USER MANUAL

Dual Valve Controller
4 – Universal Inputs, 2 – 3A Outputs
CAN (SAE J1939)

P/N: AX021800 (with 4 - +5V References, 250 kBit/s)
P/N: AX021801 (with 4 - +5V References, 500kBit/s)
P/N: AX021802 (with 4 - +5V References, 1Mbit/s)

P/N: AX021810 (without Voltage References, 250kBit/s)
P/N: AX021811 (without Voltage References, 500kBit/s)
P/N: AX021812 (without Voltage References, 1Mbit/s)

Axiomatic Technologies Oy
Höytämöntie 6
33880 LEMPÄÄLÄ, Finland
Tel. +358 103 375 750
salesfinland@axiomatic.com
www.axiomatic.fi

Axiomatic Technologies Corporation
1445 Courtneypark Dr. E.
Mississauga, ON Canada L5T 2E3
Tel. 1 905 602 9270
sales@axiomatic.com
www.axiomatic.com

ACRONYMS

CAN	Controller Area Network
CANopen®	CAN-based higher layer protocol supported by CAN in Automation (CiA)
DM	Diagnostic message. Defined in J1939/73 standard
EA	Axiomatic Electronic Assistant. PC application software from Axiomatic, primarily designed to view and program Axiomatic control setpoints through CAN bus using J1939 Memory Access Protocol
ECU	Electronic control unit
EMI	Electromagnetic Interference
LSB	Less Significant Byte
NTC	Negative Temperature Coefficient
PC	Personal Computer
PGN	Parameter Group Number. Defined in J1939/73 standard
PID	Proportional–integral–derivative (regulator)
PTC	Positive Temperature Coefficient
PWM	Pulse-width modulation
RS232	PC serial port interface
RTD	Resistance Temperature Detector
SAE J1939	CAN-based higher-level protocol designed and supported by Society of automobile Engineers (SAE)
USB	Universal Serial Bus.
UTP	Un-shielded twisted pair

TABLE OF CONTENTS

1 INTRODUCTION

The following User Manual describes architecture, functionality, and application programming of the 4 Universal Input 2 – 3A Output Dual Valve Controller. It also contains technical specification and installation instructions to help users build a custom solution on the base of this controller.

The user should check whether the application firmware installed on the controller is covered by this user manual. It can be done using Axiomatic EA software. The user manual is valid for application firmware with the same major version number as the user manual. For example, this user manual is valid for any controller application firmware V8.xx. Updates specific to the user manual are done by adding letters to the user manual version number, see Version History section.

The controller supports SAE J1939 CAN interface. It is assumed, that the user is familiar with the J1939 group of standards; the terminology from these standards is widely used in this manual. Support for CANopen and other high-level CAN protocols can be available on request.

The baud rate of the CAN interface is not adjustable. The user should order the controller part number with the necessary baud rate. Application firmware for a unit with one baud rate cannot be flashed in a unit with a different baud rate.

2 CONTROLLER DESCRIPTION

The controller is designed to independently control two proportional or on/off solenoid valves using PWM control from a variety of input sources. It accepts voltage, current, resistance, frequency, PWM, and discrete levels from four universal inputs. Signals transmitted on the CAN bus can also be used as input sources.

Besides reading signals transmitted on the CAN bus, the controller can also transmit CAN messages carrying signals internally generated by the controller. They include values acquired from the universal inputs, output currents, output fault conditions, etc.

Due to high versatility of the controller, it can be used, with some minor restrictions, to control not only solenoid valves but also non-inductive loads, for example: automotive lamps. It can also act as a signal to CAN converter, converting voltage, current, etc. from universal inputs to CAN messages.

2.1 Hardware Block Diagram

The controller contains: four independent universal input channels, two PWM outputs to drive proportional or on-off valves up to 3A each, four +5V reference voltage power supplies (only for P/N: AX021800, AX021801 and AX021802) and a CAN port. An embedded 32-bit microcontroller provides necessary functionality of the controller.

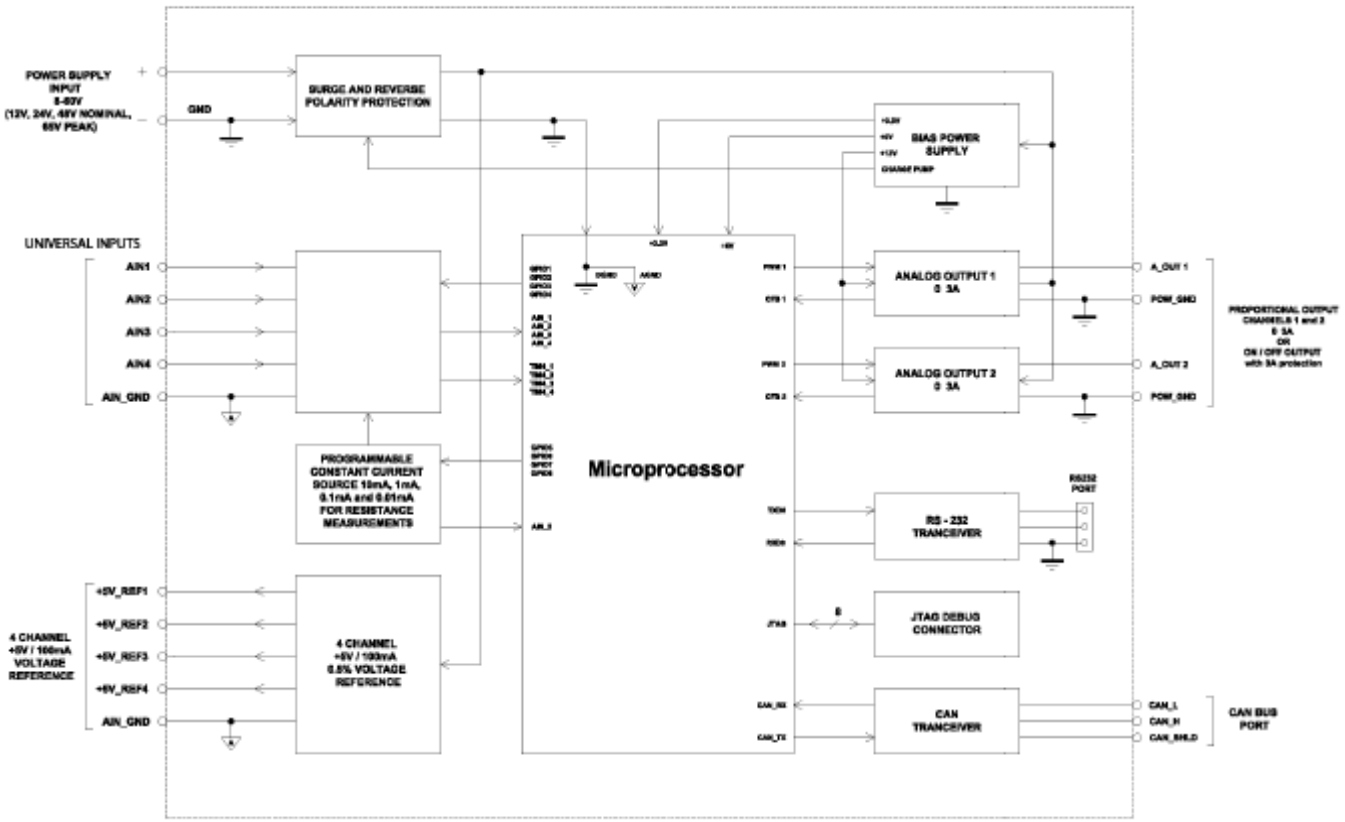


Figure 1. The Controller Hardware Block Diagram.

The controller has a wide range of build-in protections features. The controller power supply has a transient and reverse polarity protection, all PWM outputs have short circuit and power supply rail protection, +5V reference voltage power supplies have overvoltage and overcurrent protection (only for P/N: AX021800, AX021801 and AX021802).

2.2 Software Organization

The 4 Universal Input 2 – 3A Output Dual Valve Controller belongs to a family of Axiomatic smart controllers with programmable internal architecture. This architecture allows building a control algorithm based on a set of predefined internal configurable function blocks without the need for custom software.

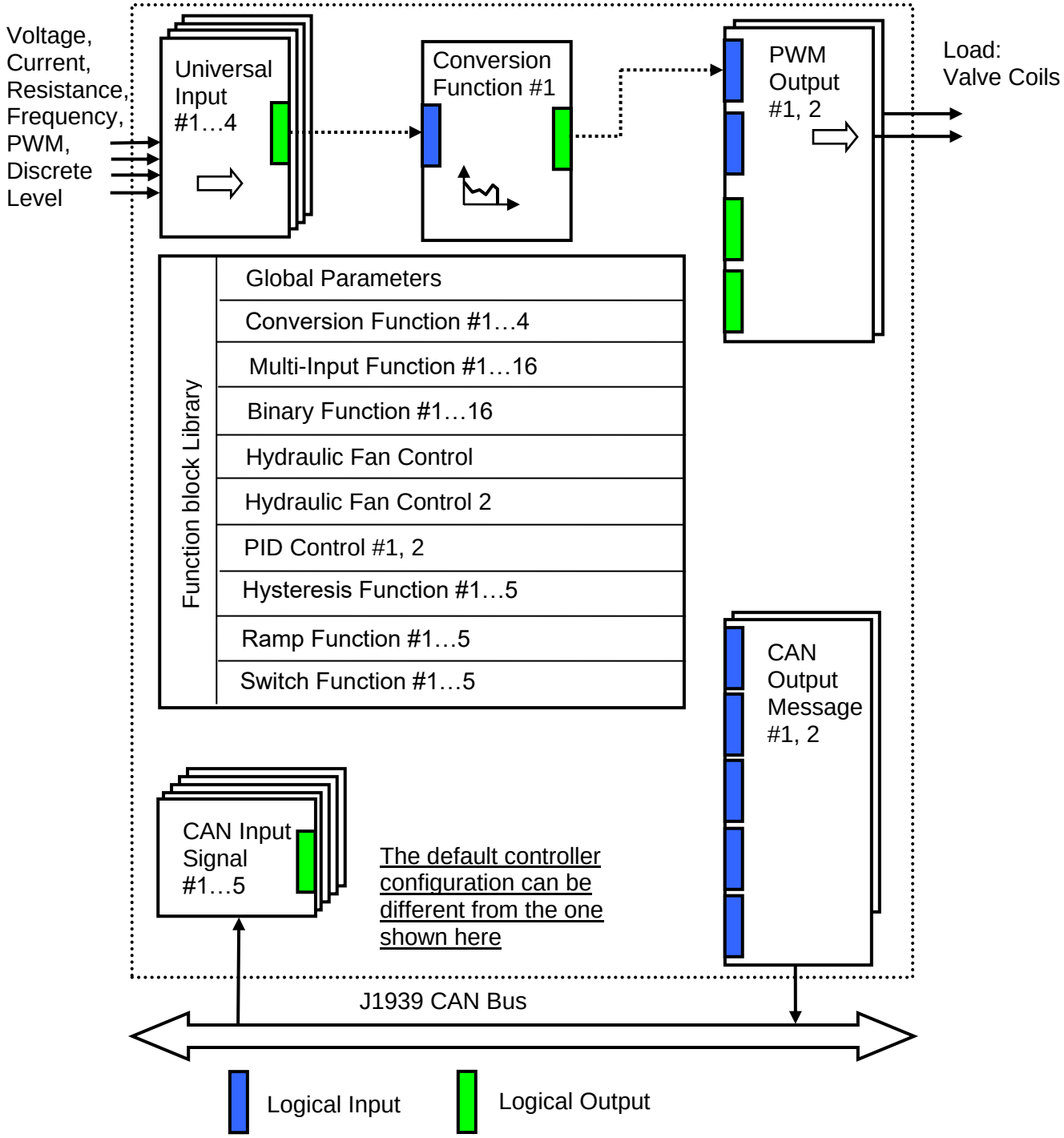
In addition to generic function blocks, the controller has a set of two application specific function blocks designed for hydraulic fan control applications.

The user can set the controller structure and configure the controller function blocks using PC-based Axiomatic Electronic Assistant (EA) software through CAN interface, without disconnecting the controller from the user's system.

Starting from the firmware version 8.00 and EA 4.4.42.0, the controller application firmware can be updated the same way using EA in the field, see Flashing New Firmware section.

3 CONTROLLER ARCHITECTURE

From the software perspective, the controller consists of a set of internal function blocks, which can be individually programmed and arbitrarily connected together to achieve the required system functionality, Figure 2.



As an example, the Universal Inputs #1 function block is connected to the Conversion Function #1 and the Conversion Function #1 is connected to the PWM Output #1 function block, providing a path for the input signal from input to output through the Conversion Function #1 function block.

Figure 2. The Controller Internal Structure

Each function block is absolutely independent and has its own set of programmable parameters, or setpoints. The setpoints can be viewed and changed through CAN using Axiomatic Electronic Assistant (EA) software.

3.1 Function Blocks

There are two types of the controller function blocks. One type represents the controller hardware resources, for example: universal inputs or PWM outputs. The other type is purely logical – these function blocks are included to program the user defined functionality of the controller. The number and functional diversity of these function blocks are only limited by the system resources of the internal microcontroller. They can be added or modified on the customer's request to accommodate user-specific requirements.

The user can build virtually any type of a custom control by logically connecting inputs and outputs of the function blocks. This approach gives the user an absolute freedom of customization and an ability to fully utilize the controller hardware resources in a user's application.

Depending on the block functionality, a function block can have: logical inputs, logical outputs or any combinations of them. The connection between logical inputs and outputs is defined by logical input setpoints. The following rules apply:

- A logical input can be connected to any logical output using a logical input setpoint.
- Two or more logical inputs can be connected to one logical output.
- Logical outputs do not have their own setpoints controlling their connectivity. They can only be chosen as signal sources by logical inputs.

To provide data flow between logical inputs and outputs, all logical outputs are normalized to [0;1] data range using the following equation:

$$Yn = (Y - Ymin) / (Ymax - Ymin),$$

where: Yn – normalized output value,
Y – original output value,
Ymax – maximum output value,
Ymin – minimum output value.

The original output values are restored, if necessary, at the logical inputs using the following reverse linear transformation:

$$X = Xn \bullet (Xmax - Xmin) + Xmin,$$

where: X – original restored input value,
Xn – normalized input value, Xn=Yn (input is connected to the output),
Xmax – maximum input value, Xmax=Ymax,
Xmin – minimum input value, Xmin=Ymin.

All function blocks have (Xmax, Xmin) and (Ymax, Ymin) setpoint pairs controlling the normalization process. They will be called "normalization parameters" further in the setpoint descriptions.

For discrete logical inputs and outputs, the normalization parameters are not required, since the discrete signals can take only two values: {0,1}. When a regular logical output of a function block is connected to a discrete logical input, it is assumed that the input values below 0.5 represent state 0 and above 0.5 – state 1:

Discrete Logical Input	Logical State
< 0.5	0
≥ 0.5	1

For additional flexibility, in a majority of function blocks, logical input signals can be inverted using the following inversion function:

$Xn' = Inv(Xn,I)$, $I \in \{Yes,No\}$,

$Inv(Xn,I)=\{1-Xn$, if $I=Yes$; Xn, if $I=No\}$

In addition to signal values in the range of [0;1], the logical inputs and outputs also carry information on the state of the data source. This information can show that the source is not available or there is an error in data, or the data source is in a special state.

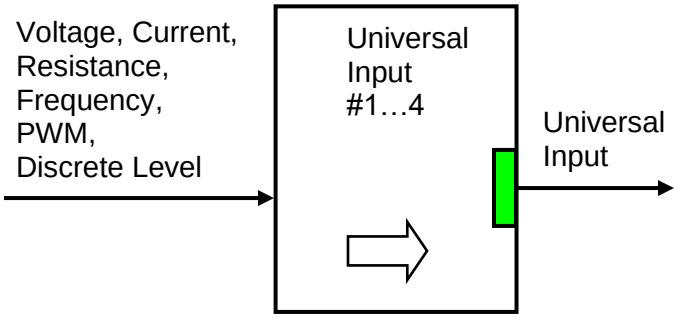
When the data source does not carry a valid data, the output signal value is always set to 0 and the inversion function also returns 0 independently whether the inversion was set or not. In this case, instead of the signal value, the logical signal carries a signal state code, associated with its signal state, see the table below:

Signal State	Signal Value, Xn	Signal State Code	Inverted Signal Value	
			Xn'=Inv(Xn,Yes)	Xn'=Inv(Xn,No)
Valid Data	[0;1]	0	1-Xn	Xn
Special	0	0…4294967295 (0…0xFFFFFFFF) – Special State Code	0	0
Error	0	0…4294967295 (0…0xFFFFFFFF) – Error Code	0	0
Not Available	0	0	0	0

The states of the data source other than the “Valid Data” are primary used by CAN function blocks to report that a CAN input signal is absent on the bus, is out of range, etc. Other function blocks usually use only the “Error” state to show an error condition.

3.2 Universal Input

There are four Universal Input function blocks, each controlling its own universal physical input of the controller.



A Universal Input function block has one logical output providing a normalized input signal from the physical input to other function blocks of the controller. The function block setpoints are presented in the following table:

Name	Default Value	Range	Units	Description
Input Parameter[1]	Voltage	{Input Disabled, Voltage, Current, Resistance, Discrete Voltage Level, Frequency, PWM Duty Cycle}	–	Type of the universal input electrical parameter to be measured
Voltage Range[4]	0…5V	{0…10V, 0…5V, 0…2.5V, 0…1V}	–	Signal range for Voltage measurements
Current Range[4]	0…20mA	{0…20mA, 4…20mA}	–	Signal range for Current measurements
Resistance Range[2,4,6]	Auto Range	{ Auto Range, 0…150Ohm, 0…2kOhm, 0…20kOhm, 0…250kOhm}	–	Signal range for Resistance measurements
Frequency Range[3,4]	10Hz…1kHz	{10Hz…1kHz, 100Hz…10kHz}	–	Signal frequency range for Frequency and PWM Duty Cycle measurements
Pull-Up/Pull-Down Resistor	Disabled	{Disabled, 10kOhm Pull-Up, 10kOhm Pull-Down}	–	Connection of the pull-up/pull-down resistor for: Discrete Voltage Level, Frequency and PWM Duty Cycle measurements
Analog Input Filter	Both: 50Hz and 60Hz noise rejection	{Disabled, 50Hz noise rejection, 60Hz noise rejection, Both: 50Hz and 60Hz noise rejection}	–	Input filter for: Voltage, Current and Resistance measurements
Debounce Input Filter	1.78µs	{Disabled, 111ns, 1.78µs, 14.22µs}	–	Debounce input digital filter for Frequency and PWM Duty Cycle measurements
Digital Input Polarity	Active High	{Active High, Active Low}	–	Input polarity for Discrete Voltage Level and PWM Duty Cycle measurements
Vmax – Maximum Input Voltage	5.0	[0…10], but Vmax>Vmin	V	Normalization parameters for Voltage measurements
Vmin – Minimum Input Voltage	0.0	[0…10], but Vmin<Vmax	V	
Imax – Maximum Input Current	20.0	[0…20], but Imax>Imin	mA	Normalization parameters for Current measurements

Name	Default Value	Range	Units	Description
Imin – Minimum Input Current	0.0	[0…20], but Imin<Imax	mA	
Rmax – Maximum Input Resistance	250.0	[0…250], but Rmax>Rmin	kOhm	Normalization parameters for Resistance measurements
Rmin – Minimum Input Resistance	0.0	[0…250], but Rmin<Rmax	kOhm	
Fmax – Maximum Input Frequency[5]	1000.0	[0…10000], but Fmax>Fmin	Hz	Normalization parameters for Frequency measurements
Fmin – Minimum Input Frequency[5]	0.0	[0…10000], but Fmin<Fmax	Hz	
Dmax – Maximum Duty Cycle	100.0	[0…100], but Dmax>Dmin	%	Normalization parameters for PWM Duty Cycle measurements
Dmin – Minimum Duty Cycle	0.0	[0…100], but Dmin<Dmax	%	

[1] Due to hardware limitations, the following restrictions apply:

- PWM Duty Cycle or Frequency input mode can be configured only on two (out of four) universal inputs: input #1 and #3.
- If PWM Duty Cycle or Frequency mode is chosen on input #1, all other inputs cannot be used as analog inputs (for measuring Voltage, Current or Resistance). In this case, the default value of the Input Parameter is set to Input Disabled.

[2] Resistance below 20 Ohm is measured as 0 Ohm when the Resistance Range is set to Auto Range or 0…150Ohm range.

[3] Frequencies below 9.5Hz for 10Hz…1kHz range (95Hz for 100Hz…10kHz range) are measured as 0 Hz.

[4] Signal range should comply with normalization parameters. Setting, for example, voltage range to 0…1V and Vmin=5V, Vmax=10V will result in the logical output being equal to 0.0 independently of the input voltage.

[5] Normalization parameters for Frequency measurements do not apply to PWM duty cycle measurements and do not affect choosing the Frequency Range in this mode.

[6] In the majority of cases, users should use the Auto Range for the Resistance measurements. For some applications, to increase resolution and accuracy, a manual selection of the resistance range is preferable.

3.2.1 Voltage Input

To acquire a voltage signal, the user should set up: Input Parameter – to Voltage, Voltage Range – to the expected signal range, Vmin and Vmax – to the minimum and maximum voltage acquired by the function block.

Usually, Vmin and Vmax are set to cover the entire signal range. For example, for Voltage Range equal to 0…5V: Vmin=0 [V] and Vmax=5 [V]. For some applications, however, they can be set inside the signal range. For example, if there is a +5V potentiometer input, setting Vmin=0.1[V] and Vmax=4.9 [V] will ensure that the minimum and maximum potentiometer positions will be clearly identified.

The voltage signal, as well as all other analog signals, is sampled every 1.1(1) ms. By default, it is filtered by the running average filter, which is set up using the Analog Input Filter setpoint. The parameters of the filter are provided below:

Analog Input Filter	Number of points	Averaging Period [ms]
Disabled	-	-

Analog Input Filter	Number of points	Averaging Period [ms]
50Hz noise rejection	18	20
60Hz noise rejection	15	16.6(6)
Both: 50Hz and 60Hz noise rejection	90	100

3.2.2 Current Input

The current signal is acquired the same way as a voltage signal. The user should set up: Input Parameter – to Current, Current Range – to the expected current signal range, Imin and Imax – to the minimum and maximum current that will be output as a logical signal by the function block.

The user should also define the filter parameter using the Analog Input Filter setpoint.

Please, remember that the unit acquires current by measuring a voltage drop on an internal reference resistor. The value of this resistor provided in the Technical Specification should be within an acceptable range for the current source.

3.2.3 Resistance Input

The Universal Input function block can be set up to measure resistance by setting the Input Parameter setpoint to Resistance, Resistance Range to Auto Range or a specific range and Rmin, Rmax normalization parameters to the required resistance range.

Analog input filter is also used for resistance measurements. It is recommended that the Analog Input Filter setpoint be set to the value rejecting both: 50Hz and 60Hz industrial noise.

When the Resistance Range setpoint is set to the Auto Range, a special algorithm is used to maintain monotonicity of the conversion function during switching between resistance ranges. The actual resistance range used for measuring resistance in the Auto Range mode can be found in the following table:

Range	Resistance in the Auto Range Mode
0…150 Ohm	<100 Ohm
0…2 kOhm	100 Ohm …1.1 kOhm
0…20 kOhm	1.1 kOhm …11.1 kOhm
0…250 kOhm	>11.1 kOhm

When the range switching occurs between the [Rmin; Rmax] in the Auto Range mode, it is sometimes possible to increase accuracy and resolution by keeping one specific range for the entire [Rmin; Rmax] input range. For example, if Rmin=80 Ohm and Rmax=130 Ohm, it is recommended that the user change Resistance Range setpoint from Auto Range to 0…150 Ohm range.

3.2.4 Frequency and PWM Input

The user can set up the Universal Input to measure frequency or PWM input signal using the Input Parameter setpoint. The user should define the frequency range of the input signal by the Frequency Range setpoint and set up the Fmin, Fmax or Dmin, Dmax normalization parameters.

The polarity of the input signal is set up by the Digital Input Polarity setpoint. The user can also apply a pull-up or pull-down resistor by the Pull-Up/Pull-Down Resistor setpoint and change the

debounce input filter settings using the Debounce Input Filter setpoint to filter out parasitic spikes that can be present in the noisy input signal.

Be aware, that the debounce filter settings can affect accuracy of the frequency and PWM signal acquisition at the high frequency. For example, for the 10 kHz PWM signal, setting the Debounce Input Filter to 14.22µs will result in the 14.22% additional error in the output data.

For the Frequency and PWM Duty Cycle input modes the Universal Input function block will output an error code if the frequency of the input signal is beyond the selected frequency range. The signal value, in this case, will be 0.

Frequency Range	Input Frequency	Error Code
10Hz…1kHz	< 9.155 Hz	0
	≥ 1.2 kHz	1
100Hz…10kHz	< 91.55 Hz	0
	≥ 12 kHz	1

This error code can be acquired through the CAN bus when the logical output of the Universal Input is connected to the CAN Output Message function block.

For the Duty Cycle measurements, a special algorithm will identify a loss of the PWM frequency carrier as 0% or 100% valid PWM signal depending on the Digital Input Polarity setpoint and the actual digital state of the input.

3.2.5 Discrete Voltage Level Input

The discrete voltage level input mode is the simplest mode of the Universal Input function block. It is intended to input control signals mainly from switches and buttons.

To activate this mode the user should set the Input Parameter setpoint to the Discrete Voltage Level and define the polarity of the input signal by the Digital Input Polarity setpoint.

The user can also apply a pull-up or pull-down resistor by the Pull-Up/Pull-Down Resistor setpoint.

The debouncing time for the input signal in this mode is fixed and set to 100ms.

3.3 Conversion Function

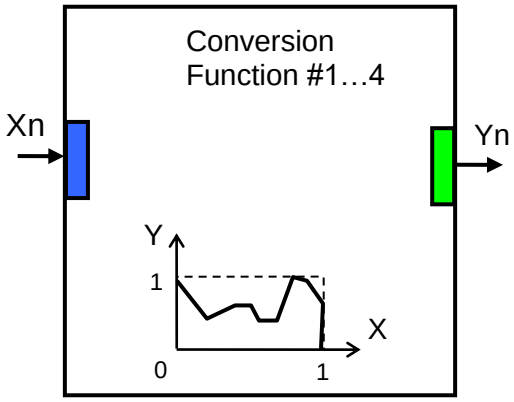
A Conversion Function block allows the user to perform a linearization of an input signal, apply a user-defined control profile, and to do a hotshot control, if necessary. There are four Conversion Function blocks available in the current version of the controller.

Each function block has one logical input, one output and implements a function:

$$Yn = F(Xn),$$

where:

Xn – normalized input signal (can be inverted by the inversion function),
Yn – normalized output signal.



The function Yn=F(Xn) is defined using a piecewise linear approximation in up to 11 points. Each point is presented by three parameters:

$$P_i = (State_i, Xn_i, Yn_i),\ i = 0\ldots 10,$$

where: P_i – i-th point of the function F,
$State_i$ – state of the i-th point. $State_i \in$ {Off, On},
Xn_i – normalized input value at the i-th point.
Yn_i – normalized output value at the i-th point.
If the $State_i$=Off, the point is not active and is not used in the function approximation.

The function values (Xn,Yn) between active points (with $State_i$=On) are defined the following way:

$$Yn = A_j \bullet Xn + B_j,\ j = 0\ldots N,\ N \le 10,$$

$$A_j = (Yn_j - Yn_{(j+1)}) / (Xn_j - Xn_{(j+1)}),$$

$$B_j = (Yn_{(j+1)} \bullet Xn_j - Yn_j \bullet Xn_{(j+1)}) / (Xn_j - Xn_{(j+1)}),$$

$$Xn \in [Xn_j ; Xn_{(j+1)}[,\ State_j=On,\ State_{(j+1)}=On.$$

where: A_j, B_j – linear approximation coefficients between j and (j+1) active points.
N – number of active points.

The Conversion Function block is also capable to implement a hotshot control. For this purpose the user can specify two values for the last, 10-th, function point. The first value is a normalized output value at the 10-th point and the second one is the value that will be assigned to the output if the input remains Xn =1.0 for a hotshot time.

The Conversion Function block has the following set of setpoints:

Name	Default Value	Range	Units	Description
Input Source	Not Connected	Any logical output of any function block or “Not Connected”	–	Defines a source of the input signal Xn
Input Inversion	No	{Yes, No}	–	Specifies, whether the input signal Xn is inverted
Point 0 State	On	–	–	$State_0$. Read only parameter
Point 0 X	0	–	–	Xn_0. Read only parameter

Name	Default Value	Range	Units	Description
Point 0 Y	0	[0;1]	–	Yn_0
Point 1 State	Off	{Off, On}	–	$State_1$
Point 1 X	0.1	[Xn_0; Xn_2]	–	Xn_1
Point 1 Y	0	[0;1]	–	Yn_1
Point 2 State	Off	{Off, On}	–	$State_2$
Point 2 X	0.2	[Xn_1; Xn_3]	–	Xn_2
Point 2 Y	0	[0;1]	–	Yn_2
Point 3 State	Off	{Off, On}	–	$State_3$
Point 3 X	0.3	[Xn_2; Xn_4]	–	Xn_3
Point 3 Y	0	[0;1]	–	Yn_3
Point 4 State	Off	{Off, On}	–	$State_4$
Point 4 X	0.4	[Xn_3; Xn_5]	–	Xn_4
Point 4 Y	0	[0;1]	–	Yn_4
Point 5 State	Off	{Off, On}	–	$State_5$
Point 5 X	0.5	[Xn_4; Xn_6]	–	Xn_5
Point 5 Y	0	[0;1]	–	Yn_5
Point 6 State	Off	{Off, On}	–	$State_6$
Point 6 X	0.6	[Xn_5; Xn_7]	–	Xn_6
Point 6 Y	0	[0;1]	–	Yn_6
Point 7 State	Off	{Off, On}	–	$State_7$
Point 7 X	0.7	[Xn_6; Xn_8]	–	Xn_7
Point 7 Y	0	[0;1]	–	Yn_7
Point 8 State	Off	{Off, On}	–	$State_8$
Point 8 X	0.8	[Xn_7; Xn_9]	–	Xn_8
Point 8 Y	0	[0;1]	–	Yn_8
Point 9 State	Off	{Off, On}	–	$State_9$
Point 9 X	0.9	[Xn_8; Xn_{10}]	–	Xn_9
Point 9 Y	0	[0;1]	–	Yn_9
Point 10 State	On	–	–	$State_{10}$. Read only parameter
Point 10 X	1	–	–	Xn_{10}. Read only parameter
Point 10 Y	0	[0;1]	–	Yn_{10}
Hotshot Delay	0	0…10000	ms	Undefined if 0
Hotshot Y	0	[0;1]	–	Yn_{10}, if Xn=1.0 for Time>Hotshot Delay, and Hotshot Delay $\neq$ 0

3.4 Hydraulic Fan Control

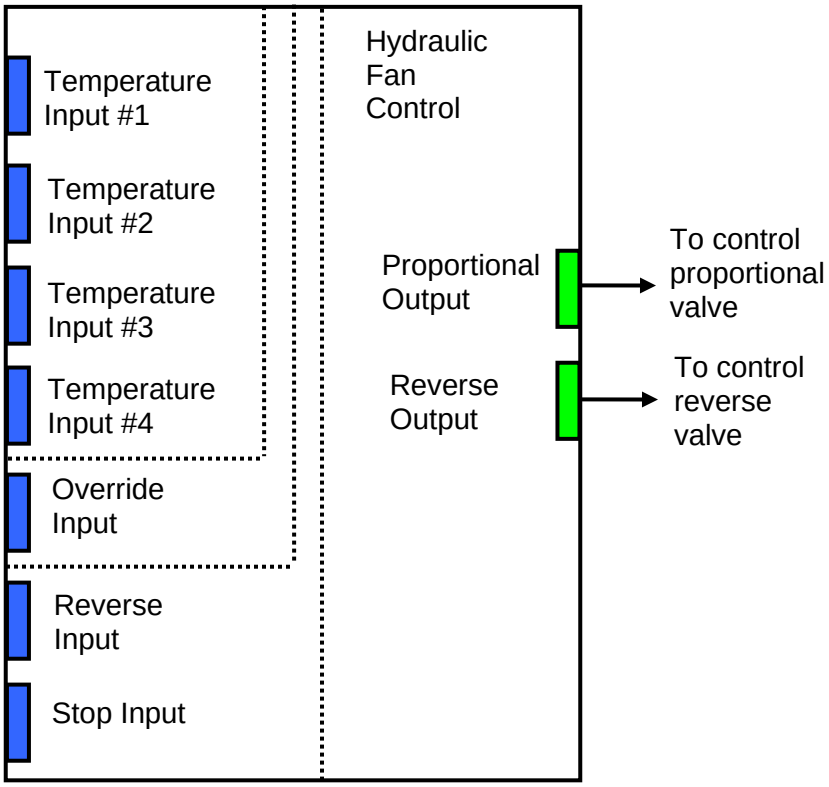
The Hydraulic Fan Control function block represents an application specific function block added to the controller to support hydraulic fan control applications. It is now obsolete and replaced with a new Hydraulic Fan Control 2 function block.

This function block is kept for compatibility with previous versions only and is not recommended for the new design.

The function block provides signals to control proportional and reverse valves of a hydraulic fan. It collects data from up to four temperature inputs and one override input and, based on the temperature data, calculates the proportional valve control signal that defines the fan speed.

A reverse input is used to put the fan into a reverse mode and a stop input returns the fan into its initial state.

All logical inputs can be inverted.



The Hydraulic Fan Control function block has the following set of setpoints:

Name	Default Value	Range	Units	Description
Temperature Input #1 Source	Not Connected	Any logical output of any function block or “Not Connected”	–	Source of the temperature input #1 signal
Temperature Input #1 Inversion	No	{Yes, No}	–	Specifies, whether to invert the temperature input #1 signal

Name	Default Value	Range	Units	Description
Temperature Input #1 T_max	120.0	[-100;500][1], but T_max>T_high	Deg. C	Temperature that corresponds to the maximum signal value (equal to 1.0) on the temperature input #1
Temperature Input #1 T_high	100.0	[-100;500][1], but T_low<T_high<T_max	Deg. C	Temperature on the input #1, at which the fan should run at the maximum speed
Temperature Input #1 T_low	80.0	[-100;500][1], but T_min<T_low<T_high	Deg. C	Temperature on the input #1, at which the fan should run at the minimum speed
Temperature Input #1 T_min	20.0	[-100;500][1], but T_min<T_low	Deg. C	Temperature that corresponds to the minimum signal value (equal to 0.0) on the temperature input #1
Temperature Input #2 Source	Not Connected	Any logical output of any function block or “Not Connected”	–	Source of the temperature input #2 signal
Temperature Input #2 Inversion	No	{Yes, No}	–	Specifies, whether to invert the temperature input #2 signal
Temperature Input #2 T_max	120.0	[-100;500][1], but T_max>T_high	Deg. C	Temperature that corresponds to the maximum signal value (equal to 1.0) on the temperature input #2
Temperature Input #2 T_high	100.0	[-100;500][1], but T_low<T_high<T_max	Deg. C	Temperature on the input #2, at which the fan should run at the maximum speed
Temperature Input #2 T_low	80.0	[-100;500][1], but T_min<T_low<T_high	Deg. C	Temperature on the input #2, at which the fan should run at the minimum speed
Temperature Input #2 T_min	20.0	[-100;500][1], but T_min<T_low	Deg. C	Temperature that corresponds to the minimum signal value (equal to 0.0) on the temperature input #2
Temperature Input #3 Source	Not Connected	Any logical output of any function block or “Not Connected”	–	Source of the temperature input #3 signal
Temperature Input #3 Inversion	No	{Yes, No}	–	Specifies, whether to invert the temperature input #3 signal
Temperature Input #3 T_max	120.0	[-100;500][1], but T_max>T_high	Deg. C	Temperature that corresponds to the maximum signal value (equal to 1.0) on the temperature input #3
Temperature Input #3 T_high	100.0	[-100;500][1], but T_low<T_high<T_max	Deg. C	Temperature on the input #3, at which the fan should run at the maximum speed
Temperature Input #3 T_low	80.0	[-100;500][1], but T_min<T_low<T_high	Deg. C	Temperature on the input #3, at which the fan should run at the minimum speed

Name	Default Value	Range	Units	Description
Temperature Input #3 T_min	20.0	[-100;500][1], but T_min<T_low	Deg. C	Temperature that corresponds to the minimum signal value (equal to 0.0) on the temperature input #3
Temperature Input #4 Source	Not Connected	Any logical output of any function block or “Not Connected”	–	Source of the temperature input #4 signal
Temperature Input #4 Inversion	No	{Yes, No}	–	Specifies, whether to invert the temperature input #4 signal
Temperature Input #4 T_max	120.0	[-100;500][1], but T_max>T_high	Deg. C	Temperature that corresponds to the maximum signal value (equal to 1.0) on the temperature input #4
Temperature Input #4 T_high	100.0	[-100;500][1], but T_low<T_high<T_max	Deg. C	Temperature on the input #4, at which the fan should run at the maximum speed
Temperature Input #4 T_low	80.0	[-100;500][1], but T_min<T_low<T_high	Deg. C	Temperature on the input #4, at which the fan should run at the minimum speed
Temperature Input #4 T_min	20.0	[-100;500][1], but T_min<T_low	Deg. C	Temperature that corresponds to the minimum signal value (equal to 0.0) on the temperature input #4
Temperature Input Priority	No Priority	{No Priority, Temperature Input #1, Temperature Input #2, Temperature Input #3, Temperature Input #4}	–	Specifies the priority temperature input that will be used to control the fan speed, if other temperatures are within the range (T<T_high)
Override Input Source	Not Connected	Any logical output of any function block or “Not Connected”	–	Source of an override input signal used to control the fan speed
Override Input Inversion	No	{Yes, No}	–	Specifies, whether to invert the override input signal
Reverse Input Source	Not Connected	Any logical output of any function block or “Not Connected”	–	Source of a reverse input signal, which activates the fan reverse control sequence
Reverse Input Inversion	No	{Yes, No}	–	Specifies, whether to invert the reverse input signal
Stop Input Source	Not Connected	Any logical output of any function block or “Not Connected”	–	Source of a stop input signal. The signal safely brings the fan into its initial idle state switching off both: proportional and reverse valves.
Stop Input Inversion	No	{Yes, No}	–	Specifies, whether to invert the stop input signal
Max Fan Stop Time[2]	5.0	[0;1000]	s	Time between closing the proportional valve and switching the reverse valve at

Name	Default Value	Range	Units	Description
				the maximum fan speed
Fan Switch Time	0.5	[0;20]	s	Time between switching the reverse valve and opening the proportional valve
Fan Reverse Hold Time	10	[0;100]	s	Time the fan will be reversing after the user removes the reverse input signal

[1] The temperature ranges are [0;200] for EA prior to Version 3.0.32.0.
[2] Time between closing the proportional valve and switching the reverse valve is proportional to the proportional valve signal defining the fan speed at the time immediately preceding the switching operation.

3.5 Hydraulic Fan Control 2

The Hydraulic Fan Control 2 function block represents a new hydraulic fan control application specific function block that supersedes an old Hydraulic Fan Control function block.

This function block provides signals to control proportional and reverse switch valves of a hydraulic fan. It collects data from up to four temperature inputs and, based on the temperature data, calculates the proportional valve control signal that defines the fan speed.

A reverse input is used to but the fan into a reverse mode to blow-off the dust from the engine, and a stop input brings the fan to the full stop.

All logical inputs can be inverted.

The proportional output, PropOut, controls the proportional valve the way that the minimum (zero) signal output corresponds to the zero-fan speed and the maximum (one) signal output – to the maximum fan speed.

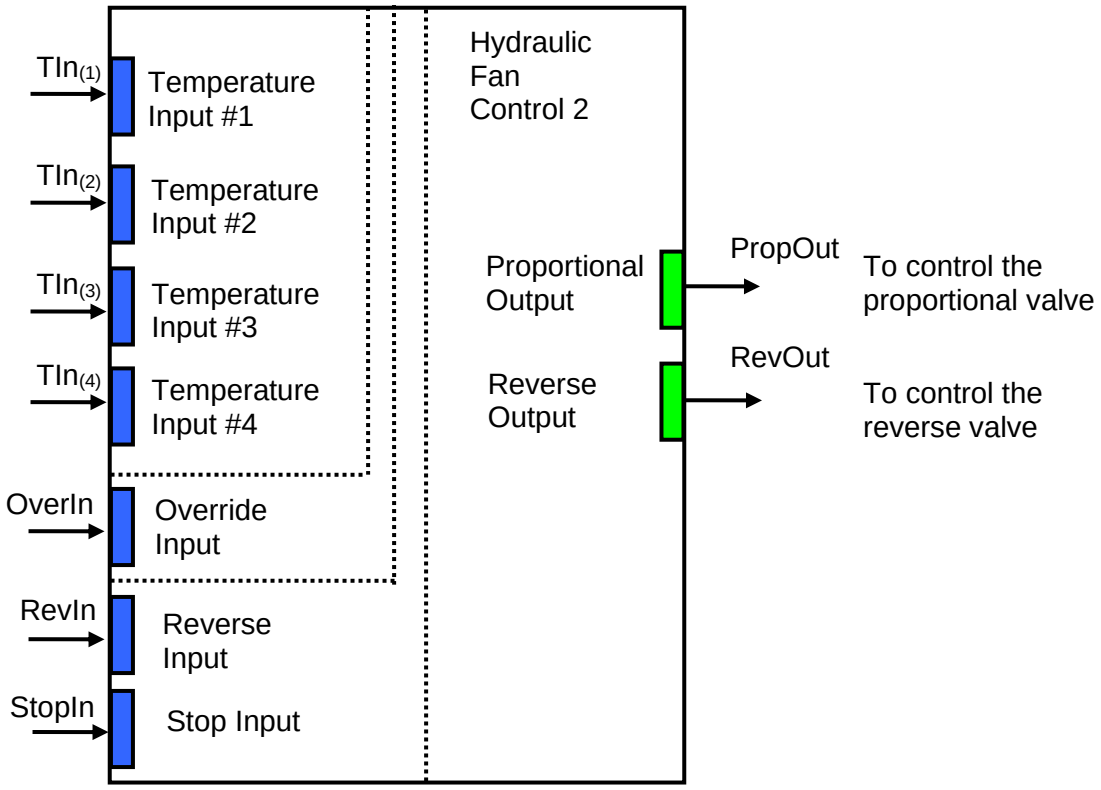
The on/off output, RevOut, controls the reverse switch valve. Zero on its output corresponds to the normal (direct) direction and one – to the reverse direction.

The temperature control is performed in the normal (direct) mode, when RevOut=0. The temperatures from the inputs are acquired and compared to determine which temperature signal should be used to control the proportional valve. Then the selected temperature signal is normalized to be within the boundaries of the speed control signal and is sent to the proportional output.

When the reverse mode is activated, the reverse switch output and the proportional output are set to one (RevOut=1, PropOut=1) bringing the fan to full speed in the reverse direction. There is no temperature control in this mode, since it is activated only for a small period of time to remove the dust from the system.

When the fan is switched between the direct and reverse modes, the fan is first brought to the full stop (PropOut=0) before changing the rotation direction (changing RevOut from 0 to 1 or from 1 to 0) to avoid overstress of the hydraulic and mechanical parts of the fan.

The Hydraulic Fan Control 2 function block is presented below:



The Hydraulic Fan Control 2 function block has the following set of setpoints:

Name	Default Value	Range	Units	Description
Temperature Input #1 Source	Not Connected	Any logical output of any function block or “Not Connected”	–	Source of the temperature input #1 signal
Temperature Input #1 Inversion	No	{Yes, No}	–	Specifies, whether to invert the temperature input #1 signal
Temperature Input #1 T_max	120.0	[-100; 500], but T_max>T_high	Deg. C	Temperature that corresponds to the maximum signal value (equal to 1.0) on the temperature input #1
Temperature Input #1 T_high	100.0	[-100; 500], but T_low<T_high<T_max	Deg. C	Temperature on the input #1, at which the fan should run at the maximum speed
Temperature Input #1 T_low	80.0	[-100; 500], but T_min<T_low<T_high	Deg. C	Temperature on the input #1, at which the fan should run at the minimum speed
Temperature Input #1 T_min	20.0	[-100; 500], but T_min<T_low	Deg. C	Temperature that corresponds to the minimum signal value (equal to 0.0) on the temperature input #1
Temperature Input #2 Source	Not Connected	Any logical output of any function block or “Not Connected”	–	Source of the temperature input #2 signal

Name	Default Value	Range	Units	Description
Temperature Input #2 Inversion	No	{Yes, No}	–	Specifies, whether to invert the temperature input #2 signal
Temperature Input #2 T_max	120.0	[-100; 500], but T_max>T_high	Deg. C	Temperature that corresponds to the maximum signal value (equal to 1.0) on the temperature input #2
Temperature Input #2 T_high	100.0	[-100; 500], but T_low<T_high<T_max	Deg. C	Temperature on the input #2, at which the fan should run at the maximum speed
Temperature Input #2 T_low	80.0	[-100; 500], but T_min<T_low<T_high	Deg. C	Temperature on the input #2, at which the fan should run at the minimum speed
Temperature Input #2 T_min	20.0	[-100; 500], but T_min<T_low	Deg. C	Temperature that corresponds to the minimum signal value (equal to 0.0) on the temperature input #2
Temperature Input #3 Source	Not Connected	Any logical output of any function block or “Not Connected”	–	Source of the temperature input #3 signal
Temperature Input #3 Inversion	No	{Yes, No}	–	Specifies, whether to invert the temperature input #3 signal
Temperature Input #3 T_max	120.0	[-100; 500], but T_max>T_high	Deg. C	Temperature that corresponds to the maximum signal value (equal to 1.0) on the temperature input #3
Temperature Input #3 T_high	100.0	[-100; 500], but T_low<T_high<T_max	Deg. C	Temperature on the input #3, at which the fan should run at the maximum speed
Temperature Input #3 T_low	80.0	[-100; 500], but T_min<T_low<T_high	Deg. C	Temperature on the input #3, at which the fan should run at the minimum speed
Temperature Input #3 T_min	20.0	[-100; 500], but T_min<T_low	Deg. C	Temperature that corresponds to the minimum signal value (equal to 0.0) on the temperature input #3
Temperature Input #4 Source	Not Connected	Any logical output of any function block or “Not Connected”	–	Source of the temperature input #4 signal
Temperature Input #4 Inversion	No	{Yes, No}	–	Specifies, whether to invert the temperature input #4 signal
Temperature Input #4 T_max	120.0	[-100; 500], but T_max>T_high	Deg. C	Temperature that corresponds to the maximum signal value (equal to 1.0) on the temperature input #4
Temperature Input #4 T_high	100.0	[-100; 500], but T_low<T_high<T_max	Deg. C	Temperature on the input #4, at which the fan should run at the maximum speed

Name	Default Value	Range	Units	Description
Temperature Input #4 T_low	80.0	[-100; 500], but T_min<T_low<T_high	Deg. C	Temperature on the input #4, at which the fan should run at the minimum speed
Temperature Input #4 T_min	20.0	[-100; 500], but T_min<T_low	Deg. C	Temperature that corresponds to the minimum signal value (equal to 0.0) on the temperature input #4
Temperature Input Priority	No Priority	{No Priority, Temperature Input #1, Temperature Input #2, Temperature Input #3, Temperature Input #4}	–	Specifies the priority temperature input that will be used to control the fan speed, if other temperatures are within the range (T<T_high)
Override Input Source	Not Connected	Any logical output of any function block or “Not Connected”	–	Source of an override input signal used to control the fan speed
Override Input Inversion	No	{Yes, No}	–	Specifies, whether to invert the override input signal
Reverse Input Source	Not Connected	Any logical output of any function block or “Not Connected”	–	Source of a reverse input signal, which activates the fan reverse control sequence
Reverse Input Inversion	No	{Yes, No}	–	Specifies, whether to invert the reverse input signal
Stop Input Source	Not Connected	Any logical output of any function block or “Not Connected”	–	Source of a stop input signal. The signal brings the fan to the full stop ramping the proportional output down to zero (PropOut=0). The reverse valve output is not affected by this input.
Stop Input Inversion	No	{Yes, No}	–	Specifies, whether to invert the stop input signal
FanSpeedMax – Max Fan Speed Control Signal	1.0	[0;1], but FanSpeedMax> FanSpeedMin	–	Signal value on the proportional output corresponding to the maximum fan speed in the linear valve control region.
FanSpeedMin – Min Fan Speed Control Signal	0.1	[0;1], but FanSpeedMin< FanSpeedMax	–	Signal value on the proportional output corresponding to the minimum fan speed in the linear valve control region.
RampUp – Fan Speed Ramp Up Time	10.0	[0;10000]	s	Time, during which the proportional output ramps from FanSpeedMin to FanSpeedMax value.
RampDown – Fan Speed Ramp Down Time	10.0	[0;10000]	s	Time, during which the proportional output ramps from FanSpeedMax to FanSpeedMin value.
Tsw – Switch Delay Time	1.0	[0;10000]	s	Time between the complete closing of the proportional

Name	Default Value	Range	Units	Description
				valve (PropOut = 0) and switching the direction of the reverse valve.
Trev_hold – Reverse Hold Time	0.0	[0;10000]	s	Time the fan will be reversing after the user removes the reverse input signal. Useful when the reverse input signal comes from a pushbutton.

The temperature control part of the fan control (proportional part) is presented in the following block diagram:

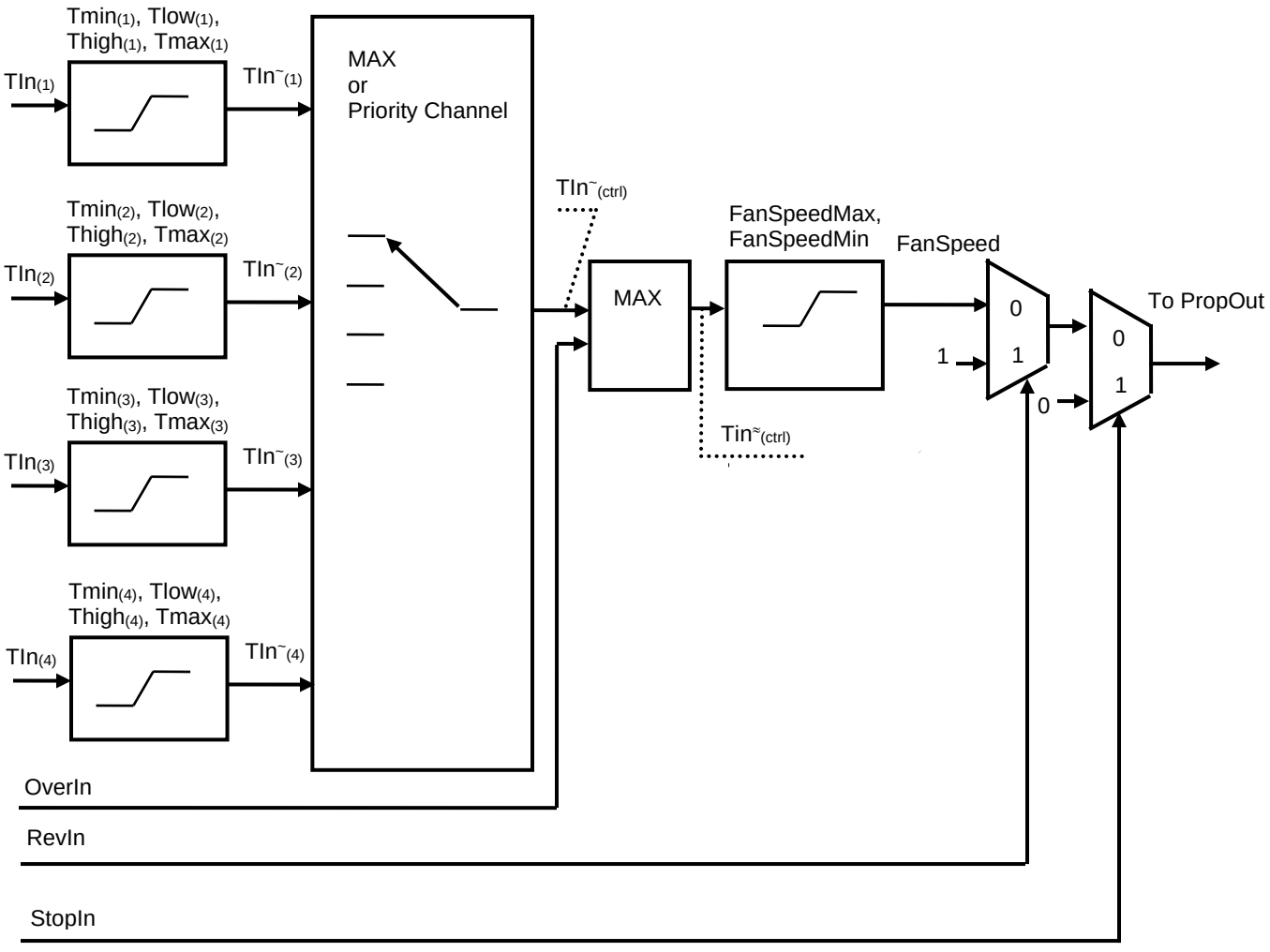
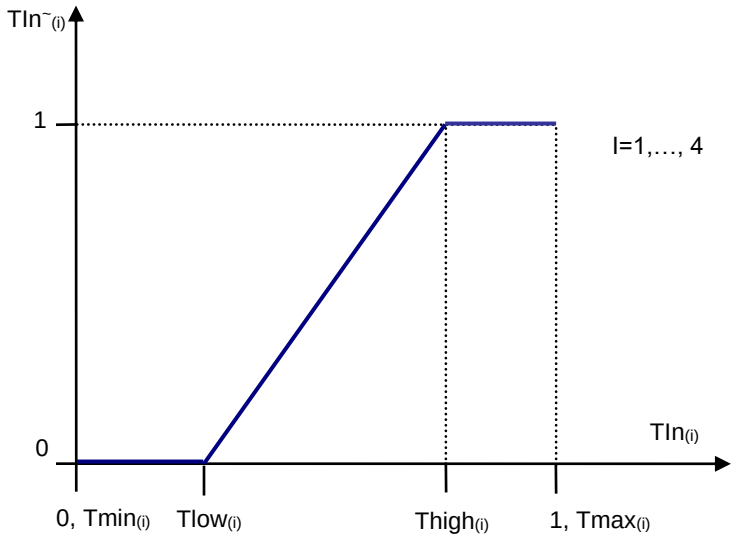


Figure 3. Temperature Control Part of the Hydraulic Fan Control 2

The PropOut follows FanSpeed, when RevIn=0, and is equal to 1, when RevIn=1. It is equal to 0, when StopIn=1.



All unused channels have $TIn_{(i)}=0$.

If no channel has a priority:

$TIn^{\sim}_{(ctrl)}=max[TIn^{\sim}_{(i)}]$, i=1,…, 4

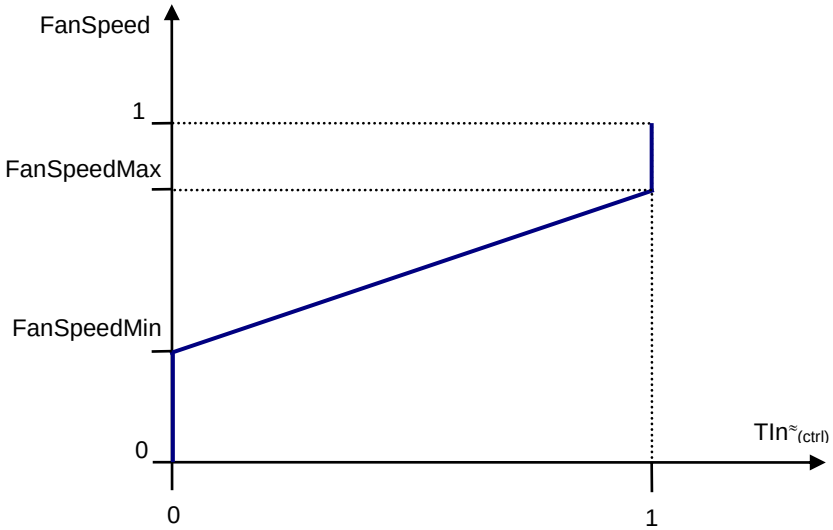
If channel # i has a priority:

$TIn^{\sim}_{(ctrl)}= TIn^{\sim}_{(i)}$, if $max[TIn^{\sim}_{(j)}]<1$, j=1,…, 4
$TIn^{\sim}_{(ctrl)}=1$, if $max[TIn^{\sim}_{(j)}]=1$

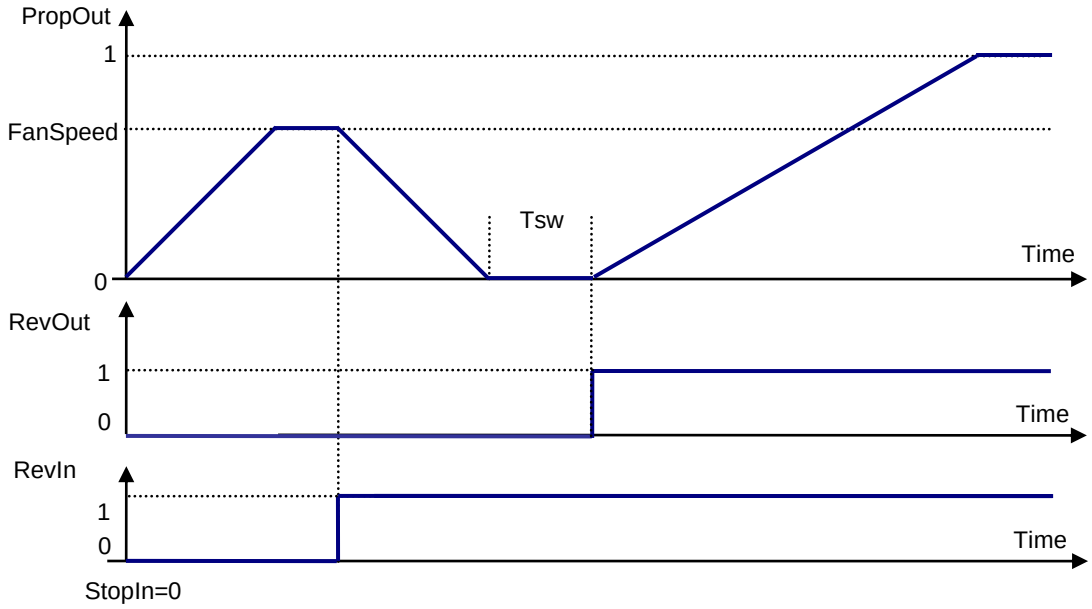
After applying the OverIn signal, the result will be:

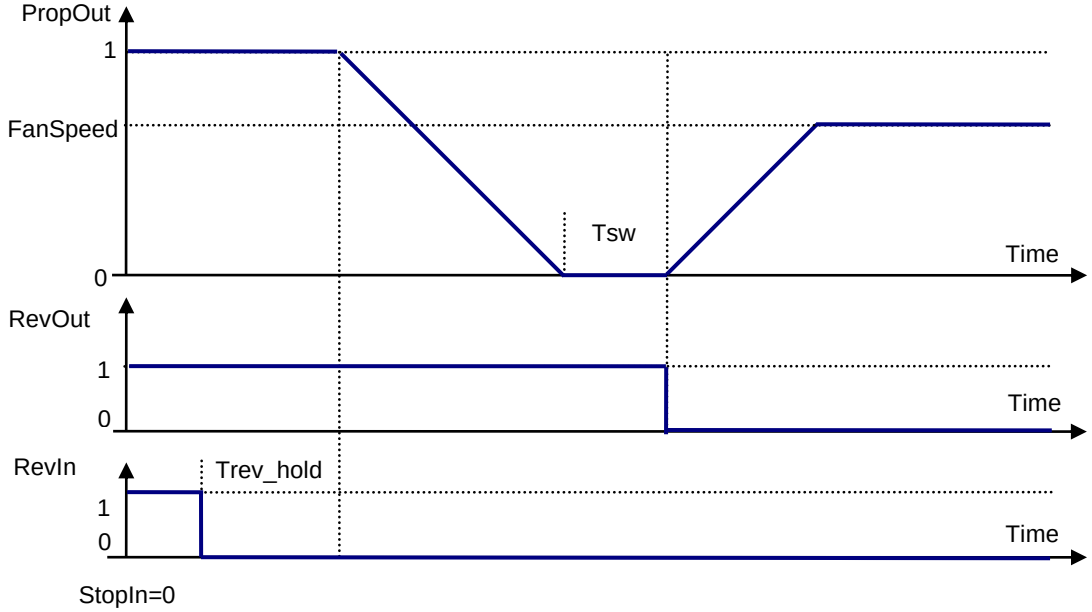
$TIn^{\approx}_{(ctrl)}=max[TIn^{\sim}_{(ctrl)}, OverIn]$

This signal will be used for defining the FanSpeed signal:

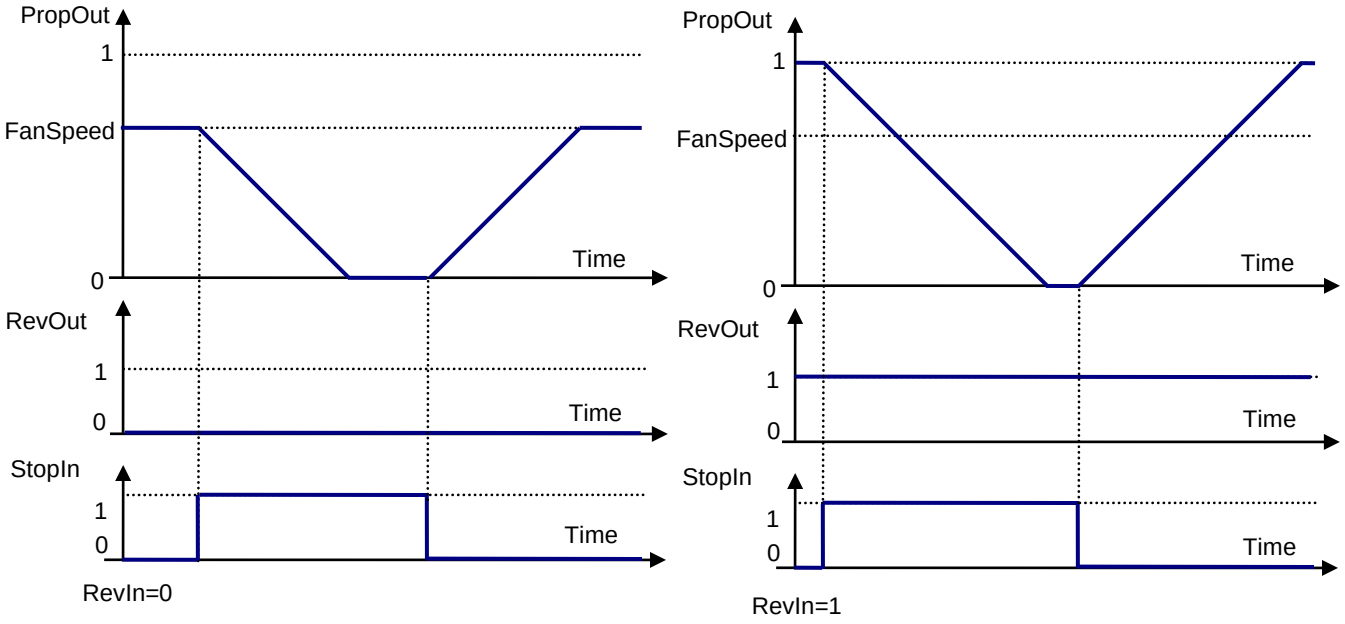


In dynamics, the PropOut ramps up and down depending on the RevIn signal as shown bellow:





When the StoIn signal is applied, the PropOut ramps down to PropOut=0 bringing the fan to the full stop independently on the RevIn signal:



An application example of the internal controller configuration with the Hydraulic Fan Control 2 function block is presented on Figure 4.

In this example, the first three universal inputs are connected to three different temperature sensors with a positive temperature coefficient (PTC) and a close to linear temperature-resistance characteristics (platinum RTDs, for example). The fourth universal input is connected to a switch and is used to activate the reverse mode.

The outputs are connected the following way: the first PWM output is used to control the proportional valve defining the fan speed, and the second one controls the switch valve changing the fan direction.

The Universal Input normalization setpoints and the Hydraulic Fan Control 2 Temperature input setpoints are linked together the following way:

$Rmin_i=R(T\_min_i)$, $Rmax_i=R(T\_max_i)$,

where i – number of the temperature input (i=1,…,3 – for this example).

If, for example, the R=R(T) characteristics is the following:

T (Deg. C)	0	20	40	60	80	100	150
R (Ohm)	100.00	107.79	115.54	123.24	130.89	138.50	157.31

then: Rmin=0.1 kOhm, T_min=0 Deg.C and Rmax=0.15731 kOhm, T_max=150 Deg.C

The user then can chose T_high and T_low depending on what temperature range the fan will be running at.

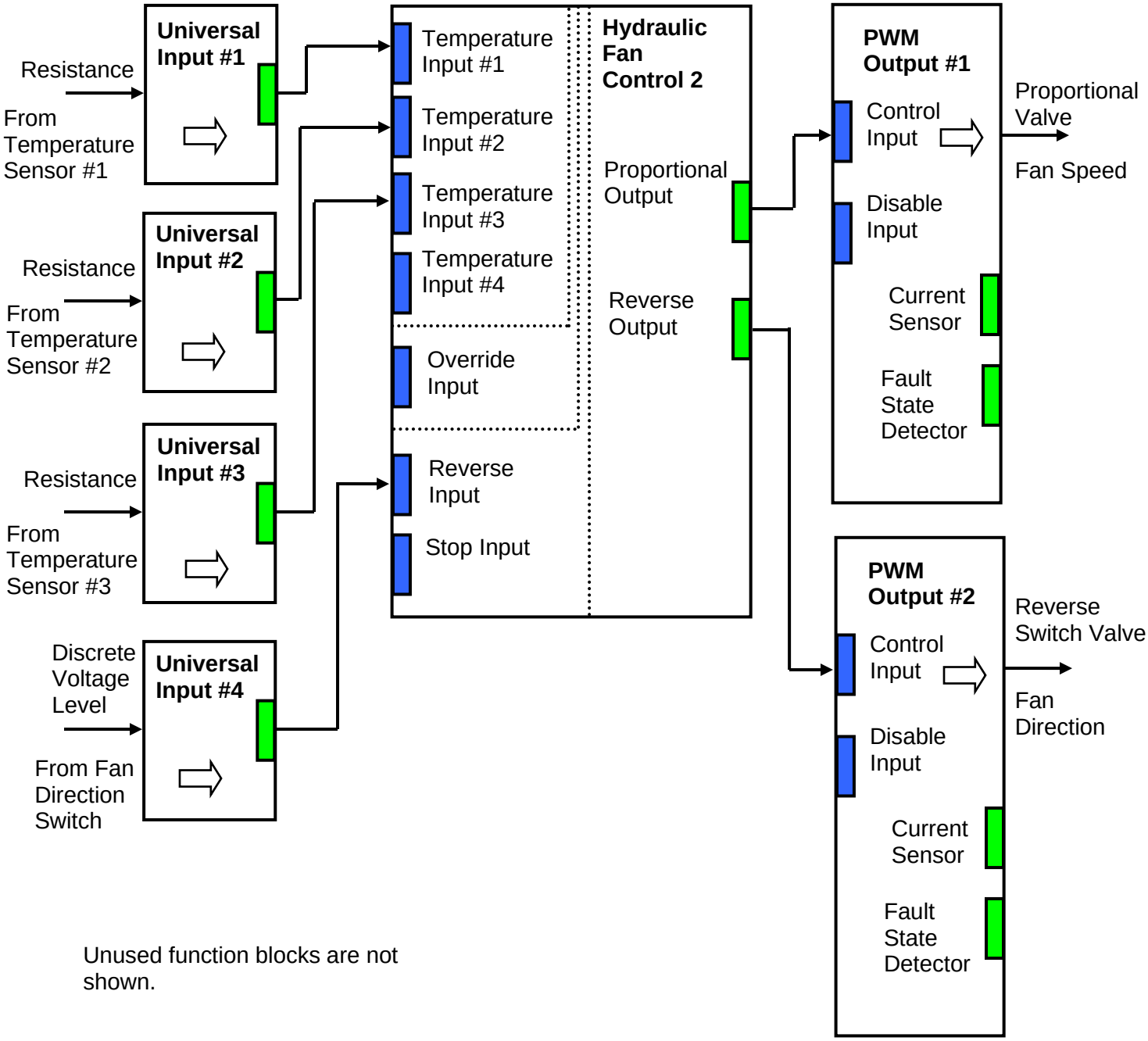
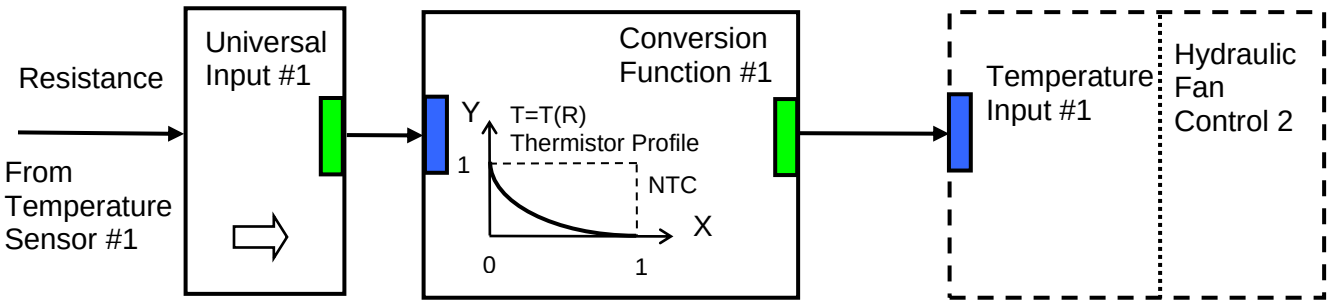


Figure 4. Example of a Hydraulic Fan Control Configuration

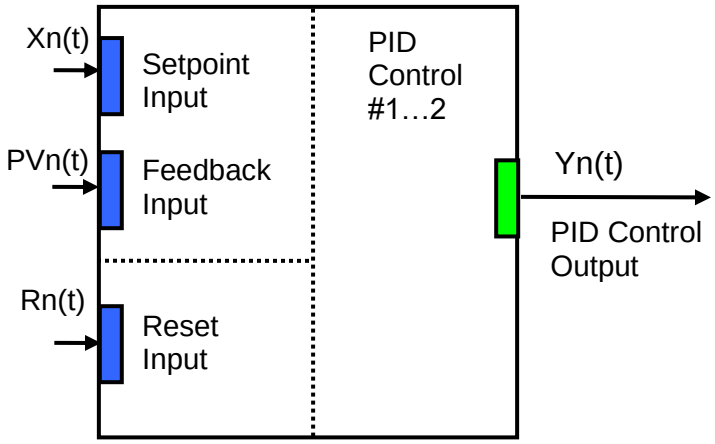
In case the temperature sensors have negative temperature-resistance characteristics (NTC) and (or) the sensors have a high nonlinearity in the working range, the Conversion Functions should be put between the Universal Input and the Hydraulic Fan Control 2 function blocks:



3.6 PID Control

To provide the user with means to build generic closed loop PID regulators, two PID Control function blocks were added to the controller.

A PID Control function block has: setpoint and feedback inputs, manual control mode and a reset input to bring the regulator into its initial state. The user can also adjust the time resolution for fast or slow responding closed loop systems.



The normalized output of the PID Control function block Yn(t), as a function of time, can be described by the following formula:

$$Yn(t)=Clip(Y(t)),$$

$$Y(t) = P\bullet[\, e(t) + 1/T_I\bullet\int e(t)dt - T_D\bullet dPVn(t)/dt \,],$$

where:

$Clip(Y(t))=\{Y(t), \text{ if } 0\le Y(t)\le 1;\ 0, \text{ if } Y(t)<0;\ 1, \text{ if } Y(t)>1\}$ – clipping function;
$e(t) = Xn(t) – PVn(t)$ – error function, where
$Xn(t)$ – normalized setpoint variable, set by the Setpoint Input,
$PVn(t)$ – normalized process variable, set by the Feedback Input,
P – proportional gain,
T_I – integral time,
T_D – derivative time.

All PID Control logical inputs can be inverted.

To avoid saturation of the output due to the integral term of the PID regulator, an anti-windup algorithm is implemented. The integrator is stopped when the output saturates, and the error function moves the output to further saturation:

- Y(t)>1 and e(t)>0 or
- Y(t)<0 and e(t)<0.

When the Reset Input is activated, an integral part of the PID regulator is reset to zero and the output of the PID Control function block is brought to zero, too:

$\int e(t)dt=0$, $Y(t) = 0$, when $Rn(t) \geq 0.5$,

where:

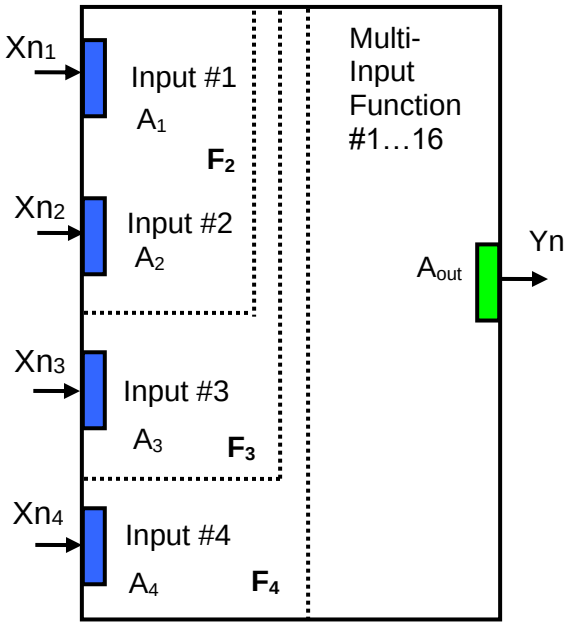
Rn(t) – normalized reset variable, set by the Reset Input.

The PID Control function block setpoints are described in the following table:

Name	Default Value	Range	Units	Description
Setpoint Input Source	Not Connected	Any logical output of any function block or “Not Connected”	–	Source of a setpoint input signal
Setpoint Input Inversion	No	{Yes, No}	–	Specifies, whether to invert the setpoint signal
Feedback Input Source	Not Connected	Any logical output of any function block or “Not Connected”	–	Source of a feedback input signal
Feedback Input Inversion	No	{Yes, No}	–	Specifies, whether to invert the feedback signal
Reset Input Source	Not Connected	Any logical output of any function block or “Not Connected”	–	Source of a reset input signal. The signal brings the regulator into its initial state.
Reset Input Inversion	No	{Yes, No}	–	Specifies, whether to invert the reset signal
Manual Control	No	{Yes, No}	–	Put the PID control in a manual control mode. In this mode the PID regulator is off and the PID output is equal to the value of the “Manual Control Output” setpoint
Manual Control Output	0.5	[0;1]	–	Output of the PID control in the manual control mode
Proportional Gain	1.0	[0;100000]	–	Proportional PID parameter
Integral Time Constant	0.1	[0;100000]	s	Integral PID parameter
Derivative Time Constant	0.01	[0;100000]	s	Derivative PID parameter. Derivation from the process variable is used.
Time Resolution	0.001	[0.001;10]	s	Time interval between PID control cycles

3.7 Multi-Input Function

There are sixteen Multi-Input Function blocks added to the controller to increase its flexibility to support different user-defined control algorithms. A Multi-Input Function block takes up to four input signals, scales them, and performs consequent arithmetic or logical operations. Then it outputs the result, which can be scaled as well.



The normalized output signal Yn of the Multi-Input Function block can be presented by the following formula:

$$Yn=Clip(Y),$$

$$Y = A_{out} \bullet F_4[F_3[F_2[A_1 \bullet Xn_1, A_2 \bullet Xn_2], A_3 \bullet Xn_3], A_4 \bullet Xn_4],$$

where:

$Clip(Y)=\{Y, \text{ if } 0 \le Y \le 1; 0, \text{ if } Y<0; 1, \text{ if } Y>1\}$ – clipping function;
Xn_i, i=1,…,4 – normalized signal value of the i-th input source (can be inverted);
A_i, i=1,…,4 – scale coefficient of the i-th input signal;
A_{out} – scale coefficient of the output signal.
$F_i[x, y]$, i=2,…,4 – binary function of the i-th input signal. The function takes two arguments: x and y, where x is a result of the previous F_{i-1} binary function or a scaled 1-st input signal value and y is a scaled i-th input signal value. The function does not exist for the 1-st input signal.

If any of the input sources are not connected, the formula is truncated to perform operations with only connected input sources. For example:

$Y= A_{out} \bullet F_2[A_1 \bullet Xn_1, A_2 \bullet Xn_2]$, if the 3-rd input source is ”Not Connected”

$Y=A_{out} \bullet F_3[F_2[A_1 \bullet Xn_1, A_2 \bullet Xn_2], A_3 \bullet Xn_3]$, if the 4-th input source is” Not Connected”

In case the 1-st or the 2-nd input source is not connected, the output signal of the function block is not available and its signal value is set to Y=0.

The Multi-Input Function block has the following set of setpoints:

Name	Default Value	Range	Units	Description
Input #1 Source	Not Connected	Any logical output of any function block or “Not Connected”	–	Source of the input #1 signal
Input #1 Inversion	No	{Yes, No}	–	Specifies, whether to invert the input #1 signal
Input #1 Scale	1.0	[-1…1] or Any value*	–	Input #1 signal scale coefficient
Input #2 Source	Not Connected	Any logical output of any function block or “Not Connected”	–	Source of the input #2 signal
Input #2 Inversion	No	{Yes, No}	–	Specifies, whether to invert the input #2 signal
Input #2 Scale	1.0	[-1…1] or Any value*	–	Input #2 signal scale coefficient
Input #2 Function	+	{+, ∗,÷, Max, Min, OR, AND, XOR, <, ≤, =, >, ≥}	–	Binary function of the input #1 scaled signal and the input #2 scaled signal
Input #3 Source	Not Connected	Any logical output of any function block or “Not Connected”	–	Source of the input #3 signal
Input #3 Inversion	No	{Yes, No}	–	Specifies, whether to invert the input #3 signal
Input #3 Scale	1.0	[-1…1] or Any value*	–	Input #3 signal scale coefficient
Input #3 Function	+	{+, ∗,÷, Max, Min, OR, AND, XOR, <, ≤, =, >, ≥}	–	Binary function of the Input #2 function result and the input #3 scaled signal
Input #4 Source	Not Connected	Any logical output of any function block or “Not Connected”	–	Source of the input #4 signal
Input #4 Inversion	No	{Yes, No}	–	Specifies, whether to invert the input #4 signal
Input #4 Scale	1.0	[-1…1] or Any value*	–	Input #4 signal scale coefficient
Input #4 Function	+	{+, ∗,÷, Max, Min, OR, AND, XOR, <, ≤, =, >, ≥}	–	Binary function of the Input #3 function result and the input #4 scaled signal
Output Scale	1.0	[-1…1] or Any value*	–	Output signal scale coefficient

*Any scale value can be programmed using EA version 3.0.29.0 or later.

The binary functions $F_i[x,y]$ have the following implementation specifics.

In the division function, to avoid ambiguity in dividing by 0, the dividend and the divisor are not allowed to be less than δ:

$$F_i^{(\div)}[x,y] = \max(x,\delta)/\max(y,\delta), \qquad i=2,\ldots,4$$

where: δ =1.0E-6 is a specially introduced computational constant.

For logical functions {OR, AND, XOR} values $X_i \geq 0.5$ (i=1,…,4) are treated as 1 (true) and $X_i < 0.5$ – as 0 (false).

To minimize influence of computational errors during normalization, comparison functions {≤, =, ≥ } are defined the following way:

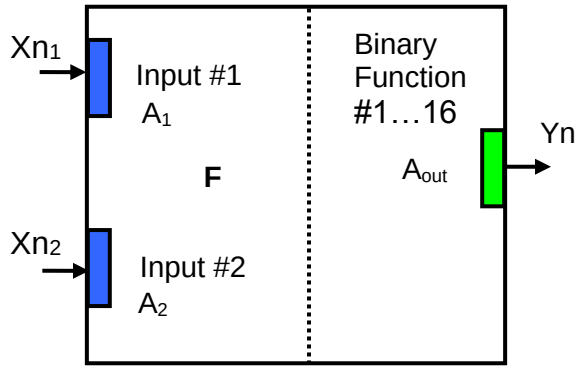
$F_i^{(\leq)}[x,y] = \{1, \text{if } x \leq y+\delta;\ 0, \text{if } x > y+\delta\}$,

$F_i^{(=)}[x,y] = \{1, \text{if } |x-y| \leq \delta;\ 0, \text{if } |x-y| > \delta\}$,

$F_i^{(\geq)}[x,y] = \{1, \text{if } x \geq y-\delta;\ 0, \text{if } x < y-\delta\}$.

3.8 Binary Function

A simplified version of the Multi-Input Function is a Binary Function. There are sixteen Binary Function blocks added to the controller. Each Binary Function block takes two logical input signals, scales them, and performs arithmetic or logical operations similar to the Multi-Input Function block. Then it outputs the result, which can be scaled as well.



The normalized output signal Yn of the Binary Function block can be presented by the following formula:

$Yn = Clip(Y)$,

$Y = A_{out} \bullet F[A_1 \bullet Xn_1, A_2 \bullet Xn_2]$

where:

Clip(Y)={Y, if 0≤Y≤1; 0, if Y<0; 1, if Y>1} – clipping function;
Xn_1, Xn_2 – normalized signal values of the input sources (can be inverted);
A_1, A_2 – input scale coefficients.
A_{out} – output scale coefficient.
F[x, y] – binary function of the scaled input signals: $x=A_1 \bullet Xn_1$, $y=A_2 \bullet Xn_2$.

In case one of the input sources is not connected, the output signal of the function block is not available and its signal value is equal to Yn=0. The Binary Function block has the following set of setpoints:

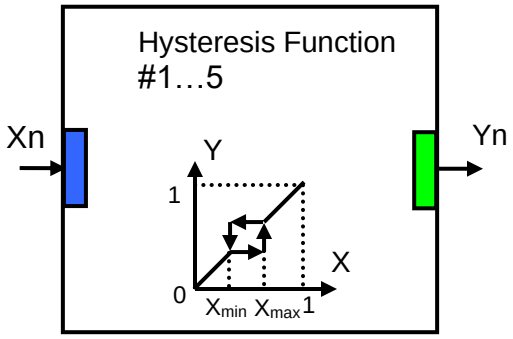
Name	Default Value	Range	Units	Description
Input #1 Source	Not Connected	Any logical output of any function block or “Not	–	Source of the input #1 signal

Name	Default Value	Range	Units	Description
		Connected”		
Input #1 Inversion	No	{Yes, No}	–	Specifies, whether to invert the input #1 signal
Input #1 Scale	1.0	Any value	–	Input #1 signal scale coefficient
Input #2 Source	Not Connected	Any logical output of any function block or “Not Connected”	–	Source of the input #2 signal
Input #2 Inversion	No	{Yes, No}	–	Specifies, whether to invert the input #2 signal
Input #2 Scale	1.0	Any value	–	Input #2 signal scale coefficient
Function	+	{+, ∗,÷, Max, Min, OR, AND, XOR, <, ≤, =, >, ≥}	–	Binary function of the input #1 scaled signal and the input #2 scaled signal
Output Scale	1.0	Any value	–	Output signal scale coefficient

For details of the F[x,y] implementation, see the Multi-Input Function block.

3.9 Hysteresis Function

To provide hysteresis functions, five Hysteresis Function blocks were added to the controller.



Each block has one logical input, one output and performs the following function:

Yn=Xn, Direction=FromMinToMax, if Xn≤Xmin,
Yn=Xmin, if Xmin<Xn≤Xmax and Direction=FromMinToMax
Yn=Xmax, if Xmin<Xn≤Xmax and Direction=FromMaxToMin
Yn=Xn, Direction=FromMaxToMin, if Xn>Xmax,

where: Xn – normalized input signal (can be inverted),
Yn – normalized output signal,
Xmin, Xmax – hysteresis function setpoints,
Direction – internal variable storing the direction of the input signal change.

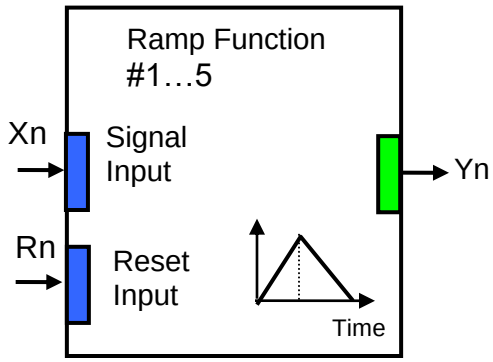
The setpoints of the Hysteresis Function block is presented below:

Name	Default Value	Range	Units	Description
Input Source	Not Connected	Any logical output of any function block or “Not Connected”	–	Source of the input signal
Input Inversion	No	{Yes, No}	–	Specifies, whether to invert the input signal
Xmin – Hysteresis Function Begin Point	0.0	[0;1], but Xmin≤Xmax	–	Hysteresis Function Begin Point
Xmax – Hysteresis Function End Point	0.0	[0;1], but Xmax≥Xmin	–	Hysteresis Function End Point

3.10 Ramp Function

There are five Ramp Function blocks in the controller. These function blocks provide a user-defined ramping for input signals. While simple ramping of the PWM outputs can be achieved by the internal settings of the PWM Output function block, the Ramp Function block gives the user an additional flexibility to apply individual up and down ramps to internal logical signals. This function block can also be used as a timer when ramping a constant logical signal.

Each Ramp Function block has two logical inputs and one logical output:



In normal operation, when reset signal is not applied (Rn=0), the output of the Ramp Function Yn follows its input Xn, ramping up and down with predefined rates $Ramp_{up}$ and $Ramp_{down}$:

$$\begin{aligned} &Yn(t+dt)=Yn(t), && \text{if } Xn(t)=Yn(t),\\ &Yn(t+dt)=Yn(t) + Ramp_{up} \cdot dt, && \text{if } Xn(t)>Yn(t),\\ &Yn(t+dt)=Yn(t) - Ramp_{down} \cdot dt, && \text{if } Xn(t)<Yn(t). \end{aligned}$$

When the Ramp Function is reset (Rn=1) the output is immediately forced to zero: Yn(t)=0.

For convenience, both input signals: Xn and Rn can be inverted.

By default, the output of the Ramp Function is continuous. It can be made discrete, On/Off, using the output signal type setpoint. In this case, the output of the Ramp Function will be:

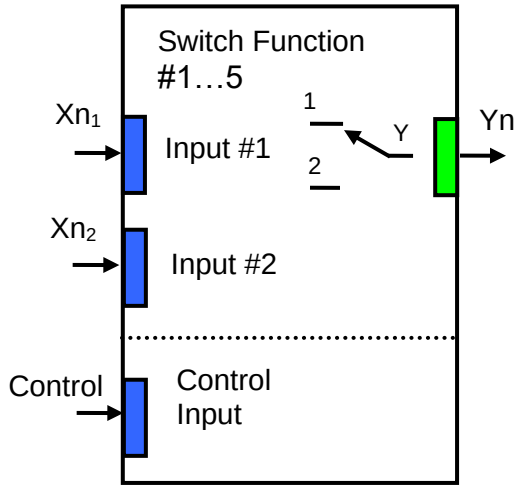
$$Yn^{(On/Off)} = \{0, \text{ if } Yn<0.5;\ 1, \text{ if } Yn\geq 0.5\}$$

The setpoints of the Ramp Function are presented in the following table:

Name	Default Value	Range	Units	Description
Signal Input Source (EA alias: Input Source)	Not Connected	Any logical output of any function block or “Not Connected”	–	Source of the input signal
Signal Input Inversion (EA alias: Input Inversion)	No	{Yes, No}	–	Specifies, whether to invert the input signal
Reset Input Source	Not Connected	Any logical output of any function block or “Not Connected”	–	Source of a reset input signal. The signal brings the output to zero.
Reset Input Inversion	No	{Yes, No}	–	Specifies, whether to invert the reset signal
Output Signal Type	Continuous	{Continuous, On/Off}	–	Type of the output signal
Ramp Up	0.0	[0…10000]	s	Ramp Up Time. Time during which the output ramps up from 0 to 1.
Ramp Down	1.0	[0…10000]	s	Ramp Down Time. Time during which the output ramps down from 1 to 0.

3.11 Switch Function

This simple logical function was added to the controller to allow users to switch between two logical signals using a third logical signal as a control. Each of the five Switch functions has two logical inputs, one control input and one logical output.



The output of the Switch Function can be described using the following equation:

$$Yn=\{Xn_1, \text{ if Control} < 0.5; Xn_2, \text{ if Control} \geq 0.5\}.$$

This function block transfers one of the input signals to the output without any processing. As the result, it preserves the original signal states and signal state codes, including error codes, of the Xn_1 and Xn_2 input signals.

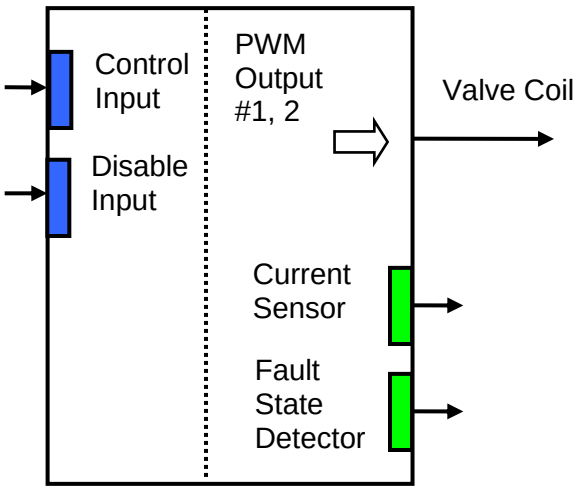
Since no processing is provided, the inversion of the input signals is not supported. The control input signal, however, can be inverted, if necessary.

The setpoints of the Switch Function block is presented in the following table:

Name	Default Value	Range	Units	Description
Control Input Source (EA alias: Control Signal Source)	Not Connected	Any logical output of any function block or "Not Connected"	–	Source of the control input signal
Control Input Inversion (EA alias: Control Signal Source Inversion)	No	{Yes, No}	–	Specifies, whether to invert the control input signal
Input #1 Source	Not Connected	Any logical output of any function block or "Not Connected"	–	Source of the input #1 signal
Input #2 Source	Not Connected	Any logical output of any function block or "Not Connected"	–	Source of the input #2 signal

3.12 PWM Output

Two PWM Output function blocks represent hardware PWM output stages of the controller. Each function block has a control and a disabled inputs to control the load, and two logical outputs: one providing data from the current sensor connected to the load, and the other – from the fault state detector. The logical inputs can be inverted.



The user can select: the output mode, minimum and maximum output values, dither parameters, and ramps. Also, PID coefficients can be set to control the output current in the "Output Current" mode. For the current sensor, the user can define an averaging time to minimize the effect of the output dither on the sensor readings.

The PWM Output function block has the following set of setpoints:

Name	Default Value	Range	Units	Description
Output Mode	Output Current	{Output Disable, Discrete On/Off, Output Current, Output Voltage, Output PWM Duty Cycle}	–	Specifies a control mode of the controller PWM output stage
Reverse Action	No	{Yes, No}	–	Defines a reverse control (increasing the input signal will decrease the output value: from Imax to Imin, etc.)
Control Input Source	Universal Input with the same number as the PWM output	Any logical output of any function block or “Not Connected”	–	Defines a source of the control input signal. This signal controls the PWM output
Control Input Inversion	No	{Yes, No}	–	Specifies, whether the control input signal is inverted
Disable Input Source	Not Connected	Any logical output of any function block or “Not Connected”	–	Defines a source of the disable input signal. This signal immediately brings the output to its original “Disabled” state[5].
Disable Input Inversion	No	{Yes, No}	–	Specifies, whether the disable input signal is inverted
Imax – Max Output Current	1.0	[0; 3], but Imax>Imin	A	Normalization parameters for Output Current mode. Imax is limited, if DithAmp is high[1].
Imin – Min Output Current	0	[0;3], but Imin<Imax	A	
Vmax – Max Output Voltage	24.0	[0; 60], but Vmax>Vmin	V	Normalization parameters for Output Voltage mode.
Vmin – Min Output Voltage	0.0	[0; 60], but Vmin<Vmax	V	
Dmax – Max PWM Duty Cycle	100.0	[0; 100], but Dmax<Dmin	%	Normalization parameters for Output PWM Duty Cycle mode.
Dmin – Min PWM Duty Cycle	0.0	[0; 100], but Dmin<Dmax	%	
RampUp – Ramp Up Time	10.0	[0; 100000]	ms	Time, during which the output ramps from its minimum to maximum value.
RampDown – Ramp Down Time	10.0	[0; 100000]	ms	Time, during which the output ramps from its maximum to minimum value.
DithFreq – Dither Frequency	100.0	[20; 400]	Hz	Frequency of the superimposed dither[2]
DithAmp – Dither Amplitude	5.0	[0; 40]	%	Point-to-point amplitude of the superimposed dither. Defined in % of the maximum output value[4]. Limited in the Output Current mode, if Imax is high[1].

Name	Default Value	Range	Units	Description
Proportional Gain	0.8	[0; 1000]	–	Proportional PID parameter. Password Protected[3].
Integral Time Constant	0.03	[0; 10]	s	Integral PID parameter. Password Protected[3].
Derivative Time Constant	0.001	[0; 10]	s	Derivative PID parameter Password Protected[3].
Current Sensor Averaging Time	100	[0; 1000]	ms	Current sensor output will be updated every specified averaging period of time with an average value calculated on the previous averaging time interval.
Current Sensor Max	3.0	–	A	Normalization parameters for the current sensor output. Read only.
Current Sensor Min	0	–	A	

[1]Due to a limited dynamic range of the current control circuit, Imax and DithAmp values should satisfy the following equation:

$$Imax \bullet (1 + DithAmp/100) \leq 3.15,$$

where: 3.15 – internal control constant.
[2]A global parameter for all PWM outputs.
[3]To avoid accidental changing of the PID parameters, they are password protected. The password is: PIDSetupNow, case sensitive. PID control loop is only used in the Output Current mode.
[4]The maximum output value is defined by the Output Mode. It is equal to: Imax in the Output Current mode, Vmax – in the Output Voltage mode and Dmax – in the Output PWM Duty Cycle mode.
[5]This state corresponds to the zero output, if Reverse Action is not activated (default). If the Reverse Action is activated, the output will be set to the maximum value, which is the value of the output when the control signal is equal to zero in this mode.

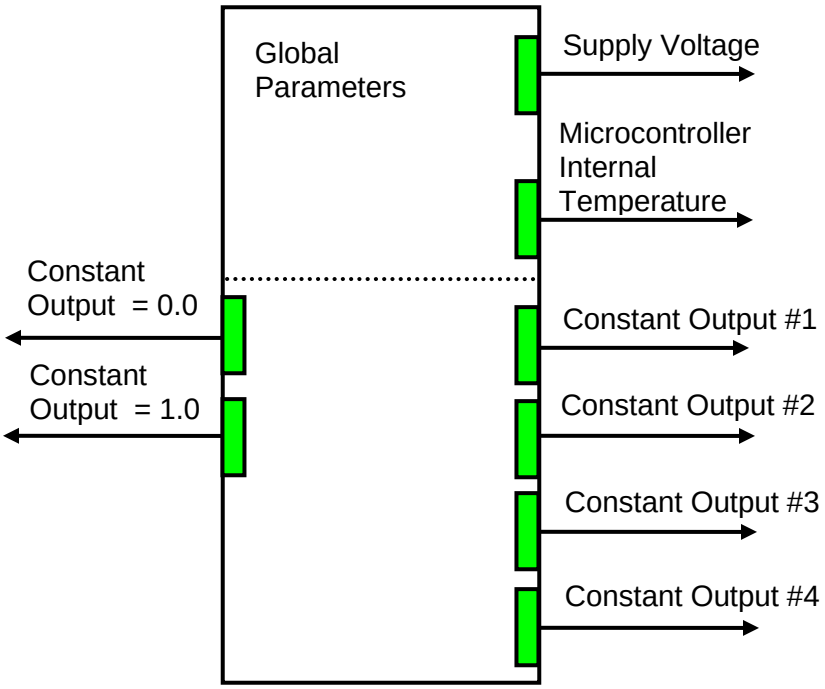
3.12.1 Fault State Detector

A fault-state detector changes its state from 0 to 1, when the PWM output is connected to ground or to the battery terminal. In case the PWM output is shorted to ground, it may be necessary to drive the output with some control signal to detect the fault condition.

The fault-state detector does not come on in the overvoltage/undervoltage condition when the supply voltage goes beyond the specified power supply voltage range (6…60 VDC), resulting the PWM outputs to be temporary shut down as a protective measure.

3.13 Global Parameters

The Global Parameters function block gives the user access to the controller supply voltage and the microcontroller internal temperature as well as to a set of six constant logical outputs. These outputs can be used by other function blocks as constant input sources. For example, they can be used to set up threshold values for Multi-Input or Binary functions.



Four out of six constant logical outputs are user programmable. Other two represent logical one and logical zero outputs.

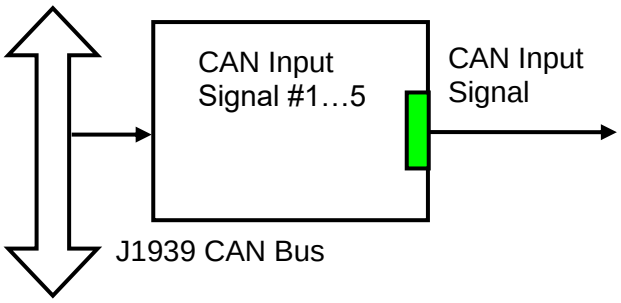
The setpoints for this function block are presented in the following table:

Name	Default Value	Range	Units	Description
Constant Output #1	0.0	[0…1]	–	Logical output with a constant value
Constant Output #2	0.0	[0…1]	–	Logical output with a constant value
Constant Output #3	0.0	[0…1]	–	Logical output with a constant value
Constant Output #4	0.0	[0…1]	–	Logical output with a constant value
Vsmax – Max Supply Voltage	70	–	V	Normalization parameters for the controller supply voltage. Read only parameters.
Vsmin – Min Supply Voltage	0	–	V	
Tmax – Max Microcontroller Temperature	150	–	°C	Normalization parameters for the microcontroller embedded temperature sensor. Read only parameters.
Tmin – Min Microcontroller Temperature	-50	–	°C	

3.14 CAN Input Signals

There are five CAN Input Signal function blocks supported by the controller. Each function block provides the controller with a logical interface to CAN application specific signals transmitted on the CAN bus. It can be programmed to read a single-frame CAN messages with virtually any CAN

signal data format and then output the signal data to its logical output for processing by other function blocks of the controller.



The function block has an ability to filter out signals transmitted only from a selected address. This way, it can be bound to a specific ECU on the CAN network. It can also automatically reset the input signal data in case the signal has been absent on the network for more than a specific period of time. CAN application specific signals transmitted by the controller itself are also processed by this function block.

The setpoints of a CAN Input Signal function block are presented in the following table:

Name	Default Value	Range	Units	Description
Signal Type	Undefined	{Undefined, 1-Bit Discrete, 2-Bit Discrete, 4-Bit Discrete, 1-Byte Continuous, 2-Byte Continuous, 4-Byte Continuous}	–	Type of the CAN input signal
PGN	65280	Any J1939 PGN value	–	PGN of the single frame CAN messages carrying the CAN input signal
PGN From Selected Address	No	{No, Yes}	–	Only CAN messages from the selected address will be accepted, if “Yes”
Selected Address	0	[0; 253]	–	Address of the ECU transmitting CAN messages carrying the CAN input signal
Data Position Byte	1	[1; 8]	–	Input signal data position byte within the CAN message data frame. LSB for continuous input signals
Data Position Bit	1	[1; 8]	–	Less significant input signal data position bit within the “Data Position Byte” for discrete input signals[1]
Resolution	1	Any value	Signal Units / Bit	CAN continuous signal resolution
Offset	0	Any value	Signal Units	CAN continuous signal offset
Signal Max Value	1	Any value, but: Signal Max Value > Signal Min Value	Signal Units	Normalization parameters for the CAN input signal. Valid

Name	Default Value	Range	Units	Description
Signal Min Value	0	Any value, but: Signal Min Value < Signal Max Value	Signal Units	only for continuous signals
Autoreset Time	500	[0; 10000]	ms	Time interval, after which the output signal will be automatically reset to “Not Available”, if a new CAN message, carrying the signal, has not arrived. If 0 – autoreset is disabled.

[1]Discrete input signals should be within the “Data Position Byte” borders, not split between the adjacent bytes.

According to the J1939/71 standard, CAN signals can carry not only signal values, but also special indicators, including: error indicator, “signal not available” indicator, etc. CAN signal types, supported by the controller, have the following CAN signal code mapping to the controller logical signals:

CAN Signal Type	CAN Signal Code	Logical Signal		
		State	Value	Signal State Code
1-Bit Discrete*	0…1	Valid Data	0…1 (=CANSignalCode)	0
2-Bit Discrete	0…1	Valid Data	0…1 (=CANSignalCode)	0
	2	Error	0	0
	3	Not Available	0	0
4-Bit Discrete*	0…1	Valid Data	0…1 (=CANSignalCode)	0
	2…13 (0x02…0x0D)	Special	0	0…11 =CANSignalCode-2
	14 (0x0E)	Error	0	0
	15 (0x0F)	Not Available	0	0
1-Byte Continuous	0…250 (0…0xFA)	Valid Data	[0;1] - normalized signal code	0
	251…253 (0xFB…0xFD)	Special	0	0…2 =CANSignalCode-251
	254 (0xFE)	Error	0	0
	255 (0xFF)	Not Available	0	0
2-Byte Continuous	0…64255 (0…0xFAFF)	Valid Data	[0;1] - normalized signal code	0
	64256…65023 (0xFB00…0xFDFF)	Special	0	0…267 =CANSignalCode-64256
	65024…65279 (0xFExx)	Error	0	0…255 =CANSignalCode-65024
	65280…65535 (0xFFxx)	Not Available	0	0
4-Byte Continuous	0…4211081215 (0… 0xFAFFFFFF)	Valid Data	[0;1] - normalized signal code	0
	4211081216… 4261412863 (0xFB000000…	Special	0	0…50331647 =CANSignalCode-4211081216

CAN Signal Type	CAN Signal Code	Logical Signal		
		State	Value	Signal State Code
	0xFDFFFFFF)			
	4261412864… 4278190079 (0xFExxxxxx)	Error	0	0…16777215 =CANSignalCode-4261412864
	4278190080… 4294967295 (0xFFxxxxxx)	Not Available	0	0

*CAN signal code mapping for these types is specific to this control.

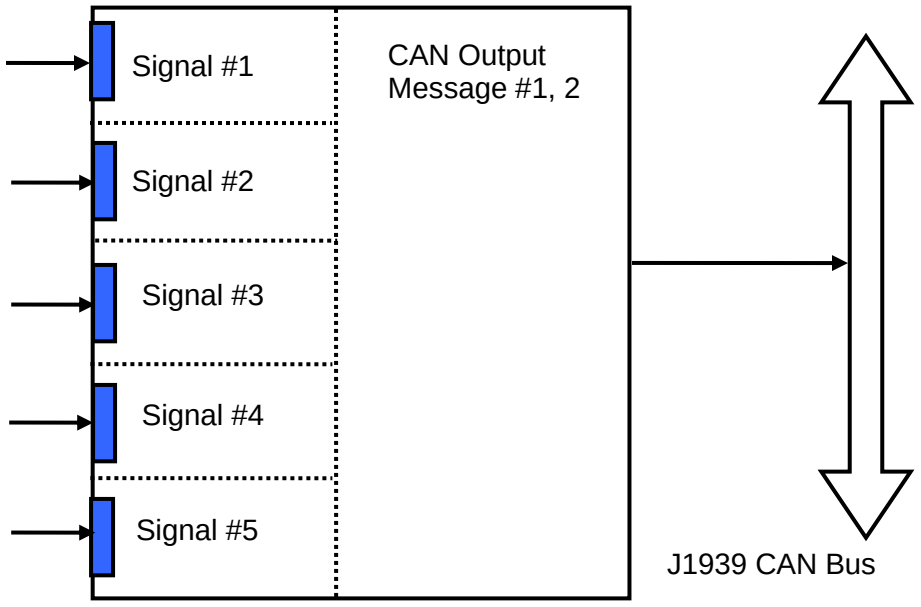
This mapping closely follows the J1939/71 standard for the 2-bit Discrete and all continuous CAN signal types, dividing the CAN code in similar ranges to represent different states of the signal. For the 1-bit and 4-bit Discrete signal types there are no generic rules specified by the J1939/71 standard to encode special indicators. The control uses its own mapping scheme for these types.

The J1939 standard does not specify how to encode the error codes and parameter specific indicators within the special indicator ranges. The control uses its own simple way of encoding, converting parameter specific and error indicators into absolute signal state codes. This allows us to receive and transmit the same codes using different CAN signal types in a consistent way.

For example, if the logical signal is in the “Error” state with the error code equal to 1, the CAN signal code carrying this error will be 650251 (0xFE01) for the “2-Byte Continuous” CAN signal type or 4261412865 (0xFE00 0001) – for the “4-Byte Continuous” CAN signal type. See also the CAN Output Messages for reverse conversion of the logical signals into the CAN signal codes.

3.15 CAN Output Messages

The two CAN Output Message function blocks allow the controller to send two independent single frame application specific CAN messages to the CAN bus. The messages can be sent continuously or upon request. Each message contains up to five user defined CAN signals. The signals can be inverted, if necessary.



The messages do not have a specific destination address. In case the PGN of the message is presented in the PDU1 format, the message is sent to the global address.

The setpoints of a CAN Output Message function block are presented in the following table:

Name	Default Value	Range	Units	Description
PGN	65281 (Out1), 65282 (Out2)	Any J1939 PGN value	–	CAN output message PGN
Transmission Enable	No	{Yes, No}		Enables the CAN output message transmission
Transmission Rate	0	[0;10000]		CAN output message transmission rate. If 0 – transmission is upon request
Signal #1 Type	Undefined	{Undefined, 1-Bit Discrete, 2-Bit Discrete, 4-Bit Discrete, 1-Byte Continuous, 2-Byte Continuous, 4-Byte Continuous}	–	Type of the CAN output signal #1
Signal #1 Source	Not Connected	Any logical output of any function block or “Not Connected”	–	Source of the CAN output signal #1
Signal #1 Inversion	No	{Yes, No}	–	Specifies, whether to invert the output signal #1
Signal #1 Data Position Byte	1	[1; 8]	–	Signal #1 data position byte within the CAN message data frame. LSB for continuous output signals
Signal #1 Data Position Bit	1	[1; 8]	–	Less significant signal #1 data position bit within the “Signal #1 Data Position Byte” for discrete output signals[1]
Signal #1 Resolution	1	Any value, except 0	Signal Units / Bit	CAN output signal #1 resolution. Valid only for continuous signals
Signal #1 Offset	0	Any value	Signal Units	CAN output signal #1 offset. Valid only for continuous signals
Signal #1 Max Value	1	Any value, but: Signal #1 Max Value > Signal #1 Min Value	Signal Units	Normalization parameters for the CAN output signal #1. Valid only for continuous signals
Signal #1 Min Value	0	Any value, but: Signal #1 Min Value < Signal #1 Max Value	Signal Units	
Signal #2 Type	Undefined	{Undefined, 1-Bit Discrete, 2-Bit Discrete, 4-Bit Discrete, 1-Byte Continuous, 2-Byte Continuous, 4-Byte Continuous}	–	Type of the CAN output signal #2

Name	Default Value	Range	Units	Description
Signal #2 Source	Not Connected	Any logical output of any function block or “Not Connected”	–	Source of the CAN output signal #2
Signal #2 Inversion	No	{Yes, No}	–	Specifies, whether to invert the output signal #2
Signal #2 Data Position Byte	1	[1; 8]	–	Signal #2 data position byte within the CAN message data frame. LSB for continuous output signals
Signal #2 Data Position Bit	1	[1; 8]	–	Less significant signal #2 data position bit within the “Signal #2 Data Position Byte” for discrete output signals[1]
Signal #2 Resolution	1	Any value, except 0	Signal Units / Bit	CAN output signal #2 resolution. Valid only for continuous signals
Signal #2 Offset	0	Any value	Signal Units	CAN output signal #2 offset. Valid only for continuous signals
Signal #2 Max Value	1	Any value, but: Signal #2 Max Value > Signal #2 Min Value	Signal Units	Normalization parameters for the CAN output signal #2. Valid only for continuous signals
Signal #2 Min Value	0	Any value, but: Signal #2 Min Value < Signal #2 Max Value	Signal Units	
Signal #3 Type	Undefined	{Undefined, 1-Bit Discrete, 2-Bit Discrete, 4-Bit Discrete, 1-Byte Continuous, 2-Byte Continuous, 4-Byte Continuous}	–	Type of the CAN output signal #3
Signal #3 Source	Not Connected	Any logical output of any function block or “Not Connected”	–	Source of the CAN output signal #3
Signal #3 Inversion	No	{Yes, No}	–	Specifies, whether to invert the output signal #3
Signal #3 Data Position Byte	1	[1; 8]	–	Signal #3 data position byte within the CAN message data frame. LSB for continuous output signals
Signal #3 Data Position Bit	1	[1; 8]	–	Less significant signal #3 data position bit within the “Signal #3 Data Position Byte” for discrete output signals[1]
Signal #3 Resolution	1	Any value, except 0	Signal Units / Bit	CAN output signal #3 resolution. Valid only for continuous signals
Signal #3 Offset	0	Any value	Signal	CAN output signal #3 offset.

Name	Default Value	Range	Units	Description
			Units	Valid only for continuous signals
Signal #3 Max Value	1	Any value, but: Signal #3 Max Value > Signal #3 Min Value	Signal Units	Normalization parameters for the CAN output signal #3. Valid only for continuous signals
Signal #3 Min Value	0	Any value, but: Signal #3 Min Value < Signal #3 Max Value	Signal Units	
Signal #4 Type	Undefined	{Undefined, 1-Bit Discrete, 2-Bit Discrete, 4-Bit Discrete, 1-Byte Continuous, 2-Byte Continuous, 4-Byte Continuous}	–	Type of the CAN output signal #4
Signal #4 Source	Not Connected	Any logical output of any function block or “Not Connected”	–	Source of the CAN output signal #4
Signal #4 Inversion	No	{Yes, No}	–	Specifies, whether to invert the output signal #4
Signal #4 Data Position Byte	1	[1; 8]	–	Signal #4 data position byte within the CAN message data frame. LSB for continuous output signals
Signal #4 Data Position Bit	1	[1; 8]	–	Less significant signal #4 data position bit within the Signal #4 Data Position Byte for discrete output signals[1]
Signal #4 Resolution	1	Any value, except 0	Signal Units / Bit	CAN output signal #4 resolution. Valid only for continuous signals
Signal #4 Offset	0	Any value	Signal Units	CAN output signal #4 offset. Valid only for continuous signals
Signal #4 Max Value	1	Any value, but: Signal #4 Max Value > Signal #4 Min Value	Signal Units	Normalization parameter for the CAN output signal #4. Valid only for continuous signals
Signal #4 Min Value	0	Any value, but: Signal #4 Min Value < Signal #4 Max Value	Signal Units	
Signal #5 Type	Undefined	{Undefined, 1-Bit Discrete, 2-Bit Discrete, 4-Bit Discrete, 1-Byte Continuous, 2-Byte Continuous, 4-Byte Continuous}	–	Type of the CAN output signal #5
Signal #5 Source	Not Connected	Any logical output of any function block or “Not Connected”	–	Source of the CAN output signal #5
Signal #5 Inversion	No	{Yes, No}	–	Specifies, whether to invert the output signal #5

Name	Default Value	Range	Units	Description
Signal #5 Data Position Byte	1	[1; 8]	–	Signal #5 data position byte within the CAN message data frame. LSB for continuous output signals
Signal #5 Data Position Bit	1	[1; 8]	–	Less significant signal #5 data position bit within the “Signal #5 Data Position Byte” for discrete output signals[1]
Signal #5 Resolution	1	Any value, except 0	Signal Units / Bit	CAN output signal #5 resolution. Valid only for continuous signals
Signal #5 Offset	0	Any value	Signal Units	CAN output signal #5 offset. Valid only for continuous signals
Signal #5 Max Value	1	Any value, but: Signal #5 Max Value > Signal #5 Min Value	Signal Units	Normalization parameter for the CAN output signal #5. Valid only for continuous signals.
Signal #5 Min Value	0	Any value, but: Signal #5 Min Value < Signal #5 Max Value	Signal Units	

[1]CAN discrete signals should be within the “Data Position Byte” borders, not split between the adjacent bytes.

The logical signals can carry not only signal values but also error and special codes reflecting different states of the logical signal. The logical signals are converted into CAN signal codes the same way as in the CAN Input Signal function block, closely following the J1939/71 standard when possible. See the table below:

CAN Signal Type	Logical Signal			CAN Signal Code
	State	Value	Signal State Code	
1-Bit Discrete	Valid Data	[0;1]	0	0, if Value<0.5 1, if Value≥0.5
	Special*	0	0…4294967295 (0…0xFFFFFFFF)	1
	Error*	0	0…4294967295 (0…0xFFFFFFFF)	1
	Not Available*	0	0	1
2-Bit Discrete	Valid Data	[0;1]	0	0, if Value<0.5 1, if Value≥0.5
	Special*	0	0…4294967295 (0…0xFFFFFFFF)	3 (Same as “Not Available”)
	Error	0	0…4294967295 (0…0xFFFFFFFF)	2
	Not Available	0	0	3
4-Bit Discrete*	Valid Data	[0;1]	0	0, if Value<0.5 1, if Value≥0.5
	Special	0	0…4294967295 (0…0xFFFFFFFF)	2…13 (0x02…0x0D) =SignalStateCode+2, if

CAN Signal Type	Logical Signal			CAN Signal Code
	State	Value	Signal State Code	
				SignalStateCode<12 =13, if SignalStateCode ≥12
	Error	0	0…4294967295 (0…0xFFFFFFFF)	14 (0x0E)
	Not Available	0	0	15 (0x0F)
1-Byte Continuous	Valid Data	[0;1]	0	0…250 (0…0xFA) – calculated from the Value using normalization parameters
	Special	0	0…4294967295 (0…0xFFFFFFFF)	251…253 (0xFB…0xFD) = SignalStateCode+251, if SignalStateCode<3, =253, if SignalStateCode ≥3
	Error	0	0…4294967295 (0…0xFFFFFFFF)	254 (0xFE)
	Not Available	0	0	255 (0xFF)
2-Byte Continuous	Valid Data	[0;1]	0	0…64255 (0…0xFAFF) – calculated from the Value using normalization parameters
	Special	0	0…4294967295 (0…0xFFFFFFFF)	64256…65023 (0xFB00…0xFDFF) = SignalStateCode+64256, if SignalStateCode<768, =65023, if SignalStateCode ≥768
	Error	0	0…4294967295 (0…0xFFFFFFFF)	65024…65279 (0xFExx) = SignalStateCode+65024, if SignalStateCode<256, =65279, if SignalStateCode ≥256
	Not Available	0	0	65535 (0xFFFF)
4-Byte Continuous	Valid Data	[0;1]	0	0…4211081215 (0… 0xFAFFFFFF) – calculated from the Value using normalization parameters
	Special	0	0…4294967295 (0…0xFFFFFFFF)	4211081216… 4261412863 (0xFB000000… 0xFDFFFFFF) =SignalStateCode+4211081216, if SignalStateCode<50331648, =4261412863, if SignalStateCode ≥50331648
	Error	0	0…4294967295 (0…0xFFFFFFFF)	4261412864… 4278190079 (0xFExxxxxx) =SignalStateCode+4261412864, if SignalStateCode<16777216, =4278190079, if SignalStateCode ≥16777216
	Not Available	0	0	4294967295 (0xFFFFFFFF)

*Conversion rules are specific to this control. They are not defined by the J1939/71 standard.

4 NETWORK SUPPORT

The controller is designed to work on the J1939 CAN network. When connected to the network or upon power up, it automatically recognizes the network connection, claims a network address, and then starts a network communication.

Several CAN baud rates are supported. The most common J1939 250kBit/s baud rate is supported by units with P/N AX021800, AX021810. Units with P/N AX021801 and AX02181 support J1939 500kBit/s baud rate. For customers requiring the maximum CAN bandwidth, a non-standard 1MBit/s baud rate is supported by units with P/N AX021802, AX021812.

The network part of the controller is compliant with Bosch CAN protocol specification, Rev.2.0, Part B, and the following J1939 standards:

ISO/OSI Network Model Layer	**J1939 Standard**
Physical	*For P/N: AX021800, AX021810* J1939/11 – Physical Layer, 250K bit/s, Twisted Shielded Pair. Rev. SEP 2006. J1939/15 - Reduced Physical Layer, 250K bits/sec, Un-Shielded Twisted Pair (UTP). Rev. AUG 2008.
	For P/N: AX021801, AX021811 J1939/14 - Physical Layer, 500 Kbps, OCT2011.
Data Link	J1939/21 – Data Link Layer. Rev. DEC 2006
	The controller supports Transport Protocol for Commanded Address messages (PGN 65240) and software identification -SOFT messages (PGN 65242). It also supports responses on PGN Requests (PGN 59904).
Network	J1939, Appendix B – Address and Identity Assignments. Rev. FEB 2010. J1939/81 – Network Management. Rev. 2003-05.
	The controller is an Arbitrary Address Capable ECU. It can dynamically change its network address in real time to resolve an address conflict with other ECUs. The controller supports: Address Claimed Messages (PGN 60928), Requests for Address Claimed Messages (PGN 59904) and Commanded Address Messages (PGN 65240).
Transport	N/A in J1939.
Session	N/A in J1939.
Presentation	N/A in J1939.
Application	J1939/71 – Vehicle Application Layer. Rev. FEB 2010
	The controller can receive application specific PGNs with input signals and transmit application specific PGNs with up to five output signals. All application specific PGNs are user programmable.
	J1939/73 – Application Layer – Diagnostics. Rev. FEB 2010
	Memory access protocol (MAP) support: DM14, DM15, DM16 messages used by EA to program setpoints.

4.1 J1939 Name and Address

Upon connecting to the network, before sending and receiving any application data, the controller claims its network address with the unique J1939 Name. The Name fields are presented in the table bellow.

Field Name	Field Length	Field Value	User Programmable
Arbitrary Address Capable	1 bit	1 (Capable)	No
Industry Group	3 bit	0 (Global)	No
Vehicle System Instance	4 bit	0 (First Instance)	No
Vehicle System	7 bit	0 (Nonspecific System)	No
Reserved	1 bit	0	No
Function	8 bit	66 (I/O Controller)	No
Function Instance	5 bit	14 (Fifteenth Instance)	No
ECU Instance	3 bit	0 (First Instance)	Yes
Manufacturer Code	11 bit	162 (Axiomatic Technologies Corp.)	No
Identity Number	21 bit	Calculated on the base of the microcontroller unique ID	No

The user can change the controller ECU Instance using EA to accommodate multiple controllers on the same CAN network.

The controller takes its network address from a pool of addresses assigned to self configurable ECUs. The address is preset to 151, but the controller can change it during an arbitration process or upon receiving a commanded address message. The new address value is then stored in a non-volatile memory and is used during the next address claim procedure. The user can also change the controller network address using EA, if necessary.

4.2 Slew Rate Control

Controllers supporting the standard 250kBit/s baud rate have an ability to adjust the CAN transceiver slew rate for better performance on the physical network. The *Slew Rate* setpoint can be set according to the following table:

Setpoint Value	Slew Rate
Fast	19 V/µs
Slow	4 V/ µs

For the majority of J1939 250kBit/s CAN applications the slow slew rate is preferable due to the reduced EMI of the transceiver.

On the contrary, the fast slew rate is always set for controllers supporting the higher baud rates of 500kBit/s and 1Mbit/s. The user will not be able to change the slew rate from fast to slow using EA in this case.

4.3 Network Bus Terminating Resistors

The controller does not have an embedded 120Ohm CAN bus terminating resistor. The appropriate resistors should be installed externally on both ends of the CAN twisted pair cable according to the J1939/11 or J1939/15 standards.

Even if the length of the CAN network is short and the signal reflection from both ends of the cable can be ignored, at least one 120Ohm resistor is required for the majority of CAN transceivers to operate properly.

4.4 Network Setpoint Group

The following table summarizes the EA programmable setpoints controlling the controller CAN network functionality:

Name	Default Value	Range	Units	Description
ECU Instance Number	0	[0…7]	–	ECU Instance field of the J1939 ECU Name.
ECU Address	151	[0…253]		ECU Address
Slew Rate	*For 250kBit/s units:* Slow	{Slow, Fast}	–	Slew rate control of the CAN transceiver
	For 500kBit/s and 1Mbit/s units: Fast	Fast		

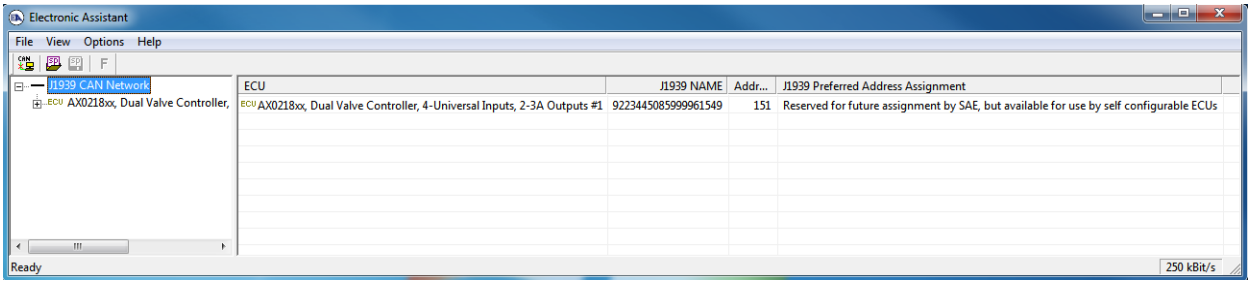
5 SETPOINT PROGRAMMING

The controller setpoints can be viewed and programmed using the standard J1939 memory access protocol through the CAN bus using Axiomatic PC-based Electronic Assistant (EA) software.

5.1 Axiomatic Electronic Assistant Software

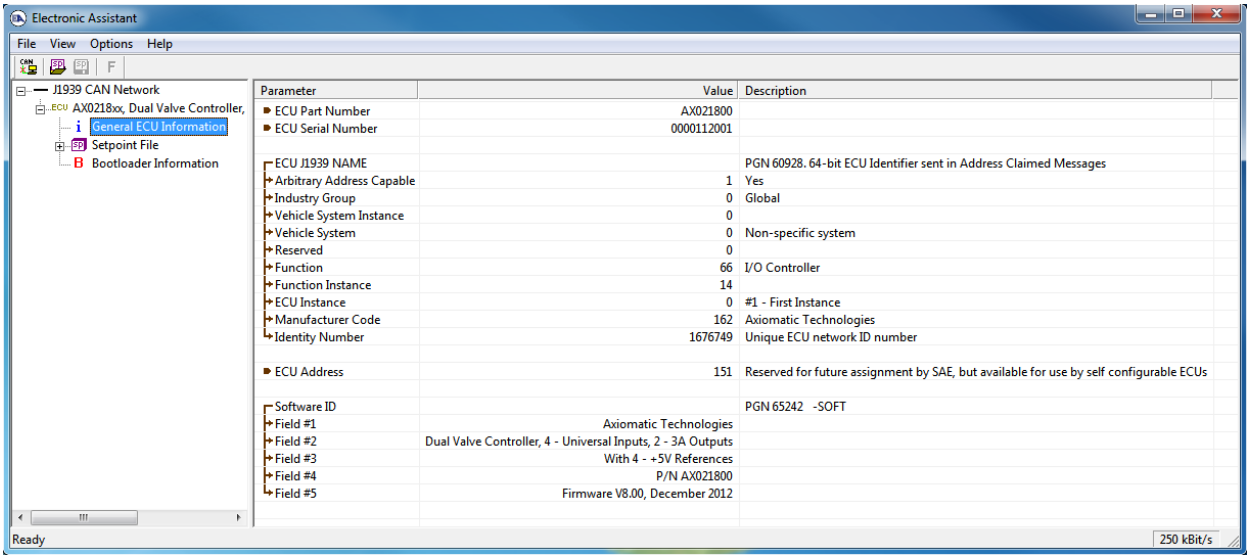
Axiomatic provides PC-based Electronic Assistant (EA) software, together with a USB-CAN converter, as a kit P/N AX070502 or AX070506K, to communicate with a wide range of Axiomatic controls, including this controller. Please also refer to the EA user manual UMAX07050X for the description of the EA and for the network connection troubleshooting.

To connect to the control, the user should first select the proper baud rate in EA, according to the controller part number. Upon connection, EA will show the controller on the list of controls that are present on the J1939 CAN network. If there is only one controller on the network, the following screen will appear for a 250kBit/s controller:



For 500kBit/s and 1Mbit/s controllers the baud rate in the bottom-right part of the screen will be different.

The user can then browse through the ECU parameters, read general ECU information, and Bootloader Information, view and modify setpoints:

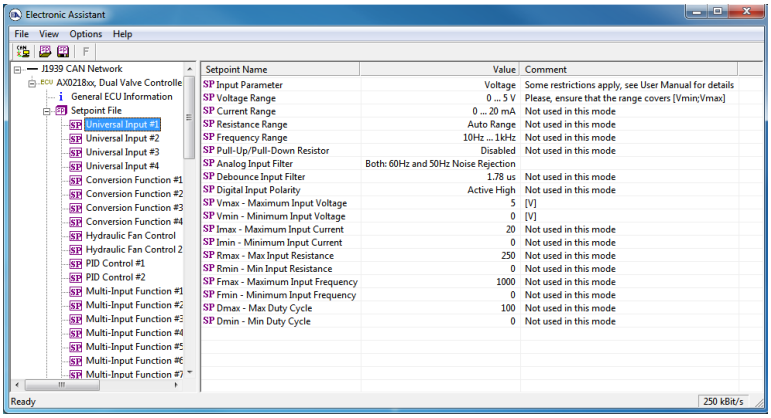


The setpoints are grouped on the basis of their functionality. Please, refer to the appropriate sections of this manual describing the required function block or a setpoint group.

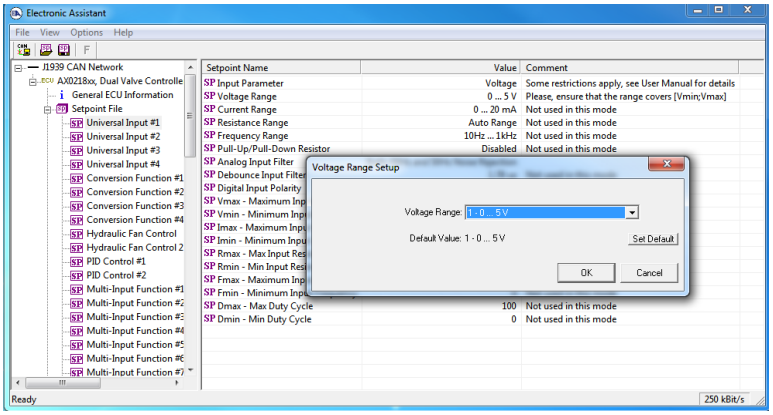
In the General ECU Information group, the user will see the version number of the application firmware. Please, make sure that the user manual version number matches with the most significant part of the application firmware version number. Otherwise, a different user manual is required to work with the controller.

5.2 Function blocks in EA

Each controller function block is presented by its own setpoint group in the Setpoint File main group. Individual setpoints of the function blocks can be accessed through these setpoint groups:



The user can view and, when necessary, change setpoints by double-clicking on the appropriate setpoint name. The setpoint pop-up dialog window will appear:



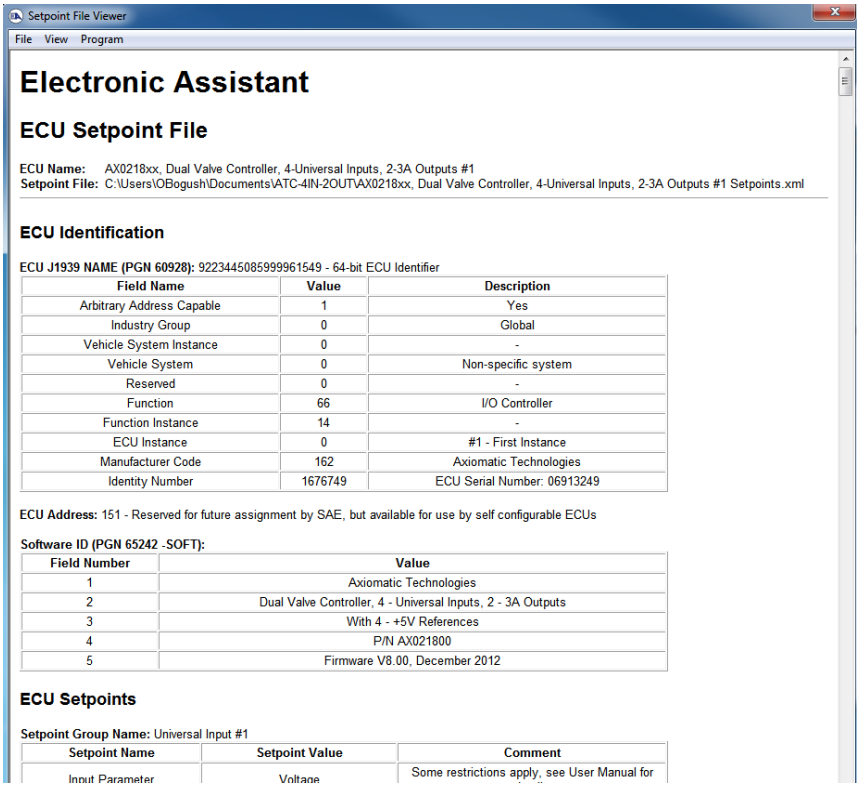
The controller will perform an internal reset of all function blocks after each change of the setpoints. If the new setpoint affects the network identification, the controller will reclaim its network address with a new network identification message, see J1939 Name and Address.

All controller function blocks are described in subsections of the Controller Architecture section. The *Network* setpoint group is described in the Network Setpoint Group subsection of the Network Support section of this manual.

5.3 Setpoint File

The EA can store all controller setpoints in one setpoint file and then flash them into the controller in one operation.

The setpoint file is created and stored on disk using a command *Save Setpoint File* from the EA menu or toolbar. The user then can open the setpoint file, view or print it and flash the setpoint file into the controller.


Setpoint File Viewer

File View Program

Electronic Assistant

ECU Setpoint File

ECU Name: AX0218xx, Dual Valve Controller, 4-Universal Inputs, 2-3A Outputs #1
Setpoint File: C:\Users\OBogush\Documents\ATC-4IN-2OUT\AX0218xx, Dual Valve Controller, 4-Universal Inputs, 2-3A Outputs #1 Setpoints.xml

ECU Identification

ECU J1939 NAME (PGN 60928): 9223445085999961549 - 64-bit ECU Identifier

Field Name	Value	Description
Arbitrary Address Capable	1	Yes
Industry Group	0	Global
Vehicle System Instance	0	-
Vehicle System	0	Non-specific system
Reserved	0	-
Function	66	I/O Controller
Function Instance	14	-
ECU Instance	0	#1 - First Instance
Manufacturer Code	162	Axiomatic Technologies
Identity Number	1676749	ECU Serial Number: 06913249

ECU Address: 151 - Reserved for future assignment by SAE, but available for use by self configurable ECUs

Software ID (PGN 65242 -SOFT):

Field Number	Value
1	Axiomatic Technologies
2	Dual Valve Controller, 4 - Universal Inputs, 2 - 3A Outputs
3	With 4 - +5V References
4	P/N AX021800
5	Firmware V8.00, December 2012

ECU Setpoints

Setpoint Group Name: Universal Input #1

Setpoint Name	Setpoint Value	Comment
Input Parameter	Voltage	Some restrictions apply, see User Manual for

The network identification and “read-only” setpoints are not transferrable using this operation. Also, the controller will perform one or several internal resets of all function blocks during the setpoint flashing operation.

There can be small differences in setpoints between different versions of the application firmware. It is recommended that the user manually inspect all setpoints after flashing if the setpoint file was created by a different version of the application firmware.

5.4 Default Setpoint Settings

The controller is preprogrammed by the manufacturer with default setpoint values. These values can be found for each internal function block in the Controller Architecture section of this manual.

In the default configuration, both PWM Output function blocks are set to output current and are connected to the Universal Input function blocks. The universal inputs are programmed to accept voltages in the 0…5 voltage range, see the block diagram on Figure 5.

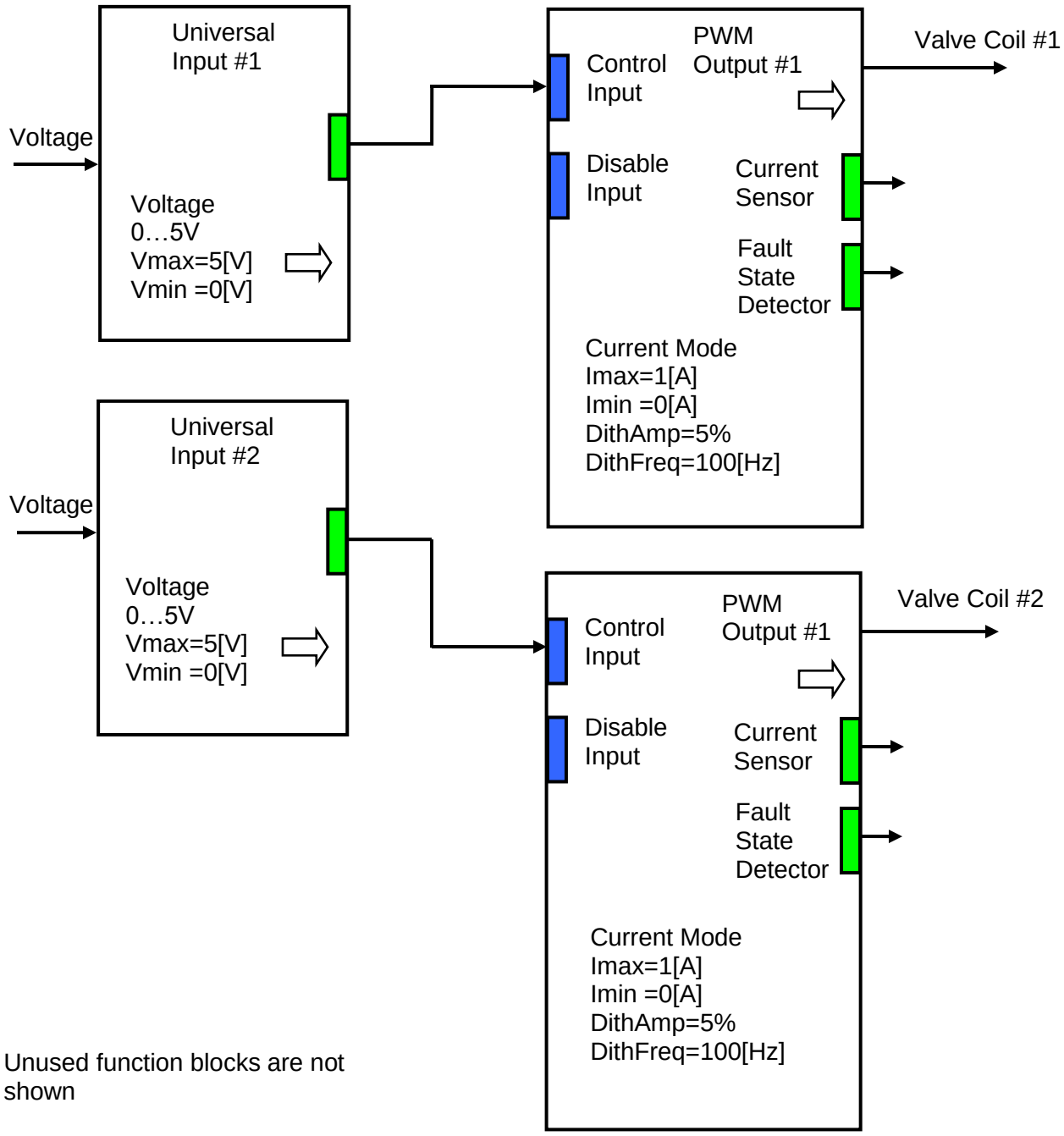


Figure 5. Default Controller Configuration

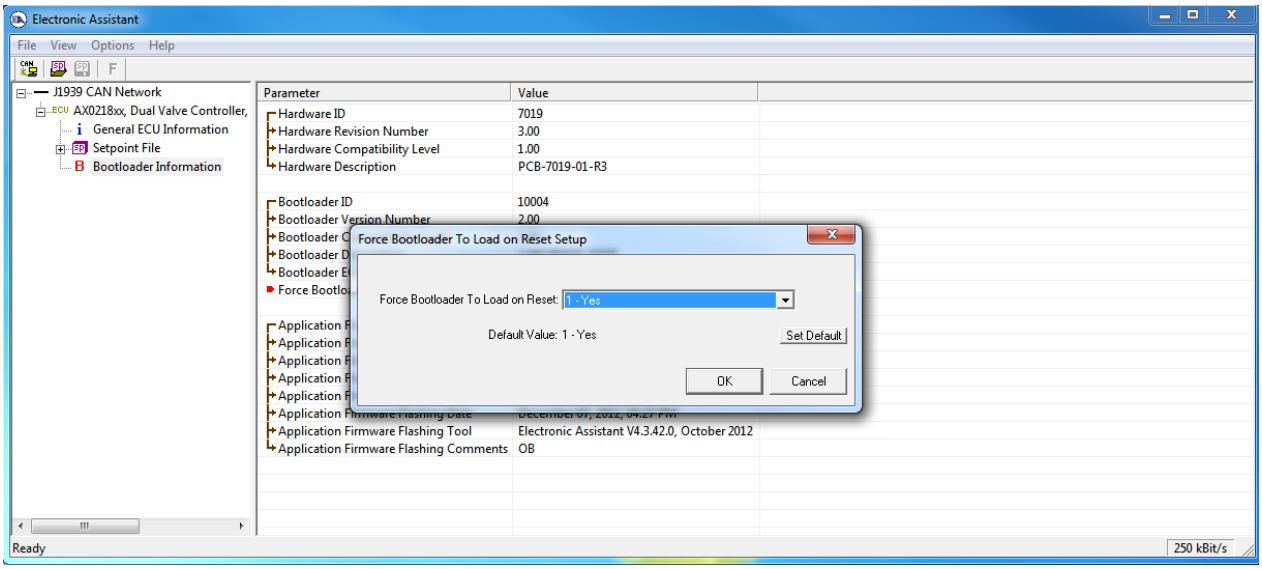
As the result, this simple configuration outputs currents from 0 to 1A, when voltages at the physical inputs are changes from 0 to 5V.

This default controller configuration is set only as an example. The user should use EA to program a user-specific controller configuration on the base of the required controller functionality.

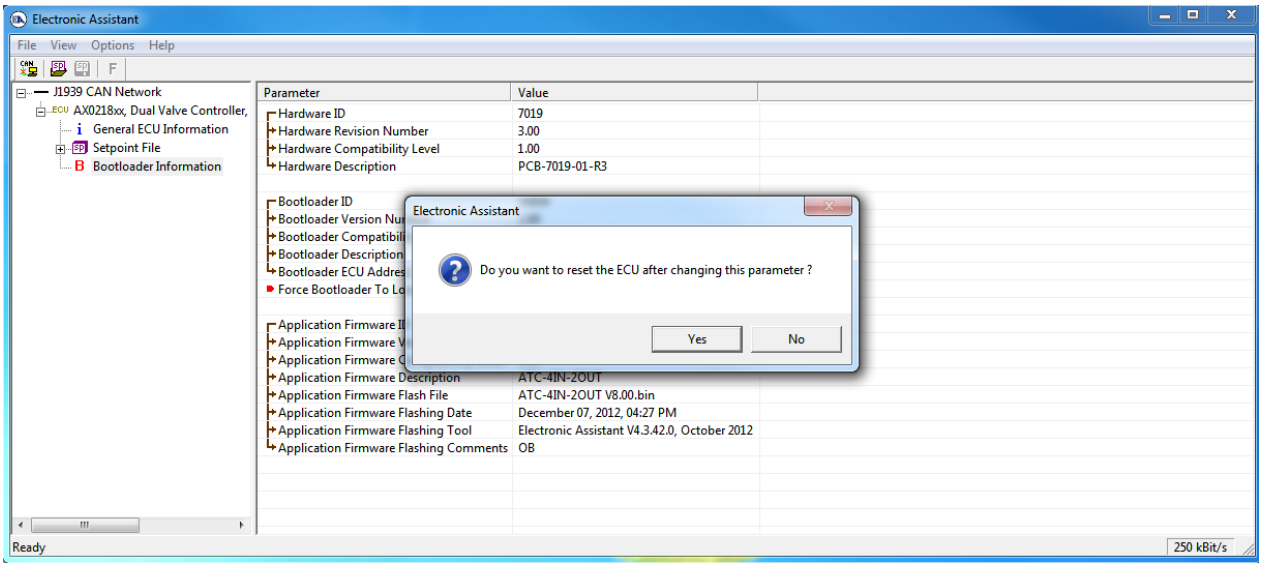
6 FLASHING NEW FIRMWARE

The controller application firmware can be updated in the field starting from V8.00. The Axiomatic EA 4.4.42.0 or later will be necessary to perform this operation. The user should contact Axiomatic to obtain a flash file with the new firmware before starting the flashing operation. Changing baud rate of the unit by flashing a new firmware supporting a different baud rate is not allowed.

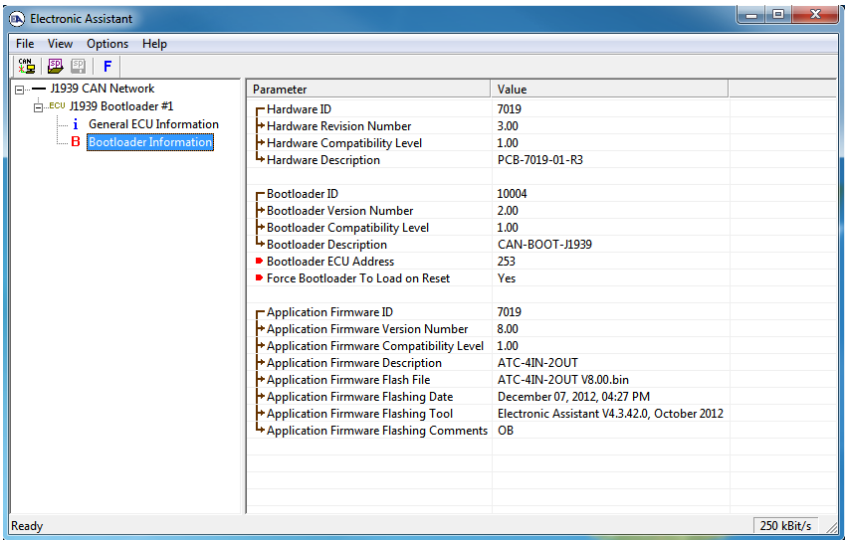
To flash the new firmware, the user should activate the embedded bootloader. To do so, start the EA and in the *Bootloader Information* group pane click on the *Force Bootloader to Load on Reset* parameter. The following dialog will appear:



The EA will prompt the user to change the *Force Bootloader to Load on Reset* parameter flag to *Yes*. This will automatically activate the bootloader on the next ECU reset. After accepting the change, the next screen will ask the user if the reset is actually required. Select *Yes*.

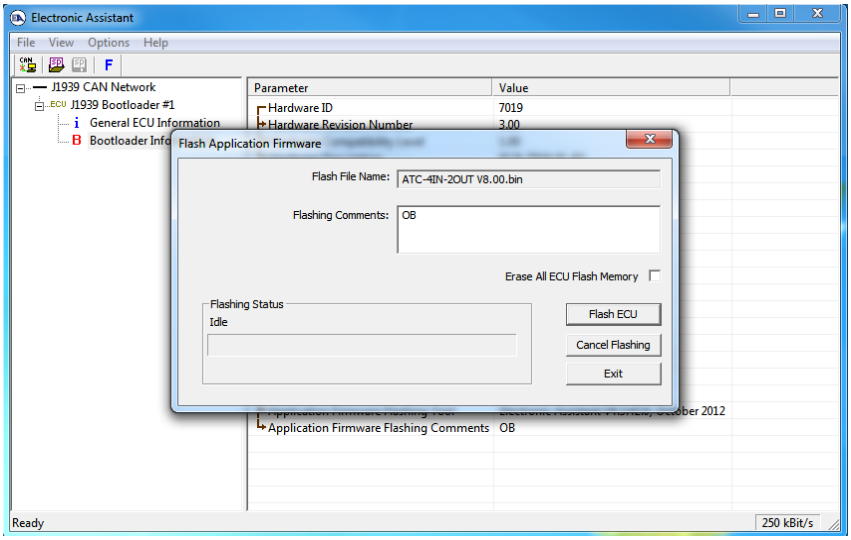


After automatic reset, instead of *AX0218XX, Dual Valve Controller, 4-Universal Inputs, 2-3A Outputs*, the user will see *J1939 Bootloader* ECU in the *J1939 CAN Network* top level group in the EA. This means that the bootloader is activated and ready to accept the new firmware. All the bootloader specific information: controller hardware, bootloader details and the currently installed application firmware remains the same in the bootloader mode and the user can read it in the *Bootloader Information* group pane.



At this point, the user can return to the installed controller firmware by changing the *Force Bootloader to Load on Reset* flag back to *No* and resetting the ECU.

To flash the new firmware, the user should click on F toolbar icon or from the *File* menu select the *Open Flash File* command. The *Open Application Firmware Flash File* dialog will appear. Pick up the flash file with the new controller firmware and confirm selection by pressing the *Open* button. The *Flash Application Firmware* dialog window will appear[1].

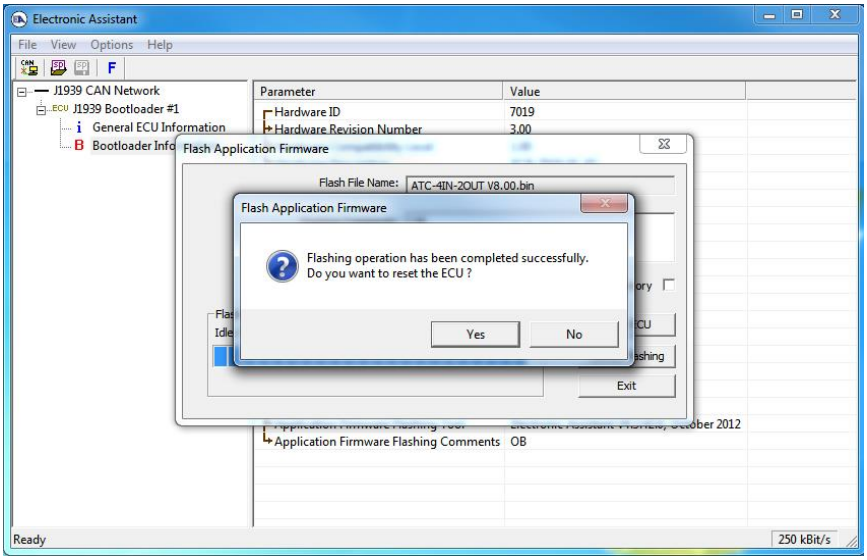


[1] In this example, instead of the new firmware the old firmware is being simply re-flashed.

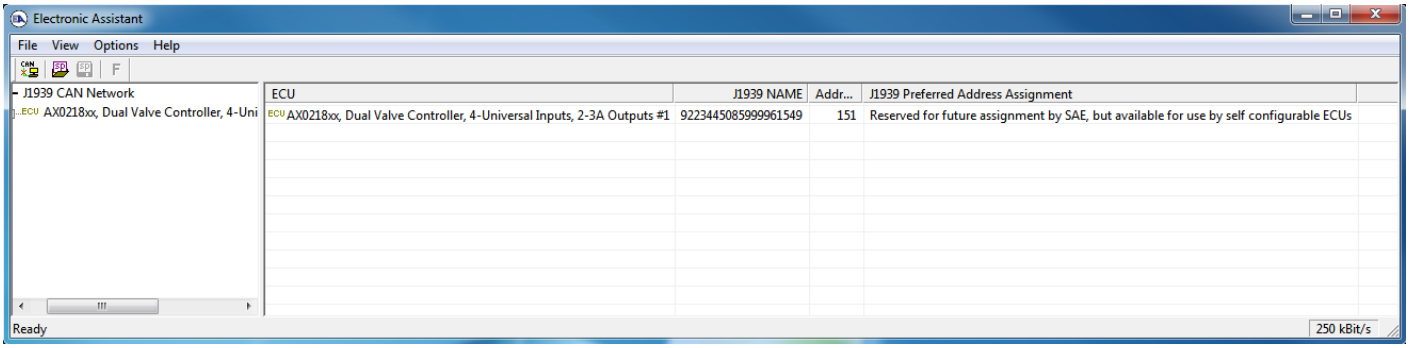
Now the user can add any comments to the flashing operation in the *Flashing Comments* field. They will be stored in the *Bootloader Information* group after flashing.

The user can also check the *Erase All ECU Flash Memory* flag to erase all setpoint values set by the old firmware and force the controller to load default setpoints after flashing the new firmware.

Select the *Flash ECU* button to start flashing. A reminder that the old application firmware will be destroyed by the flashing operation will appear. Press *Ok* to continue and watch the dynamics of the flashing operation in the *Flashing Status* field. When flashing is done, the following screen will appear prompting the user to reset the ECU.



Select *Yes* and see the ECU running the new firmware. This will indicate that the flashing operation has been performed successfully.



For more information, see the *J1939 Bootloader* section of the EA user manual.

7 TECHNICAL SPECIFICATIONS

Refer to Technical Datasheets # TDAX021800, TDAX021801, TDAX021802 and TDAX021810, TDAX021811, TDAX021812 for more details.

Specifications are indicative and subject to change. Actual performance will vary depending on the application and operating conditions. Users should satisfy themselves that the product is suitable for use in the intended application.
All our products carry a limited warranty against defects in material and workmanship. Please refer to our Warranty, Application Approvals/Limitations and Return Materials Process as described on https://www.axiomatic.com/service/.

Input Specifications

Power Supply Input - Nominal	12, 24, 48VDC nominal (8…60 VDC power supply range)
Protection	Reverse polarity protection is provided. Overvoltage protection up to 65V is provided. Overvoltage (undervoltage) shutdown of the output load is provided.
CAN	SAE J1939 Commands. For baud rate see the table: **AX021800, AX021810** — 250 kBit/s — J1939/11, J1939/15. Most common **AX021801, AX021811** — 500 kBit/s — J1939/14. New standard **AX021802, AX021812** — 1Mbit/s — Non-standard CANopen® is available on request.
Universal Signal Inputs	4 universal inputs are provided. Refer to the following table “Input-User Selectable Options”. All input modes are user selectable.

Input – User Selectable Options	
Analog Input Functions	Voltage Input, Current Input or Resistive Input
Voltage Input	0-1V (Impedance 1 MOhm) 0-2.5V (Impedance 1 MOhm) 0-5V (Impedance 200 KOhm) 0-10V (Impedance 133 KOhm)
Current Input	0-20 mA (Resistance 124 Ohm) 4-20 mA (Resistance 124 Ohm)
Resistive Input	25Ω to 250 kΩ
Digital Input Functions	Discrete Input, PWM Input, Frequency Input
Digital Input Level	5V CMOS
PWM Input	0 to 100% 10 Hz to 1kHz 100 Hz to 10 kHz
Frequency Input	The controller can interface to sensors with a pulse output. 10 Hz to 1kHz 100 Hz to 10 kHz NOTE: • PWM/Frequency input mode can be configured on only two (out of four) inputs: input #1 and #3. • If PWM/Frequency mode is chosen on input #1, then all other inputs cannot be used as analog inputs (for measuring Voltage, Current or Resistance). • If PWM/Frequency mode is chosen on input # 3, all other inputs remain as analog inputs (Voltage, Current or Resistive) or digital inputs.
Digital Input	Active High or Active Low
Input Impedance	1 MOhm High impedance, 10KOhm pull-down, 10KOhm pull-up to +5V
Input Accuracy	≤ 1%
Input Resolution	12-bit

Output Specifications

CAN	SAE J1939 Messages. For baud rate see the table: **AX021800, AX021810** — 250 kBit/s — J1939/11, J1939/15. Most common **AX021801, AX021811** — 500 kBit/s — J1939/14. New standard **AX021802, AX021812** — 1Mbit/s — Non-standard

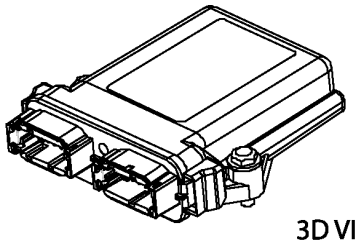
Outputs	2 outputs are provided. Up to 3A Half bridge with High Side, Current Sensing, Grounded Load The user can select the following options for output using the EA. • Output Disable • Discrete Output • Output Current (PID loop*, with current sensing) • Output Voltage • Output PWM Duty Cycle *Parameters are password protected. Refer to the user manual for details.
Output Accuracy	Output Current mode ≤2% Output Voltage mode ≤3% Output PWM Duty Cycle mode ≤3%
+5V Reference Voltages **(only for AX021800, AX021801, AX021802)**	4 reference voltages are provided. +5V, 100 mA (current limited to 115 mA)
RS232 Service Port **(only for AX021810, AX021811, AX021812)**	One RS232 service port. The user can monitor the internal state of the controller including the outputs of all internal function blocks through this port. The port can be also used for flashing of the new firmware (contact Axiomatic for details).
Protection for Output + Terminal	Fully protected against short circuit to ground and short circuit to power supply rail. Unit will fail safe in the case of a short circuit condition, self-recovering when the short is removed.

General Specifications

Operating Conditions	-40 to 85 °C (-40 to 185 °F)
Weight	0.55 lbs. (0.250 kg)
Protection	IP67 rating for the product assembly NOTE: TE Deutsch equivalent connectors are rated at IP67 for submersion (3 ft., 0.9 m) and IP69K for high pressure, high temperature wash down applications.
Quiescent Current	0.03A @24VDC
Microcontroller	32-bit, 128 KByte or larger program memory
Control Logic	User programmable functionality using Axiomatic Electronic Assistant
Communications	1 CAN port (SAE J1939). For baud rate see the table: **AX021800, AX021810** — 250 kBit/s — J1939/11, J1939/15. Most common **AX021801, AX021811** — 500 kBit/s — J1939/14. New standard **AX021802, AX021812** — 1Mbit/s — Non-standard 1 RS232 Service Port **(only for AX021810, AX021811, AX021812)**
User Interface	Axiomatic Electronic Assistant for *Windows* operating systems It comes with a royalty-free license for use. To use the Axiomatic Electronic Assistant, an USB-CAN converter links the device's CAN port to a *Windows*-based PC. An Axiomatic USB-CAN Converter AX070501 is available as part of the Axiomatic Configuration KIT. **P/N: AX070502 or AX070506K**, the Axiomatic Configuration KIT includes the following. USB-CAN Converter P/N: AX070501 1 ft. (0.3 m) USB Cable P/N: CBL-USB-AB-MM-1.5 12 in. (30 cm) CAN Cable with female DB-9 P/N: CAB-AX070501 AX070502IN CD P/N: CD-AX070502, includes: Axiomatic Electronic Assistant software; EA & USB-CAN User Manual UMAX07050X; USB-CAN drivers & documentation; CAN Assistant (Scope and Visual) software & documentation; and the SDK Software Development Kit.

8 INSTALLATION INSTRUCTIONS

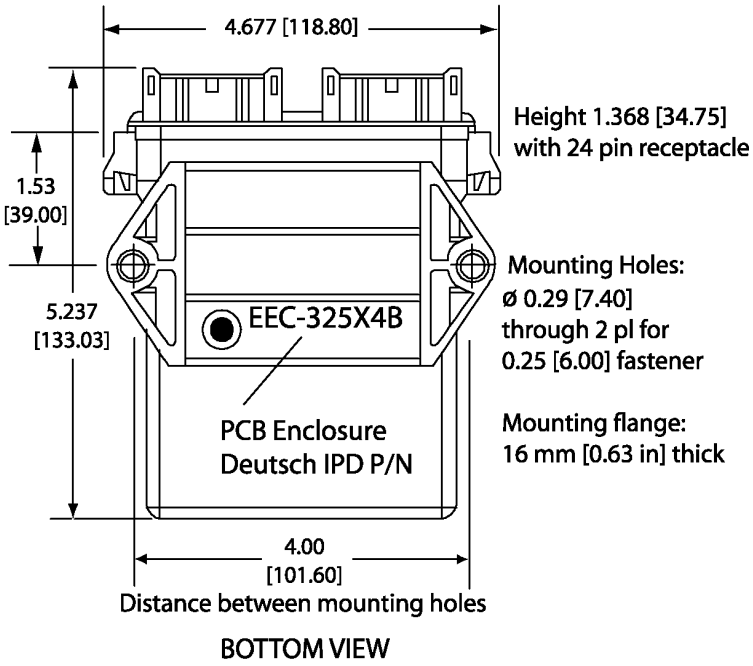
8.1 Dimensions and Pinout



3D VIEW
Housing with 24 Pin Receptacle

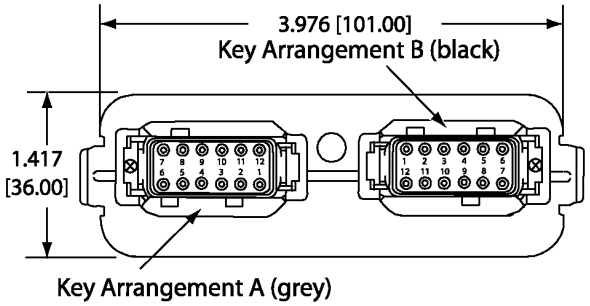
HOUSING DIMENSIONS

Housing Material: High Temperature Nylon (Black)



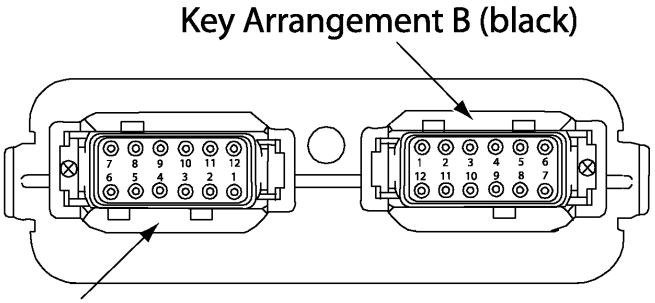
BOTTOM VIEW

FRONT VIEW 24-PIN RECEPTACLE (NOT TO SCALE)



Mating Plug Assemblies for 24-pin receptacle:
Deutsch IPD P/N: DTM06-12SA and DTM06-12SB
with wedgelocks WM12S and contacts
(Contact factory for contact specification.)

Dimensions: inches [mm]
excluding mating plug(s)



FRONT VIEW 24 PIN RECEPTACLE

TE Deutsch equivalent DTM series 24 pin receptacle (DTM13-12PA-12PB-R008)
Mating plug: TE Deutsch equivalent DTM06-12SA and DTM06-12SB with 2 wedgelocks (WM12S)
and 24 contacts (0462-201-20141). 20 AWG wire is recommended for use with contacts 0462-201-20141.
Use dielectric grease on the pins when installing the controller.

Pin out: AX021800

Grey Connector		Black Connector	
Pin #	**Function**	**Pin #**	**Function**
1	CAN Shield	1	+5V Reference 1
2	Earth (Chassis) GND	2	Universal Input 1
3	Solenoid Valve Output 2 – (internally connected to Power GND)	3	Analog GND 1
4	Solenoid Valve Output 1 – (internally connected to Power GND)	4	Analog GND 2
5	Power GND	5	Universal Input 2
6	Power GND	6	+5V Reference 2
7	Power +	7	+5V Reference 3
8	Power +	8	Universal Input 3
9	Solenoid Valve Output 1 +	9	Analog GND 3
10	Solenoid Valve Output 2 +	10	Analog GND 4
11	CAN High	11	Universal Input 4
12	CAN Low	12	+5V Reference 4

Pin out: AX021810

Grey Connector		Black Connector	
Pin #	**Function**	**Pin #**	**Function**
1	CAN Shield	1	Proprietary (Leave Unconnected)
2	Earth (Chassis) GND	2	Universal Input 1
3	Solenoid Valve Output 2 – (internally connected to Power GND)	3	Analog GND 1
4	Solenoid Valve Output 1 – (internally connected to Power GND)	4	Analog GND 2
5	Power GND	5	Universal Input 2
6	Power GND	6	RS-232 RxD
7	Power +	7	RS-232 TxD
8	Power +	8	Universal Input 3
9	Solenoid Valve Output 1 +	9	Analog GND 3
10	Solenoid Valve Output 2 +	10	Analog GND 4
11	CAN High	11	Universal Input 4
12	CAN Low	12	RS-232 GND

Pins: 6,7,12 on the black connector of the AX021810 unit can be used for connection to the internal service port.

8.2 Installation

- Mounting holes are sized for ¼ inch or M6 bolts. The bolt length will be determined by the end-user's mounting plate thickness. The mounting flange of the controller is 0.63 inches (16 mm) thick.

- If the module is mounted without an enclosure, it should be mounted to reduce the likelihood of moisture entry.

- Install the unit with appropriate space available for servicing and for adequate wire harness access (6 inches or 15 cm) and strain relief (12 inches or 30 cm).

- The CAN wiring is considered intrinsically safe. The power wires are not considered intrinsically safe and so in hazardous locations, they need to be located in conduit or conduit trays at all times. The module must be mounted in an enclosure in hazardous locations for this purpose.

- All field wiring should be suitable for the operating temperature range of the module.

- It is necessary to terminate the network with external termination resistors. The resistors are 120 Ohm, 0.25W minimum, metal film or similar type. They should be placed between CAN_H and CAN_L terminals at both ends of the network.

All chassis grounding should go to a single ground point designated for the machine and all related equipment.

9 VERSION HISTORY

User Manual Version	Firmware version	Axiomatic Electronic Assistant (EA) version	Date	Author	Modifications
1.00	1.xx	3.0.20.0	Sep. 17, 2008	Olek Bogush	Initial release – draft version.
2.00	2.xx	3.0.20.1 or higher	Sep. 30, 2008	Olek Bogush	• In the Hydraulic Fan Control function block, Reset Input Signal was renamed to Stop Input Signal. This signal now safely brings the fan into its initial idle state. The Fan Stop Time setpoint was renamed to the Max Fan Stop Time setpoint. • In the PWM Output function block default value of the dither amplitude was changed to 5%.
2.00a	2.xx	3.0.20.3 or higher	Oct. 15, 2008	Olek Bogush	• In the PWM Output function block, setpoint values for: Vmax – Max Output Voltage and Vmin – Min Output Voltage were changed. • In the Conversion Function function block, a comment was added to the default value of the input source setpoint. • In the CAN output message function block, a default value of the PGN setpoint was added for the second function block. • On Fig.1, an omitted logical input was added to a CAN output message function block.
2.00a	2.xx	3.0.20.3 or higher	Oct. 15, 2008	A. Wilkins	Marketing Review
2.00b	2.xx	3.0.20.3 or higher	Jan. 15, 2009	Olek Bogush	• In Introduction, some paragraphs were changed to better describe the controller. • In Control Architecture, a phrase that logical function blocks can be changed on the customer's request was added. • In Universal Input, notes for signal range setpoints were added. • In PWM Outputs, a new paragraph, Fault State Detector, was added.
2.00c	2.xx	3.0.23.0 or higher	Jan. 27, 2009	Olek Bogush	• In Universal Input, default value of the Pull-Up/Pull-Down Resistor setpoint was changed to Disabled. • In Universal Input, a default value and restrictions for the Input Parameter setpoint were further clarified.
3	3.xx	3.0.23.1 or higher	Feb. 11, 2009	Olek Bogush	• In Controller Architecture paragraph, added explanation of the data source state. • In Universal Input function block, added

					error codes for Frequency/PWM modes. • Added description of the algorithm used by the PID Control function block. • In Multi-Input function block, added functions: {<, ≤, =, >, ≥} and a description of the block algorithm. • Added two constant logical outputs in the Global Parameters function block.
3a	3.xx	3.0.23.1 or higher	Mar. 12, 2009	Olek Bogush	• In the Introduction and the Controller Architecture some paragraphs were clarified. • In the Conversion Function, PID Control and Multi-Input Function function blocks descriptions of the algorithms were corrected.
3b	3.xx	3.0.23.1 or higher	June 24, 2009	Olek Bogush	• On Fig.1. Hydraulic Valve Control function block name was changed to Hydraulic Fan Control. • Added clarification of the message destination address in the CAN Output Message function block.
3b	3.xx	3.0.23.1 or higher	July 16, 2009	A. Wilkins	• Added dimensions and pinout
4	4.xx	3.0.27.1 or higher	Aug 7, 2009	Olek Bogush	• Changed a default value from 0 to 1 for CAN signal resolution setpoints in CAN Output Message and CAN Input Signal function blocks. • In CAN Input Signal function block Autoreset Time description has been changed.
4a	4.xx	3.0.27.1 or higher	Aug 18, 2009	Olek Bogush	• Dither Amplitude setpoint in the PWM Output function block has been clarified.
4b	4.xx	3.0.27.1 or higher	Oct 19, 2009	Olek Bogush	• In the Multi-Input Function function block the range of scale values was changed from [-1;1] to “Any value” for EA version 3.0.29.0 or later.
5	4.xx-5.xx	3.0.27.1 or higher	Oct 22, 2009	Olek Bogush	• No changes in functionality. The firmware version was stepped up from version 4.01 to 5.00 without any changes in functionality.
5a	4.xx-5.xx	3.0.27.1 or higher	Dec 15, 2009	Olek Bogush	• In the “Controller Architecture” section changed name of the Fig.1. • Added block-diagram symbols for all function blocks. • Added detailed description of the data source states other than “Valid Data”. • Added rules for conversion of the logical signals and CAN signal codes into each other. • Added Default Setpoint Settings subsection.

5b	4.xx-5.xx	3.0.27.1 or higher	Apr 30, 2010	Olek Bogush	• Default values for the temperature input setpoints were corrected in the Hydraulic Fan Controller function block. • J1939 standard document revisions were updated in the Network Support section. • Added technical specifications as Appendix A.
6	6.xx	3.0.32.0 or higher	May 18, 2010	Olek Bogush	• The User Manual was made generic for all units with AX0218XX part numbers. • Footnotes were changed. Now they contain a document name and a version number. • Figure 1, Controller Internal Structure, was modified. • Description of the AX021810 was added in different sections of this document. • Added Resistance Range setpoint to the Universal Input function block. • Added extended description of the Universal Input function block. • Added Hydraulic Fan Control 2 function block together with an application example. • Widened temperature ranges for the Hydraulic Fan Control (old). • In the Hydraulic Fan Control (old) description made a note that this function block is now obsolete.
6a	6.xx	3.0.32.0 or higher	May 25, 2010	Olek Bogush	• The Frequency and PWM input of the Universal Input function block was clarified. • The inversion function formula Inv(...) was removed from all logical inputs of all function blocks for simplicity. It was mentioned instead that the inputs can be inverted. • Added small clarification to the Conversion Function. • Added small clarification to the PID Control. • A caption was added to the diagram describing the temperature control part of the Hydraulic Fan Control 2 function block.
7	7.xx	3.0.32.1 or higher	June 8, 2010	Olek Bogush	• Added 14 extra Multi-Input Function function blocks. • Added 16 new Binary Function function blocks. • Added 5 new Hysteresis Function function blocks. • Added 5 new Ramp Function function blocks. • Added 5 new Switch Function function blocks. • Added 3 additional CAN Input Signal function blocks.

					• Added 2 settable logical constant output sources, and two predefined logical constant output sources: Constant Output =0.0, Constant Output =1.0 to the Global Parameters function block. • In the PID Control function block, the anti-windup algorithm was changed from simply clamping the integral part to stopping the integration process when the output of the function block saturates and the control error contributes to its further saturation. • Clarified the Disable Input in the PWM Output function block. • Changed Figure 1, Controller Internal Structure, to reflect the new function blocks added to the controller. • Added the number of the function blocks available in the controller inside the graphical presentation of the function blocks. • Inversion function was clarified in the Controller Architecture section. • Corrected the default value of the Input Source setpoint in the Conversion Function function block.
7a	7.xx	3.0.32.1 or higher	June 15, 2010	Olek Bogush	• Corrected Inverted Signal Value in a table describing signal inversion. • Changed some function block drawings. • Added EA aliases for some setpoint names in Ramp and Switch Function function blocks. • Clarified descriptions of the Multi-Input and Binary Functions. • Corrected Formula for the PID Control. • Added hyperlinks to function block names.
7b	7.xx	3.0.32.1 or higher	Aug. 3, 2010	Olek Bogush	• A typo in number of conversion functions in figure 1 was fixed.
7c	7.xx	3.0.32.1 or higher	Oct. 14, 2010	Olek Bogush	• Explained discrete logical inputs in the Controller Architecture section. • Clarified the Reset Input of the PID Control function block.
8	8.xx	EA 4.4.42.0 or higher	Dec. 6, 2012	Olek Bogush	• Added support for controllers with different baud rate. Changed the front page of the manual, Technical Specification section, and applied appropriate changes throughout the manual. • Added support for the J1939 bootloader. Added Flashing New Firmware section. • Setpoint Programming section was rewritten. It now includes: Axiomatic Electronic Assistant Software, Function

					Blocks in EA, and Setpoint File subsections. • Introduction section was rewritten. • Controller Description section was added between Introduction and Controller Architecture sections. This section contains Hardware Block Diagram and Software Organization subsections. • In the Network Support section changed description of how the Identity Number is calculated in the V8.xx of the firmware. Modified the slew rate control description to address multiple baud rates. • Changed “functional blocks” to “function blocks” throughout the text. • Moved: Technical Specifications, Installation Instructions and Version History sections to the end of the User Manual.
V9	9.xx		Dec. 31, 2023	M Ejaz, Sue Thomas	Marketing review, legacy updates, new address

OUR PRODUCTS

AC/DC Power Supplies

Actuator Controls/Interfaces

Automotive Ethernet Interfaces

Battery Chargers

CAN Controls, Routers, Repeaters

CAN/WiFi, CAN/Bluetooth, Routers

Current/Voltage/PWM Converters

DC/DC Power Converters

Engine Temperature Scanners

Ethernet/CAN Converters, Gateways, Switches

Fan Drive Controllers

Gateways, CAN/Modbus, RS-232

Gyroscopes, Inclinometers

Hydraulic Valve Controllers

Inclinometers, Triaxial

I/O Controls

LVDT Signal Converters

Machine Controls

Modbus, RS-422, RS-485 Controls

Motor Controls, Inverters

Power Supplies, DC/DC, AC/DC

PWM Signal Converters/Isolators

Resolver Signal Conditioners

Service Tools

Signal Conditioners, Converters

Strain Gauge CAN Controls

Surge Suppressors

OUR COMPANY

Axiomatic provides electronic machine control components to the off-highway, commercial vehicle, electric vehicle, power generator set, material handling, renewable energy and industrial OEM markets. ***We innovate with engineered and off-the-shelf machine controls that add value for our customers.***

QUALITY DESIGN AND MANUFACTURING

We have an ISO9001:2015 registered design/manufacturing facility in Canada.

WARRANTY, APPLICATION APPROVALS/LIMITATIONS

Axiomatic Technologies Corporation reserves the right to make corrections, modifications, enhancements, improvements, and other changes to its products and services at any time and to discontinue any product or service without notice. Customers should obtain the latest relevant information before placing orders and should verify that such information is current and complete. Users should satisfy themselves that the product is suitable for use in the intended application. All our products carry a limited warranty against defects in material and workmanship. Please refer to our Warranty, Application Approvals/Limitations and Return Materials Process at https://www.axiomatic.com/service/.

COMPLIANCE

Product compliance details can be found in the product literature and/or on axiomatic.com. Any inquiries should be sent to sales@axiomatic.com.

SAFE USE

All products should be serviced by Axiomatic. Do not open the product and perform the service yourself.

This product can expose you to chemicals which are known in the State of California, USA to cause cancer and reproductive harm. For more information go to www.P65Warnings.ca.gov.

SERVICE

All products to be returned to Axiomatic require a Return Materials Authorization Number (RMA#) from sales@axiomatic.com. Please provide the following information when requesting an RMA number:

- Serial number, part number
- Runtime hours, description of problem
- Wiring set up diagram, application and other comments as needed

DISPOSAL

Axiomatic products are electronic waste. Please follow your local environmental waste and recycling laws, regulations and policies for safe disposal or recycling of electronic waste.

CONTACTS

Axiomatic Technologies Corporation
1445 Courtneypark Drive E.
Mississauga, ON
CANADA L5T 2E3
TEL: +1 905 602 9270
FAX: +1 905 602 9279
www.axiomatic.com
sales@axiomatic.com

Axiomatic Technologies Oy
Höytämöntie 6
33880 Lempäälä
FINLAND
TEL: +358 103 375 750
www.axiomatic.com
salesfinland@axiomatic.com