block Capabilities in Hybrid/Batch Applications

NOTICE

This document was developed by a Fieldbus Foundation study team to illustrate possible use of FOUNDATION™ fieldbus function block technology in hybrid/batch applications and is not a normative technical specification. The information presented in this document is for the general education of the reader. The reader is expected to exercise sound professional judgment in using any of the information presented in a particular application.

The Fieldbus Foundation has not investigated or considered the affect of any patents on the ability of the reader to use any of the information in a particular application. The reader is responsible for reviewing any possible patents that may affect any particular use of the information presented.

Any references to commercial products in this document are examples only and the Fieldbus Foundation does not endorse any referenced commercial product. Any trademarks or trade names referenced belong to the respective owner of the mark or name. The Fieldbus Foundation makes no representation regarding the availability of any referenced commercial product at any time. The manufacturer's instructions on use of any commercial product or display of any trademark or trade name must be followed at all times, even if in conflict with the information in this document.

This document is provided on an "as is" basis and may be subject to future additions, modifications, or corrections without notice.

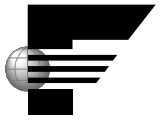
DISCLAIMER OF WARRANTIES

THE FIELDBUS FOUNDATION HEREBY DISCLAIMS ALL WARRANTIES OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING ANY WARRANTY OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE, FOR THIS DOCUMENT. IN NO EVENT WILL THE FIELDBUS FOUNDATION BE RESPONSIBLE FOR ANY LOSS OR DAMAGE ARISING OUT OF OR RESULTING FROM ANY DEFECT, ERROR OR OMISSION IN THIS DOCUMENT OR FROM ANYONE'S USE OF OR RELIANCE ON THE INFORMATION IN THIS DOCUMENT.

DOCUMENT: AG-170 (Formerly FF-950)

REVISION: 1.1

ISSUE DATE: 4 December 2002



Fieldbus Foundation

This page is intentionally blank.

Table of Contents

1. Purpose

This document describes possible applications in order to facilitate understanding of the uses for the Flexible Function Block and the various methods required to create them.

1.1 Scope

The applications describe analog, discrete and hybrid applications for Flexible Function Blocks. The specification consists of examples for the use of normative statements made in the FF-89x Function Block Application Process specifications. This specification is informative.

1.2 FF References

Number	Revision	Date	Title
FF-804	FS 1.0	August 15, 2000	Multi-Variable Optimization Addendum
FF-890	FS 1.5	November 5, 2001	Function Block Application Process Part 1 (Architecture)
FF-891	FS 1.5	October 28, 2001	Function Block Application Process Part 2 (10 Standard Blocks)
FF-892	FS 1.5	November 5, 2001	Function Block Application Process Part 3 (Additional Blocks)
FF-893	FS 1.0	March 14, 2000	Function Block Application Process Part 4 (Multiple I/O)
FF-894	FS 1.0	September 21, 2001	Function Block Application Process Part 5 (Flexible Block)

1.3 Definitions

All definitions are in the Function Block Application Process specifications listed above in FF References, except:

Profile: A concise descriptive sketch of a much more detailed application. (Webster - 5: A concise biographical sketch.)

1.4 Acronyms and Abbreviations

The following acronyms and abbreviations are used in this document.

DC: Device Control function block.

DD: Device description.

FFB: Flexible Function Block.

FOD: Fixed Object Dictionary.

FPR: Fixed Object Dictionary for a Programmable Resource.

VOD: Variable Object Dictionary.

VPR: Variable Object Dictionary for a Programmable Resource.

VRB: Resource Block for a Programmable Resource.

MIO: Multivariable Input/Output.

MVC: Multivariable Container.

OD: Object Dictionary for one VFD.

URL: Uniform Resource Locator. Internet RFC 1738 defines the syntax and semantics.

VFD: Virtual Field Device.

1.5 Synopsis of Specifications

The following specifications contain material that is essential to understand before the application profiles can be understood. An overview of that material is given for the general reader. Knowledge of FF-890 and FF-891 is presumed, as they are central to the function blocks available from all H1 device vendors.

1.5.1 FF-893 Multiple I/O Blocks

The standard I/O blocks defined in FF-891 have exactly one physical element for each block. FF-893 introduces I/O blocks that have multiple physical elements. For example, the Multiple Discrete Input block has 8 discrete outputs. The primary purpose for these blocks is to serve as an interface between FF and the remote I/O units used by PLC systems. MIO blocks solve the problem of assigning channel numbers to individual bits or analog values in remote I/O units. The MIO CHANNEL parameter refers to an entire remote I/O unit.

The MIO blocks are specific subclasses of the Flexible Function Block class. Similar blocks may be built with any number of I/O variables.

1.5.2 FF-804 Multi-Variable Optimization

The limited information bandwidth of H1 may make it necessary to consolidate some of the variables in a device for transmission in a single message. For example, the eight discrete outputs of an MDI block may be published in a single message instead of eight separate messages. This makes the values available to any other device on the H1 bus that wishes to subscribe to the single message published by the device that contains the MDI block. The subscribing device has a list that matches the list of variables published, except that it directs the values to inputs of blocks within the subscribing device. Individual published values may be ignored if the subscribing device does not use them.

In the publishing device, output values are gathered into a Multi-Variable Container for broadcast in a message. The location of each value is determined by its OD index, so the values are not limited to one function block within the device. The size of the container is limited by the maximum message size on the H1 bus. In the subscribing device, the MVC contains the OD indexes of the function block inputs that are to receive the data. The lists are kept synchronized by a revision number.

Published MVC messages are restricted to objects of the class Output, and subscriber MVC lists may only contain objects of the class Input. This scheme is intended only for linking function block outputs in one device to function block inputs in another.

The MVC may also be used to gather function block objects of the classes Input, Output, and/or Contained to be sent as a Report Distribution message. Any Host device may subscribe to this message. It functions like a Variable List (e.g., View) object except that it does not have to be requested, it may contain values from several blocks, and it can be scheduled at any multiple of the Macrocycle. It is also true that an MVC Report Distribution message may consist of a standard function block View.

The choice of method is determined by considerations including network loading, device functionality and host support.

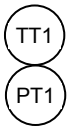
1.5.3 FF-892 FBAP Part 3

This specification contains many useful function blocks, but the one used in this document is the Device Controller (DC). It is intended to control any two or three state (e.g., position or speed) physical device, in the sense that it accepts a setpoint and causes the device to drive to that setpoint. Time is allowed for the transition, but alarms are generated if the physical device fails to reach the desired state or loses that state after the transition is complete. The DC block has inputs for control of the setpoint by external logic or commands from a host, as well as permissive, interlock and shutdown (emergency stop) logic functions. An operator may temporarily bypass a faulty limit switch after visual confirmation of the state of the physical device. The parameter DC_STATE displays one of 14 states that describe the current control condition (e.g., Open, Opening, Delaying, Failed to Open, Failed to leave Closed, Locked Out). The parameter FAIL gives specific reasons for failures.

1.6 Drawing Conventions

1.6.1 Process Diagram

A Process Diagram is a sketch of process piping and vessels with sensors and actuators, along with some descriptive text. No field wiring is shown because many fieldbus arrangements are possible, and wiring clutters up the drawing with detail that is not essential to the application. The ‘tags’ have three characters: first is the process variable type that is being sensed or controlled, second is T for transmitter or C for control, and third is a loop number. An analog device tagged ‘PC1’ will have a sensor ‘PT1’ linked to it over fieldbus, unless otherwise specified in the text.



This diagram represents two sensors within the same physical fieldbus device.

This diagram represents a fieldbus device attached to an analog actuator. It contains a control block and an analog output block. It may also contain discrete input blocks for position sensors.

This diagram represents a fieldbus device attached to a discrete actuator. It contains a discrete output block, and may contain a discrete control block. It may also contain a multi-state discrete input block for position sensors.

1.6.2 Field Wiring

This heading is used for descriptive text concerning the field wiring. A diagram is present when the wiring method is essential to the application.

1.6.3 Function Block Diagram

A Function Block Diagram shows the FF function blocks used in the application. Links between blocks are shown in one of three ways:

Solid line - a conventional link between single output and single input.

Double line - a cascade control connection with connections in the forward and backward directions.

Dotted line - a link made using the Multi-Variable Optimization.

Links may or may not be made over fieldbus, depending on the application. Multi-Variable Optimization links are never made within the same device because communication is not involved.

Function block parameters of the class Contained are not shown in the diagram because it is only intended to show links. Function block parameters are listed under the heading ‘Block Access’ in the specifications or in this document when describing an application FFB.

No attempt is made to group blocks into devices. Many arrangements are possible with FOUNDATION fieldbus.

2. Application Overview

The applications described in this document are intended to illustrate the capabilities of the Flexible Function Block defined in FF-894, as well as the Multiple I/O blocks defined in FF-893 and the Multi-Variable Optimization defined in FF-804.

The method chosen for describing each application in Section 3 is to specify a function block in much the same manner as the standard function blocks in FF-891 and FF-892. Each specified block is an Application Specific function block and should not be confused with a standard block.

Section 4 places more emphasis on the process and less on the details of the blocks.

A Flexible Function Block may have its function defined in one of two ways:

1. A user familiar with Distributed Control Systems may choose to write a program that is compiled for the target device and downloaded to a Domain in that device. The code does not have to be compiled. It could be written in something like BASIC and downloaded as text to a Domain in a device that interprets it.

2. A user familiar with Programmable Logic Controllers may choose to use a PLC programming tool to develop an application program in any of the IEC 61131-3 languages that the device can execute. The program is also downloaded to a Domain, but the operation of the program is controlled by a Program Invocation object.

In either case, the device that accepts human programming input may have to be matched to the device. The connection to the device may be proprietary, using a non-FOUNDATION fieldbus Ethernet port, or it may use the standard FF means to modify the device. The programming device may also be required to modify the Object Dictionary of the device in order to create the required FFB. This depends on the complexity of the device.

1. A field device may be built with an Object Dictionary that cannot be modified by user programming. In that case, each FFB has a predefined set of parameters much like the predefined set of registers in a PLC. The DD for the device will define the names of the parameters. The vendor needs a DD compiler but the user does not, unless the user can change parameter names.

2. A programming device may be built for the purpose of programming field devices, which can generate a new OD and DD for the specific application. Both the programming device and field device require a higher level of complexity than those with predefined OD structures.

Once the application and the FFB are defined in the device, it is necessary to configure values for the Block object and the parameter objects. It is also necessary to link input and output parameters if other function blocks are involved. Finally it may be necessary to create the Variable List objects that are the four Views of the block. It must be possible to do these things with standard FF configuration tools because they are interoperable communication functions of the device.

The following sections are notes on general properties uncovered by the effort to specify specific blocks.

2.1 Parameters

If a parameter is defined as an Input or Output, then its data type must be one of those listed in FF-890, section 5.13. These parameters are coupled between blocks by the linking system, which can only handle the standard data types. Parameters used only for client/server communication may have structures defined by the manufacturer, or by the user but only if the device supports variable OD and DD. New parameters must be designed following the rules defined in FF-894, Parameters with Special Semantics.

2.2 Block Execution

The non-programmable resource device blocks (DCS style) are executed as specified in the Block object. The algorithm is executed as part of block execution.

The programmable resource device (PLC style) executes the program cyclically. The rate is defined by the manufacturer. The FFBs are executed as specified in the Block object. This means that the program could run any number of times between block executions, or exactly once. The block snaps the internal registers to/from the communication objects (parameters) only when it executes.

2.3 Views

View objects are defined for the FFB in this document. They may not be mandatory. A View object is a sub-class of the FMS Variable List object, which consists of a list of object indexes. FMS has services to read or write variable lists. Thus a block can be configured in a few writes by using the static View lists. The size numbers in every block access table keep track of the size of a variable list message, which is limited to about 100 octets. Hosts with the required FMS services may configure new or existing variable lists in any device that has the matching FMS services.

2.4 Function Block Notes

The Function Block Note headings are used to describe all function blocks. The contents vary with individual blocks. The following notes apply to all FFB specified in this document except as noted in the application text.

2.4.1 Supported Modes

O/S and Auto. If the block has control outputs, it may support Manual. In Manual mode all of the outputs may be turned on and off regardless of the state of the input. Blocks that have internal setpoints may support Cas and/or RCas, as indicated by the required parameters being listed in the Block Access table.

2.4.2 Alarm Types

The presence of alarms is indicated by the required parameters being listed in the Block Access table.

2.4.3 Mode Handling

Standard but complicated by multiple setpoints and outputs. Since there is only one mode, it applies to all of them.

2.4.4 Status Handling

Standard, unless described in the text of an application. The rules defined in FF-890 may not be broken.

2.4.5 Initialization

Standard.

2.4.6 Power Failure Recovery

Retain and restore any internal values necessary to sustain specified operation after a power blink.

3. Flexible Function Block Applications

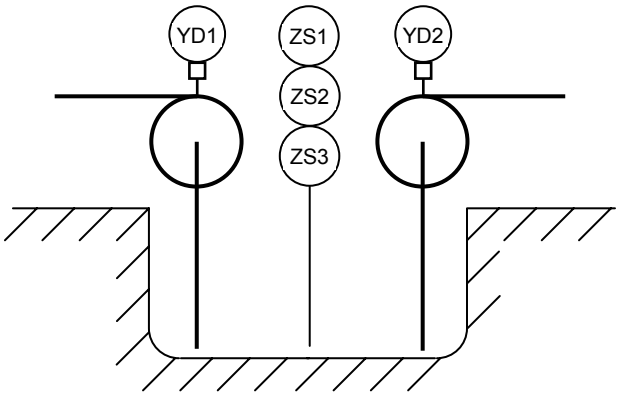
3.1 Snap Control

3.1.1 Overview

This application controls the level in a sump at three discrete points using two pumps. Normally, the pumps are alternated to extend their life. When the level is excessive both pumps are turned on.

This application uses the Fixed OD FFB. This block has a predefined OD that is described in the fixed DD and the Capabilities file for the device. The algorithm used by the block is configured by typing a list of structured text like commands intro string parameters. Functions include Boolean logic as well as various timers enabling logic and sequence. Additional function blocks such as analog alarm can be instantiated to cater e.g. for a scheme where a level transmitter is used instead of switches. The device can execute logic based on its own local I/O as well as signals received over the Fieldbus. In this application all I/O are local thus allowing the device to operate autonomously even if the H1 communication fails, provided power is still available. This is an example of an application where conventional discrete on/off signals have to be interfaced to the Fieldbus environment. This application is typical, and proves the viability of the many similar mixed applications that exist during the transition to pure Fieldbus systems, such as pressure switches, push buttons, on/off valves, motor control centers, variable speed drives, and electrical actuators, motor operated and conveyors etc.

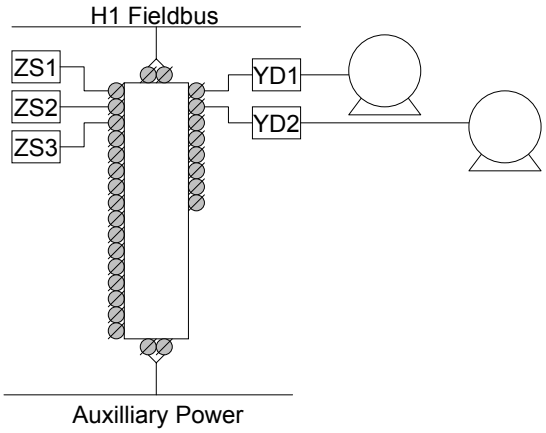
3.1.2 Process Diagram



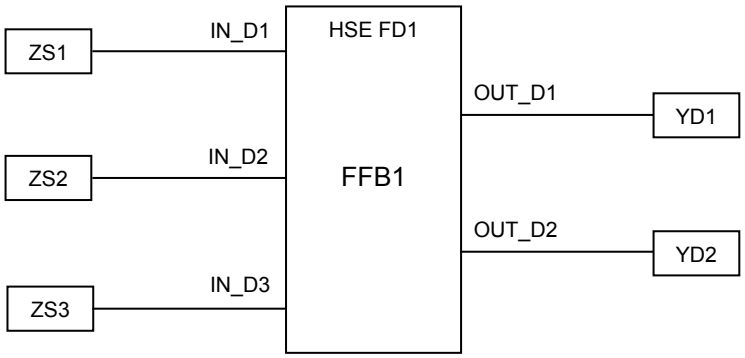
YD1 and YD2 control the two sump pumps. ZS1 closes at the very high level, ZS2 closes at the high level and ZS3 opens when the level drops below the low level setting.

3.1.3 Field Wiring

A single field device located near the sump, to minimize the wire run, can contain this application. Low-cost wiring, carrying voltage and current supplied by the device, would connect it to the level switches and motor starters.



3.1.4 Function Block Diagram



The block uses three of the discrete inputs for the states of Low, High and Very High. It uses two of the discrete outputs for the two pumps. When the Low state is true, both digital outputs are turned off. When the High state becomes true, the previous output remains off and the other output turns on. When the Very High state becomes true, both outputs are turned on and an alarm may be generated.

Manual mode is supported to allow the pumps to be turned on or off at the FFB.

The block has no setpoint parameter because the level setpoints are determined by the physical placement or adjustment of the level switches. Input and output parameters are not required by a local device. They are shown so that they may be read over the bus. Conventional discrete I/O thus become available in regular Fieldbus blocks and integrate into the control strategy just like Fieldbus devices enabling consistent implementation The logic is configured into the block as text strings and is checked internally by the device itself. This scheme allows configuration to be done from any Fieldbus host configuration tool based on regular device DD and CD files, without any need for a special configuration application and without the need to manage application specific DD and CF files for every block.

3.1.5 Block Access

The access table defines the required parameters.

Index	Parameter	VIEW_1	VIEW_2	VIEW_3	VIEW_4
1	ST_REV	2	2	2	2
2	TAG_DESC				
3	STRATEGY				2
4	ALERT_KEY				1
5	MODE_BLK	4		4	
6	BLOCK_ERR	2		2	
7	ALGORITHM_SEL				4
8	CONTENTS_REV				4
	Subtotals	8	2	8	13

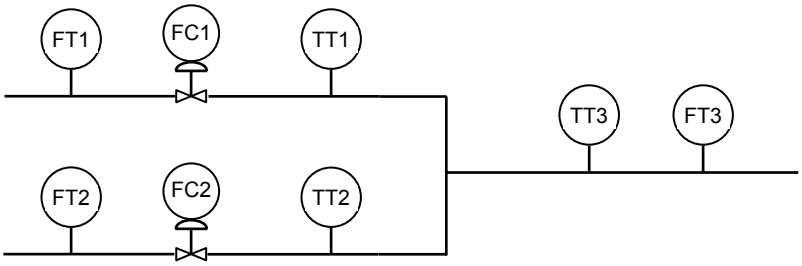
Index	Parameter	VIEW_1	VIEW_2	VIEW_3	VIEW_4
9	IN_D1	2			
10	IN_D2	2			
11	IN_D3	2			
12	OUT_D1	2			
13	OUT_D2	2			
	From left column	8	2	8	13
	Totals	18	2	8	13

3.2 Multivariable Matrix Control

3.2.1 Overview

This application is a simple example of matrix control. Two fluids are mixed so that the desired outlet stream temperature and flow rate are controlled. The matrix is configured inside of one FFB. Because matrix control requires accurate measurements, a second FFB is configured with mathematical expressions to check that mass and energy are conserved. It also checks that there is zero flow if the valve is closed, and that the valve is not closed when the associated flow setpoint is above zero.

3.2.2 Process Diagram



FT1 and FC1 form a loop controlling the associated flow, as does FT2 and FC2. TT1 and TT2 measure the stream temperatures just before they enter the mixing junction and after work done by the valve has changed the temperature. TT3 measures the result of mixing. FT3 is used to determine the amount of heat in the outlet flow.

3.2.3 Field Wiring

The field wiring is H1 FOUNDATION fieldbus. Not shown is a FOUNDATION fieldbus HSE fieldbus linking device, which may be located in any convenient non-hazardous area. The linking device requires two or three H1 Ports and the ability to run the flexible function blocks described below. Field device connection to bus segments depends on plant conditions:

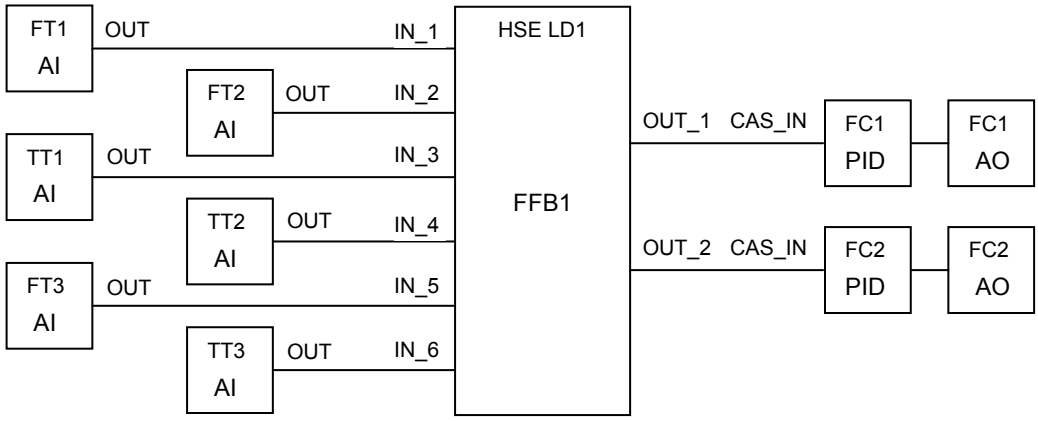
If the field devices are not multiple measurement devices and the plant wiring policy for one bus is one valve and up to two more devices, then the segments could be arranged as follows:

Segment 1: FC1, FT1, TT1, Segment 2: FC2, FT2, TT2, Segment 3: FT3, TT3

If multiple measurement field devices are used, and both flow and temperature can be measured downstream of the valve (valve drop will change the temperature), then FT and TT can be combined in MVT devices as follows:

Segment 1: FC1, MVT1, Segment 2: FC2, MVT2, MVT3

3.2.4 Function Block Diagrams

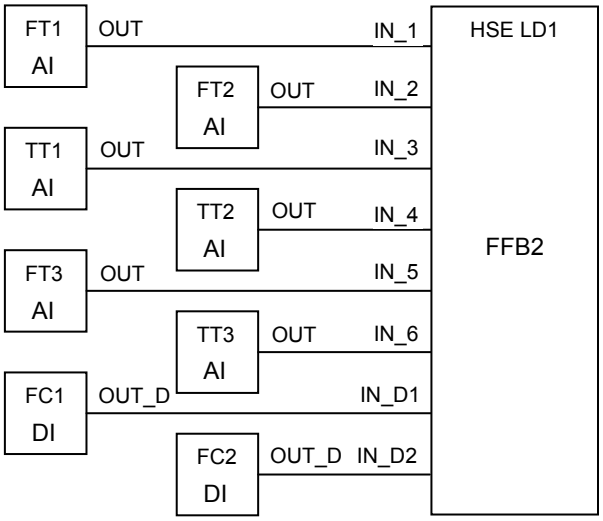


FFB1 is linked to the process devices as shown above. The cascade links between PID and AO are not shown because they are internal to the control valve instruments. No BKCAL link is shown between FFB1 and the PID blocks because the OUT parameters are calculated. The status of the outputs must be Good Non-cascade to tell the PID that no back calculation is required. There would be contained parameters for the setpoints of the flow and temperature of the outlet stream, in addition to the universal parameters. There may also be contained status and alarm parameters.

The block algorithm is a proprietary matrix calculation that operates on the 6 inputs to produce the 2 outputs that become the setpoints for the process flow controllers. An external proprietary system may be used to generate, modify or delete the algorithm code.

The only supported modes are O/S, Manual and Auto. When the block is in Manual mode it holds whatever value is in the outputs. The outputs may be written with values from the HMI. When the block transitions to Auto mode, the outputs change to the calculated values. A contained parameter may determine how fast they change to new values. It is possible to put one of the PID blocks into Auto mode to ignore the FFB output, perhaps to ‘base load’ one of the input streams. Since there is no BKCAL link, the FFB is unaware that it has no control of one (or both) of the streams, unless it has some logic to detect the difference between the output to the PID cascade input ant the corresponding flow measurement.

The following drawing is for another FFB in the same HSE linking device, or a separate device.



FFB2 is linked to the process devices as shown above. These are the same signals used by FFB1 with the addition of a discrete input from each of the control valves. The discrete input is a Boolean that is true when the valve is closed. There would be contained parameters for the status and alarms in addition to the universal parameters.

The block algorithm is a set of simple equations that sum the flows from FT1 and FT2, and compare the sum to FT3. An alarm is generated if the difference exceeds the alarm trip point. Similarly, the heat content of streams 1 and 2 is compared to the outlet heat flow. This generates an alarm if the flow measurements do not have an alarm, because otherwise the heat calculation would be invalid. On those occasions when a valve is closed, as indicated by a true discrete input, then an alarm is generated if the associated flow is not zero. An alarm may also be generated if the flow is zero and the valve is not closed. The PID blocks in the field devices can best generate alarms for deviation between the flow setpoint and the measured flow.

The equations may mix analog and discrete values in mathematical or logical expressions. This can be done with a simplified version of BASIC or any other language of the vendor's choice. An external proprietary system may be used to generate, modify or delete the algorithm code.

The only supported modes are O/S, Manual and Auto. Manual mode is optional since the block has no outputs, but it can be used to stop alarm generation in the event that something becomes marginal.

3.2.5 Block Access for FFB1

The access table defines the required parameters.

Index	Parameter	VIEW_1	VIEW_2	VIEW_3	VIEW_4
1	ST_REV	2	2	2	2
2	TAG_DESC				
3	STRATEGY				2
4	ALERT_KEY				1
5	MODE_BLK	4		4	
6	BLOCK_ERR	2		2	
7	ALGORITHM_SEL				4
8	CONTENTS_REV				4
9	IN_1	5		5	
10	IN_2	5		5	
11	IN_3	5		5	
12	IN_4	5		5	
13	IN_5	5		5	
14	IN_6	5		5	
15	OUT_1	5		5	
16	BAL_TIME_1				4
17	OUT_HI_LIM_1		4		
18	OUT_LO_LIM_1		4		
19	OUT_2	5		5	
20	BAL_TIME_2				4
21	OUT_HI_LIM_2		4		
22	OUT_LO_LIM_2		4		
23	SHED_OPT_2				1
24	SP_1	5		5	
25	RCAS_IN_1			5	
26	RCAS_OUT_1			5	
27	SP_RATE_DN_1				4
28	SP_RATE_UP_1				4
29	SP_HI_LIM_1		4		
30	SP_LO_LIM_1		4		
31	SP_2	5		5	
32	RCAS_IN_2			5	
33	RCAS_OUT_2			5	
34	SP_RATE_DN_2				4
35	SP_RATE_UP_2				4
36	SP_HI_LIM_2		4		
37	SP_LO_LIM_2		4		
	Subtotals	58	34	78	38

Index	Parameter	VIEW_1	VIEW_2	VIEW_3	VIEW_4
38	UPDATE_EVT				
39	BLOCK_ALM				
40	ALARM_SUM	8		8	
41	ACK_OPTION				2
42	ALARM_HYS				4
43	IN_5_HI_HI_PRI				1
44	IN_5_HI_HI_LIM				4
45	IN_5_HI_PRI				1
46	IN_5_HI_LIM				4
47	IN_5_LO_PRI				1
48	IN_5_LO_LIM				4
49	IN_5_LO_LO_PRI				1
50	IN_5_LO_LO_LIM				4
51	IN_1_DV_HI_PRI				1
52	IN_1_DV_HI_LIM				4
53	IN_1_DV_LO_PRI				1
54	IN_1_DV_LO_LIM				4
55	IN_6_HI_HI_PRI				1
56	IN_6_HI_HI_LIM				4
57	IN_6_HI_PRI				1
58	IN_6_HI_LIM				4
59	IN_6_LO_PRI				1
60	IN_6_LO_LIM				4
61	IN_6_LO_LO_PRI				1
62	IN_6_LO_LO_LIM				4
63	IN_2_DV_HI_PRI				1
64	IN_2_DV_HI_LIM				4
65	IN_2_DV_LO_PRI				1
66	IN_2_DV_LO_LIM				4
67	IN_5_HI_HI_ALM				
68	IN_5_HI_ALM				
69	IN_5_LO_ALM				
70	IN_5_LO_LO_ALM				
71	IN_1_DV_HI_ALM				
72	IN_1_DV_LO_ALM				
73	IN_6_HI_HI_ALM				
74	IN_6_HI_ALM				
75	IN_6_LO_ALM				
76	IN_6_LO_LO_ALM				
77	IN_2_DV_HI_ALM				
78	IN_2_DV_LO_ALM				
	Subtotals	8	0	8	66
	Totals	66	34	86	104

3.2.6 Block Access for FFB2

The access table defines the required parameters.

Index	Parameter	VIEW_1	VIEW_2	VIEW_3	VIEW_4
1	ST_REV	2	2	2	2
2	TAG_DESC				
3	STRATEGY				2
4	ALERT_KEY				1
5	MODE_BLK	4		4	
6	BLOCK_ERR	2		2	
7	ALGORITHM_SEL				4
8	CONTENTS_REV				4
9	IN_1	5		5	
10	IN_2	5		5	
11	IN_3	5		5	
12	IN_4	5		5	
13	IN_5	5		5	
14	IN_6	5		5	
15	IN_D1	2		2	
16	IN_D2	2		2	
17	UPDATE_EVT				
18	BLOCK_ALM				
19	ALARM_SUM	8		8	
20	ACK_OPTION				2
21	ALARM_HYS				4
	Subtotals	50	2	50	19

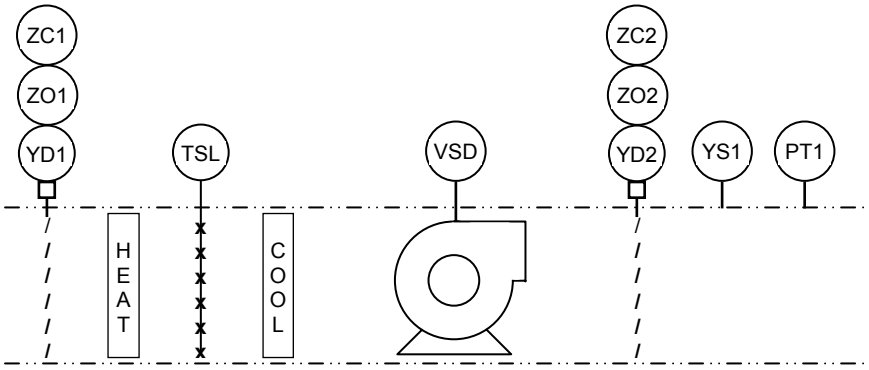
Index	Parameter	VIEW_1	VIEW_2	VIEW_3	VIEW_4
22	FLOW_DV_HI_PRI				1
23	FLOW_DV_HI_LIM				4
24	FLOW_DV_LO_PRI				1
25	FLOW_DV_LO_LIM				4
26	HEAT_DV_HI_PRI				1
27	HEAT_DV_HI_LIM				4
28	HEAT_DV_LO_PRI				1
29	HEAT_DV_LO_LIM				4
30	FT1_DV_HI_PRI				1
31	FT1_DV_HI_LIM				4
32	FT1_DV_LO_PRI				1
33	FT1_DV_LO_LIM				4
34	FT2_DV_HI_PRI				1
35	FT2_DV_HI_LIM				4
36	FT2_DV_LO_PRI				1
37	FT2_DV_LO_LIM				4
38	FLOW_DV_HI_ALM				
39	FLOW_DV_LO_ALM				
40	HEAT_DV_HI_ALM				
41	HEAT_DV_LO_ALM				
42	FT1_DV_HI_ALM				
43	FT1_DV_LO_ALM				
44	FT2_DV_HI_ALM				
45	FT2_DV_LO_ALM				
	Subtotals	0	0	0	40
	Totals	50	2	50	59

3.3 Variable Speed Drive Control

3.3.1 Overview

This application is a partial example of HVAC Air Handler control. The example is centered on the blower and it's Variable Speed Drive (VSD). YD1 is an inlet damper that is normally wide open, but closes if TSL detects a low temperature after the heating coil due to some failure. YD2 is an outlet damper that closes if YS1 detects smoke in the air duct, perhaps from failure to lubricate the fan bearings. PT1 senses the pressure in the air duct, usually in inches of water or 0.01 Bar. The VSD turns the fan at the speed necessary to maintain a set duct pressure. It is interlocked with the dampers so that it stops quickly if one of them closes. A complete example would include temperature controls.

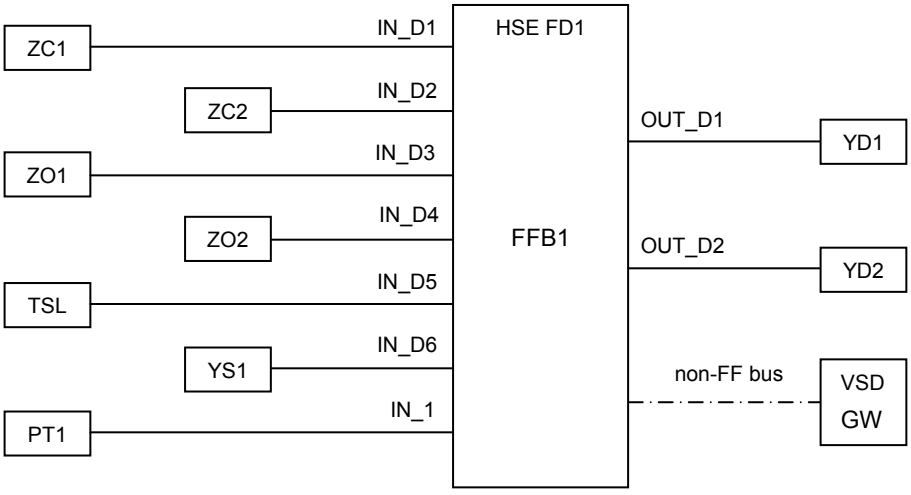
3.3.2 Process Diagram



3.3.3 Field Wiring

The field wiring is not H1 FOUNDATION fieldbus. It does not carry modulated AC transmissions. Similarly, the instruments are not required to serve as ladder rungs for maintenance technicians and they may use non-standard signal levels. Not shown is a device located on or near the air duct that connects to FOUNDATION fieldbus HSE fieldbus. It also connects to the non-Fieldbus instruments and is a gateway to whatever communication method that is used by the VSD vendor.

3.3.4 Function Block Diagram



FFB1 is linked to the process devices as shown above. It is shown as a field device FD1 and not as a linking device because it has no H1 ports. The labeled input and output parameters are not linked to any FF device. Instead, the bell wire is hooked to screw terminals on FD1 and the converted values are shown to other FF devices with status in the I/O parameters. The VSD connection is not labeled because it is proprietary, and so is invisible to FF devices.

The block algorithm may be user configurable, but is more likely to be developed by the HVAC vendor. The major function is to act as a gateway for the VSD command and status values. These would be displayed in many contained FFB variables. The individual discrete and analog I/O points must also be converted. If the external devices have no status information, they can be displayed in contained variables. The HSE Field Device may also have other I/O as required to control temperature, humidity, CO2, etc. or detect clogged filters, vibration, etc.

The block parameter list below contains a set of parameters for a standard SP and Remote Cascade. This applies to the duct pressure setpoint. If this value never changes, it can be made into one simple contained value. If a host computer does not ever set the setpoint, then the 3 remote cascade parameters can be removed. But if a host does set the value of SP, it is much safer to use the cascade initialization handshake. This forces the host software to look at the device before it writes a value. It is essential if the host has an integrating controller that calculates the value to be written.

Standard alarms are provided for IN_1, the duct pressure input. There is a deviation alarm and one level of absolute alarm. There may be alarms without any standard parameters for drive faults and interlock events.

The only supported modes are O/S, Auto and Rcas. There are no outputs that can be set in a Manual mode.

If the block has only Contained parameters, none of the information can be linked to other FF devices.

3.3.5 Block Access for FFB1

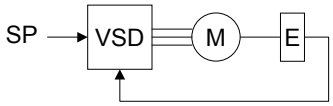
The access table defines the required parameters.

Index	Parameter	VIEW_1	VIEW_2	VIEW_3	VIEW_4
1	ST_REV	2	2	2	2
2	TAG_DESC				
3	STRATEGY				2
4	ALERT_KEY				1
5	MODE_BLK	4		4	
6	BLOCK_ERR	2		2	
7	ALGORITHM_SEL				4
8	CONTENTS_REV				4
9	IN_1	5		5	
10	IN_D1	2		2	
11	IN_D2	2		2	
12	IN_D3	2		2	
13	IN_D4	2		2	
14	IN_D5	2		2	
15	IN_D6	2		2	
16	OUT_D1	2		2	
17	OUT_D2	2		2	
18	SHED_OPT_1				1
19	SP_1	5		5	
20	RCAS_IN_1			5	
21	RCAS_OUT_1			5	
22	SP_RATE_DN_1				4
23	SP_RATE_UP_1				4
24	SP_HI_LIM_1		4		
25	SP_LO_LIM_1		4		
26	UPDATE_EVT				
27	BLOCK_ALM				
28	ALARM_SUM	8		8	
29	ACK_OPTION				2
30	ALARM_HYS				4
31	IN_1_HI_PRI				1
32	IN_1_HI_LIM				4
33	IN_1_LO_PRI				1
34	IN_1_LO_LIM				4
35	IN_1_DV_HI_PRI				1
36	IN_1_DV_HI_LIM				4
37	IN_1_DV_LO_PRI				1
38	IN_1_DV_LO_LIM				4
	Subtotals	42	10	52	48

Index	Parameter	VIEW_1	VIEW_2	VIEW_3	VIEW_4
39	HAND_OFF_AUTO	1		1	
40	START	1		1	
41	STOP	1		1	
42	CLEAR_FAULT	1		1	
43	ACTIVE_FAULT	1		1	
44	DRIVE_STATUS	1		1	
45	FREQUENCY			4	
46	VOLTAGE			4	
47	CURRENT			4	
48	POWER			4	
49	TORQUE			4	
50	SPEED	4		4	
51	VOLTS/HERTZ			4	
52	BUS_VOLTS			4	
53	POWER_FACTOR			4	
54	HEATSINK_TEMP			4	
55	ACCEL_RATE				4
56	DECEL_RATE				4
57	MIN_FREQUENCY				4
58	MAX_FREQUENCY				4
59	MAX_MOTOR_AMPS				4
60	CURRENT_LIMIT				4
61	BRAKING_TIME				4
62	BRAKING_VOLTS				4
63	RESTART_TRIES				4
64	RESTART_DELAY				4
65	COMPENSATION				4
66	IN_1_HI_ALM				
67	IN_1_LO_ALM				
68	IN_1_DV_HI_ALM				
69	IN_1_DV_LO_ALM				
	Subtotals	10	0	46	44
	Totals	52	10	98	92

A Variable Speed Drive (also known as a Adjustable Speed Drive in IEC 61800-2, Variable Frequency Drive, Inverter, Adjustable Speed Drive and Adjustable Frequency Drive etc.)

3.3.6 Additional thoughts on Fieldbus Drives



A motor is traditionally not seen as a process control instrument and therefore the adoption of FOUNDATION fieldbus in a starter or drive still has not happened. However, pumps and fans/blowers are increasingly taking the place of control valves and dampers/louvers and motors also power conveyor belts and other equipment in process plants. Moreover, process plants have many motors for agitators etc. Since process plants only want a single bus technology in their plant, at least as far as possible, Fieldbus drives and starters are highly desirable and there are obviously plenty of applications for it. The versatile communication mechanisms offered by FOUNDATIONfieldbus include client/server (acyclic) and scheduled (equidistant, isosynchronous) publisher/subscriber (cyclic) communications as well as report distribution.

The FOUNDATION fieldbus technology essentially consists of two parts: communication networking and a function block programming language for building control strategies. The FFB comes in handy when there is a need to deviate from these two aspects:

- Incorporate devices on foreign networks

- Incorporate devices with foreign programming languages

E.g. the FFB is useful in a gateway to incorporate data from a bus technology such as DeviceNet, and to incorporate a language such as one of the IEC 61131-3 languages e.g. structured text. The drive application makes use of the FFB in both of these capacities. One block is used for network interfacing and another block is used for discrete interlocks. Ideally a native FOUNDATION fieldbus drive shall be used but when an existing drive using other bus technology shall be interfaced a gateway with FFB is necessary. For the purpose of comparison a possible arrangement for a drive transducer block is also studied.

The parameters of an Allen-Bradley 1336 PLUS II were studied as a typical example. This drive has some 334 parameters but many are related to local display and hardwired I/O. Since the purpose of Fieldbus is to eliminate hardwired I/O and local operation instead interfacing data using networking all the hardwiring and conversion related parameters were omitted in the gateway flexible function block. There are also several values that are "internal" that may not be of interest. This "internal" was arbitrarily decided by precedence set by standard function blocks. E.g. deviation/error and integral contribution values in a PID are not available as parameters in the PID block. A drive may have a built in process controller but to fully utilize the flexibility of the FOUNDATION fieldbus function block programming language this should be done in a standard function block and therefore the process control parameters are not exposed in the flexible function block. "Logging" of last faults is not done in Fieldbus devices but in the host computer.

3.3.6.1 Flexible Function Block interface to Variable Speed Drive on foreign network

3.3.6.1.1 Setpoint selection (desired speed/frequency)

In conventional drives the desired setpoint (speed/frequency) can be set in many different ways. E.g. by DI selection of pre-set values, 4-20 mA and other analog inputs, potentiometer and pulse etc. and of course by writing the setpoint via communication. In this case a FFB is used to interface to a foreign protocol that ultimately writes the setpoint to the drive. The FFB would receive the setpoint in percentage % e.g. from a PID block or possibly via an AO block in order to avoid problems in the cascade initialization.

These parameters are selected from the Allen-Bradley 1336 PLUS II drive based on parameter spreadsheet and manual downloaded from freely accessible web site.

3.3.6.1.2 Parameters

Parameters in addition to standard FFB parameters

OUTPUT_VOLTAGE	RO	Volt
%_OUTPUT_CURR	RO	%
%_OUTPUT_POWER	RO	%
CONTROL_SELECT	R/W	Enumerated
STOP_SELECT_1	R/W	Enumerated
BUS_LIMIT_EN	R/W	Enumerated
DC_HOLD_TIME	R/W	sec
DC_HOLD_LEVEL	R/W	%
RUN_ON_POWER_UP_ENABLE	R/W	Enumerated
RESET_RUN_TIME	R/W	sec
MINIMUM_FREQ	R/W	Hz
BASE_FREQUENCY	R/W	Hz
BASE_VOLTAGE	R/W	Volt
MAXIMUM_FREQ	R/W	Hz
MAXIMUM_VOLTAGE	R/W	Volt
OUTPUT_POWER	RO	kW
JOG_FREQUENCY	R/W	Hz
STOP_MODE_USED	RO	Enumerated
SKIP_FREQ_1	R/W	Hz
SKIP_FREQ_2	R/W	Hz
SKIP_FREQ_3	R/W	Hz
SKIP_FREQ_BAND	R/W	Hz
CURRENT_LIMIT	R/W	%
OVERLOAD_MODE	R/W	Enumerated
OVERLOAD_AMPS	R/W	Ampere
FLT_CLEAR_MODE	R/W	Enumerated
LINE_LOSS_FAULT_ENABLE	R/W	Enumerated
MOTOR_TYPE	R/W	Enumerated
SLIP_FLA	R/W	Hz
DWELL_FREQUENCY	R/W	Hz
DWELL_TIME	R/W	sec
PWM_FREQUENCY	R/W	kHz
ENCODER_PPR	R/W	"pulses per revolution"
START_BOOST	R/W	Volt
BREAK_FREQUENCY	R/W	Hz
BREAK_VOLTAGE	R/W	Volt
CLEAR_FAULT	R/W	Enumerated
STOP_SELECT_2	R/W	Enumerated
DC_BUS_VOLTAGE	RO	Volt
OUTPUT_CURRENT	RO	Ampere
S_CURVE_TIME	R/W	sec
S_CURVE_ENABLE	R/W	Enumerated
COMMON_BUS	R/W	Enumerated
DRIVE_STATUS_1	RO	BitEnumerated
DRIVE_ALARM_1	RO	BitEnumerated
this is the final value or something	RO	Hz
OUTPUT_FREQ	RO	Hz
DRIVE_DIRECTION	R/W	Enumerated

HEATSINK_TEMP	RO	degC
FIRMWARE_VER	RO	None
CURRENT_ANGLE	RO	Deg (angle)
SPEED_CONTROL	R/W	Enumerated
TRAVERSE_INC	R/W	sec
MAX_TRAVERSE	R/W	Hz
P_JUMP	R/W	Hz
BLWN_FUSE_FLT	R/W	Enumerated
CUR_LIM_TRIP_EN	R/W	Enumerated
RUN_BOOST	R/W	Volt
POWER_OL_COUNT	RO	%
RESET_RUN_TRIES	R/W	"Tries"
LOW_BUS_FAULT_ENABLE	R/W	Enumerated
MOTOR_MODE	RO	Enumerated
POWER_MODE	RO	Enumerated
FLT_MOTOR_MODE	RO	Enumerated
FLT_POWER_MODE	RO	Enumerated
FAULT_FREQUENCY	RO	Hz
FAULT_STATUS_1	RO	BitEnumerated
RATED_VOLTS	RO	Volt
RATED_CT_AMPS	RO	Ampere
RATED_CT_KW	RO	kW
MAXIMUM_SPEED	R/W	Hz
ENCODER_TYPE	R/W	Enumerated
MOTOR_POLES	RO	"Poles"
FLYING_START_TYPE	R/W	Enumerated
FSTART_FORWARD	R/W	Hz
FSTART_REVERSE	R/W	Hz
FREQ_LIM	R/W	Hz
CURRENT_LIM	R/W	%
TORQUE_LIM	R/W	Ampere
TORQUE_CURRENT	RO	Ampere
FLUX_CURRENT	RO	Ampere
SPEED_KP	R/W	None
SPEED_KI	R/W	None
SPEED_ADDER	RO	Hz
BOOST_SLOPE	R/W	None
RATED_AMPS	RO	Ampere
RATED_KW	RO	kW
FAULT_ALARMS_1	RO	Enumerated
MOTOR_NP_RPM	R/W	"RPM"
MOTOR_NP_HERTZ	R/W	Hz
MOTOR_NP_VOLTS	R/W	
MOTOR_NP_AMPS	R/W	
FLUX_AMPS_REF	R/W	Ampere
KP_AMPS	R/W	None
IR_DROP_VOLTS	R/W	Volt
SLIP_COMP_GAIN	R/W	None

RATED_VT_AMPS	RO	Ampere
RATED_VT_KW	RO	kW
FLUX_UP_TIME	R/W	sec
MOTOR_OL_FAULT_ENABLE	R/W	Enumerated
MOTOR_OL_COUNT	RO	%
VT_SCALING	R/W	Enumerated
GROUND_WARNING_ENABLE	R/W	Enumerated
DC_BUS_MEMORY	RO	Volt
SHEAR_PIN_FAULT_ENABLE	R/W	Enumerated
ADAPTIVE_I_LIM	R/W	Enumerated
LLOSS_RESTART	R/W	Enumerated
DRIVE_STATUS_2	RO	BitEnumerated
CNTRL_BOARD_REV	RO	None
SLIP_ADDER	RO	Hz
LINE_LOSS_MODE	R/W	Enumerated
TEMP_LIM	R/W	degC
MOTOR_THERM_FLT_ENABLE	R/W	Enumerated
DRIVE_ALARM_2	RO	BitEnumerated
MEAS_VOLTS	RO	Volt
ELAPSED_RUN_TIME	R/W	hour
ENC_COUNT_SCALE	R/W	None
ENCODER_COUNTS	R/W	"Counts"
FAULT_STATUS_2	RO	BitEnumerated
FAULT_ALARMS_2	RO	BitEnumerated
BUS_REGULATION_ENABLE	R/W	Enumerated
LOAD_LOSS_DETECT_ENABLE	R/W	Enumerated
LOAD_LOSS_LEVEL	R/W	%
LOAD_LOSS_TIME	R/W	sec
CURRENT_LMT_EN	R/W	Enumerated
TRAVERSE_DEC	R/W	sec
SYNC_TIME	R/W	sec
SYNC_LOSS_SEL	R/W	Enumerated
SYNC_LOSS_GAIN	R/W	None
SYNC_LOSS_TIME	R/W	sec
SYNC_LOSS_COMP	R/W	Volt
APPLICATION_STS	RO	BitEnumerated
RUN_ACCEL_VOLTS	R/W	%
SPEED_BRAKE_EN	R/W	Enumerated
LINE_LOSS_VOLTS	R/W	Volt
LOSS_RECOVER	R/W	Volt
RIDE_THRU_VOLTS	R/W	Volt
MIN_BUS_VOLTS	R/W	Volt
STABILITY_GAIN	R/W	None
MAX_BUS_VOLTS	R/W	Volt
MAX_ENC_COUNTS	R/W	
PHASE_LOSS_ENABLE	R/W	Enumerated
PHASE_LOSS_LIM	R/W	
PRECHARGE_FAULT_ENABLE	R/W	Enumerated

PWM_COMP_TIME	R/W	None
BREAK_FREQ	R/W	Hz

3.3.6.2 Standard Variable Speed Drive transducer block

This idea builds on the PS FF-902 and 903 documents. The suggested transducer block encapsulates the device specific characteristics, excluding the application specific characteristics that rightfully belong in function blocks.

3.3.6.2.1 Setpoint selection (desired speed/frequency)

In conventional drives the desired setpoint (speed/frequency) can be set in many different ways. E.g. by DI selection of pre-set values, 4-20 mA and other analog inputs, potentiometer and pulse etc. and of course by writing the setpoint via communication. In the FOUNDATION fieldbus function block diagram programming language setpoint originates from the CAS_IN and SP parameters in the AO block. The AO block setpoint is usually received in percentage % e.g. from a PID block. This is in the AO block converted to desired frequency range using XD_SCALE and subsequently passed to the transducer block. Acceleration and deceleration time is replaced by the setpoint rate of change in the AO block.

In traditional drives there are various alarms for current overload etc. In Fieldbus these become diagnostics since transducers do not have alarm parameters. Depending on the implementation "device failure" can be set in AO READBACK parameter status to indicate to control strategy that the drive is not OK. Further details are indicated using the XD_ERROR parameter.

Manufacturers can from this standard block create enhanced transducer blocks adding parameters e.g. to measure motor winding temperatures etc. In traditional drives many commands are given through hardwired analog, discrete and pulse inputs. In a Fieldbus drive commands shall be given over the network through the AO block. However, drives are not precluded from having physical inputs like in the past that then may be reflected as parameters in the transducer block. Similarly in traditional drives analog, discrete and pulse outputs indicate drive status. In a Fieldbus drive status shall be indicated over the network by transducer block parameters and through status in the AO block.

3.3.6.2.2 Parameters

Parameters in addition to standard transducer block parameters

TRD_native	Units	R/W	Mode
OUTPUT_VOLTAGE	Volt	RO	
%_OUTPUT_CURR	%	RO	
%_OUTPUT_POWER	%	RO	
CONTROL_SELECT	Enumerated	R/W	
STOP_SELECT_1	Enumerated	R/W	
DC_HOLD_TIME	sec	R/W	
DC_HOLD_LEVEL	%	R/W	
RESET_RUN_TIME	sec	R/W	
MINIMUM_FREQ	Hz	R/W	
BASE_FREQUENCY	Hz	R/W	
BASE_VOLTAGE	Volt	R/W	
MAXIMUM_FREQ	Hz	R/W	OOS
MAXIMUM_VOLTAGE	Volt	R/W	
OUTPUT_POWER	kW	RO	
JOG_FREQUENCY	Hz	R/W	
STOP_OPTS_USED	Enumerated	RO	
SKIP_FREQ_1	Hz	R/W	
SKIP_FREQ_2	Hz	R/W	
SKIP_FREQ_3	Hz	R/W	
SKIP_FREQ_BAND	Hz	R/W	
CURRENT_LIMIT	%	R/W	
OVERLOAD_OPTS	Enumerated	R/W	
OVERLOAD_AMPS	Ampere	R/W	
FLT_CLEAR_OPTS	Enumerated	R/W	
MOTOR_TYPE	Enumerated	R/W	
SLIP_FLA	Hz	R/W	
DWELL_FREQUENCY	Hz	R/W	
DWELL_TIME	sec	R/W	
PWM_FREQUENCY	kHz	R/W	OOS
ENCODER_PPR	"pulses per revolution"	R/W	
START_BOOST	Volt	R/W	
BREAK_FREQUENCY	Hz	R/W	
BREAK_VOLTAGE	Volt	R/W	
CLEAR_FAULT	Enumerated	R/W	
STOP_SELECT_2	Enumerated	R/W	
DC_BUS_VOLTAGE	Volt	RO	
OUTPUT_CURRENT	Ampere	RO	
S_CURVE_TIME	sec	R/W	
FINAL_VALUE	Hz	RO	
OUTPUT_FREQ	Hz	RO	
DRIVE_DIRECTION	Enumerated	R/W	
HEATSINK_TEMP	degC	RO	
FIRMWARE_VER	None	RO	
CURRENT_ANGLE	Deg (angle)	RO	
SPEED_CONTROL	Enumerated	R/W	OOS
TRAVERSE_INC	sec	R/W	
MAX_TRAVERSE	Hz	R/W	

P_JUMP	Hz	R/W	
BLWN_FUSE_FLT	Enumerated	R/W	
CUR_LIM_TRIP_EN	Enumerated	R/W	
RUN_BOOST	Volt	R/W	
POWER_OL_COUNT	%	RO	
RESET_RUN_TRIES	"Tries"	R/W	
MOTOR_OPTS	Enumerated	RO	
POWER_OPTS	Enumerated	RO	
FLT_MOTOR_OPTS	Enumerated	RO	
FLT_POWER_OPTS	Enumerated	RO	
FAULT_FREQUENCY	Hz	RO	
RATED_VOLTS	Volt	RO	
RATED_CT_AMPS	Ampere	RO	
RATED_CT_KW	kW	RO	
MAXIMUM_SPEED	Hz	R/W	
ENCODER_TYPE	Enumerated	R/W	OOS
MOTOR_POLES	"Poles"	RO	
FLYING_START_TYPE	Enumerated	R/W	
FSTART_FORWARD	Hz	R/W	
FSTART_REVERSE	Hz	R/W	
FREQ_LIM	Hz	R/W	
CURRENT_LIM	%	R/W	
TORQUE_LIM	Ampere	R/W	
TORQUE_CURRENT	Ampere	RO	
FLUX_CURRENT	Ampere	RO	
SPEED_KP	None	R/W	
SPEED_KI	None	R/W	
SPEED_ADDER	Hz	RO	
BOOST_SLOPE	None	R/W	
RATED_AMPS	Ampere	RO	
RATED_KW	kW	RO	
MOTOR_NP_RPM	"RPM"	R/W	OOS
MOTOR_NP_HERTZ	Hz	R/W	OOS
MOTOR_NP_VOLTS		R/W	OOS
MOTOR_NP_AMPS		R/W	OOS
FLUX_AMPS_REF	Ampere	R/W	
KP_AMPS	None	R/W	
IR_DROP_VOLTS	Volt	R/W	
SLIP_COMP_GAIN	None	R/W	
RATED_VT_AMPS	Ampere	RO	
RATED_VT_KW	kW	RO	
FLUX_UP_TIME	sec	R/W	
MOTOR_OL_COUNT	%	RO	
VT_SCALING	Enumerated	R/W	OOS
DC_BUS_MEMORY	Volt	RO	
ADAPTIVE_I_LIM	Enumerated	R/W	
LLOSS_RESTART	Enumerated	R/W	

CNTRL_BOARD_REV	None	RO	
SLIP_ADDER	Hz	RO	
LINE_LOSS_OPTS	Enumerated	R/W	
TEMP_LIM	degC	R/W	
MEAS_VOLTS	Volt	RO	
ELAPSED_RUN_TIME	hour	R/W	
ENC_COUNT_SCALE	None	R/W	
ENCODER_COUNTS	"Counts"	R/W	
LOAD_LOSS_LEVEL	%	R/W	
LOAD_LOSS_TIME	sec	R/W	
CURRENT_LMT_EN	Enumerated	R/W	
TRAVERSE_DEC	sec	R/W	
SYNC_TIME	sec	R/W	
SYNC_LOSS_GAIN	None	R/W	
SYNC_LOSS_TIME	sec	R/W	
SYNC_LOSS_COMP	Volt	R/W	
APPLICATION_STS	BitEnumerated	RO	
RUN_ACCEL_VOLTS	%	R/W	
LINE_LOSS_VOLTS	Volt	R/W	
LOSS_RECOVER	Volt	R/W	
RIDE_THRU_VOLTS	Volt	R/W	
MIN_BUS_VOLTS	Volt	R/W	
STABILITY_GAIN	None	R/W	
MAX_BUS_VOLTS	Volt	R/W	
MAX_ENC_COUNTS		R/W	
PHASE_LOSS_LIM		R/W	
PWM_COMP_TIME	None	R/W	
BREAK_FREQ	Hz	R/W	
DRIVE_OPTS			
FAULT_OPTS			
XD_ERROR			

3.3.6.3 Reference

IEC 61800-2 AC Adjustable Speed Drive - general requirements

3.4 Four Discrete Valve Control

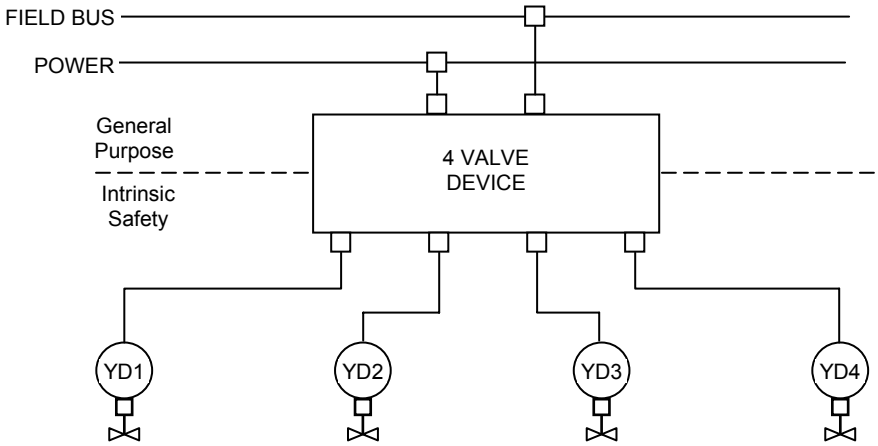
3.4.1 Overview

This application is concerned with a basic part of batch processing. Block valves are used to change the configuration of the process unit relative to its piping. When many block valves are used it becomes important to transfer information on the bus efficiently, to conserve bandwidth. A single fieldbus device is described that uses conventional intrinsically safe wiring to connect to four valves (or other devices) and their limit sensors. The Multi-Variable Optimization is used to report the status of all four devices in a single message. A host computer sends commands to one valve at a time. It is possible to broadcast a shutdown command in one message that is received by all devices on that bus that are configured to hear it.

3.4.2 Process Diagram

No diagram is required. The use of the valves is generic.

3.4.3 Field Wiring

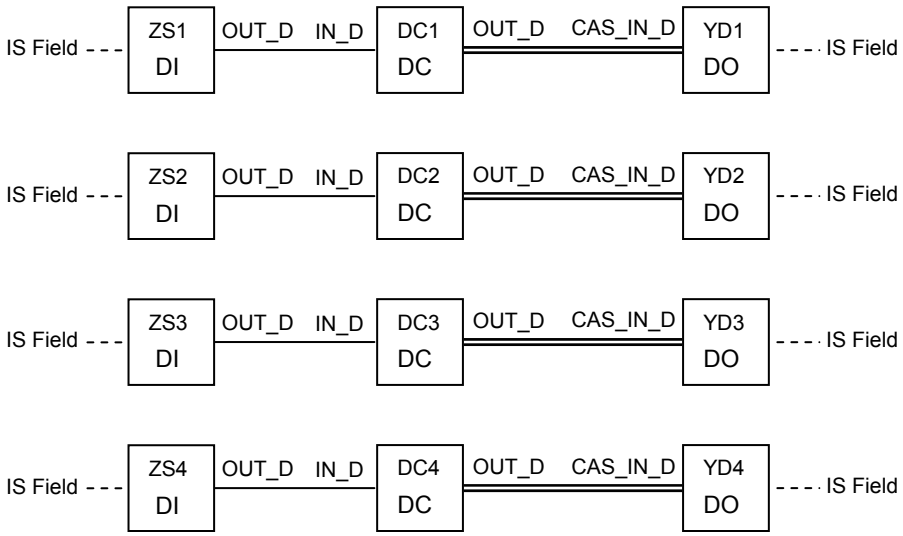


The wiring from the device to the valves may be conventional intrinsically safe multicore cable. It could also be some simple discrete fieldbus suitable to the application. The device contains barrier or other fieldbus hardware.

Separate power wiring is shown to allow the number of devices on the fieldbus to increase without being limited by the power required to run the device and four valves. Power is not necessarily bussed or run alongside of the fieldbus.

The contents of the MVC are broadcast as a Report at some interval that is a multiple of the bus macrocycle. The multiplier may be one, depending on the number of devices on the bus and the other work being done by the host.

3.4.4 Function Block Diagram



The device contains four instances of a standard discrete control loop, each consisting of a multi-state discrete input for the position sensors, a device control block (see FF-892), and a multi-state discrete output for the actuator. Three state actuators may be used. The twelve blocks share a common OD so that an MVC can contain values from all of them. The MVC list is shown below. There is no FFB, so there is no Block Access list.

3.4.5 MVC Object List

Index	Parameter	Size
	The following are from DC1	
1	ST_REV	2
2	MODE_BLK	4
3	BLOCK_ERR	2
4	IN_D	2
5	SP_D	2
6	RCAS_OUT_D	2
7	OUT_D	2
8	SHUTDOWN_D	2
9	DC_STATE	1
10	FAIL	2
	The following are from DC2	
11	ST_REV	2
12	MODE_BLK	4
13	BLOCK_ERR	2
14	IN_D	2
15	SP_D	2
16	RCAS_OUT_D	2
17	OUT_D	2
18	SHUTDOWN_D	2
19	DC_STATE	1
20	FAIL	2
	Subtotals	42

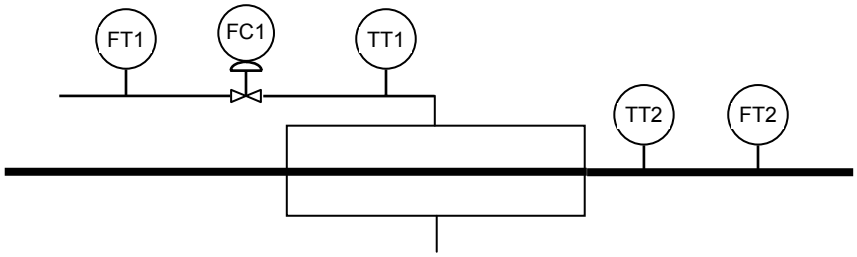
Index	Parameter	Size
	The following are from DC3	
21	ST_REV	2
22	MODE_BLK	4
23	BLOCK_ERR	2
24	IN_D	2
25	SP_D	2
26	RCAS_OUT_D	2
27	OUT_D	2
28	SHUTDOWN_D	2
29	DC_STATE	1
30	FAIL	2
	The following are from DC4	
31	ST_REV	2
32	MODE_BLK	4
33	BLOCK_ERR	2
34	IN_D	2
35	SP_D	2
36	RCAS_OUT_D	2
37	OUT_D	2
38	SHUTDOWN_D	2
39	DC_STATE	1
40	FAIL	2
	Subtotals	42
	Totals	84

3.5 Extended PID with Autotuner

3.5.1 Overview

This application is a multi-phase heat exchanger. It requires an advanced self-tuning PID, adaptive gain and calculated feed-forward values. The advanced PID is contained in and FFB and Multi-Variable Optimization is used to transfer parameters to and from another FFB that contains the adaptive calculations.

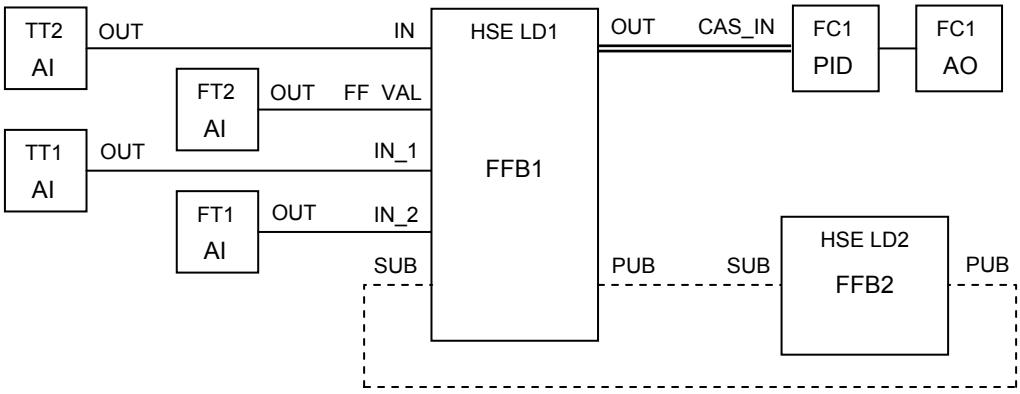
3.5.2 Process Diagram



3.5.3 Field Wiring

The field wiring has no impact on this application.

3.5.4 Function Block Diagram



FFB1 is a standard PID with some additional parameters. It behaves like a PID, but it has additional calculations. FFB2 contains calculations that adaptively modify the tuning and feedforward parameters of FFB1. FFB1 and FFB2 are linked by MVC object lists that are published once during each macrocycle. The additional parameters for FFB1 are listed below, along with the MVC lists.

If the autotune algorithm runs in a host, it is unlikely that the host can publish an MVC link. Instead, the FMS Write Variable List capability can be used to write data to a set of unrelated objects.

3.5.5 Block Access for FFB1

The access table defines the required parameters.

Index	Parameter	VIEW_1	VIEW_2	VIEW_3	VIEW_4
1	Standard PID	43	43	83	104
2	GAIN_IN			5	
3	RESET_IN			5	
4	RATE_IN			5	
5	REL_GAIN_IN			5	
6	DTIME_IN			5	
7	PV_FTIME_IN			5	
	Subtotals	43	43	113	104

Index	Parameter	VIEW_1	VIEW_2	VIEW_3	VIEW_4
8	IMPULSE_IN			5	
9	FLOAT1_OUT			5	
10	FLOAT2_OUT			5	
11	PV_OUT			5	
12	SP_OUT			5	
13	MODE_OUT_D			2	
	Subtotals	0	0	27	0
	Totals	43	43	140	104

3.5.6 MVC Lists

The table defines the contents of the published containers.

Index	Parameter	Size
	The following are from FFB1	
1	OUT	5
2	PV_OUT	5
3	SP_OUT	5
4	FF_VAL	5
5	IN_1	5
6	IN_2	5
7	MODE_OUT_D	2
	Total	32

Index	Parameter	Size
	The following are from FFB2	
1	GAIN_OUT	5
2	RESET_OUT	5
3	RATE_OUT	5
4	REL_GAIN_OUT	5
5	DTIME_OUT	5
6	PV_FTIME_OUT	5
7	IMPULSE_OUT	5
8	FLOAT1_OUT	5
9	FLOAT2_OUT	5
	Total	45

3.6 Fermentation Zymolysis Control

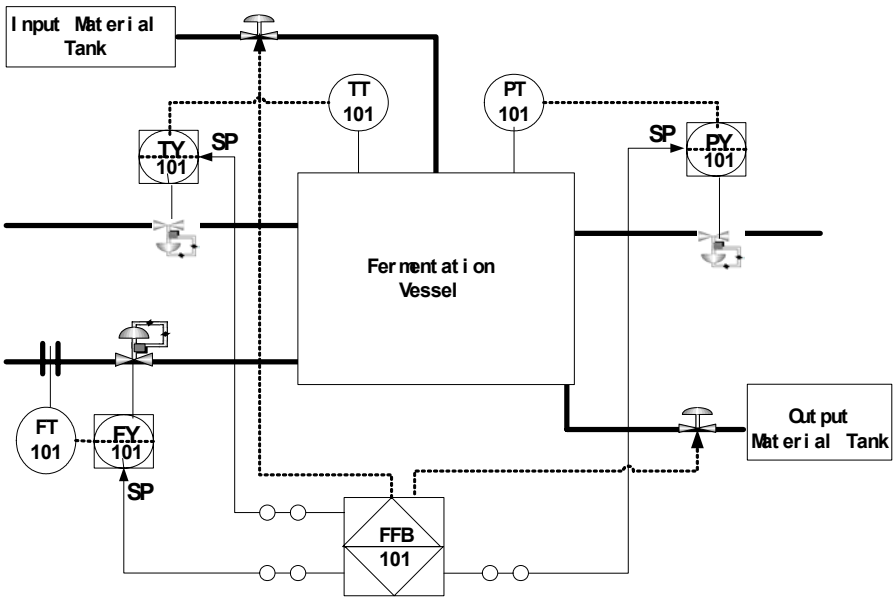
3.6.1 Overview

This application is a fermenting process that vitamin C with four analog points and four discrete points controlled by two sets of switch and three PID control loops.

The fermenting process of vitamin C experiences four phases: input raw material, ferment it, maintain pressure, and output material. The switch is used to trigger input or output of the material alternatively, the three PID control loops are used to control the temperature, pressure and flux respectively. Through the parameter SETPOINT of the PID function block, the application can control the process according to the fermentation diagram.

A HSE field device containing a FFB application is used; other FF H1 field devices with three PID loops are used. A gateway called linking device is used to integrate the HSE and H1 networks. The algorithm programmed by IEC 61131-3 language is contained in a Domain of the HSE's FBAP VFD. This application uses the Fixed OD FFB. This block has a predefined OD that is described in the fixed DD and the Capabilities file for the VFD that contains it. The algorithm used by the block is loaded into a Domain in HSE device.

3.6.2 Process Diagram



3.6.3 Description

The block has two discrete inputs for the states of input and output, two discrete outputs for the switches, one analog input for the measurement of PH and three analog outputs for the CAS_IN parameter of the three PID function blocks. When the input or output state is true, the corresponding digital output is turned on. The SETPOINTS of the three PID control loops are not constant. They vary in different fermenting phases according to the fermentation diagram.

3.6.3.1 Supported Modes

O/S, Manual and Auto. In Manual mode the outputs may be turned on and off regardless of the state of the input.

3.6.3.2 Alarm Types

Defined by the DD for this block.

3.6.3.3 Mode Handling

Standard.

3.6.3.4 Status Handling

Standard discrete output status. A transition to Bad input status changes Auto to Manual. The standard alarm for non-operator mode change is generated. A transition to Good input status does not change the mode.

3.6.3.5 Initialization

Standard.

3.6.3.6 Power Failure Recovery

Retain and restore the output states.

3.6.4 Block Access

The access table defines the required parameters.

Index	Parameter Mnemonic	VIEW_1	VIEW_2	VIEW_3	VIEW_4
1	ST_REV	2	2	2	2
2	TAG_DESC				
3	STRATEGY				2
4	ALERT_KEY				1
5	MODE_BLK	4		4	
6	BLOCK_ERR	2		2	
7	ALGORITHM_SEL				4
8	CONTENTS_REV				4
	Subtotals	8	2	8	13

Index	Parameter Mnemonic	VIEW_1	VIEW_2	VIEW_3	VIEW_4
9	DI_INPUT	2			
10	DI_OUTPUT	2			
11	DO_SW1	2			
12	DO_SW2	2			
13	AI_PH	5		5	
14	AO_TEMP	5		5	
15	AO_PRES	5		5	
16	AO_FLUX	5		5	
	From left column	8	2	8	13
	Totals	36	2	28	13

3.7 Distillation Startup and Shutdown

3.7.1 Overview

This application controls the startup and shutdown sequences for a simple distillation column with cascade control of steam flow and reflux flow based on tray temperatures.

The control is done in two flexible function blocks in one HSE device. The device has access to ten H1 field devices and three pumps. The Sequential FFB contains 24 steps. The Execution FFB has 17 steps to check initial conditions.

3.7.2 Description of the Hybrid Step

The unit of FFB standard logic and calculation functionality is the Hybrid Step.

The hierarchy looks like this:

Device
FFB
Hybrid Step
Statements
Expressions and Boolean Operators
Variables, Operators and Functions

A Fieldbus device contains, among many other things, one or more Flexible Function Blocks (FFB).

There are only two kinds of standard FFB that can execute Hybrid Steps:

1: The Execution FFB executes all of the Hybrid Steps within it at each FFB execution.

2: The Sequential FFB executes one or more Hybrid Steps within it at each FFB execution, as determined by the linkage of the steps. It is intended to be very similar to SFC execution.

The Execution or Sequential FFB contains one or more Hybrid Steps.

3.7.2.1 Statements

The Hybrid Step contains Statements.

There is a selection of ten statements:

1. Name - Assigns a User name to a Hybrid Step. Names must be unique within one FFB. The name is used in HMI messages from the step. The name also represents the state of the step when enclosed in quotes and used as a variable name in an expression.

2. Set - Contains an expression that turns the step on if it evaluates to true. This statement is not used in a Sequential FFB unless a set of parallel steps has a single transition under the bar.

3. Clear - Contains an expression that turns the step off if it evaluates to true. If this statement is not present then the step is turned off when the Set statement evaluates to false. It corresponds to the Transition Condition in a Sequential FFB.

4. Previous - Contains the names of steps that must be active and done in order to allow this step to become active with Set and deactivate the named steps. It corresponds to the parallel bar under a set of parallel steps. Not used if the only previous step is the one before this one. It is not used in an Execution FFB.

5. Next - Contains the name of a step and an expression, as <name> IF <expression>. If the expression is true then the named step is activated and this step is deactivated. More than one Next statement may be in the Hybrid Step, where they form a list that is evaluated in order when the block is done. If none of the listed expressions are true, the following step becomes active. It corresponds to the single bar with multiple transitions following an SFC step, but has deterministic behavior. The AND operator may be used with step names to start multiple steps, corresponding to an opening parallel bar. Not used if the only next step is the step following this one. It is not used in an Execution FFB.

6. Watchdog - Contains a time expression (h, m, s) defining the value of the execution timer that will cause a Watchdog Timeout alarm to be generated. If triggered, the alarm clears when the step is turned off.

The following are Action statements:

7. Rise - Contains a list of expressions to be evaluated when the step changes state to ON.

8. On - Contains a list of expressions to be evaluated when the state of the step is ON.

9. Fall - Contains a list of expressions to be evaluated when the step changes state to OFF.

10. Off - Contains a list of expressions to be evaluated when the state of the step is OFF. It can not be used in a Sequential FFB, because an inactive step is not evaluated.

3.7.2.2 Expressions

A Statement contains one or more expressions, as described above.

Expressions may be combined in statements using the AND or OR operators.

An expression contains operators and variables. An evaluated expression has a value that may be assigned to a named variable. A named variable may be local or remote. A local variable must be in the list of variables that is local to the FFB that contains the step. These variables may be in the VFD Object Dictionary and visible on Fieldbus, or not.

Remote variables in other function blocks, possibly in other devices, require communication functions to read and write values. The remote variable may be linked to the FFB containing the step. In this case, the local name of the link object is used. An Input link will read, and an Output link will write to a remote value. If the remote variable is not linked to the FFB, then communication requires that a client/server relationship with the remote VFD containing the variable be set up before FFB execution begins.

3.7.2.3 Functions

A communication function requires a client/server read/write connection to the VFD containing the Tag. The following communication functions may be used:

readmode(Tag) - the value of this function is set to the value of the actual mode. Converts the mode bitstring to a number representing the highest priority (active) mode.

readbits(Tag.Parameter) - the value of this function is set to the value of the Tag.Parameter. Fails if the parameter data type is not bitstring. Converts bitstring 8 and 16 to bitstring 32. Fails if status is bad if type Input or Output. If type is Contained, fails if tag block mode is O/S.

readstring(Tag.Parameter) - the value of this function is set to the value of the Tag.Parameter. Fails if the parameter data type is not visible or octet string. Converts blank filled visible strings to null terminated strings, may handle Unicode. Fails if status is bad if type Input or Output. If type is Contained, fails if tag block mode is O/S.

readvalue(Tag.Parameter) - the value of this function is set to the value of the Tag.Parameter. Fails if the parameter data type is string. Converts all parameter values to Float. Only evaluates once if it is the initialValue of a ramp function. Fails if status is bad if type is Input or Output. If type is Contained, fails if tag block mode is O/S.

readstatus(Tag.Parameter) - the value of this function is set to the status of the Tag.Parameter. Fails if the parameter data type is not Input or Output. Converts the status bits to a number in the range 1 to 4.

readsubstatus(Tag.Parameter) - the value of this function is set to the substatus of the Tag.Parameter at the FFB evaluation rate divided by s if the action is true. Fails if the parameter data type is not Input or Output. Converts the substatus bits to a number in the range 1 to 16.

readlimitstatus(Tag.Parameter) - the value of this function is set to the limit status of the Tag.Parameter. Fails if the parameter data type is not Input or Output. Converts the limit status bits to a number in the range 1 to 4.

setvalue(Tag.Parameter, value) - writes the value to Tag.Parameter if the Action is true. Fails if the parameter is read-only.

setmode(Tag, mode) - writes the mode to Tag.Parameter if the Action is true. Writes to the Target mode and suspends further evaluation of the step's actions until the Actual mode equals the Target. This is necessary because the block being written must evaluate before the Actual mode changes in order to let further writes succeed. Fails if the mode is not permitted.

rampvalue(Tag.Parameter, target, rate) - the function acts to write values to the Parameter that move it to the target value at the specified rate. After it has written a value equal to the target, writing stops and the function begins reading the Parameter. The value of the function becomes True if the values match, and further communication stops. Fails if the Parameter does not match after ten reads.

exprampvalue(Tag.Parameter, target, rate) - as above but the rate is an exponential time constant.

The following functions are useful for steps in Sequential FFBs:

logstep - sends an information report that silently logs timestamp, FFB tag, "Starting" <step name>.

showstep - sets an alarm with the above information, and clears the alarm when the step clears.

opaction("string", variable) - like showstep, but adds "string" after the step name and has a return value which is placed in variable.

ontime - Returns the elapsed time in the step, in seconds.

The following functions are useful for local variables within an FFB:

status(variable) - the value of this function is set to the status of the named variable. Fails if the variable data type is not Input or Output. Converts the status bits to a number in the range 1 to 4.

substatus(variable) - the value of this function is set to the substatus of the variable. Fails if the variable data type is not Input or Output. Converts the substatus bits to a number in the range 1 to 16.

limitstatus(variable) - the value of this function is set to the limit status of the variable. Fails if the variable data type is not Input or Output. Converts the limit status bits to a number in the range 1 to 4.

waitfor(hms, expression) - the value of this function is false until the expression has been true for hms time. The timer resets when the step turns on or the expression is false.

within(variable, target, tolerance) - the value of this function is true if the value is within tolerance of the target. The magnitude of the difference is used for a high or low comparison.

ramp(variable, target, rate) - the function acts to write values to the variable that move it to the target value at the specified rate. The value of the function becomes true when the present value equals the target.

expramp(variable, target, rate) - the function acts to write values to the variable that move it to the target value at an exponential rate, using the specified rate as the time constant. The variable is set equal to the target after a time equal to six times the time constant has passed. The value of the function becomes true/1 when the present value equals the target.

3.7.2.4 Evaluation Rules

The FFB block mode controls evaluation of the steps. Evaluation is normal for the FFB in Auto mode, but stops in Manual or O/S mode. Any step may be turned on or off when the block is not in Auto mode. This capability may be used to test Execution steps or change the overall state of the sequence in a Sequential FFB. If the state of a step is changed while in Manual, the appropriate Rise or Fall action will be evaluated first when the mode is changed to Auto. If the mode was O/S then the previous state is set to the present state, so Rise and Fall will not work when the mode is changed to Auto.

The Sequential FFB will turn on the Initial step if no step is active, in any mode except O/S.

The expressions in a statement are evaluated in order. An expression may suspend evaluation while it waits for a reply or an event. The last expression to be evaluated determines the Boolean value of the statement.

The statements are evaluated in order, except that Clear precedes the actions in an Evaluation Step, so that Set and Clear cause no change to the state of the step if both are True.

All functions are evaluated in the order they are encountered in an action, set or clear statement.

All Tag.Parameter and Tag mode functions that reference other VFDs require a client/server communication connection to the VFD containing the tag, as well as read or write services as appropriate. These functions fail if the communication request returns an error response or if communication fails. All write functions fail if Tag.GRANT_DENY has the Program Denied bit set. Other failure conditions may be specified for a specific function. If any function fails, an alarm is generated with the FFB tag and step name. The function is retried at a five second interval, and clears the alarm when the function no longer fails. Further evaluation of the active step is suspended, except for the watchdog timer, while the failure is not cleared.

The Boolean functions accept numeric as well as Boolean values. A numeric value is True if it is greater than zero, otherwise it is False.

3.7.2.5 Evaluation of Sequential steps

All steps are initially inactive except for the step designated by the name “Initial Step”.

When the Clear statement of the Initial Step becomes true and the Fall action is complete, then the process of passing control from one step to one or more other steps begins. The Next statements are evaluated once to determine the name of the next step. The AND operator may produce multiple step names; otherwise only one is selected. If there is no following step then the Initial Step becomes the next step. During a single evaluation of the FFB, the active step is turned off (becomes inactive) and the named next steps are turned on (activated).

If a step has a list of names in the Previous statement, operation depends on the Set statement. When the Set statement is not present then the steps above a parallel bar have individual Clear statements. They enter a waiting state after Clear becomes true and Fall actions are complete. The step with the Previous statement turns off the previous steps and turns on its step when all named states are waiting.

If there is a Set statement, then the named steps can not have Clear statements. When all of the named steps are active and the Set statement becomes true, then the previous steps are turned off and this step turns on. If a previous step had a Fall statement, it needs to be handled.

When a step becomes activated by clearing a previous transition, start the timer for this step, check for a Watchdog statement and set its alarm time if present, and begin evaluation of the Rise action statement.

Once the Rise action is complete, evaluate the ON action. When one pass is complete, evaluate the Clear transition expression. Keep cycling between ON and Clear until the Clear expression is true, then just evaluate Fall until it is complete. When Fall is complete, evaluate any Next statements and then transition to the following step(s). Off delays may continue to run until timed out, otherwise there is no reason to evaluate this step any longer. There is no OFF action.

3.7.3 Sequential Control

The following are hybrid steps in the Sequential FFB

3.7.3.1 Startup

Initial Step
Name = Stopped
Rise = showstep;
Clear = Start switch

Initialize Equipment
Name = Initialize
Rise = opaction(“Do initialization checklist”, Didit)
Clear = Didit
Next = CheckEquip IF PotLevel>30

Raise Pot Level to 35% with Feed
Name = Fill Pot
Watchdog = 5M
Rise = showstep; setmode(FeedFC,Man); FeedPump=ON
ON = ramp(FeedFC.Out,30,1)
Clear = PotLevel>35

Stop Feed Flow
Name = Stop Filling Pot
Watchdog = 1M
Rise = showstep
ON =
Fall = FeedPump=OFF
Clear = ramp(FeedFC.Out,-5,1) AND FeedFlow<0.2

Verify that equipment is ready
Name = Check Equipment
Watchdog = 1M
Rise = showstep
Clear = EquipCheck

Verify that condenser is set up
Name = Condenser Setup
Rise = opaction(“Turn on condenser water”, Didit)
Clear = Didit

Open steam valve to 35%
Name = Begin Steam
Watchdog = 1M
Rise = showstep; setmode(SteamFC,Man)
Clear = ramp(SteamFC.Out,35,1) AND SteamFlow>20

Wait for Tray 2 > 95 Degrees
Name = Begin Vapor
Watchdog = 10M
Rise = showstep; setmode(StripperTC,Auto)
Clear = TrayTempTwo>95

Close steam cascade to tray 2 and stabilize
Name = Cascade Steam
Watchdog = 2M
Rise = showstep; setmode(SteamFC,Cas); setvalue(StripperTC.SP,105)
Clear = wait(1M,SteamFlow>20)

Start feed pump, auto, 4 L/M, pot level on auto at 50%
Name = Begin Feed
Watchdog = 1M
Rise = showstep; setmode(PotLC,Auto); setmode(FeedFC,Auto); setvalue(PotLC.SP,50);
FeedPump=ON; BottomsPump=ON
Clear = ramp(FeedFC.SP,4,0.1) AND FeedFlow>3.9

Wait for drum to rise 10%
Name = Begin Condensing
Watchdog = 10M
Rise = showstep; set Temp01=RefluxLevel
Clear = (RefluxLevel-Temp01)>10 OR RefluxLevel>80

Turn on reflux pump, set RefluxLC to auto, set RefluxFC to auto at 1 L/m
Name = Begin Reflux
Watchdog = 1M
Rise = showstep; setmode(RefluxLC,Auto); setmode(RefluxFC,Auto); setvalue(RefluxLC.SP,50); RefluxPump=ON
Clear = ramp(RefluxFC.SP,1,0.025) AND RefluxFlow>0.9

Stabilize reflux drum level
Name = Stabilize Reflux
Watchdog = 2M
Rise = showstep
Clear = wait(1,within(RefluxLevel, 50,2))

Ramp up reflux to 2 lpm
Name = Ramp Up Reflux
Watchdog = 1M
Rise = showstep; setmode(RectifierTC,Auto)
ON = ramp(RefluxFC.SP,2,0.025)
Clear = RefluxFlow>1.9

Close Tray Temp cascade to reflux flow
Name = Cascade Reflux
Rise = showstep; setmode(RefluxFC,Cas); setvalue(RectifierTC.SP,100)
Clear = StopSwitch
Next = Stop Steam AND Stop Feed

3.7.3.2 Shutdown

Shut down steam and shut down feed in parallel
Name = Stop Steam
Watchdog = 1M
Rise = showstep; setmode(SteamFC,Man); setmode(StripperTC,Auto); setvalue(StripperTC.Out,0)

Clear = ramp(SteamFC.Out,-5,2) AND SteamFlow<1
Next = Stop Reflux

Name = Stop Feed
Watchdog = 1M
Rise = showstep; setmode(FeedFC,Man)
Fall = FeedPump=OFF
Clear = ramp(FeedFC.Out,-5,2) AND FeedFlow<0.1

Open reflux cascade and shut reflux valve
Previous = Stop Steam AND Stop Feed
Name = Stop Reflux
Rise = showstep; setmode(RefluxFC,Man); setmode(RectifierTC,Auto); setvalue(RectifierTC.Out,0)
Clear = ramp(RefluxFC.Out,0,2) AND RefluxFlow<0.1
Next = Reduce Drum Level AND Reduce Pot Level

Wait for 35% drum level, then close product valve and stop reflux pump
Name = Reduce Drum Level
Watchdog = 5M
Rise = showstep; setvalue(RefluxLC.SP,40)
ON = timedog (5M)
Clear = RefluxLevel<36

Name = Stop Product
Watchdog = 1M
Rise = showstep; setmode(RefluxLC,Man)
Fall = RefluxPump=OFF
Clear = ramp(RefluxLC.Out,-5,2)
Next = Cooling Down

Wait for 35% pot level, then close bottoms valve and stop bottoms pump
Name = Reduce Pot Level
Watchdog = 5M
Rise = showstep; setvalue(PotLC.SP,40)
ON = timedog (5M)
Clear = PotLevel<36

Name = Stop Bottoms
Watchdog = 1M
Rise = showstep; setmode(PotLC,Man)
Fall = BottomsPump=OFF
Clear = ramp(PotLC.Out,-5,2)

When Tray 2 < 70, shut off pumps and set controllers to manual and 0 output
Previous = Stop Product AND Stop Bottoms
Name = Cooling Down
Watchdog = 10M
Rise = showstep
Clear = TrayTempTwo<70

Shut off condenser cooling water
Name = Condenser Shutdown
Rise = opaction("Turn off condenser water")
ON = timedog (1M)
Clear = Didit
Next = Stopped

3.7.3.3 List of Variables

List of Tags and Parameters
FeedFC Mode, SP, Out
SteamFC Mode, SP, Out
RefluxFC Mode, SP, Out
PotLC Mode, SP, Out

RefluxLC Mode, SP, Out
RectifierTC Mode, SP
StripperTC Mode, SP

List of Inputs
PotLevel
FeedFlow
SteamFlow
TrayTempTwo
RefluxLevel
RefluxFlow
EquipCheck [from Execution FFB]

List of Outputs
FeedPump - latch
RefluxPump - latch
BottomsPump - latch

List of Internal Variables
StartSwitch
StopSwitch
DidIt
Temp01

3.7.4 Distillation Initialization Check Logic

The following are hybrid steps in the Execution FFB.

3.7.4.1 Transmitters

Name = Check Pot Level
SET = PotLevel > 30 AND status(PotLevel) == GOODNC

Name = Check Feed Flow
SET = “Check Pot Level” AND FeedFlow <= 0 AND status(FeedFlow) == GOODNC

Name = Check Steam Flow
SET = “Check Feed Flow” AND SteamFlow <= 0 AND status(SteamFlow) == GOODNC

Name = Check Tray Temp Two
SET = “Check Steam Flow” AND TrayTempTwo < 80 AND status(TrayTempTwo) == GOODNC

Name = Check Reflux Level
SET = “Check Tray Temp Two” AND status(RefluxLevel) == GOODNC

Name = Check Reflux Flow
SET = “Check Reflux Level” AND RefluxFlow <= 0 AND status(RefluxFlow) == GOODNC

3.7.4.2 Pumps

Name = Check Feed Pump
SET = FeedPump == OFF

Name = Check Reflux Pump
SET = “Check Feed Pump” AND RefluxPump == OFF

Name = Check Bottoms Pump
SET = “Check Reflux Pump” AND BottomsPump == OFF

3.7.4.3 Controllers

Name = Check Feed FC
SET = readmode (FeedFC) == MAN AND readstatus (FeedFC.IN) == GOODNC AND readvalue (FeedFC.SP) == 0 AND
readvalue (FeedFC.OUT) == 0

Name = Check Steam FC
SET = “Check Feed FC” AND readmode (SteamFC) == MAN AND readstatus (SteamFC.IN) == GOODNC AND readvalue (SteamFC.SP) == 0 AND readvalue (SteamFC.OUT) == 0

Name = Check Reflux FC
SET = “Check Steam FC” AND readmode (RefluxFC) == MAN AND readstatus (RefluxFC.IN) == GOODNC AND readvalue (RefluxFC.SP) == 0 AND readvalue (RefluxFC.OUT) == 0

Name = Check Pot LC
SET = “Check Reflux FC” AND readmode (PotLC) == MAN AND readstatus (PotLC.IN) == GOODNC AND readvalue (PotLC.SP) == 50 AND readvalue (PotLC.OUT) == 0

Name = Check Reflux LC
SET = “Check Pot LC” AND readmode (RefluxLC) == MAN AND readstatus (RefluxLC.IN) == GOODNC AND readvalue (RefluxLC.SP) == 50 AND readvalue (RefluxLC.OUT) == 0

Name = Check Rectifier TC
SET = “Check Reflux LC” AND readmode (RectifierTC) == MAN AND readstatus (RectifierTC.IN) == GOODNC AND readvalue (RectifierTC.SP) == 0 AND readvalue (RectifierTC.OUT) == 0

Name = Check Stripper TC
SET = “Check Rectifier TC” AND readmode (StripperTC) == MAN AND readstatus (StripperTC.IN) == GOODNC

3.7.4.4 Combined Result

Name = Check Equip
SET = “Check Reflux Flow” AND “Check Bottoms Pump” AND “Check Stripper TC”

3.7.5 List of Variables

List of Tags and Parameters

FeedFC Mode, SP, Out	Init check Man, 0, 0, PV Status=Good NC
SteamFC Mode, SP, Out	Init check Man, 0, 0, PV Status=Good NC
RefluxFC Mode, SP, Out	Init check Man, 0, 0, PV Status=Good NC
PotLC Mode, SP, Out	Init check Man, 50, 0, PV Status=Good NC
RefluxLC Mode, SP, Out	Init check Man, 50, 0, PV Status=Good NC
RectifierTC Mode	Init check Man, PV Status=Good NC
StripperTC Mode	Init check Man, PV Status=Good NC

List of Field Inputs

PotLevel	Init check Status=Good NC
FeedFlow	Init check Status=Good NC
SteamFlow	Init check Status=Good NC
TrayTempTwo	Init check Status=Good NC
RefluxLevel	Init check Status=Good NC
RefluxFlow	Init check Status=Good NC

List of Inputs from Sequential FFB

FeedPump	Init check Off
RefluxPump	Init check Off
BottomsPump	Init check Off

3.8 Integration path for a legacy system and multiple field HART interface

3.8.1 Overview

This application controls the water level in Tank 1 and Tank 2 by using

a. Profibus-DP and Modbus/RTU Variable Speed Drive (VSD) for Pump 1 and Pump 2 respectively

b. HART level instrument

Variable speed drive will be used for pumps supplying water to Tank 1 and extracting water from Tank 2 to maintain the set level. Modbus/RTU will be integrated using a HSE "hard" gateway device. Profibus-DP will be integrated using a "soft" gateway based on OPC servers for Fieldbus and Profibus in conjunction with an OPC bridge. Watchdog logic in the control strategy in both the Fieldbus and Profibus ends will be used to check the integrity of the OPC link and to take safe action in case of fault. In the Fieldbus-end a flexible function block will be employed for this purpose. The OPC servers and bridging software runs on computer workstations. The HSE gateway device has custom blocks for configuration of the Modbus communication.

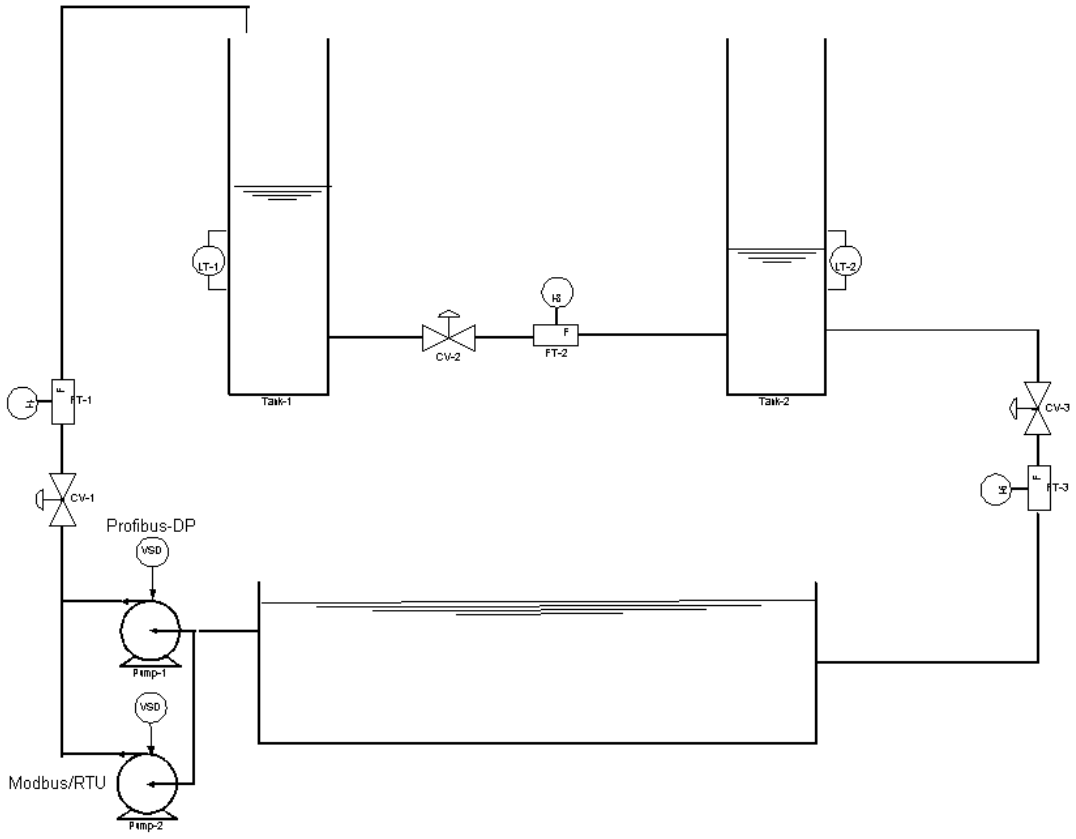
HART level transmitter will be used for measuring the water level and then for feeding the level to a control valve. HART will be integrated using a H1 gateway device. The H1 gateway device has custom blocks for configuration of the HART communication.

An H1 remote-I/O device will be used for the discrete input of level switches used for overfill and dry-run protection.

This application illustrates complete system integration of legacy protocols under HSE and OPC together with H1 devices. The application illustrates the type of level control in intermediate storage tanks commonplace in the industry. Variable speed drives were chosen to reflect the trend away from fixed speed pumps and control valves. A mixture of Fieldbus with different legacy protocols and conventional signals will inevitably be experienced in the transition period to systems that are purely Fieldbus. Both hard and soft gateways will have to be employed. This application is an example of this.

3.8.2 Process diagram

More instruments may be added if necessary e.g. HART temperature transmitters and level switches.



3.8.3 Field Wiring

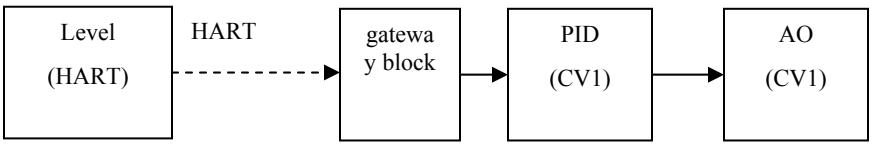
The field wiring is mainly H1 FOUNDATION fieldbus. At the host-level the HSE devices and host computer workstations are connected on Ethernet where an industrial grade switch will be used. Some sections will be Modbus and Profibus due to the usage of variable speed drive. HART is used for some transmitters. Three different kinds of gateways will be used to connect FOUNDATION fieldbus to the other 3 buses.

3.8.4 Function Block Diagram

The strategy consists of four control modules

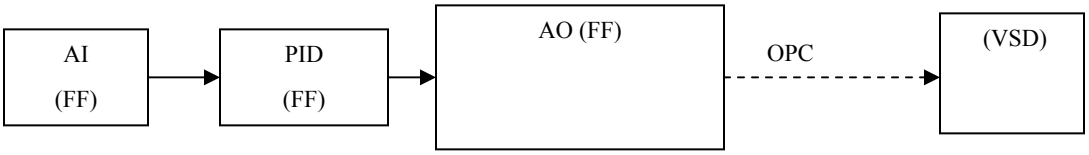
3.8.4.1 Level control using HART level transmitters

A HART pressure transmitter does the level measurement. Custom blocks are used in the HART gateway device to configure access to process and non-process data. The measured value is made available as a regular function block output which in turn is connected to standard AI and PID blocks. Initially a regular Fieldbus control valve positioner will be used but the strategy may later be combined with the other gateway to utilize a variable speed drive for the actuation. It is also intended that the operator workstation will show the non-process data in the HART device, available in the gateway block as contained parameters.



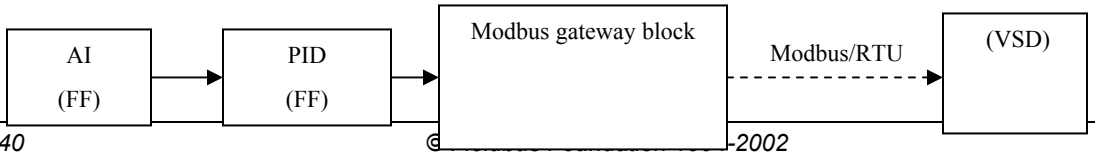
3.8.4.2 Level control using Profibus-DP variable speed drive

A regular Fieldbus pressure transmitter with a standard AI block is used for the level measurement. Control is performed in a standard PID block and passed to a standard AO block in the HSE linking device. The Fieldbus OPC server will then read the AO block OUT parameter, pass it to the OPC bridge, which in turn uses the Profibus OPC server to write the value as the setpoint for the variable speed drive. It is also intended that the operator workstation will show non-process data from the variable speed drive.



3.8.4.3 Level control using Modbus/RTU variable speed drive

A regular Fieldbus pressure transmitter with a standard AI block is used for the level measurement. Control is performed in a standard PID block and passed to a custom Modbus block in the HSE linking device. The HSE gateway device will then write the value as the setpoint for the variable speed drive. It is also intended that the operator workstation will show non-process data from the variable speed drive.



3.8.4.4 OPC watchdog

In the Fieldbus-end a flexible function block including a watchdog timer will be employed to detect an OPC link failure. The strategy will utilize a scheme where a discrete value is received on one parameter, toggled, and written to another. If the signal being received is not toggled within a certain configurable timeout, the OPC link can be considered broken and failsafe action will be initiated.

3.8.5 Block Access

This application uses standard, custom and fixed-OD flexible function blocks. Therefore no user-selectable parameter sets apply.

3.9 PROCESS COOLING WATER SYSTEM

3.9.1 Scenario

This application describes the control system for a closed loop Process Cooling Water (PCW) system. The process area is remote from the operator and is generally unmanned.

The load comprises an undefined number of processes. The heat load is variable and therefore the PCW flow demand will vary.

The product value is high. It can be destroyed by a high process temperature. Consequently, reliability is paramount. There are redundant pumps and heat exchangers. The control system is also required to be robust. Common mode failures must be minimized.

All communications utilize FOUNDATION fieldbus H1 or HSE. One exception is the hardwired interlock between the PCW Receiver Low-Low level switch and the pump shutdowns. This interlock must function even if the drives are in local manual control.

The HSE networks connecting the pump drives are redundant for control system robustness. No single component failure can shutdown all three drive control systems.

3.9.2 Receiver Level Control

This is a closed loop system and water loss is minimal. The receiver level is allowed to vary between 4 feet and 5 feet. LC-100 Opens the two position valve, LV-100 at 4 feet and closes it when the level rises to 5 feet. Both values are adjustable from the central control room. Also the controller can be switched to manual from the control room and LV-100 can be opened and closed manually.

A HAND-OFF-AUTO (H-O-A) station allows local manual intervention. When the H-O-A switch is moved out of Auto, local control overrides the automatic control system and an alarm is generated to alert the central control room.

During normal operation, the makeup valve, LV-100, will not be open more than two or three minutes at any one time. If the valve is open 5 minutes, an abnormal process condition exists, and an alarm is generated to alert the operator.

The high level switch, LSHH-100, also generates an alarm to alert the central control room.

The low level switch, LSLL-100, generates an alarm to alert the central control room and is also interlocked (hard wired) to the three pump drives to shut them down on Low-Low level.

3.9.3 Distribution Pump Control

There are three distribution pumps, P-100A, P-100B and P-100C. Normal operation is two pumps online with a third on standby. Initially, the control room operator will chose the operational and standby pumps. The system will rotate the standby pump weekly. If one of the operational pumps fails, the control system will automatically start the standby unit and alert the operator. Normal rotation will stop and only be restarted by the operator.

Pump failure will be determined by

The operational pumps not being able to maintain the pressure setpoint.

Pump failure alarm from the frequency drive.

Each of the pump drives has redundant HSE stacks. The two HSE inputs go to a signal selector which selects the first good signal as the cascade input to the drive speed control.

The primary pressure controller is PC-100 is located in PIT-100 or in the linking device, PX-100. Constant PCW discharge header pressure is maintained by varying the pumps speed. Operational pumps should share the load equally. If two pumps are unable to maintain the pressure setpoint, the standby pump will automatically be started by the FFB in the linking devices and an alarm generated to alert the operator.

If PIT-/ PC-100 fails, control automatically switches to end of loop pressure transmitter and controller, PIT-101/PC-101. The pressure at PIT-101 will be different from PIT-100 due to pipe losses and elevation differences. To maintain a stable process, the setpoint of PC-101 will be the pressure at PIT-101 at the time of failure. Each transmitter will monitor each other's status and initiate the control transfer on bad or uncertain status of the other. An alarm is generated at the time of control switch over to alert the operator.

Local H-O-A switches in the variable frequency drive cabinets allow local control. When the H-O-A is not in the Auto position, an alarm is generated to alert the operator. When the H-O-A is in Hand, the pump speed can be varied locally at the variable frequency drive console. Automatic control is overridden.

The pumps drives have hard wire interlocks from the PCW Receiver Low-Low level switch, LSLL-100, to shutdown the pumps at low-low level whether in automatic or local manual control.

3.9.4 Process Cooling Water Temperature Control

The PCW is cooled by two parallel heat exchangers, HEX-100A and HEX-100B. Both exchangers are normally online. However, each exchanger is sized to maintain the full system load if the other must be taken out of service

The primary temperature controller is TC-100 located in TIT-100 which controls TV-100A and TV-100B so that they are equally open. Chilled water, CHW, at 40 degrees F is circulated on the tube side. Flow is varied to maintain a discharge PCW temperature of 65 degrees F. The two heat exchangers should share the load equally.

If TIT-100/ TC-100 fails, control is automatically switched to the backup temperature controllers, TC-100A and TC-100B by the input selector switches, TS-100A and TSD-100B. The temperature setpoints will remain at their last value on failure of the primary controller. An alarm is generated at the time of control switch over to alert the operator.

The system is also able to continue to function on a single heat exchanger control if one of the inner loops fails. The failed loop control valve will maintain its last setting on failure of the transmitter or controller.

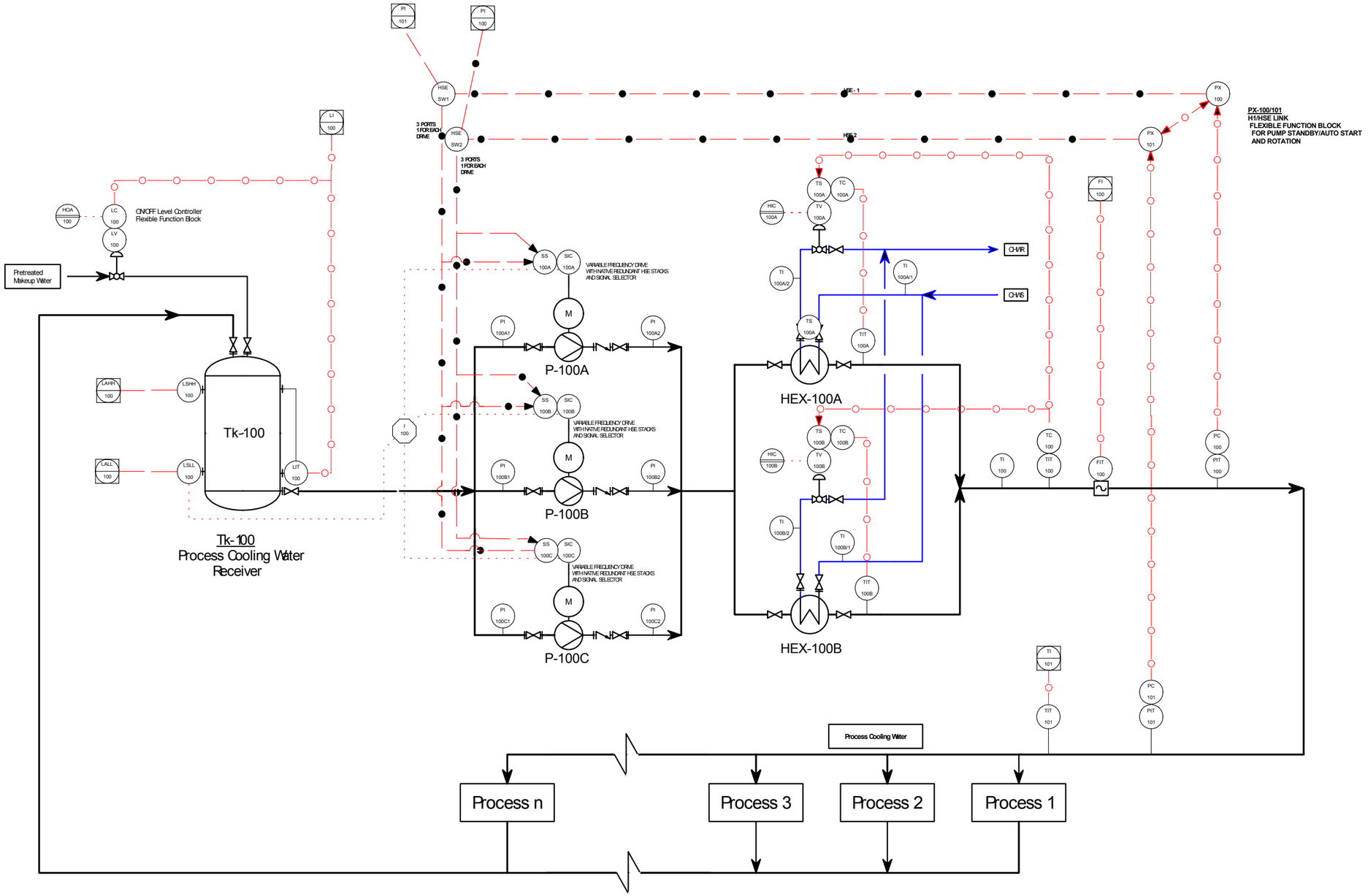
Local H-O-A switches allow individual local control of each heat exchanger. When the H-O-A is not in the Auto position, an alarm is generated to alert the operator. When the H-O-A is in Hand, the CHW flow through the heat exchanger can by varied by an adjacent manual loading station. Automatic control is overridden.

3.9.5 Variable Frequency Drives Monitoring

As a minimum, the following points will be visible, on demand, to the operator in the central control room:

- Motor speed (RPM) or Output frequency
- Output Voltage
- Output current
- Run status
- Major fault status
- Minor fault status
- Current fault condition
- Last fault condition
- Second to last fault condition
- Third to last fault condition
- Fourth to last fault condition
- Motor FLA current setting
- H-O-A Not-in Auto

Both a motor fault and the H-O-A switch not in Auto will generate alarms.


PI 101
PI 100
HSE SW1
HSE SW2
3 PORTS 1 FOR EACH DRIVE
HSE-1
HSE-2
PX 100
PX 101
PX-100/101
H1/HSE LINK
FLEXIBLE FUNCTION BLOCK
FOR PUMP STANDBY/AUTO START
AND ROTATION
LI 100
HOA 100
LC 100
LV 100
ON/OFF Level Controller
Flexible Function Block
Pretreated
Makeup Water
LAHH 100
LSHH 100
LALL 100
LSLL 100
LIT 100
Tk-100
Tk-100
Process Cooling Water
Receiver
I 100
SS 100A
SIC 100A
VARIABLE FREQUENCY DRIVE
WITH NATIVE REDUNDANT HSE STACKS
AND SIGNAL SELECTOR
M
PI 100A1
PI 100A2
P-100A
SS 100B
SIC 100B
PI 100B1
PI 100B2
P-100B
SS 100C
SIC 100C
PI 100C1
PI 100C2
P-100C
TS 100A
TC 100A
HIC 100A
TV 100A
TI 100A/2
TI 100A/1
TIT 100A
HEX-100A
CHWR
CHWS
TS 100B
TC 100B
HIC 100B
TV 100B
TI 100B/2
TI 100B/1
TIT 100B
HEX-100B
FI 100
TC 100
TI 100
TIT 100
FIT 100
PC 100
PIT 100
TI 101
TIT 101
PC 101
PIT 101
Process Cooling Water
Process n
Process 3
Process 2
Process 1

4. Detailed Plant Applications

The following applications build on the simpler applications in Section 3, concentrating on process details more than the layout of each FFB. They have been submitted by different contributors using different formats.

4.1 Multivariable Matrix Control Application

4.1.1 Overview

This application is an example of a multivariable matrix control for a gas processing plant. The scope of this section is to illustrate the application of a MVMC in general, the specifics of the application will be defined by the actual process dynamics.

A gas processing plant separates raw natural gas into methane and heavier hydrocarbon liquids for petrochemical feedstock. Refrigeration and generated by a turbo-expander, chill the inlet gas to cryogenic temperatures. The two products are then separated in a demethanizer column.

The main control objective of the application is to maximize liquid recovery while maintaining product specifications. Colder operating temperatures will increase NGL liquids recovery but at the risk of violating product specification. Careful regulation of heat input at the column side reboiler is critical in the maintenance of product quality and quantity.

The product specifications are, the maximum Mol % of methane in the NGL liquids recovery, and the Overhead gas Higher Heating value.

The maximum Mol % of methane in the NGL liquids recovery is set by controlling the ratio of methane to ethane (Mol % C1/C2). This is normally achieved by manipulating the flow through the bottom reboiler, manipulated variable No. 2.

Another objective is to minimize the effects of inlet flow changes on column pressure. This will help to reduce alarms, trips and emissions.

One of the constraints in the example process is that inlet backpressure must be minimized. This is due to wellhead constraints. A consequence of this is that the plant must tolerate significant swings in feed rate and composition, while safely producing products that meet contract specifications. Inlet flow rates are therefore to be considered an uncontrollable disturbance.

The heat medium temperature and upper tray temperature were used as feedforward variables, to improve the pressure response.

The models in a real application can be created from historical data, or from actual observation and analysis of the process.

This application will demonstrate how a FFB-MVMC could be used to regulate complex, non-linear processes with variable lags and dead times.

The FFB-MVMC should be able to respond efficiently to the disturbances and load changes.

Requirements and consideration for this applications:

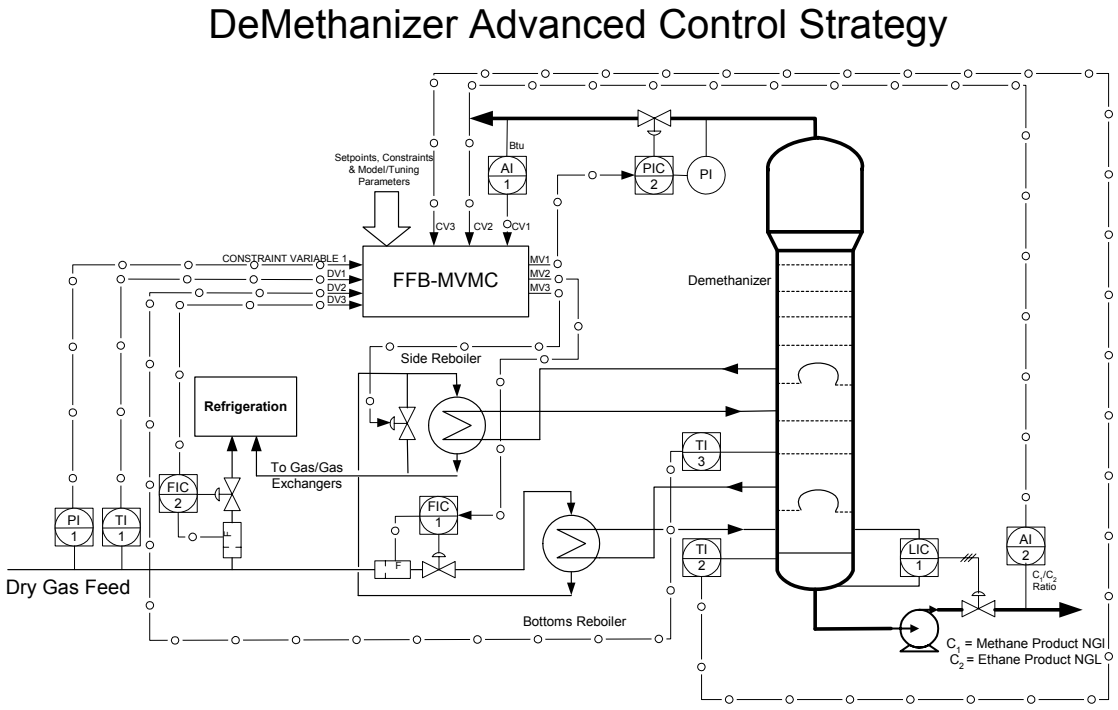
1. Process Lags need to be taken into account for this application (tower stabilization time)
2. Analyzer sample delay times.
3. Easy to install, configure and maintain.
4. It does not require a high level technical of expertise for maintenance and daily operations.

5. Capable of online changes.
6. Interoperable with all devices as described in FOUNDATION fieldbus specifications.
7. The application could work either in “Advisory” Mode or “Closed Loop Mode.
8. The FFB-MVMC Polling frequency should be variable to prevent the internal model and prediction vectors becoming too large, when controlling process with large lags and dead times.
9. In the case of the FFB-MVMC failing, the control should automatically revert to the basic PID regulatory controls.
10. The FFB-MVMC should allow the operators to enter the operating targets and constraints values for the application.
11. There should be a mode parameter for each manipulated variable. This could be changed by the operator or automatically when a regulatory layer controller is placed in local auto.
12. All input variables to the FFB-MVMC should have a mode parameter to enable them to be turned off when a variable is suspect or under maintenance.
13. The FFB-MVMC should generate diagnostic information and alarms.
14. Links to H1 for output to Manipulated Variable MV2.

BENEFITS

1. Enable the use of FFB technology for real applications.
2. Demonstrate the use of FFB HSE architecture.
3. Easy to install, configure and maintain.
4. Interoperability assured and tested by FF organization
5. This technology can be applied to many unit operations including fractionation, product treating and compression.
6. Operability of advanced control application across a multi vendor architecture using FFB HSE/H1.

4.1.2 Process Diagram

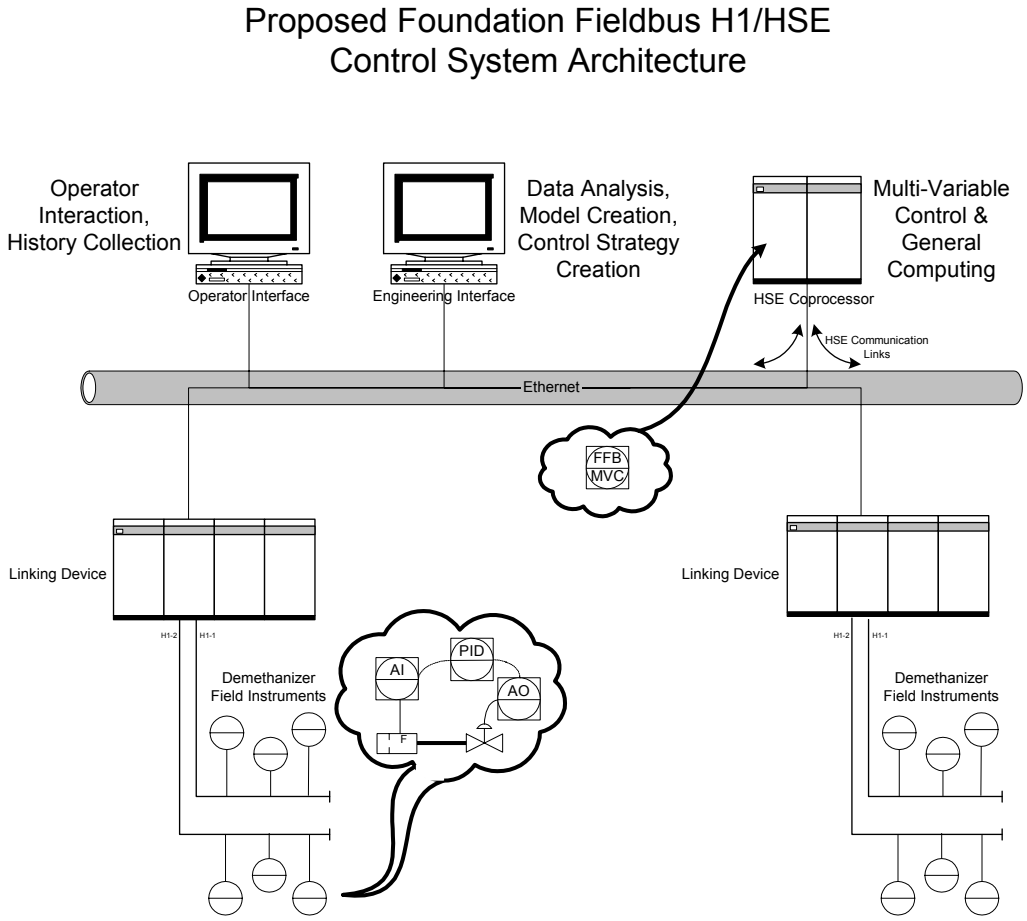


4.1.3 Matrix Diagram

		Overhead Pressure PV-2	Hot Gas Flow FV-1	Reboiler Bypass Gas Flow FV-3	Unit Inlet Pressure PI-1	Middle Tray Temperature TI-3	Unit Inlet Flow FI-1
		MV1	MV2	MV3	DV1	DV2	DV3
Btu Analyzer AI-1	CV1						
Ratio of Methane /Ethane (C1/C2) AI-2	CV2						
Bottom temperature TI-2	CV3						

One of the constraints of the system is the Unit Inlet Pressure PI-1.

4.1.4 System Architecture



4.1.5 Field Wiring

The field wiring is H1 FOUNDATION fieldbus. Not shown is a FOUNDATION fieldbus HSE linking device, which may be located in any convenient non-hazardous area. The linking device requires two or three H1 Ports. The FFB-MVMC will reside in an HSE device, perhaps a linking device, coprocessor, or general purpose computer. Field device connection to bus segments depends on plant wiring policy and device features.

4.1.6 FFB-MVMC Parameters

The access table defines the minimum required parameters.

Index	Parameter	VIEW_1	VIEW_2	VIEW_3	VIEW_4
38	ST_REV	2	2	2	2
39	TAG_DESC				
40	STRATEGY				2
41	ALERT_KEY				1
42	MODE_BLK	4		4	
43	BLOCK_ERR	2		2	
44	ALGORITHM_SEL				4
45	CONTENTS_REV				4
46	IN_1	5		5	
47	IN_2	5		5	
48	IN_3	5		5	
49	IN_4	5		5	
50	IN_5	5		5	
51	IN_6	5		5	
52	IN_7	5		5	
53	IN_8	5		5	
54	OUT_1				
16	OUT_2				
17	OUT_3	5		5	
16	OUT_4				
17	BAL_TIME_1				
18	BAL_TIME_2				
19	BAL_TIME_3				
20	BAL_TIME_4				4
21	BACKCAL_1				
22	BACKCAL_2				
23	BACKCAL_3				
24	BACKCAL_4				
25	OUT_HI_LIM_1		4		
26	OUT_LO_LIM_1		4		
27	OUT_HI_LIM_2	5		5	
28	OUT_LO_LIM_2				4
29	OUT_HI_LIM_3		4		
30	OUT_LO_LIM_3		4		
31	OUT_HI_LIM_4				
32	OUT_LO_LIM_4				
33	Output Status_1				
34	Output Status_2				
35	Output Status_3				
36	Output Status_4				
37	UPDATE_EVT				
38	BLOCK_ALM				
39	ALARM_SUM				
40	ACK_OPTION				

Index	Parameter	VIEW_1	VIEW_2	VIEW_3	VIEW_4
79					
80					
81					
82					
83					
84					
85					
86					
87					
88					
52					
53					
54					
55					
56					
57					
58					
59					
60					
61					
62					
63					
64					
65					
66					
67					
68					
69					
70					
71					
72					
73					
74					
75					
76					
77					
78					
79					
80					
81					
82					
83					
84					
85					

41	ALARM_HYS					86					

4.1.7 Conclusions

Testing of application to be performed on an actual plant (Difficult) or by using an interface to a dynamic simulation of the process (Most Workable). More detail will be added should this application go to the demonstration phase.

4.2 Application Profile - Cleanroom Makeup Air Unit

4.2.1 Scenario

This application profile has been developed around a hypothetical cleanroom air handling system. It is assumed that this is an installation in an existing plant with a legacy control and information network infrastructure. It must be emphasized that this is a hypothetical model and some of the control strategies have been modified to demonstrate FOUNDATION fieldbus capabilities and are not necessarily the designs which would be implemented in a real facility.

The application incorporates several FFBs, control of a variable speed drive with a legacy bus interface, feed-forward mass flow control of air moisture content, integration of the FOUNDATION fieldbus system with a legacy plant network and multiple I/O devices.

These systems have traditionally been implemented with PLCs or a combination of PLCs and single loop controllers. It is a good application for a hybrid FOUNDATION fieldbus implementation.

4.2.2 System Description

See the full page drawing at the end of this application for the following discussion.

Cleanrooms involve the circulation of large quantities of air. Most of this air is recirculated by Recirculation Air Handlers (RCUs) and discharged back into the space through high efficiency filters. The cleaner the space, the greater the air circulated. Cleanroom temperature is controlled at the RCUs.

The makeup air units' function is to replace the air lost through access points, ex-filtration and exhaust systems. The makeup air units (MAUs) maintain a small positive pressure in the space as well as control the space moisture. The quantity of makeup air is small in relation to the recirculated air, one-tenth or less.

Multiple MAUs discharge into a common supply air header which in turn feeds makeup air to many more RCUs. For completeness of this example, three MAUs are shown on the instrument schematic.

4.2.3 System Integration

A gateway will be required to link the new HSE network to the existing legacy plant control network (Modbus+, Profibus, ControlNet, etc.). It has been assumed that the information network is a conventional Ethernet and that this unit's SCADA can also be the system's OPC server. It should also be noted that the HMI's on the plant floor are clients of this server working off of the Ethernet.

4.2.4 Control Strategy/Sequence of Operations

Variable Speed Drive Legacy Gateway FFB

Ideally, the Variable Speed Drive (VSD) would have a FF HSE stack and FFB capability. However, no drive vendor has this interface, nor does it appear to be coming in the near future. Therefore, interface communications between the VSD and the Linking Device system is via a legacy bus (Modbus, Modbus+, Profibus, ControlNet, DeviceNet, etc.). An FFB in the Linking Device provides this gateway.

The VSD is an intelligent device and may expose a hundred or more parameters. Approximately twenty operating parameters and alarms are of interest to this application. These are transmitted to the SCADA by the Linking Device and its gateway.

Supply Fan START/STOP Control

The MAU supply fan is started and stopped from either the SCADA or the HAND-OFF-AUTO (HOA) switch located on the fan's VSD.

Placing the VSD-HOA in the OFF position stops the fan and disables the control system from starting the fan.

In the AUTO position, the VSD provides an input signal indicating that the fan is controlled by the control system. A "HOA not in Auto" alarm is generated if the HOA switch is not in AUTO. The fan is manually started from the SCADA and runs continuously.

The VSD is interlocked with the freezestats, TSL-1A, TSL-01B, TSL-1C and TSL-1D, and the smoke detector, YS-01, in the discharge duct. The freezestats and smoke detector shutdown the fan and close the outside and supply air dampers. Activation of any freezestat starts the preheat pump, if it is not already on and causes the preheat temperature control valve, TCV-3, to fail open. Appropriate alarms are transmitted to the SCADA.

A "Fan Failure" alarm is generated if the automatic control system starts the fan and, after a timed delay, does not receive a fan running verification signal from the VSD current switch. A "Fan Failure' alarm is also generated when the running verification signal from a control system started fan is lost without a Stop signal. A failed fan must be manually reset at the SCADA.

The outside and supply air dampers, YD-1and YD-2, are opened simultaneously when the fan is started. "Damper failure" alarms are generated if a damper is not confirmed opened by the damper position switches, ZSO-1or ZSO-2, when the fan is started. "Damper failure" alarms stops the fan after a timed delay.

The damper solenoid valves are FOUNDATION fieldbus with two discrete inputs connected to the damper position switches. The valves also include a diagnostic alarm to alert the operator if the damper actuators take more than the prescribed time to open or close.

Pressure switch, PS-1, alarms and will stop the fan when static pressure at the fan discharge exceeds 4 inches wc.

A "VSD fault" alarm is generated at the SCADA upon detection of a VSD fault through VSD network.

In HAND or AUTO control, the MAU's temperature, dewpoint and pressurization PID control loops are enabled when the fan is confirmed running and disabled when the fan is stopped. The preheat temperature control loop is disabled when outside temperature is above the preheat loop enable setpoint and enabled when the outside temperature is below the loop enable setpoint.

Fan Speed and Makeup Pressurization Control

There are three fans discharging into the supply air header. The three fans can only be controlled by one pressure controller, otherwise the controllers will continually fight each other. A simple FFB in each linking device will resolve who will control according to the following rules. When all control loops are active and their outputs all have a *good* status, the MAU-3 pressure loop will control. If MAU-3's status changes to *uncertain* or *bad,* the MAU-2 pressure loop will control the three fans, assuming communications are maintained. Similarly, if both MAU-3 and MAU-2's control outputs are either *uncertain* or *bad,* the MAU-1 pressure loop will control all available fans. Communication amongst the three controllers will be over the HSE.

When a communication or control system failure occurs, all VSD speeds should remain at their last setting before the failure. Similarly, after a power failure, all fans should ramp up to the last speed setting prior to the failure.

Preheat Temperature Control

The preheat temperature control loop, located in TCV-3, is enabled and when the outside air temperature, TIT-1, drops below 12 Deg C. It will remain energized until the outside air temperature rises above 13 Deg C.

The preheat temperature controller maintains a T-3 setpoint of 10 Deg C. This is sufficiently above the dewpoint to allow the steam injected by the humidifier to be absorbed.

Preheat Pump Control

Assuming the MCC HOA switch is in AUTO, the preheat pump, P-1, will start when the preheat temperature control loop is enabled.

If the HOA switch is switched out of AUTO, a "Not in Auto" alarm is generated at the SCADA.

The pump running status is determined by a current switch, IS-1, in the MCC. In the AUTO mode, if the pump fails to start when commanded by the control system, or it stops without a STOP command, a pump failure alarm is generated at the SCADA.

After a power failure, the pump will go to the ON or OFF state as determined by the preheat temperature control loop.

An MIO device located in the MCC provides the START and STOP outputs to the pump's latching relay. The pump RUN status, the four freezestats and "Pump not in AUTO" alarm are the discrete inputs to the MIO.

Outside Air Dewpoint and Temperature Monitoring

Outside air dewpoint and temperature are monitored at each MAU. If a transmitter fails, the control system of the failed transmitter will use either of the other transmitter signals.

Humidification Control

The steam humidification control loop is enabled when the outside air dewpoint is at or below 4 Deg C. It is disabled when the outside air dewpoint is at or above 5 Deg C.

The humidification controller modulates the steam flow control valve, ACV-1, to maintain a constant supply air dewpoint of 5 Deg C. The FFB algorithm calculates the difference in absolute humidity between the desired dewpoint and the outside air dewpoint measured by AIT-1, multiplied by the air flow measurement, FIT-1. This calculated steam addition is adjusted by a factor to maintain the supply air dewpoint, AIT-2, at the required setpoint.

> Note 1: The supply air dewpoint is the ultimate controlled variable. The calculated steam addition will not be as accurate as the dewpoint measurement. However, tests have revealed that the dewpoint transmitter has a 67% response time on the order of 150 seconds. The outside dewpoint will change relatively slowly so that the response time is not an issue. The variable that can change relatively quickly is the air flow. Although the author is unaware of any installation where this control strategy has been implemented, the feed forward loop should reduce the supply air dewpoint excursions and therefore improve stability of the cleanroom environment.

Note 2: No FF dewpoint measuring devices have been found on the market. However, there is at least one manufacturer that offers a multi-variable transmitter with RS485/422 serial interface. Dewpoint, absolute humidity and dry bulb temperature are all available in this device. If an FF device cannot be found, an FFB could be developed to link the RS485 bus to FF. Alternately, 4-20 ma signals are readily available.

Note 3: The steam control loop could be implemented simply by using a combination control valve and flow controller. These devices are sufficiently accurate for this application and it is understood that one manufacturer is planning to have this available with an FF card in the near future.

Dehumidification Control

The dehumidification controller is enabled when the outside air dewpoint is at or above 6 Deg C. It is disabled when the outside air dewpoint is at or below 5 Deg C.

The dehumidification controller modulates the brine and chilled water valves in stages, as described below, to maintain a supply air dewpoint of 5 Deg C. First, the brine control valve, TCV-2B, is modulated to 15% open or the setpoint is achieved. Then the chilled water valve, TCV-2A, is modulated until it is fully open or the setpoint is achieved. Finally the brine valve, TCV-2B, is modulated between 15% and 100% open to achieve the desired setpoint.

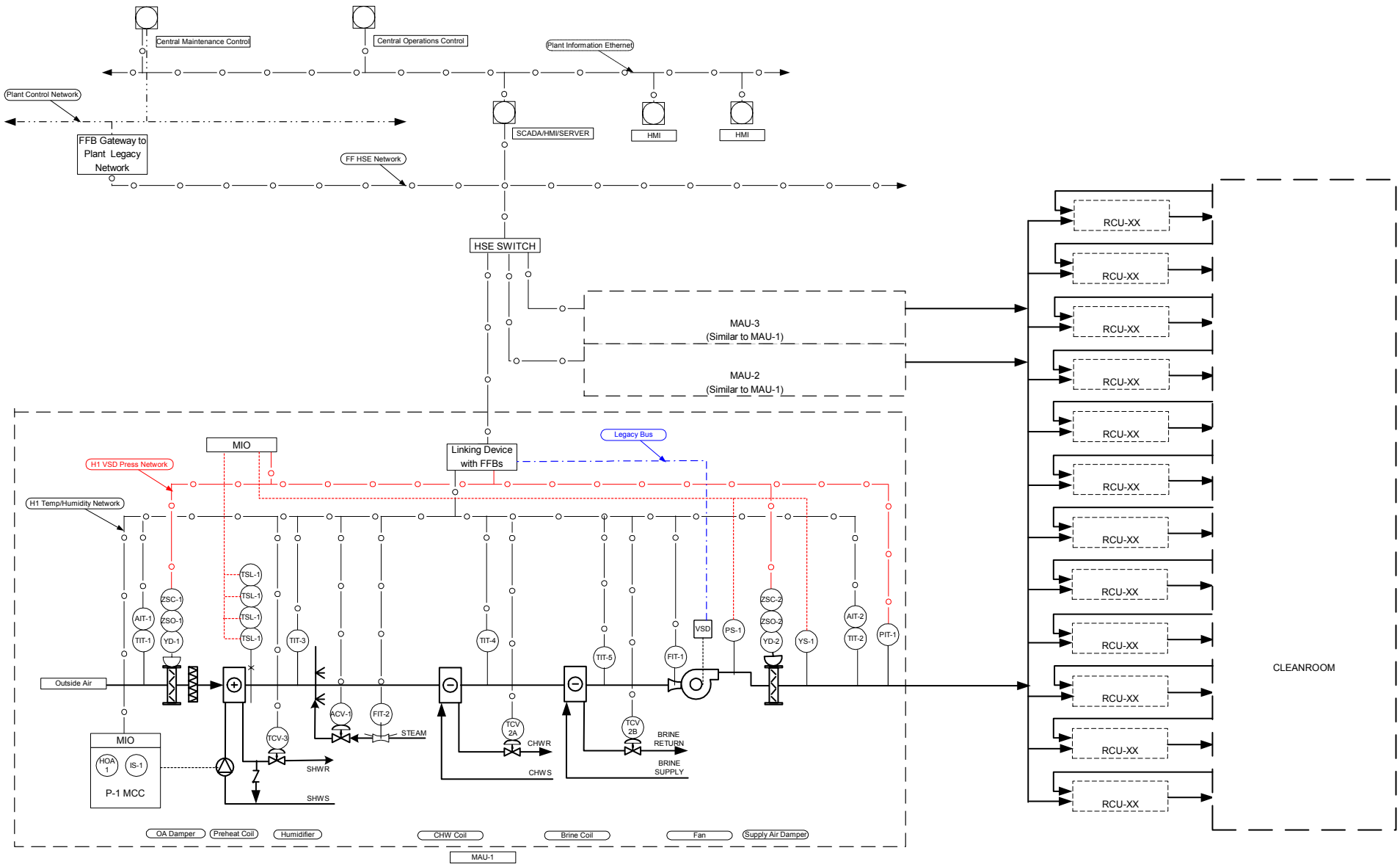
Temperature Control

The temperature controller is enabled when the dehumidification controller is disabled. It is disabled whenever the dehumidification controller is enabled.

The temperature controller modulates the brine and chilled water valves in stages similar to the dehumidification controller to keep the supply air temperature, T-2 below 15 Deg C. Note that the supply air temperature is allowed to float between 5 and 15 Deg C in this mode.

Critical Alarms

Fan failure
Freezestat Trip
VFD Fault
VFD not in auto
Preheat pump failure
Preheat pump not in auto
Smoke Detector Trip
Low preheat temp alarm
Supply Damper Not Open
Outside Air Damper Not Open
High discharge temp alarm
High discharge dewpoint
High static pressure shutdown
Low discharge dewpoint
Low Header static pressure
High header static pressure
All transmitter failures


Central Maintenance Control
Central Operations Control
Plant Information Ethernet
Plant Control Network
FFB Gateway to Plant Legacy Network
SCADA/HMI/SERVER
HMI
HMI
FF HSE Network
HSE SWITCH
MAU-3
(Similar to MAU-1)
MAU-2
(Similar to MAU-1)
RCU-XX
RCU-XX
RCU-XX
RCU-XX
RCU-XX
RCU-XX
RCU-XX
RCU-XX
RCU-XX
RCU-XX
RCU-XX
CLEANROOM
MIO
Linking Device with FFBs
Legacy Bus
H1 VSD Press Network
H1 Temp/Humidity Network
TSL-1
TSL-1
TSL-1
TSL-1
ZSC-1
ZSO-1
YD-1
AIT-1
TIT-1
TIT-3
TIT-4
TIT-5
VSD
PS-1
FIT-1
ZSC-2
ZSO-2
YD-2
YS-1
AIT-2
TIT-2
PIT-1
Outside Air
ACV-1
FIT-2
STEAM
TCV-3
SHWR
SHWS
TCV 2A
CHWR
CHWS
TCV 2B
BRINE RETURN
BRINE SUPPLY
MIO
HOA 1
IS-1
P-1 MCC
OA Damper
Preheat Coil
Humidifier
CHW Coil
Brine Coil
Fan
Supply Air Damper
MAU-1

4.3 Packaged Batch Distillation Control

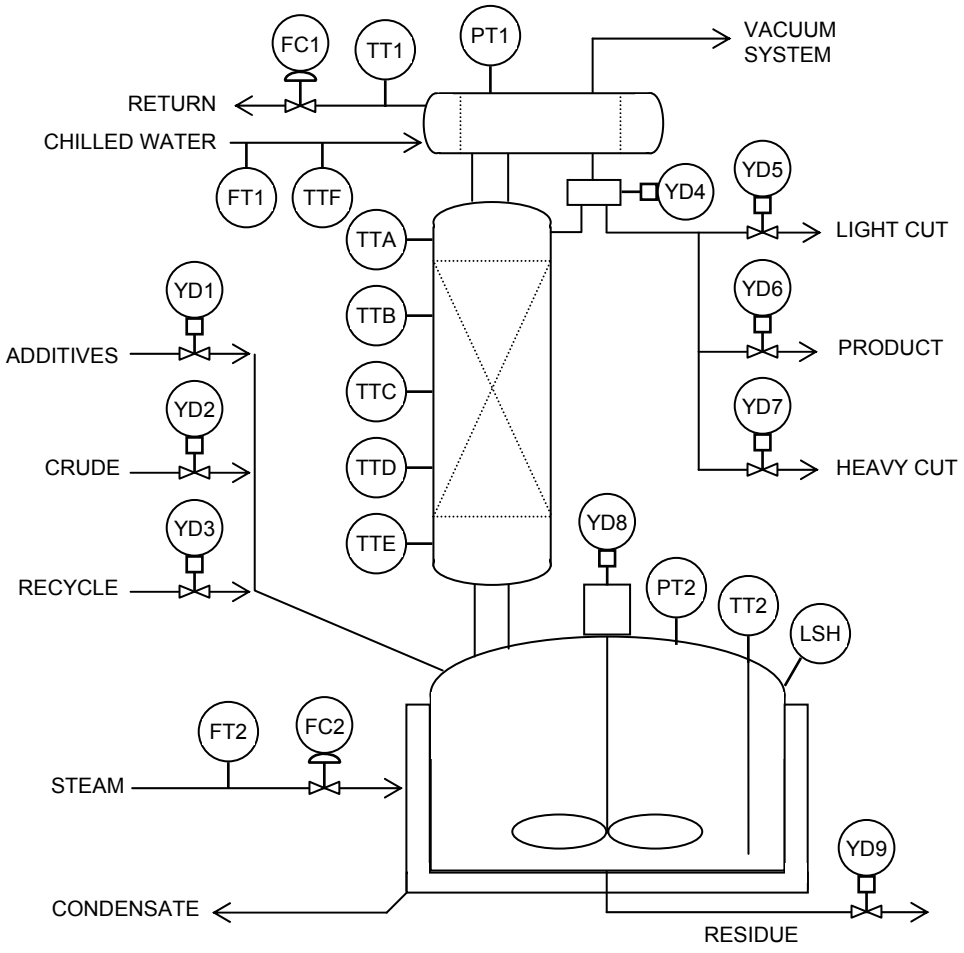
4.3.1 Overview

This application is an example of batch distillation. It is a mixture of existing processes so that it does not describe any real processes, which are all proprietary. It is a package unit, complete with control system done in several FFB.

The still pot is filled with crude material to be distilled, plus some recycle and additives. The means for measuring the amounts of these ingredients is not shown. The still pot has a steam jacket to provide heat to boil the mixture. A distillation column is attached to the pot, and outfitted with temperature sensors. A condenser vessel sits at the top of the column. The condensate flows by gravity into a reflux splitter, a device that switches the flow between column reflux and distillate receivers at a rate that will produce the desired reflux ratio. There is no condensate receiver because minimum hold-up is required. The distillate is sent to the proper receiver by the batch control system.

The unit is located in a hazardous area. The package vendor provides a programmed PLC to be mounted in a general purpose area. The package has been tested at the vendor's plant, and is known to work well for other customers.

4.3.2 Process Diagram



YD1, YD2 and YD3 are block valves for controlling the addition of ingredients.

FT1 and FC1 form a loop controlling the flow of cooling water through the condenser. This loop can be cascaded to the temperature difference TT1-TTF, with the primary control block in TT1.

TT1 and TTF allow calculation of the BTUs removed, which is proportional to the amount of vapor condensed.

PT1 measures the vapor pressure at the top of the column in absolute units.

TTA through TTE may be used to estimate the composition of the material at points within the column.

YD4 is the actuator that switches condensate to the reflux line or the distillate line. The timing of the switching point within a periodic cycle determines the reflux ratio. Minimum hold-up systems do not have enough pressure to use plug valves.

YD5, YD6, and YD7 select the desired distillate receiver. One is always open while vapor is being condensed.

FT2 and FC2 form a loop controlling the steam flow to the still pot jacket. This loop can be cascaded to one of several primary measurements, e.g. TT4, PT2-PT1.

YD8 controls the single-speed agitator.

PT2 measures the vapor pressure of the still pot in absolute units.

TT2 measures the pot contents temperature.

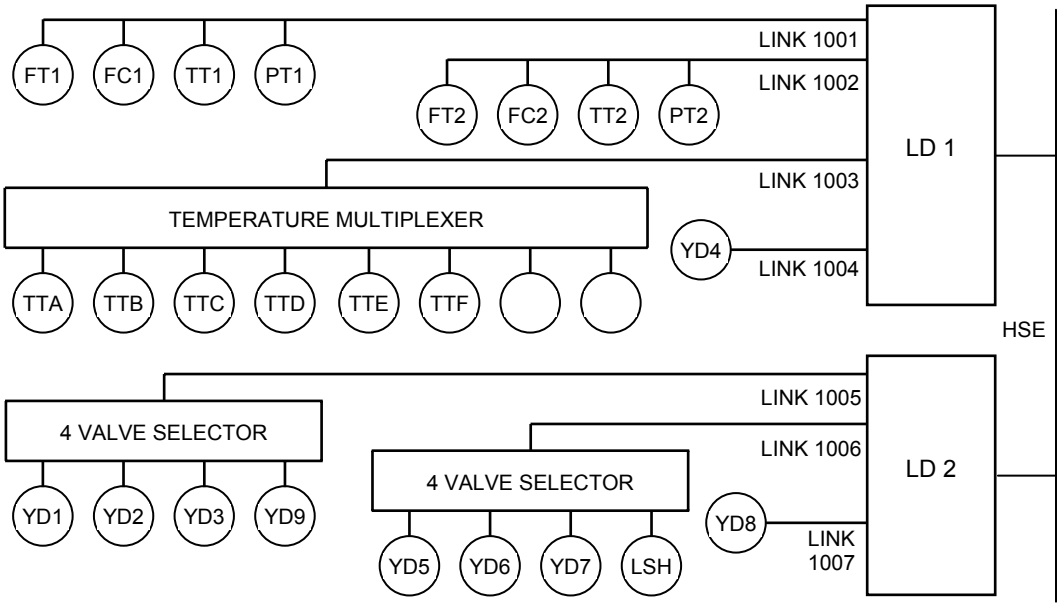
LSH closes when there is too much material in the pot.

YD9 controls the flow of residue when the batch is complete.

A real still may also have nitrogen to purge air from the system, solvent or steam to clean the vessels, and a pump for the residue line.

4.3.3 Field Wiring

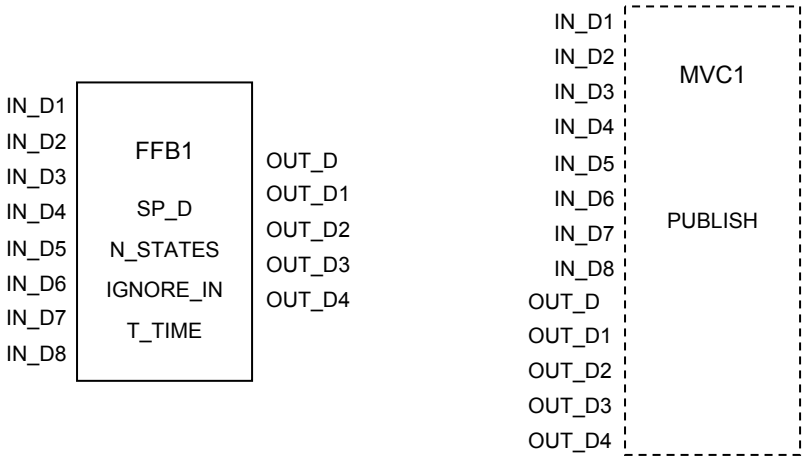
The field wiring is mostly H1 FOUNDATION fieldbus. An eight point temperature multiplexing device combines TTA through TTF into one bus drop with intrinsically safe temperature wiring. Given the availability of a "valve selector" box that can sense the position switches of 4 valves but only power the actuator of one with 100 ma max from the bus, many of the YD functions can be combined. YD4 needs a dedicated bus for power. YD8 is located in a motor control center outside of the hazardous area, so it needs a dedicated bus due to its location. An alternative solution may be less expensive. LSH is brought in as one of the position switches of a non-existent valve. The field wiring from the batch distillation package to the control room might look like this:



The seven fieldbus cables are connected to two 4-link Batch Linking Devices, so called because they have algorithms for ISA S88 functions, e.g. equipment modules and unit supervision. These connect to HSE fieldbus, which allows communication between linking devices and with the hosts that manage the process and communicate with IT functions.

4.3.4 Function Block Diagrams

4 VALVE SELECTOR



The four valve selector is designed to be mounted in a hazardous area, so it has a low power microprocessor. It contains one fixed OD FFB that acts as a five-state Device Controller and one Resource block. There are no transducer blocks and no I/O blocks. A Multi-Variable Container is used to publish the states of the device. A simple contained setpoint, SP_D, is written by a client/server transaction at those times when it is necessary to change the state of the device. Eight IN_Dx parameters show the states of the confirm switches. Five OUT_D parameters show the status of the five commanded states. The user may set the number of states between two and five in N_STATES for those cases where less that four devices are controlled. Confirm switches may be configured to be ignored in IGNORE_IN. The expected travel time may also be configured in T_TIME. The block generates alarms as required. The following enumerations define the states:

Num	SP_D	OUT_D	OUT_D1	OUT_D2	OUT_D3	OUT_D4
0	All Off	Undefined	Undefined	Undefined	Undefined	Undefined
1	Output 1	Confirmed Off	Confirmed Off	Confirmed Off	Confirmed Off	Confirmed Off
2	Output 2	Confirm Off Fail	Confirm Off Fail	Confirm Off Fail	Confirm Off Fail	Confirm Off Fail
3	Output 3	Confirm Off Lost	Confirm Off Lost	Confirm Off Lost	Confirm Off Lost	Confirm Off Lost
4	Output 4	Confirmed Active	Confirmed On	Confirmed On	Confirmed On	Confirmed On
5		Confirm Active Fail	Confirm On Fail	Confirm On Fail	Confirm On Fail	Confirm On Fail
6		Confirm Active Lost	Confirm On Lost	Confirm On Lost	Confirm On Lost	Confirm On Lost
7		Switch Fail	Switch Fail	Switch Fail	Switch Fail	Switch Fail

A confirm switch fails if it is not closed after the travel time expires. A confirm is lost if it did close, but later opened up. When the device is confirmed active, the output selected by the setpoint is confirmed. Switch fail only occurs when both confirm switches are on, unless the device detects opens and shorts in the switch wiring.

REFLUX SPLITTER

The reflux splitter is controlled with a standard Analog Output block in YD4. Splitters can be built in many ways, but the AO block provides a standard interface for the reflux ratio setting. The BKCAL_OUT parameter may provide information about the operation of the device, if it has a feedback mechanism like position switches.

AGITATOR

The agitator is controlled by a standard simple Device Control block.

TEMPERATURE MULTIPLEXER

The multiplexer results appear in an 8 output Multiple Analog Input block. They are published by a single MVC.

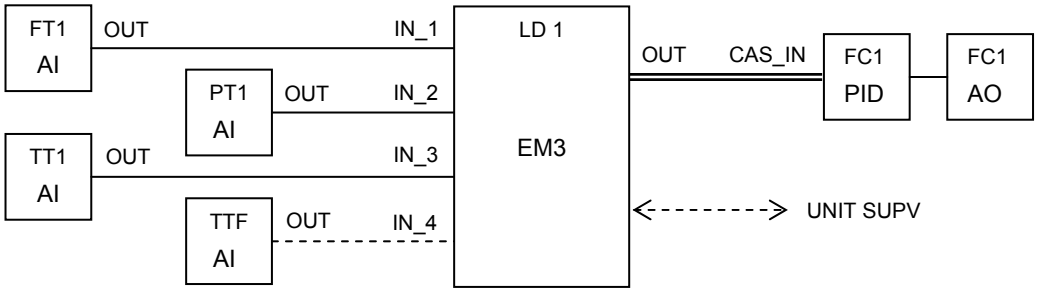
MODULES

Modules define a group of sensors and actuators with a common setpoint that can be set by Unit Supervision. A control module consists of a sensor, controller and actuator, e.g. FT1 and FC1, where Unit Supervision normally just deals with the controller. Equipment modules require at least one FFB to direct the configuration of two or more control modules according to a single command from Unit Supervision. An equipment module controls a sub-task of the unit, reducing the amount of work (and communication traffic) required from Unit Supervision. An equipment module may run one or more equipment phases.

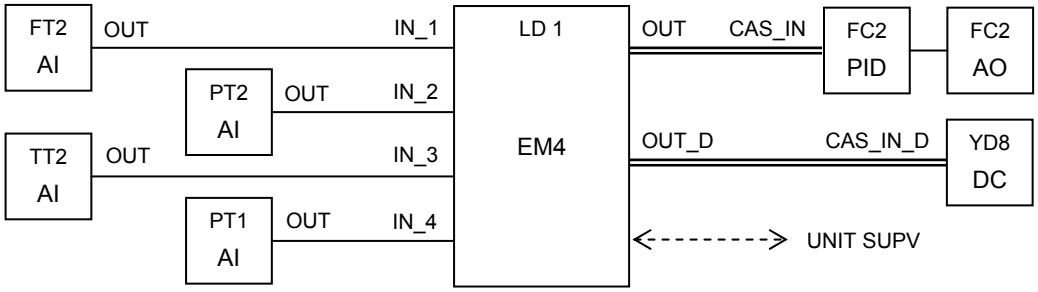
The first equipment module is the set of four valves YD1, YD2, YD3 and YD9. They determine what liquids go into and out of the still pot, not counting the column. This module is controlled by the 4 valve selector on bus 3.

The second equipment module is the set of three valves YD5, YD6 and YD7. They determine the destination for the current distillate cut. This module is controlled by the 4 valve selector on bus 4.

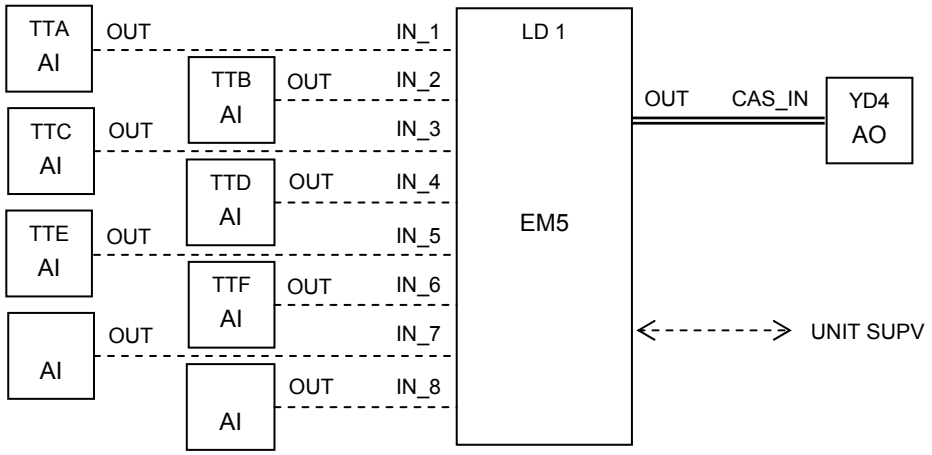
The third equipment module is the column head. The cooling water flow can be set to off, FC1 flow control, cascade to TT1, or anything else that the user finds useful. It may also control the vacuum system.



The fourth equipment module is the still pot. It can be set to off, pre-heat, heat to TT2, heat to column pressure drop PT2 minus PT1. The agitator is also controlled through this module.

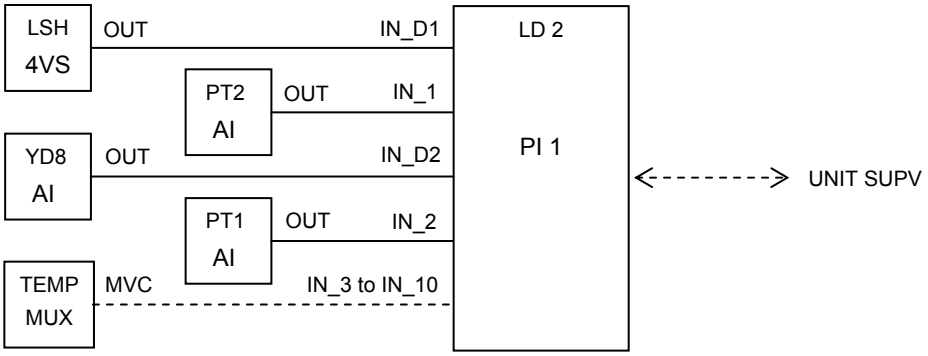


The fifth equipment module contains the column temperature multiplexer and the reflux splitter YD4. The splitter can be set to total reflux, some reflux ratio value, or controlled by a column temperature or the difference between any two column temperatures. The links are made with the Multi-Variable Optimization.



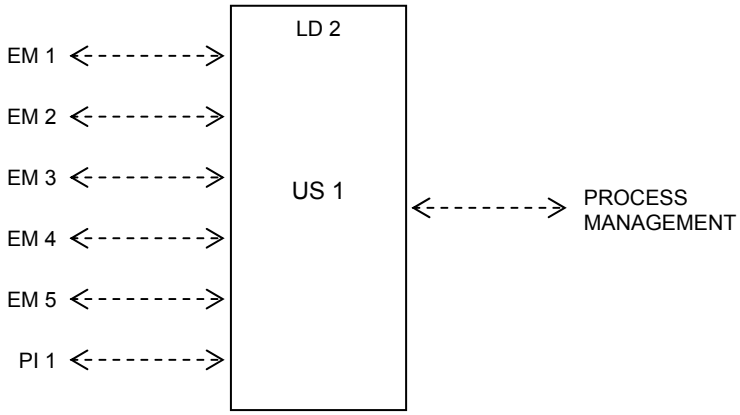
PROCESS INTERLOCKS

An FFB is used to monitor certain conditions. Unit supervision can adjust targets or turn monitors on and off as required. Watchdog timers may be set so that no phase can wait forever. If a value exceeds its limit or a timer expires this block sends an “interrupt” message to unit supervision.



UNIT SUPERVISION

This unit supervisor controls the batch distillation process. It is linked to its equipment modules by the use of Contained parameter Multi-Variable Containers that are published as Reports when required. When US 1 needs to change a setpoint in EM 3, it starts a message timer and publishes the MVC. EM 3 picks up the message as a subscriber, so that the data is written into EM 3. When the command issued by US 1 has been completed, EM 3 publishes a report that is received by US 1. If the status report is OK then the sequence that US 1 is running at the command of process management moves on to the next step.



If the US FFB is in the same physical device with an EM FFB, variables can be read and written without using fieldbus communication. Where communication is required, it is done over the HSE link.

4.4 Variable Frequency Drives (VFD) and FFB Integration Example

The purpose of the Flexible Function Block (FFB) is to provide a means for vendors to provide an interface between FOUNDATION fieldbus devices and other control equipment typically found in Process Plants. The FFB specification identifies four classifications of control integration interfaces this example addresses Programmable FPR (Fixed block par / Fixed set of algorithms). The following example is provided to illustrate how the FFB could be used to integrate FOUNDATION fieldbus based control to a device (drive) that has its own contained control yet has selectable data and is configured

4.4.1 Overview

Variable frequency (or variable speed) Drives have been around for many years, and to a large degree they were independent from the process system. Today the operational and utility cost advantage of integrating the drive with the process control system is being recognized. Drives can be used in many applications to vary the speed of a pump, web, fan, and feed line as an example.

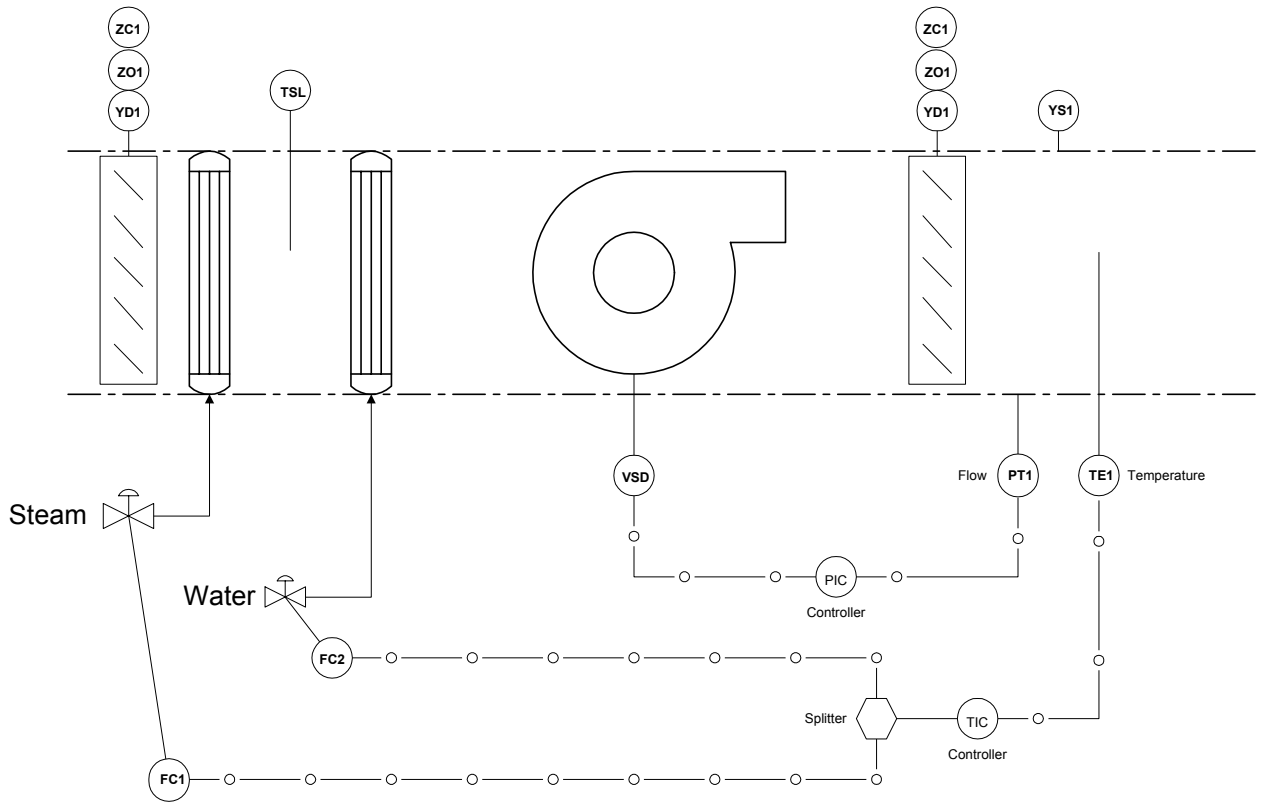
Originally drives would be connected to the control system by hardwiring between the control system I/O to input terminals on the drive with any configuration done at the drive (built in configuration interface). This provided a simple interface between the control system and the drive. Its drawback was that it consumed I/O points and was limited to a few parameters (such as; start / stop, speed set, forward / reverse). Then RS 485/422 ports were added to the drives. This interface was proprietary (required a unique driver and was usually slow) but could provide much more data on the drive's operation. Some of these interfaces provided a way to configure the drive, or the built-in drive panel could be used. Today drive manufactures are adding networks to their drive controllers, such as DeviceNet or vendor specific such as Data Highway. This means that users can access both control and data via the same interface. Tomorrow they will use FOUNDATION fieldbus because of its obvious benefits.

The importance to FFB is that drives are used in process (or process related) control applications that can be as small as a single loop, where a fieldbus device is providing the measurement of the PV and the regulatory control while the drive is manipulating the controlled variable. [When using a drive with FOUNDATION fieldbus the only way is to have a FFB HOST (or Linking Device) connected to a device (DCS/PLC) that then communicates to the drive. Although this works it is a costly solution, both in development effort and in equipment costs. Editor's note: This doesn't seem right. A simple AO block that is permanently configured in the drive will let any existing FF device control its speed. Perhaps it refers to the lack of logic algorithms presently available in the FF standard function block set.] The objective of this example is to describe a typical drive application, providing a scenario using drives with FOUNDATION fieldbus devices.

4.4.2 Process Description

This application is a partial example of an HVAC Air Handler control. The example is centered on the blower and its Variable Speed Drive (VSD). YD1 is an inlet damper that is normally wide open, but closes if TSL detects a low temperature after the heating coil due to some failure. YD2 is an outlet damper that closes if YS1 detects smoke in the air duct, perhaps from failure to lubricate the fan bearings. PT1 senses the pressure in the air duct, usually in inches of water or 0.01 Bar. The VSD turns the fan at the speed necessary to maintain a set duct pressure. The VSD is controlled by the PIC (Pressure Indicating Controller / PID function), which generates a speed set point based on the actual pressure (PV) and the required pressure (SP). The VSD is interlocked with the dampers so that it stops quickly if one of them closes.

Similarly, The temperature is measured by TE1 which is typically a thermocouple and controlled by the combination of the TIC / SPL (Split Range) functions. The control will supply an output to each of the two control valves, properly configured so that both valves can not be open at the same time.



4.4.2.1 Typical Installation Field Wiring

Field devices such as instruments, limit switches, controllers and drives can be connected / hardwired between themselves or to a system's input / output module channels. In most of these cases this field wiring is not H1 FOUNDATION fieldbus. The field wiring and type of connections will vary based on the application, which can range from a simple single drive to integrated multiple coordinated drives.

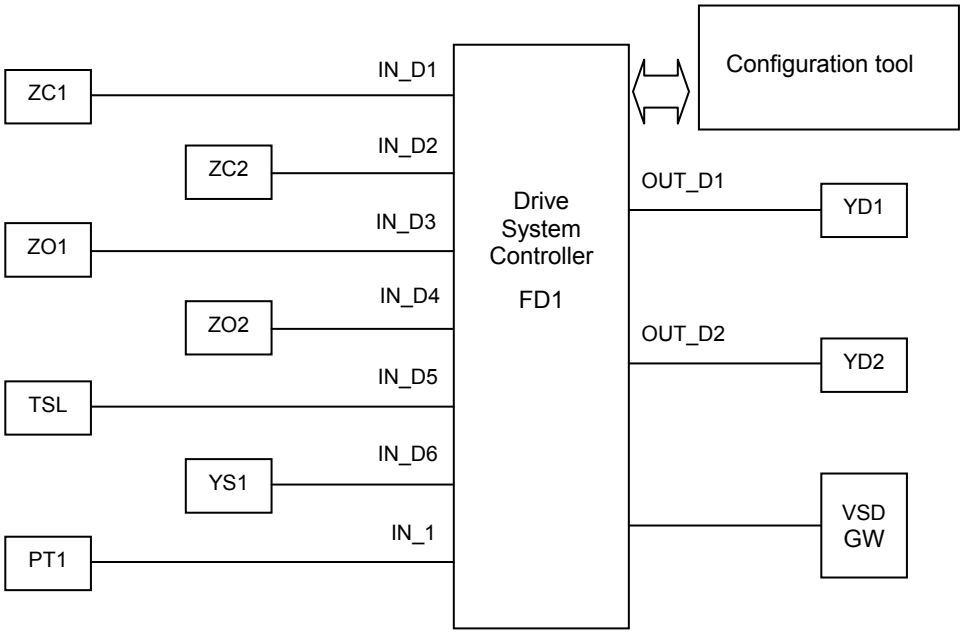
An example of a small "local" hardwired application would be a single loop controller directly connected to the field instruments and drive. Interlocks would be hardwired possibly using relays. A more sophisticated system could be implemented by combining the drive controller with a small PLC that would handle interlocks and the discrete (and in some cases analog) signals connected to its I/O modules. Then as the application requirements increase, creating a need for more complex / integrated control, a large PLC or DCS based system connected to multiple drives would be used. In all of these applications, the constant is that the VFD will need to be connected to both instrumentation and some level of controller. Hardwiring of the field signals becomes costly and does not provide sufficient data transfer.

4.4.2.2 Typical Installation Configuration

With current / typical low-end small or "Local" control applications; a single loop controller could be used. In these installations the instruments are not controllers nor do they support function blocks. Maintenance technicians would use the front panel of the drive and classic signal measurement devices. These installations may also use non-standard signal levels. However, as the applications become larger and more sophisticated, supervisory control systems (either PLC or DCS) are used. In these installations field signals, including the drives, are connected via simple I/O, vendor proprietary networks (Data Highways) or open networks (DeviceNet). Depending on the installation, the Maintenance technicians may have to use one or all of the following: the front panel of the drive, and/or classic signal measurement devices, and /or the control system programming package (for example you could monitor Ladder logic, or an array of data in a Function Block), and depending on the sophistication of the drive there may even be a dedicated software utility.

Note: there is a move today by manufactures of drives to incorporate microprocessors into their drive controllers (smart devices). These units can then control the drive, manage communications (e.g. DeviceNet) and perform interlocking Logic (Ladder). These mini drive systems can be networked together and will be accessible for configuration through system controllers or directly over some network with PC based software.

The simplified diagram below shows the field signals that are associated with our application example connected into a drive system.



The Drive System Controller is linked to the process devices as shown above. It is shown as a field device FD1 and not as a linking device because it has no H1 ports. The labeled input and output parameters are not linked to any FF device. Instead, the simple sensors are hooked to screw terminals on FD1 and the converted values are shown to other FF devices with status in the parameters of FD1. The VSD connection is not labeled because it is proprietary.

Although a simple representation, FD1 could be a drive system function block, which could reside as a communications Function Block in a supervisory controller with the field signals coming from classic I/O. If the field devices are FOUNDATION fieldbus based it could be a Flexible Function Block executing in a FOUNDATION fieldbus device that is inside of the drive cabinet.

In all cases the basic requirements are:

a) the drive must be configured or setup (this can be local or remote)

b) the drive has access to associated field signals (discrete and analog) in a real time manner

c) the drive can interact with a supervisory control system and respond to commands from it

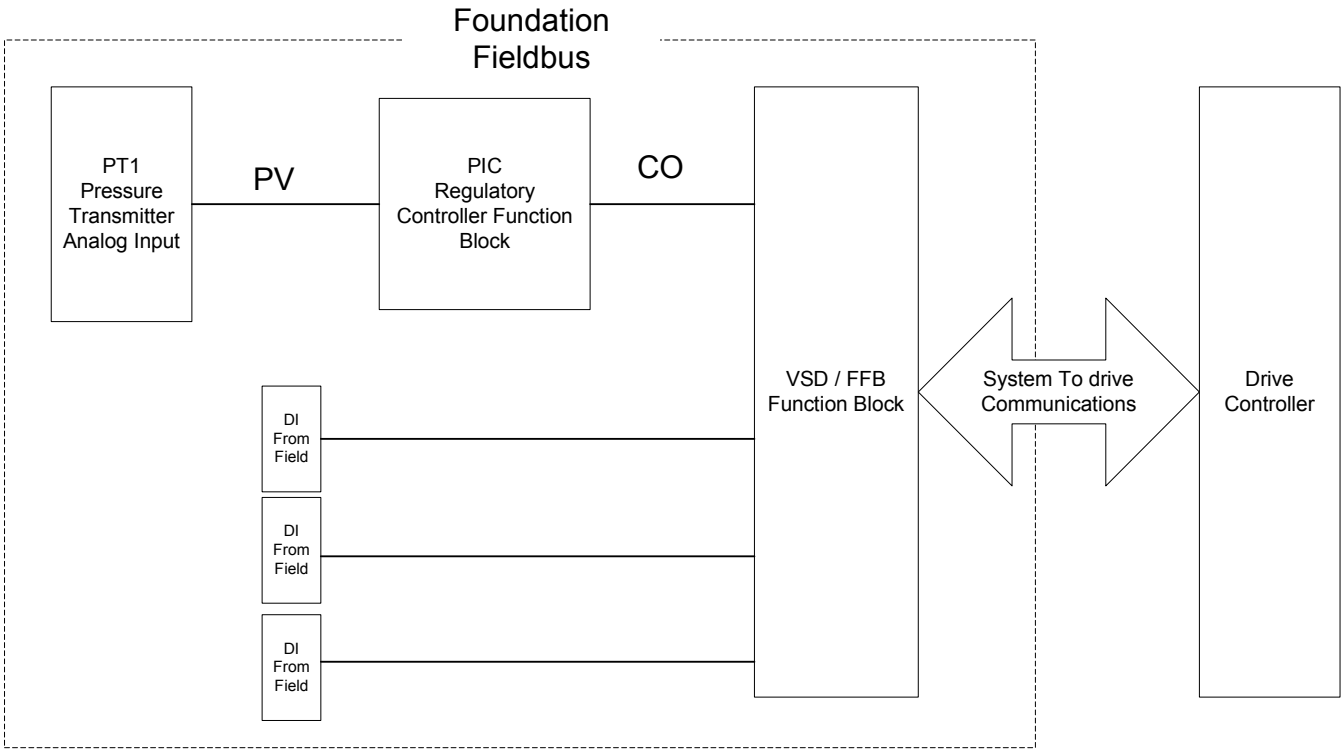
d) the drive can be monitored during operation.

4.4.3 VFD & FOUNDATION™ fieldbus

The temperature control example can have a range of system configurations, which were identified and associated with different types of traditional control systems. As we look to applying FOUNDATION fieldbus technology for drive applications, the interface to the field devices and the related control system paradigms change. Two example approaches to interfacing drives to the process control system are discussed in the context of FOUNDATION fieldbus. The first approach is to use FOUNDATION fieldbus to implement the complete process control strategy (except for any control that is built into the drive itself). The second is to use FFB instrumentation "front ends" to provide the data to a PLC (or DCS) control system.

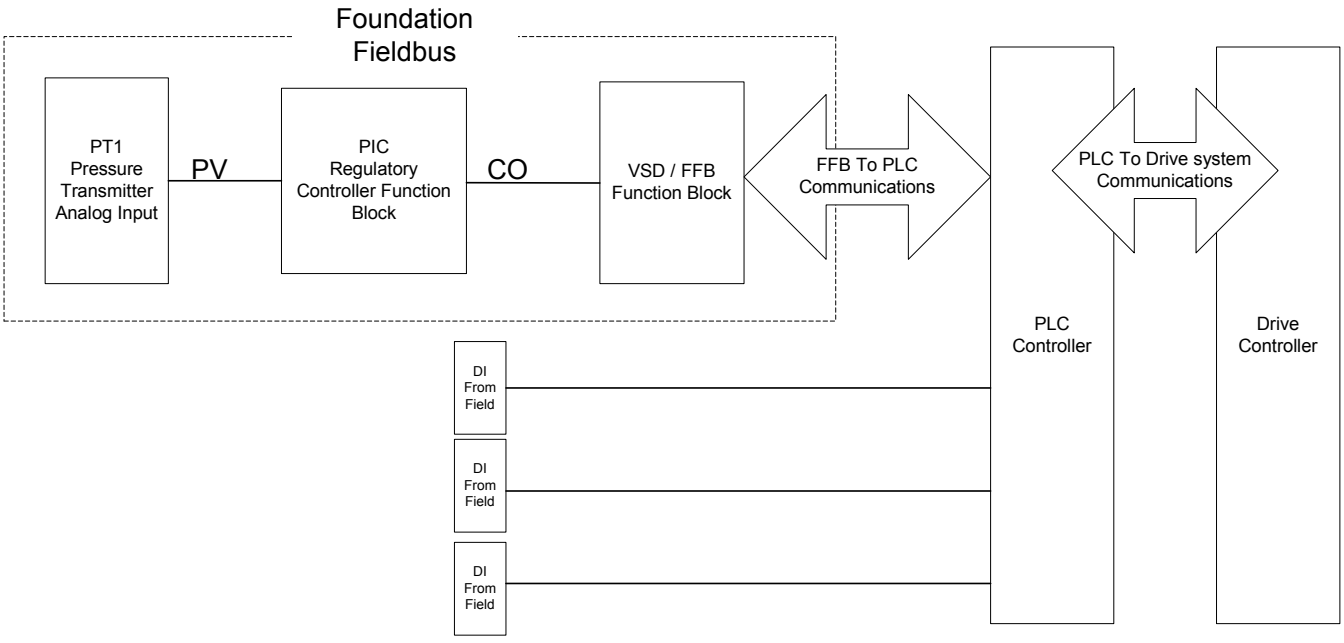
4.4.3.1 Approach I – FOUNDATION™ fieldbus Controlled

In this approach the field devices are either FOUNDATION fieldbus devices or are connected to FOUNDATION fieldbus (both analog and Discrete) via Fieldbus I/O. This would require the use of FOUNDATION fieldbus devices such as Temperature and Pressure instruments, valve actuators and discrete input I/O for non-FOUNDATION fieldbus devices. Within the FOUNDATION fieldbus environment, a Flexible Function block would be used to provide connection to the drive controller. This FFB would enable the FOUNDATION fieldbus device to connect to a communications network (e.g. DeviceNet) to communicate to with the drive.



4.4.3.2 Approach II FOUNDATION™ fieldbus connected to Supervisory system

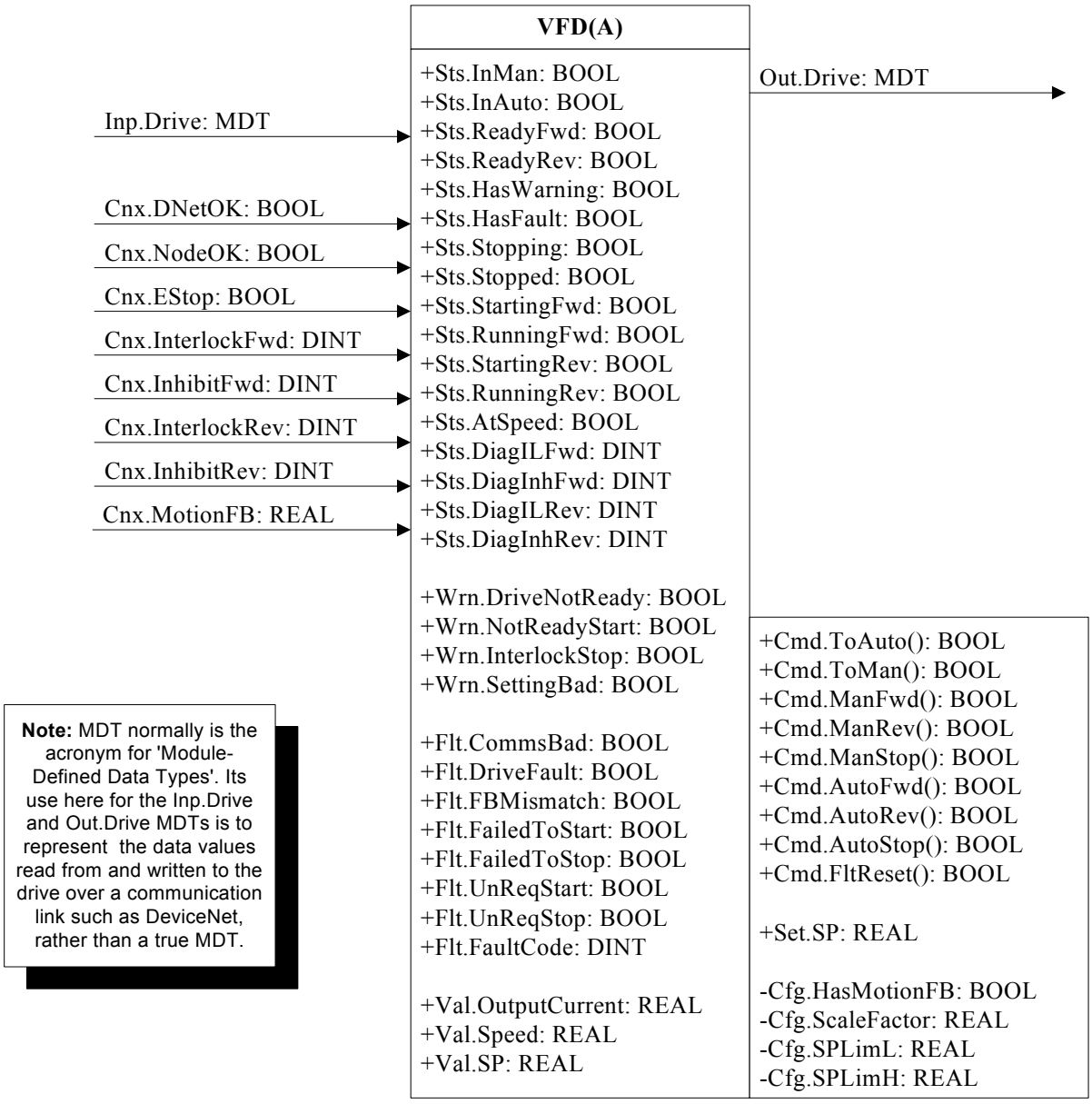
In the second approach the field instrumentation devices are either FOUNDATION fieldbus devices or are connected to FOUNDATION fieldbus (both analog and discrete) via Fieldbus I/O. This would require the use of FOUNDATION fieldbus devices, e.g. Temperature / Pressure instruments, valve actuators. Discrete input I/O for non-FOUNDATION fieldbus devices would be connected directly to the process control system. Within the FOUNDATION fieldbus environment, a Flexible Function Block would be used to provide connection to the supervisory system. This FFB would enable the FOUNDATION fieldbus devices to communicate to a PLC / DCS which would contain interlock logic and in turn communicate to the drive controller via other networks such as DeviceNet, Data Highway, Remote I/O link, and ProfiBus.



4.4.3.3 Example Flexible Function Block for VSD

Drives controllers can have parameters that can number into the hundreds. Providing these within a FFB should be a consideration but not a requirement. The selection of connected / contained parameters should be selectable depending on the application because the control requirements will change. Therefore the specific parameters in a function block can vary based the on the drive equipment and for a given application. However there is a subset of parameters that will be consistent (excluding configuration and the special applications) which should be used as the basis of the Function Block. The other attribute that needs to be considered is that in many systems the connection to the drive will be via a network. For purposes of this example we will assume that the drive would be connected either via a network (e.g. DeviceNet) directly to a drive or to a networked supervisory controller. [Huh?]

The following is an example of a function block layout, providing additional details as to the data / information that could be used to integrate a drive controller with a supervisory system. The supervisory controller, (whether it is FOUNDATION fieldbus or a PLC/ DCS) will require more than just the start / stop commands and depending on the sophistication of the system it will need to connect or have access to considerable data either for operation, monitoring or diagnostics.



Note: MDT normally is the acronym for 'Module-Defined Data Types'. Its use here for the Inp.Drive and Out.Drive MDTs is to represent the data values read from and written to the drive over a communication link such as DeviceNet, rather than a true MDT.

The block may be user configurable (a user can select which data is exposed), but is more likely to be developed (fixed to specific parameters) by the drive's vendor. Its major function is to act as a gateway between FOUNDATION fieldbus instrumentation and control devices and the drive controller. These function blocks would also support configurable connected / contained parameters. The individual discrete and analog I/O associated with the application must also be converted either by FOUNDATION fieldbus I/O or the supervisory system. If the external devices have no status information, they can be displayed in contained variables. The system may have other field devices that may also have other I/O as required to control temperature, humidity, CO2, etc. or detect clogged filters, vibration, etc. These will also have to be considered with the application. Standard alarms are provided for IN_1, the duct pressure input. There is a deviation alarm and one level of absolute alarm. There may be alarms without any standard parameters for drive faults and interlock events

4.4.4 Characteristics Table

This table provides an outline of the basic characteristics for a FFB used with a drive controller. This table compares systems presented above with each of the different architectures. "FOUNDATION fieldbus based" means the process system controller and the instrumentation are Fieldbus devices. "Supervisory based" means there is a PLC / DCS that is being used as a supervisory controller.

Characteristic	FOUNDATION fieldbus based system	Supervisory based system
Capacity		
Performance		
Geographical Dispersion		
Dependability	Availability: Reliability: Safety: Security:	Availability: Reliability: Safety: Security:
Reconfigurability		
Certifiability		
Environmental constraints		
Evolution capability		
FFB Classification	FOD Likely, FPR – Possible, VOD – Not Likely VPR – Not Likely	FOD Possible, FPR - Likley, VOD – Not Likely VPR – Not Likely

4.4.5 Modes of operation

In our example the only supported modes are O/S, Auto and RCas. However there will be a need to provide Manual or jog capability.

4.4.6 Parameter Lists

Block parameters are the data elements that are used to transfer data between the drive and the process controller. This section has been divided into several lists. The first is associated with the example application and provides a listing of the type of parameters that would need to be communicated between the drive and the process system. If the block has only contained parameters, none of the information can be linked to other FF devices, which would be unacceptable in many applications so there would need to be a mechanism to assign / configure needed data as connected of displayed data. [Huh?]

4.4.6.1 Possible FOD parameter list

The block parameter lists below contains parameters typical of a standard SP and Remote Cascade. This applies to the duct pressure setpoint. If this value never changes, it can be one simple contained value. If a host computer does not ever set the setpoint, then the 3 remote cascade parameters can be removed. But if a host does set the value of SP, it is much safer to use the cascade initialization handshake. This forces the host software to look at the device before it writes a value. It is essential if the host has an integrating controller that calculates the value to be written.

Index	Parameter	VIEW _1	VIEW _2	VIEW _3	VIEW _4
1	ST_REV	2	2	2	2
2	TAG_DESC				
3	STRATEGY				2
4	ALERT_KEY				1
5	MODE_BLK	4		4	
6	BLOCK_ERR	2		2	
7	ALGORITHM_SEL				4
8	CONTENTS_REV				4
9	IN_1	5		5	
10	IN_D1	2		2	
11	IN_D2	2		2	
12	IN_D3	2		2	
13	IN_D4	2		2	
14	IN_D5	2		2	
15	IN_D6	2		2	
16	OUT_D1	2		2	
17	OUT_D2	2		2	
18	SHED_OPT_1				1
19	SP_1	5		5	
20	RCAS_IN_1			5	
21	RCAS_OUT_1			5	
22	SP_RATE_DN_1				4
23	SP_RATE_UP_1				4
24	SP_HI_LIM_1		4		
25	SP_LO_LIM_1		4		
26	UPDATE_EVT				
27	BLOCK_ALM				
28	ALARM_SUM	8		8	
29	ACK_OPTION				2
30	ALARM_HYS				4
31	IN_1_HI_PRI				1
32	IN_1_HI_LIM				4
33	IN_1_LO_PRI				1
34	IN_1_LO_LIM				4
35	IN_1_DV_HI_PRI				1
36	IN_1_DV_HI_LIM				4
37	IN_1_DV_LO_PRI				1
38	IN_1_DV_LO_LIM				4
	Subtotals	42	10	52	48

Index	Parameter	VIEW _1	VIEW _2	VIEW _3	VIEW _4
39	HAND_OFF_AUTO	1		1	
40	START	1		1	
41	STOP	1		1	
42	CLEAR_FAULT	1		1	
43	ACTIVE_FAULT	1		1	
44	DRIVE_STATUS	1		1	
45	FREQUENCY			4	
46	VOLTAGE			4	
47	CURRENT			4	
48	POWER			4	
49	TORQUE			4	
50	SPEED	4		4	
51	VOLTS/HERTZ			4	
52	BUS_VOLTS			4	
53	POWER_FACTOR			4	
54	HEATSINK_TEMP			4	
55	ACCEL_RATE				4
56	DECEL_RATE				4
57	MIN_FREQUENCY				4
58	MAX_FREQUENCY				4
59	MAX_MOTOR_AMPS				4
60	CURRENT_LIMIT				4
61	BRAKING_TIME				4
62	BRAKING_VOLTS				4
63	RESTART_TRIES				4
64	RESTART_DELAY				4
65	COMPENSATION				4
66	IN_1_HI_ALM				
67	IN_1_LO_ALM				
68	IN_1_DV_HI_ALM				
69	IN_1_DV_LO_ALM				
	Subtotals	10	0	46	44
	Totals	52	10	98	92

4.4.6.2 Possible FPR list of parameters

The following is a partial listing of possible parameters that can be associated with drives, but organized in a FPR (Fixed parameters Programmable Algorithm) format. These are being provided as examples of the type of parameters that will need to be considered in the implementation of a FFB. The listing also provides an approach to for the user to assign or configure specific data into predefined parameter sections called "Datalink". Some drives can be configured to provide selectable data via the communications network. Using this capability a fixed block can be established yet there is the ability to customize the data that is visible on the network.

4.4.6.2.1 Input (to the process system) Parameter Block set up with configurable data

Name	Variable Name	Type	Description
Drive Enabled	VFDA_Prv.Inp.Enabled	BOOL	Drive Enabled (1 = Enabled)
Drive Running	VFDA_Prv.Inp.Running	BOOL	Drive Running (1 = Running)
Command Direction	VFDA_Prv.Inp.CmdDir	BOOL	Command Direction (1 = Forward, 0 = Reverse)
Rotation Direction	VFDA_Prv.Inp.RotDir	BOOL	Rotation Direction (1 = Forward, 0 = Reverse)
Acceleration	VFDA_Prv.Inp.Accelerating	BOOL	Acceleration (1 = Accelerating)
Deceleration	VFDA_Prv.Inp.Decelerating	BOOL	Deceleration (1 = Decelerating)
Warning	Prv.Inp.Wrn	BOOL	Warning (1 = Warning Present)
Faulted	Prv.Inp.Flt	BOOL	Faulted (1 = Faulted)
At Speed	Prv.Inp.AtSpeed	BOOL	At Speed Reference (1 = At Speed)
Local 0	Prv.Inp.Local0	BOOL	3-bit value to identify controlling port
Local 1	Prv.Inp.Local1	BOOL	3-bit value to identify controlling port
Local 2	Prv.Inp.Local2	BOOL	3-bit value to identify controlling port
Reference Source 0	Prv.Inp.LogicSts.12	BOOL	4-bit value to identify the Reference Source
Reference Source 1	Prv.Inp.LogicSts.13	BOOL	4-bit value to identify the Reference Source
Reference Source 2	Prv.Inp.LogicSts.14	BOOL	4-bit value to identify the Reference Source
Reference Source 3	Prv.Inp.LogicSts.15	BOOL	4-bit value to identify the Reference Source
Speed Feedback	Prv.Inp.SpeedFB	INT	Speed Feedback from VFD (Drive Units – 32767 = MaxFreq)
DataLink A1	Prv.Inp.DLA1	INT	DataLink parameter received from VFD (contents are configurable in VFD)
DataLink A2	Prv.Inp.DLA2	INT	DataLink parameter received from VFD (contents are configurable in VFD)
DataLink B1	Prv.Inp.DLB1	INT	DataLink parameter received from VFD (contents are configurable in VFD)
DataLink B2	Prv.Inp.DLB2	INT	DataLink parameter received from VFD (contents are configurable in VFD)
DataLink C1	Prv.Inp.DLC1	INT	DataLink parameter received from VFD (contents are configurable in VFD)
DataLink C2	Prv.Inp.DLC2	INT	DataLink parameter received from VFD (contents are configurable in VFD)
DataLink D1	Prv.Inp.DLD1	INT	DataLink parameter received from VFD (contents are configurable in VFD)
DataLink D2	Prv.Inp.DLD2	INT	DataLink parameter received from VFD (contents are configurable in VFD)

4.4.6.2.2 Output (to the drive controller) Parameter Block set up with configurable data

Name	Variable Name	Type	Description
Stop Command	.Out.Stop	BOOL	Commands the VFD to Stop (1 = stop, 0 = no operation)
Start Command	.Out.Run	BOOL	Commands the VFD to Run at the specified direction and speed (1 = start, 0 = no operation)
Jog Command	.Out.Jog	BOOL	Commands the VFD to Jog at the specified direction and speed (1 = jog, 0 = no operation)
Clear Faults Command	.Out.ClrFlt	BOOL	Commands the VFD to clear its faults (1 = clear faults, 0 = no operation)
Direction FWD Cmd	.Out.DirFwd	BOOL	Commands the VFD to run Forward (1 = Forward, 0 = allows Reverse)
Direction REV Cmd	.Out.DirRev	BOOL	Commands the VFD to run Reverse (1 = Reverse, 0 = allows Forward)
Local Command	.Out.Local	BOOL	1 = Local, 0 = Remote
MOP Increment	.Out.IncMOP	BOOL	1 = Increment MOP, 0 = no operation
Accel Rate Select	.Out.AccelSel1	BOOL	Two bits for Accel Rate: 00 = no operation, 01 = Accel Rate 1, 10 = Accel Rate 2.
…	.Out.AccelSel2	BOOL	
Decel Rate Select	.Out.DecelSel1	BOOL	Two bits for Decel Rate: 00 = no operation, 01 = Decel Rate 1, 10 = Decel Rate 2
…	.Out.DecelSel2	BOOL	
Reference Select	.Out.RefSel0	BOOL	Three bits for Reference Select: 000 = no operation, 001 = Freq Select 1 (par. 5)
…	.Out.RefSel1	BOOL	
…	.Out.RefSel2	BOOL	
MOP Decrement	.Out.DecMOP	BOOL	1 = Decrement MOP, 0 = no operation
Setpoint	.Out.SpeedSP	INT	Speed Setpoint to VFD (Drive Units – 32767 = MaxFreq)
DataLink A1	.Out.DLA1	INT	DataLink parameter sent to VFD (contents are configurable in VFD)
DataLink A2	.Out.DLA2	INT	DataLink parameter sent to VFD (contents are configurable in VFD)
DataLink B1	.Out.DLB1	INT	DataLink parameter sent to VFD (contents are configurable in VFD)
DataLink B2	.Out.DLB2	INT	DataLink parameter sent to VFD (contents are configurable in VFD)
DataLink C1	.Out.DLC1	INT	DataLink parameter sent to VFD (contents are configurable in VFD)
DataLink C2	.Out.DLC2	INT	DataLink parameter sent to VFD (contents are configurable in VFD)
DataLink D1	.Out.DLD1	INT	DataLink parameter sent to VFD (contents are configurable in VFD)
DataLink D2	.Out.DLD2	INT	DataLink parameter sent to VFD (contents are configurable in VFD)

The following set of parameters is an example of the information that the FFB would generate for use in the control system status display.

4.4.6.2.3 State Parameter Block

Name	Variable Name	Type	Description
	Prv.St	DINT	State Word (bits represent states)
0 (Reserved)	Prv.St.0	Bit in DINT	State 0: (reserved)
1 (Stopping)	Prv.St.1	Bit in DINT	State 1: VFD has been commanded to Stop, waiting for Stopped status from VFD.
2 (Stopped)	Prv.St.2	Bit in DINT	State 2: VFD reports Stopped status.
3 (StartingFWD)	Prv.St.3	Bit in DINT	State 3: VFD has been commanded to Run Forward, waiting for Running Forward status from VFD.
4 (RunningFWD)	Prv.St.4	Bit in DINT	State 4: VFD reports Running Forward status (not necessarily At Speed)
5 (Faulted)	Prv.St.5	Bit in DINT	State 5: Fault Condition detected such as a VFD Fault or an unexpected status from VFD.
6 (StartingREV)	Prv.St.6	Bit in DINT	State 6: VFD has been commanded to Run Reverse, waiting for Running Reverse status from VFD.
7 (RunningREV)	Prv.St.7	Bit in DINT	State 7: VFD reports Running Reverse status (not necessarily At Speed)

The following set of parameters is an example of the information that the FFB would generate for alarms, presented as a parameter similar to ALARM_SUM.

4.4.6.2.4 Event Parameter Block

Name	Variable Name	Type	Description
None	Prv.Ev	DINT	Event Word (bits represent events)
0 (Transition)	Prv.Ev.0	Bit in DINT	Event 0: a state transition is occurring
1 (DriveNotBusy)	Prv.Ev.1	Bit in DINT	Event 1: The Drive Running logic status bit from the VFD = 0
2 (DriveBusy)	Prv.Ev.2	Bit in DINT	Event 2: The Drive Running logic status bit from the VFD = 1
3 (FailureTimer)	Prv.Ev.3	Bit in DINT	Event 3: FailureTimer.Dn = 1
4 (StopCommand)	Prv.Ev.4	Bit in DINT	Event 4: The Stop Command for the current mode = 1
5 (FWDCommand + ReadyFWD)	Prv.Ev.5	Bit in DINT	Event 5: The ManFwd or AutoFwd command (matching the current mode) was received while the VFD is Ready to Run Forward.
6 (FltResetCommand)	Prv.Ev.6	Bit in DINT	Event 6: The Fault Reset command was received.
7 (LossOfFwdInterlock)	Prv.Ev.7	Bit in DINT	Event 7: At least 1 bit in Cnx.InterlockFwd = 1
8 (DriveFault)	Prv.Ev.8	Bit in DINT	Event 8: The VFD reported a Fault condition
9 (EStop)	Prv.Ev.9	Bit in DINT	Event 9: Cnx.EStop = 0
10 (MotionMismatch)	Prv.Ev.10	Bit in DINT	Event 10: Motion Feedback is configured, but the value does not match the current operation for speed and direction.
11 (REVCommand + ReadyREV)	Prv.Ev.11	Bit in DINT	Event 11: The ManRev or AutoRev command (matching the current mode) was received while the VFD is Ready to Run Reverse.
12 (LossOfRevInterlock)	Prv.Ev.12	Bit in DINT	Event 12: At least 1 bit in Cnx.InterlockRev = 1
13 (LossOfComms)	Prv.Ev.13	Bit in DINT	Event 13: DeviceNet Comms Not OK.