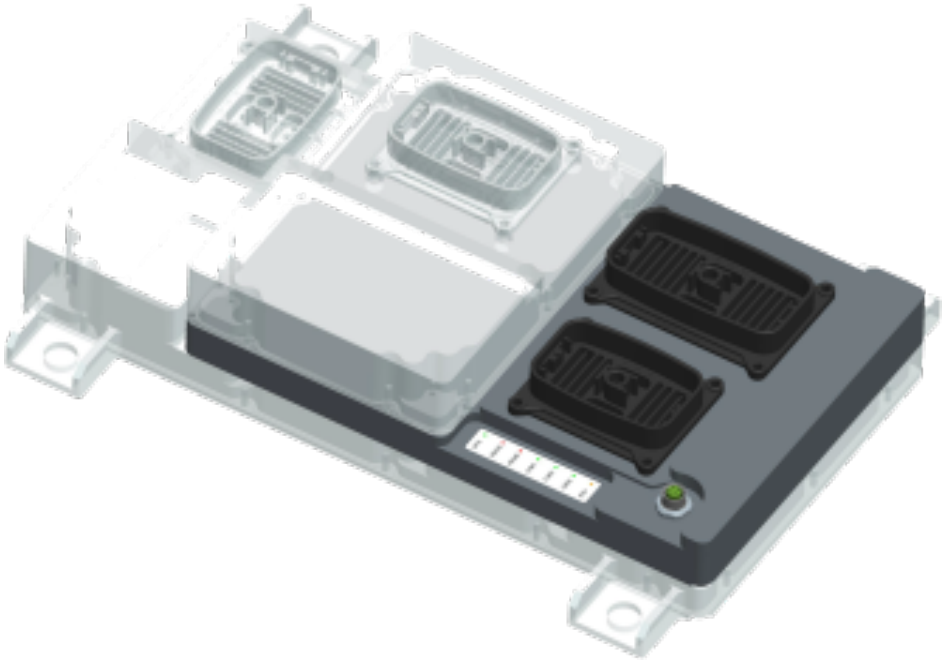
WOODWARD

Product Manual 35203V2
(Revision -, 6/2023)
Original Instructions



Large Engine System Platform

Software Features

Installation and Developer Manual

General Precautions

Read this entire manual and all other publications pertaining to the work to be performed before installing, operating, or servicing this equipment.

Practice all plant and safety instructions and precautions.

Failure to follow instructions can cause personal injury and/or property damage.

Revisions

This publication may have been revised or updated since this copy was produced. The latest version of most publications is available on the Woodward website.

http://www.woodward.com

If your publication is not there, please contact your customer service representative to get the latest copy.

Proper Use

Any unauthorized modifications to or use of this equipment outside its specified mechanical, electrical, or other operating limits may cause personal injury and/or property damage, including damage to the equipment. Any such unauthorized modifications: (i) constitute "misuse" and/or "negligence" within the meaning of the product warranty thereby excluding warranty coverage for any resulting damage, and (ii) invalidate product certifications or listings.

Translated Publications

If the cover of this publication states "Translation of the Original Instructions" please note:

The original source of this publication may have been updated since this translation was made. The latest version of most publications is available on the Woodward website.

www.woodward.com/publications

Always compare with the original for technical specifications and for proper and safe installation and operation procedures.

If your publication is not on the Woodward website, please contact your customer service representative to get the latest copy.

Revisions— A bold, black line alongside the text identifies changes in this publication since the last revision.

Manual 35203V2

Contents

WARNINGS AND NOTICES 9

CHAPTER 1. GENERAL INFORMATION 12
Introduction 12
Licensing 13
Block Dependencies 16
The LESP Packages Documentation 17
Important Notes 17

CHAPTER 2. CONTROL - CRS RAIL PRESSURE CONTROL 18
Introduction 18
CRS Rail Pressure Control Block Interface 19
CRS Rail Pressure Diagnostic Monitors Block Interface 21
CRS Rail Pressure Input Processing 23
Example Model Using the Blocks 25
Default ToolKit Pages 25
2.1 Functional Description - CRS Rail Pressure Control Logic 30
Rail Pressure Sensing 30
Introduction 30
Conversion 31
Filtering 32
Signal Diagnostics 32
Defaulting 32
Sensor Arbitration 33
Functional Description - Rail Pressure Control 34
Setpoint Strategy 34
Feedforward Strategy 35
Feedback Strategy 40
Functional Description - Rail Pressure Diagnostics 41
Range Diagnostics & Monitors 41
Zero Rail Pressure Monitor 42
Rail Pressure Control Monitor 42
Rail Pressure Sensor Deviation Monitor 42

CHAPTER 3. CONTROL - WLO FUEL METERING VALVE 44
Introduction 44
Dual Pump Flow Distributor Block Interface 45
WLO Fuel Metering Valve #1 Block Interface 49
WLO Fuel Metering Valve #2 Block Interface 52
WLO Fuel Metering Valve Circuit Diagnostics Block Interface 55
Example Model Using the Blocks 56
Default ToolKit Pages 58
Functional Description - Fuel Control Valve 60
Pump Flow Distributor 60
Fuel Control Valve 62

CHAPTER 4. CONTROL - AUX ACOUSTIC KNOCK MITIGATION 66
Introduction 66
Example Model Using the Blocks 71
Default ToolKit Pages 71
Functional Description - Knock Detection and Mitigation Logic 74
Micropilot Timing Adjustment 74
Derate Limiting 75
Configuration 75

CHAPTER 5. CONTROL – SPEED CONTROL 80
Introduction 80

Speed Control Block Interface 81
Dual Fuel Speed Control Block Interface 89
Example Model Using the Blocks 98
Default ToolKit Pages 99
Functional Description - Speed Control 104
Engine Speed Control Overview 104
Analog Speed Setting Mode 105
Digital Speed Setting Mode 108
Speed Control Logic 110
Functional Description - Fuel Limiters 119
J1939 / Modbus Speed Control Options 127
External Speed Control Option 133
Speed Control Setup 138

CHAPTER 6. CONTROL – BALANCING 153
Introduction 153
EGT Cylinder Balancing Block Interface 154
Example Model Using the Blocks 155
Default ToolKit Pages 156
Functional Description – EGT Cylinder Balancing 158

CHAPTER 7. CONTROL – AIR MANAGEMENT 163
Introduction 163
Throttle Control Block Interface 164
Example Model Using the Blocks 165
Default ToolKit Pages 166
Functional Description – Throttle Control 169

CHAPTER 8. FUNCTIONS – ENGINE STATE 177
Introduction 177
Engine State Default Strategy Block Interface 178
Engine State Block Interface 181
Example Model Using the Blocks 182
Default ToolKit Pages 183
Functional Description - Engine State 185
States 185
Default Strategies 186

CHAPTER 9. FUNCTIONS – ENGINE START 189
Introduction 189
Engine Start Control Block Interface 190
Example Model Using the Blocks 193
Default ToolKit Pages 194
Functional Description - Engine Start 198
Engine Start Controller 198

CHAPTER 10. SENSORS – INPUT PROCESSING – CRS DUAL FUEL 202
Introduction 202
CRS Dual Fuel OffEngine Sensing Block Interface 203
CRS Dual Fuel Pressure Sensing (Medium) Block Interface 209
CRS Dual Fuel Pressure Sensing (Slow) Block Interface 215
CRS Dual Fuel Speed Sensing (Medium) Block Interface 220
CRS Dual Fuel Switch Sensing (Medium) Block Interface 222
CRS Dual Fuel Temperature Sensing (Slow) Block Interface 232
Example Model Using the Blocks 251
Default ToolKit Pages 252

CHAPTER 11. SENSORS – INPUT PROCESSING - CRS 255
Introduction 255
CRS OffEngine Sensing Block Interface 256
CRS Pressure Sensing (Medium) Block Interface 261
CRS Pressure Sensing (Slow) Block Interface 266

CRS Speed Sensing (Medium) Block Interface 271
CRS Switch Sensing (Medium) Block Interface 273
CRS Temperature Sensing (Slow) Block Interface 282
Example Model Using the Blocks 301
Default ToolKit Pages 301

CHAPTER 12. SENSORS – UEGO 305
Introduction 305
UEGO1 Block Interface 306
Example Model Using the Blocks 310
Default ToolKit Pages 311
Functional Description – UEGO Sensor 317
Introduction 317

CHAPTER 13. PRODUCT SUPPORT AND SERVICE OPTIONS 320
Product Support Options 320
Product Service Options 320
Returning Equipment for Repair 321
Replacement Parts 321
Engineering Services 322
Contacting Woodward's Support Organization 322
Technical Assistance 323

TECHNICAL SPECIFICATIONS 324

REVISION HISTORY 325

DECLARATIONS 326

Illustrations and Tables

Figure 1-1. LESP Model Reference Blocks 12
Figure 1-2. LESP Model Reference Distribution & License Archives 13
Figure 1-3. Launch LESP License Manager 14
Figure 1-4. LESP License Manager 14
Figure 1-5. LESP License Manager (Select HDD) 14
Figure 1-6. LESP License Manager (Site Code & License Key) 15
Figure 1-7. Verify LESP License 15
Figure 1-8. Verify LESP License (Days Remaining) 16
Figure 1-9. Block Dependencies 16
Figure 1-10. 'C' Language Standard Setting (Simulink) 17
Figure 1-11. Note 'Sample Time Settings' to Match Trigger Rate of Block 17
Figure 2-1. LESP Model Reference CRS Rail Pressure Blocks 18
Figure 2-2. 'CRS Rail Pressure Control' Block 19
Figure 2-3. 'CRS Rail Pressure Diagnostic Monitors' Block 21
Figure 2-4. 'CRS Rail Pressure Input Processing' Block 23
Figure 2-5. Simple Example Model Using the CRS RailPressure Control Blocks 25
Figure 2-6. SID File Items Included in the Model Reference Blocks 26
Figure 2-7. Overview Page Rail Pressure Control 26
Figure 2-8. Setpoint Page Rail Pressure Control 27
Figure 2-9. FeedForward Page Rail Pressure Control 27
Figure 2-10. FeedForward Adapt Page Rail Pressure Control 28
Figure 2-11. FeedBack Controller Page Rail Pressure Control 28
Figure 2-12. FeedBack Controller Dynamics Page Rail Pressure Control 29
Figure 2-13. Example of the Toolkit Pages for Rail Pressure Diagnostics 29
Figure 2-14. Rail Pressure Sensing & Control Overview 30
Figure 2-15. Rail Pressure Sensing Strategy 31
Figure 2-16. Rail Pressure Sensor Configuration 31
Figure 2-17. Rail Pressure Filter Configurations 32
Figure 2-18. Sensor Arbitration Overview 33
Figure 2-19. Sensor Arbitration Configurations 33
Figure 2-20. Rail Pressure Control Strategy Overview 34
Figure 2-21. Setpoint Strategy 34
Figure 2-22. Setpoint Configurations 35
Figure 2-23. Feedforward Strategy 35
Figure 2-24. Feedforward Speed Compensation 36
Figure 2-25. Feedforward Table Configurations 36
Figure 2-26. Feedforward Adaption Regions 37
Figure 2-27. Feedforward Nominal Adaption 37
Figure 2-28. Feedforward Nominal Adaption Configuration 37
Figure 2-29. Nominal Feedforward Adaption Quantity 38
Figure 2-30. Feedforward Adaption Enable Condition Configurations 38
Figure 2-31. Speed Compensation Adaption 39
Figure 2-32. Speed Compensation Adaption Configuration 39
Figure 2-33. Feedback Strategy 40
Figure 2-34. Feedback Configurations 40
Figure 2-35. Rail Pressure Range Diagnostic & Monitor Configurations 41
Figure 2-36. Zero Rail Pressure Monitor Configurations 42
Figure 2-37. Rail Pressure Control Monitor Configurations 42
Figure 2-38. Rail Pressure Sensor Deviation Monitor Configurations 43
Figure 3-1. LESP Model Reference 'WLO Fuel Metering Valve' Blocks 44
Figure 3-2. 'Dual Pump Flow Distributor' Block 45
Figure 3-3. 'WLO Fuel Metering Valve #1' Block 49
Figure 3-4. 'WLO Fuel Metering Valve #2' Block 52
Figure 3-5. 'WLO Fuel Metering Valve Circuit Diagnostics' Block 55
Figure 3-6. Simple Example Model, Using the Control- WLO Fuel Metering Valves Blocks 56
Figure 3-7. Individual FMV's with 'Flow Distributor Blocks' Commented Out 57

Figure 3-8. 'Flow Distributor Block' Active with Individual 'FMV Blocks' Commented Out 57
Figure 3-9. SID File Selection and Layout for Both Options 58
Figure 3-10. Examples of the Toolkit Pages for FMV Control 59
Figure 3-11. Dual Fuel Pumps 60
Figure 3-12. Fuel Control Valve 60
Figure 3-13. Time-Based Pump Flow Distribution Strategy 61
Figure 3-14. Fuel Pump Distribution Configurations 61
Figure 3-15. Fuel Control Valve Strategy 62
Figure 3-16. Fuel Control Valve Configuration 62
Figure 3-17. Fuel Control Valve Configuration 63
Figure 3-18. Fuel Control Valve Configuration 63
Figure 3-19. Fuel Control Valve Setpoint Configurations 63
Figure 3-20. Example for Fuel Control Valve Feedforward Configuration 64
Figure 3-21. Feedback Strategy 64
Figure 3-22. Feedback Configurations 65
Figure 4-1. LESP Model Reference 'AUX Acoustic Knock Mitigation' Block 66
Figure 4-2. Simple Example Model, Using the Control - Knock Mitigation Block 71
Figure 4-3. SID File Selection Knock Logic 71
Figure 4-4. Examples of Toolkit Pages 73
Figure 4-5. Example of the Micropilot Mitigation Logic 74
Figure 4-6. Toolkit Page – Control: Knock Mitigation 75
Figure 4-7. Knock Mitigation Enable Management 75
Figure 4-8. Knock Mitigation Enable Conditions 76
Figure 4-9. Knock Thresholds Setup 76
Figure 4-10. Knock Mitigation Filtering Setup 77
Figure 4-11. Knock Mitigation Settings 77
Figure 4-12. Knock Mitigation Timing Retard Setup 78
Figure 4-13. Knock Diagnostics Setup 79
Figure 5-1. LESP Model Reference 'Control – Speed Control' Blockset 80
Figure 5-2. 'Speed Control' Block 81
Figure 5-3. 'Dual Fuel Speed Control' Block 89
Figure 5-4. 'Speed Control Setpoints' Block 97
Figure 5-5. Simple Example Model, Using the Control – Speed Control Blocks 98
Figure 5-6. SID File Selection Control - Speed Logic 99
Figure 5-7. Example Toolkit Pages 103
Figure 5-8. Speed Control Overview 105
Figure 5-9. Deadband Overview 105
Figure 5-10. Analog Speed Setting Calibration Values 106
Figure 5-11. Strategy Selection 107
Figure 5-12. Engineering Unit Setup 107
Figure 5-13. Diagnostic Setup 108
Figure 5-14. Digital Speed Setting Calibration Parameters 109
Figure 5-15. Different Rated Speed Setpoint Selection Options 110
Figure 5-16. Speed Source Select Options 111
Figure 5-17. Speed Control Main Page 113
Figure 5-18. Speed Control Configuration 113
Figure 5-19. Main Speed Control Items to Setup 114
Figure 5-20. Select View – Speed Control Page 114
Figure 5-21. Overview of Possible Pages 115
Figure 5-22. Speed Controller Pages 115
Figure 5-23. Example of Gain Scheduling Options 116
Figure 5-24. Load Dump Dependent Parameters 117
Figure 5-25. Load Jump Dependent Parameters 117
Figure 5-26. Acceleration / Deceleration Setup 118
Figure 5-27. Overview of Fuel Limiters 119
Figure 5-28. Starting Phase Fuel Demand 120
Figure 5-29. Fuel Limiters Setup – Start Fuel Limit 120
Figure 5-30. Speed Setpoint Dependent Switch Off 121
Figure 5-31. Torque Fuel Limiter 121

Figure 5-32. Manifold Air Pressure Fuel Limiter 122
Figure 5-33. Maximum Fuel Limiter 123
Figure 5-34. Power Limit Parameter Setup 124
Figure 5-35. Power Limit Dynamics Tuning Option 124
Figure 5-36. Jump Rate Limiter Settings 125
Figure 5-37. Near Fuel limit Notification 126
Figure 5-38. ModBus Configuration 127
Figure 5-39. ModBus Configuration 128
Figure 5-40. Speed Control Messages 129
Figure 5-41. Speed Control Messages Configuration 130
Figure 5-42. Enable/Disable Messages 130
Figure 5-43. Speed Control Messages Diagnostics Setup 130
Figure 5-44. TSC1 Details 131
Figure 5-45. GC1 Details 131
Figure 5-46. GC2 Details 131
Figure 5-47. ACS Details 131
Figure 5-48. GTACP Details 132
Figure 5-49. Engine Configuration Mode 133
Figure 5-50. Engine Control Mode Selection 134
Figure 5-51. Fuel Demand Selection 134
Figure 5-52. Fuel Demand Overview 135
Figure 5-53. Fuel Demand Configuration 135
Figure 5-54. Analog Input Type Selection 136
Figure 5-55. Example Setup 0-5 V to 0-850 mg/cyl Fuel Demand 136
Figure 5-56. Toolkit Configuration Page 138
Figure 5-57. Setup Guide Start Page 138
Figure 5-58. Setup Guide Navigation Buttons 139
Figure 5-59. [Speed Control] Control Mode Offline Editor Page 139
Figure 5-60. Demanded Load Setup 140
Figure 5-61. [Speed Control] Fuel Demand Page 140
Figure 5-62. [Speed Control] Speed Reference Page 141
Figure 5-63. Setup Datalink Select 142
Figure 5-64. [Speed Control] Datalink Select 142
Figure 5-65. Setup Digital Select 143
Figure 5-66. Digital Select Setup Page 143
Figure 5-67. Setup Analog Select 144
Figure 5-68. Analog Select Setup Page 145
Figure 5-69. Speed Reference Adjustments 145
Figure 5-70. Speed Reference Adjustments 146
Figure 5-71. Speed Reference Setpoints and Ramp Rates 147
Figure 5-72. [Speed Control] Speed Controller Page 147
Figure 5-73. Speed Source Selection 148
Figure 5-74. Speed Control Dynamics Select 148
Figure 5-75. Speed Controller Limits 148
Figure 5-76. Speed Controller I-Term Settings 149
Figure 5-77. [Speed Control] Power Limit Controller Page 149
Figure 5-78. [Speed Control] Fuel Limiting Page 150
Figure 5-79. Start Fuel Limiter Disable Options 151
Figure 5-80. Enable/Disable of Power Limiter 151
Figure 5-81. [Speed Control] Complete Page 152
Figure 6-1. LESP Model Reference Control - Balancing Block 153
Figure 6-2. LESP Model Reference EGT Cylinder Balancing Block 154
Figure 6-3. Simple Example Model Using the Control – Balancing Block 155
Figure 6-4. SID File Selection Control - Balancing Logic 156
Figure 6-5. Examples of the Toolkit Pages 157
Figure 6-6. Setup of the Exhaust Gas Temperature Sensors 158
Figure 6-7. Exhaust Gas Temperature Configurations 159
Figure 6-8. Exhaust Gas Temperature Diagnostics (Example Range High) 159
Figure 7-1. LESP Model Reference ‘Control – Throttle Control’ Block 163

Figure 7-2. LESP Model Reference 'Throttle Control' Block 164
Figure 7-3. Simple Example Model, Using the Control - Air Management Block 165
Figure 7-4. SID File Selection Control – Air Management Logic 166
Figure 7-5. Examples of the Toolkit Pages 168
Figure 7-6. Throttle Valve – F-Series ITB 169
Figure 7-7. Toolkit Throttle/Bypass/Wastegate Valve Configuration Page 169
Figure 7-8. Throttle/Bypass/Wastegate Valve Configuration Setup 170
Figure 7-9. Bypass Inlet Valve Feedback Setup 170
Figure 7-10. Throttle/Bypass/Wastegate Inlet Valve Diagnostics Setup 170
Figure 7-11. Bypass Inlet Valve Position Setup 171
Figure 7-12. Throttle/Bypass/Wastegate Valve Diagnostics Monitor 171
Figure 7-13. F-Series Actuator Tool 172
Figure 7-14. Throttle/Bypass/Wastegate Valve - R-Series Actuator 172
Figure 7-15. Toolkit Throttle/Bypass/Wastegate Valve Configuration Page 173
Figure 7-16. Throttle/Bypass/Wastegate Valve Configuration Setup 173
Figure 7-17. Throttle Valve Feedback Setup 173
Figure 7-18. Throttle Valve Datalink 174
Figure 7-19. Throttle Valve J1939 Command/Feedback and Additional Info 174
Figure 7-20. Throttle/Bypass/Wastegate Valve Diagnostics Setup 175
Figure 7-21. Throttle/Bypass/Wastegate Valve Feedforward Table Setup 175
Figure 7-22. Throttle/Bypass/Wastegate Valve Feedback PID Setup 175
Figure 7-23. Throttle Valve Desired AFR Setup 176
Figure 7-24. Throttle/Bypass/Wastegate Valve Service Tool 176
Figure 8-1. LESP Model Reference 'Functions – Engine State' Blocks 177
Figure 8-2. LESP Model Reference 'Functions – Engine State' Block 178
Figure 8-3. LESP Model Reference 'Functions – Engine State' Block 181
Figure 8-4. Simple Example Model, Using the Functions – Engine State Blocks 182
Figure 8-5. SID File Selection Functions – Engine States Logic 183
Figure 8-6. Examples of the Functions - Engine State Toolkit Pages 184
Figure 8-7. Engine State Overview 185
Figure 8-8. Engine Starting State Exit Conditions 186
Figure 8-9. Prestart Configurations 186
Figure 8-10. Warmup Configurations 187
Figure 8-11. Cooldown Configurations 188
Figure 8-12. Postrun Configurations 188
Figure 9-1. LESP Model Reference 'Functions – Engine Start' Blocks 189
Figure 9-2. LESP Model Reference 'Functions – Engine Start Control' Block 190
Figure 9-3. LESP Model Reference 'Functions – Engine Start Request' Block 192
Figure 9-4. Simple Example Model, Using the Functions – Engine Start Blocks 193
Figure 9-5. SID File Selection Functions – Engine Start Logic 194
Figure 9-6. Example Functions - Engine Start Toolkit Pages 197
Figure 9-7. Engine Start Request Command Diagram 198
Figure 9-8. Engine Start Controller 200
Figure 9-9. Engine Start Service Tool Page 200
Figure 9-10. Engine Start Configurations 201
Figure 10-1. LESP Model Reference 'Sensors – CRS Dual Fuel Input Processing' Block 202
Figure 10-2. LESP Model Reference 'Sensors – CRS Dual Fuel OffEngine Sensing' Block 203
Figure 10-3. LESP Model Reference 'Sensors – CRS Dual Fuel Pressure Sensing (Medium)' Block 209
Figure 10-4. LESP Model Reference 'Sensors – CRS Dual Fuel Pressure Sensing (Slow)' Block 215
Figure 10-5. LESP Model Reference 'Sensors – CRS Dual Fuel Speed Sensing (Medium)' Block 220
Figure 10-6. LESP Model Reference 'Sensors – CRS Dual Fuel Switch Sensing (Medium)' Block 222
Figure 10-7. LESP Model Reference 'Sensors – CRS Dual Fuel Temperature Sensing (Slow)' Block 232
Figure 10-8. Simple Example Model, Using the Sensors - CRS Interface Blocks 251
Figure 10-9. SID File Selection Sensors – CRS Dual Fuel Interface Logic 252
Figure 10-10. Examples of the Sensors-Input CRS Dual Fuel Toolkit Pages 254
Figure 11-1. LESP Model Reference 'Sensors – CRS Input Processing' Block 255
Figure 11-2. LESP Model Reference 'Sensors – CRS OffEngine Sensing' Block 256
Figure 11-3. LESP Model Reference 'Sensors – CRS Pressure Sensing (Medium)' Block 261
Figure 11-4. LESP Model Reference 'Sensors – CRS Pressure Sensing (Slow)' Block 266

Figure 11-5. LESP Model Reference ‘Sensors – CRS Speed Sensing (Medium)’ Block 271
Figure 11-6. LESP Model Reference ‘Sensors – CRS Switch Sensing (Medium)’ Block 273
Figure 11-7. LESP Model Reference ‘Sensors – CRS Temperature Sensing (Slow)’ Block 282
Figure 11-8. Simple Example Model, Using the Sensors - CRS Interface Block 301
Figure 11-9. SID File Selection Sensors – CRS Interface Logic 301
Figure 11-10. Examples of the Sensors-Input CRS Dual Fuel Toolkit Pages 304
Figure 12-1. LESP Model Reference ‘Sensors – UEGO’ Block 305
Figure 12-2. LESP Model Reference ‘Sensors – UEGO1’ Block 306
Figure 12-3. Simple Example Model, Using the Sensors – UEGO1 Block 310
Figure 12-4. SID File Selection Sensors – UEGO 311
Figure 12-5. Examples of the Sensors-UEGO Toolkit Pages 316
Figure 12-6. UEGO Sensor Toolkit Setup Page 317
Figure 12-7. UEGO Sensor Activation / Deactivation 317
Figure 12-8. UEGO Sensor - Enabled 318

Table 2-1. ‘CRS Rail Pressure Control’ Block 19
Table 2-2. ‘CRS Rail Pressure Diagnostic Monitors’ Block 21
Table 2-3. ‘CRS Rail Pressure Input Processing’ Block 23
Table 2-4. Rail Pressure Sensor IO Mappings 30
Table 3-1. ‘CRS Rail Pressure Diagnostic Monitors’ Block 45
Table 3-2. ‘WLO Fuel Metering Valve #1’ Block 49
Table 3-3. ‘WLO Fuel Metering Valve #2’ Block 52
Table 3-4. ‘WLO Fuel Metering Valve #2’ Block 55
Table 4-1. ‘Aux Acoustic Knock Mitigation’ Block 66
Table 5-1. ‘Speed Control’ Block 82
Table 5-2. ‘Dual Fuel Speed Control’ Block 90
Table 5-3. ‘Speed Control Setpoints’ Block 97
Table 6-1. EGT Cylinder Balancing Block 154
Table 7-1. ‘Throttle Control’ Block 164
Table 8-1. ‘Speed Control Setpoints’ Block 178
Table 8-2. ‘Speed Control Setpoints’ Block 181
Table 9-1. ‘Speed Control Setpoints’ Block 190
Table 9-2. ‘Speed Control Setpoints’ Block 192
Table 10-1. ‘CRS Dual Fuel OffEngine Sensing (Medium)’ Block 203
Table 10-2. ‘CRS Dual Fuel Pressure Sensing (Medium)’ Block 209
Table 10-3. ‘CRS Dual Fuel Pressure Sensing (Slow)’ Block 215
Table 10-4. ‘Dual Fuel Speed Sensing (Medium)’ Block 220
Table 10-5. ‘Dual Fuel Switch Sensing (Medium)’ Block 222
Table 10-6. ‘CRS Dual Fuel Temperature Sensing (Slow)’ Block 232
Table 11-1. ‘CRS OffEngine Sensing (Medium)’ Block 256
Table 11-2. ‘CRS Pressure Sensing (Medium)’ Block 261
Table 11-3. ‘CRS Pressure Sensing (Slow)’ Block 266
Table 11-4. ‘CRS Speed Sensing (Medium)’ Block 271
Table 11-5. ‘CRS Switch Sensing (Medium)’ Block 273
Table 11-6. ‘CRS Temperature Sensing (Slow)’ Block 282
Table 12-1. ‘UEGO1’ Block 306

Warnings and Notices

Important Definitions

This is the safety alert symbol used to alert you to potential personal injury hazards. Obey all safety messages that follow this symbol to avoid possible injury or death.

- **DANGER** - Indicates a hazardous situation, which if not avoided, will result in death or serious injury.
- **WARNING** - Indicates a hazardous situation, which if not avoided, could result in death or serious injury.
- **CAUTION** - Indicates a hazardous situation, which if not avoided, could result in minor or moderate injury.
- **NOTICE** - Indicates a hazard that could result in property damage only (including damage to the control).
- **IMPORTANT** - Designates an operating tip or maintenance suggestion.

⚠WARNING

Lockout/Tagout LOTO

Ensure that personnel are fully trained on LOTO procedures prior to attempting to replace or service equipment on a "live" running engine. All safety protective systems (overspeed, over temperature, overpressure, etc.) must be in proper operational condition prior to the start or operation of a running engine. Personnel should be equipped with appropriate personal protective equipment to minimize the potential for injury due to release of hot hydraulic fluids, exposure to hot surfaces and/or moving parts, or any moving parts that may be activated and are located in the area of control of the unit.

⚠WARNING

Overspeed / Overtemperature / Overpressure

The engine, turbine, or other type of prime mover should be equipped with an overspeed shutdown device to protect against runaway or damage to the prime mover with possible personal injury, loss of life, or property damage.

The overspeed shutdown device must be totally independent of the prime mover control system. An overtemperature or overpressure shutdown device may also be needed for safety, as appropriate.

⚠WARNING

Personal Protective Equipment

The products described in this publication may present risks that could lead to personal injury, loss of life, or property damage. Always wear the appropriate personal protective equipment (PPE) for the job at hand. Equipment that should be considered includes but is not limited to:

- **Eye Protection**
- **Hearing Protection**
- **Hard Hat**
- **Gloves**
- **Safety Boots**
- **Respirator**

Always read the proper Material Safety Data Sheet (MSDS) for any working fluid(s) and comply with recommended safety equipment.

⚠WARNING

Start-up

Be prepared to make an emergency shutdown when starting the engine, turbine, or other type of prime mover, to protect against runaway or overspeed with possible personal injury, loss of life, or property damage.

⚠WARNING

Automotive Applications

On- and off-highway Mobile Applications: Unless Woodward's control functions as the supervisory control, customer should install a system totally independent of the prime mover control system that monitors for supervisory control of engine (and takes appropriate action if supervisory control is lost) to protect against loss of engine control with possible personal injury, loss of life, or property damage.

NOTICE

Battery Charging Device

To prevent damage to a control system that uses an alternator or battery-charging device, make sure the charging device is turned off before disconnecting the battery from the system.

Regulatory Compliance

The LESP is partially completed Motohawk blocks meant to assist end users in designing their applications. As such, the LESP is a partially complete subsystem of individual parts. The device/system it is used in is only complete when the Motohawk application is integrated with reciprocating engines on or in completed “Engine Packages”, “Engine Skids”, “Construction, Earth Moving, Agricultural, Forestry, & Offroad Vehicles & Equipment”, “Semi-Mobile Industrial” (small installations that are fixed during operation), and stationary industrial “Fixed Installations”.

The LESP is a system of individually qualified devices; most do not have a function unless integrated into the fully operating package or vehicle.

Regulatory Compliance for each country that the finished device will ship into or be retrofit is the responsibility of the OEM or Aftermarket Retrofit Contractor. This includes the three essential items of marking the finished product, creating regulatory documentation, and any evaluation of the finished device/product.

The LESP has no regulatory marking requirements since it is a part of the system and performs no function on its own. **The OEM or Aftermarket Retrofit Contractor are responsible for any marking required by the governing body for the location or locations of operation.**

For the various parts mentioned in the following manual, please refer to the appropriate hardware manuals for regulatory compliance.

Manual #	**Description**
03425	**LECM Product Spec**
26757	**Large Engine Control Module**
35014	**LECM Platform EID MixedMode Driver**
35079	**LECM Aux RTCDC**
35079	**RT-CDC**
35100	**LECM Platform EID Ignition Driver**
35112	**LECM Aux Thermocouple**
35138	**LECM Aux Platform Knock**
35175	**LECM Platform EID Injection Driver**
35188	**LECM Aux Protection IO**
35200	**MotoHawk Developer Guide**
35203V1	**LESP Packages Developer Manual**
35203V2	**LESP Software Features Developer Manual**

Chapter 1.
General Information

Introduction

The LESP Model Reference software features pre-compiled Simulink blocks with specific functionality, which can be used in creating an application. The blocks contain logical functions that have been tested and verified on various engines to help in speeding up development work for custom applications. The model reference blocks can be found in the Simulink library browser once the license has been purchased and the software successfully installed.

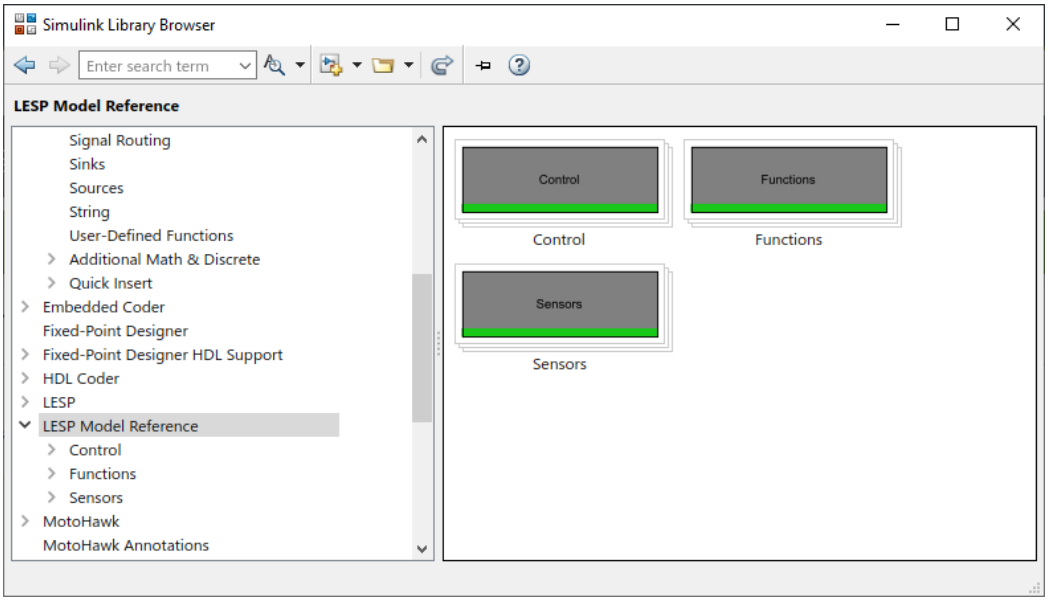


Figure 1-1. LESP Model Reference Blocks

The model reference blocks are divided into three different categories: Control, Functions, and Sensors.

Each of these blocks are pre-compiled for a particular version of Matlab, MotoHawk, MotoHawk Target, and Compiler that is also used in the application. Matlab, MotoHawk, ECU Target, and the compilers should match to ensure correct functioning inside the application.

The model reference blocks can be recognized by the green bottom colors. The blocks are sealed, preventing you from looking deeper inside the block to verify how the logic has been built. To understand the exact functionality of using the blocks, a detailed description will be provided in the following chapters of this manual. The blocks also contain several MotoHawk probes and MotoHawk calibration blocks available from the .sid file, which can be made visible using ToolKit or any other 3rd party interface tool (.a2l file based). An example model that uses the model reference blocks is available, including an example HMI tool (Toolkit-based).

Licensing

LESP model reference blocks are node-locked licensed on a PC. The blocks can be integrated into models and allow for model updates to occur on a PC with or without a license; however, to compile the application with the blocks, a licensed PC must be used. LESP model reference blocks are distributed in archives (*.zip) in conjunction with the licensing. The licensing and model reference blocks can be extracted to the local project directory and added to the Matlab path, similar to other project dependencies. Below is an example of LESP feature archives and corresponding licensing highlighted in yellow.

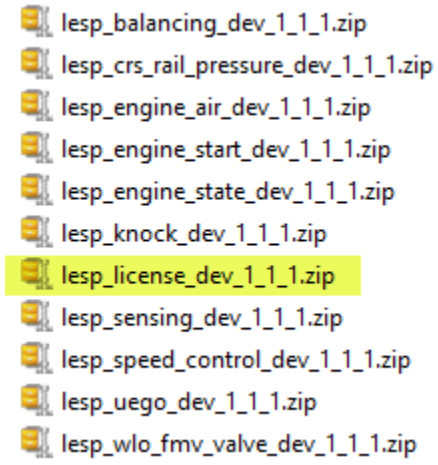


Figure 1-2. LESP Model Reference Distribution & License Archives

Feature Versioning

All LESP model reference blocks are versioned using a <BuildType>_<Major>_<Minor>_<Build> versioning scheme.

- Build Type
  - Dev = Development build, generally used for debugging or prototypes.
  - Rel = Release. Used in production and distribution.
- Major
  - Major version based on the LESP platform major version number.
- Minor
  - Minor version based on the LESP platform minor version number.
- Build
  - Unique platform build (revision) number.

IMPORTANT **A version change may not equate to a functional change in the block but could be related to other non-functional changes based on the LESP platform updates, including Matlab, MotoHawk, and Woodward ASB version updates. See release notes for LESP features for functional changes and issue resolutions.**

PC License (or License Update)

After extracting the license archive file (lesp_license_*.zip) to the local project directory on the respective PC node, the licensing wizard can be activated by ensuring the LESP license folder is on the Matlab path and executing the following command in Matlab:

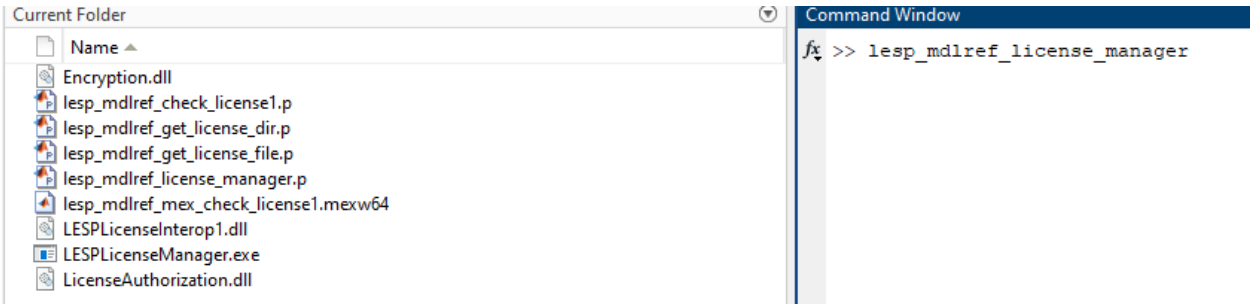


Figure 1-3. Launch LESP License Manager

This will launch the LESP license manager dialog. The dialog enables two different types of license requests based on the purchase. An annual license will last for 366 days from the date the license key is retrieved. A perpetual license does not expire. The two licenses are not compatible, and a new request will override any current active license. If purchasing an annual license, use the Request Annual License button; otherwise, use the Request Perpetual License button.

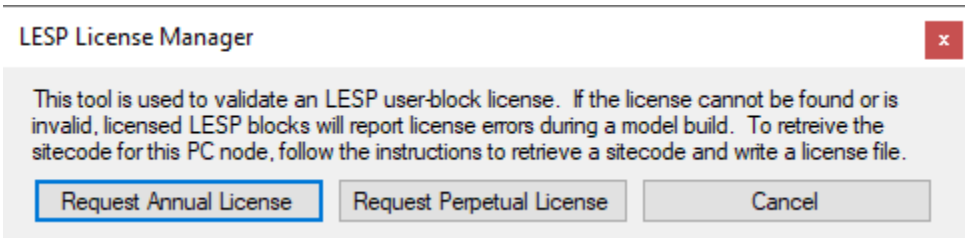


Figure 1-4. LESP License Manager

After clicking the license request button, a second dialog will open. Select the drive letter for which to associate the license and click OK.

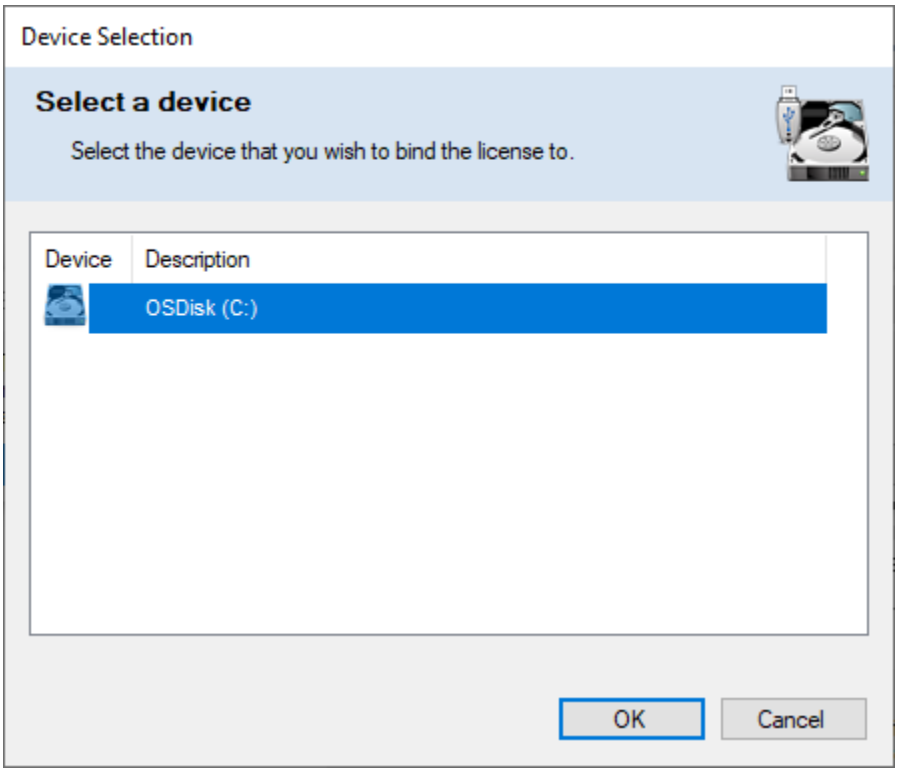


Figure 1-5. LESP License Manager (Select HDD)

A final dialog will appear with a local PC device site code. Using the site code and the serial number received after the purchase, retrieve the license key from your Woodward account. Paste the license key in the License Key field of the form and hit OK. The PC should now be licensed.

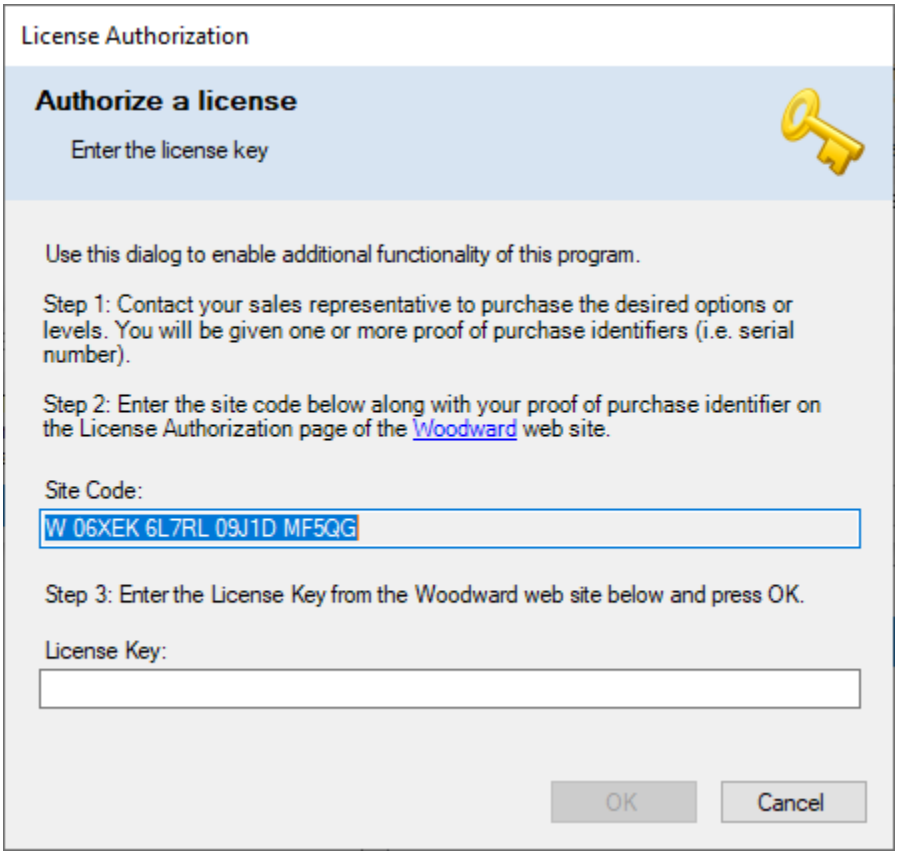


Figure 1-6. LESP License Manager (Site Code & License Key)

Verify License

The license can be validated with the lesp_mdlref_check_license1 command. A logical ‘1’ is returned if the license is valid; otherwise, a logical ‘0’ is returned.

```
>> lesp_mdlref_check_license1

ans =

  logical

   1
```

Figure 1-7. Verify LESP License

To retrieve the number of days remaining for annual licenses, add three output arguments to the check license function call as indicated below.

```
>> [license_valid, errcode, days_remaining] = lesp_mdlref_check_licensel

license_valid =

  logical

   1


errcode =

     0


days_remaining =

   352
```

Figure 1-8. Verify LESP License (Days Remaining)

IMPORTANT **LESP licensing is validated during a model compilation (build). If the license is within 30 days of expiring, warnings will be posted.**

Block Dependencies

At the time of release, all LESP feature blocks are compiled with a specific version of Matlab, MotoHawk, MotoHawk Target, and Compiler. These versions must match when the blocks are integrated in other applications. Matlab and MotoHawk versions are fixed and documented in the block mask. In addition, the blocks will report an error during a model update if the Matlab or MotoHawk versions do not match. Most blocks are distributed with multiple variants of MotoHawk targets and compilers; however, these must match the current model for which the blocks are located; otherwise, an error will occur during compilation.

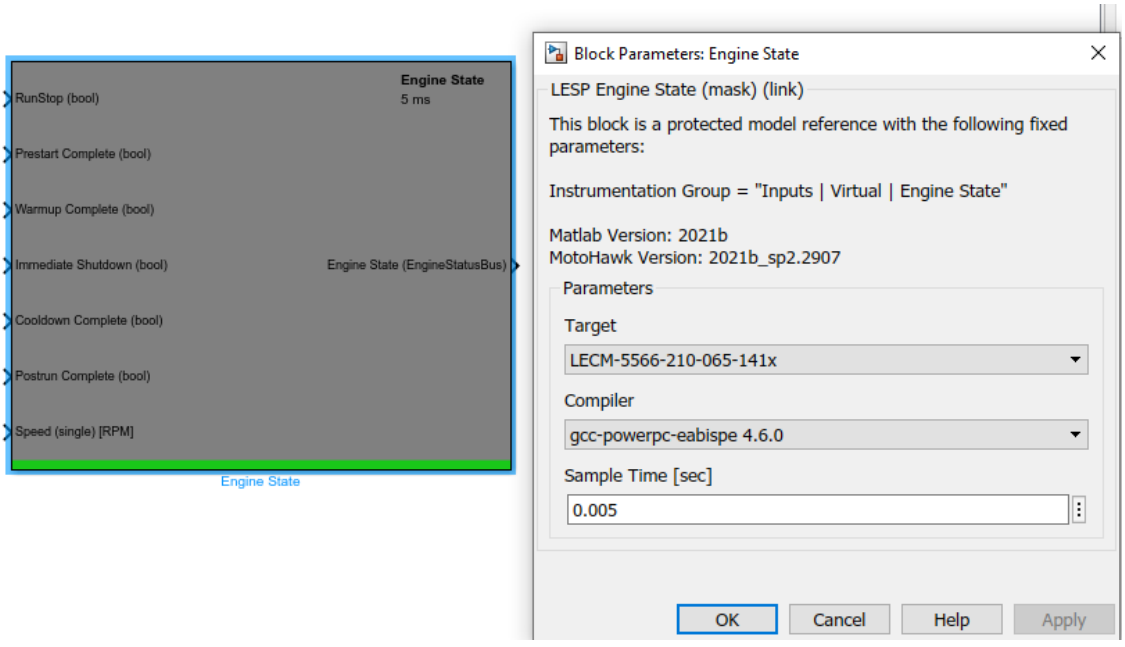


Figure 1-9. Block Dependencies

The LESP Packages Documentation

The LESP packages do contain several model references, as mentioned earlier. Each of the blocks will be described in greater detail in the next chapters of this manual.

Important Notes

When using the model reference blocks inside a new application, the guides for the framework as described in manual 35203V1 must be followed. Any LESP framework dependency variables will automatically be loaded to the base workspace during a model update.

The model reference blocks are set to the C99 (ISO) settings for the ‘C’ language standard, so be sure to have the application also setup in such way, otherwise an error will be generated during the building process.

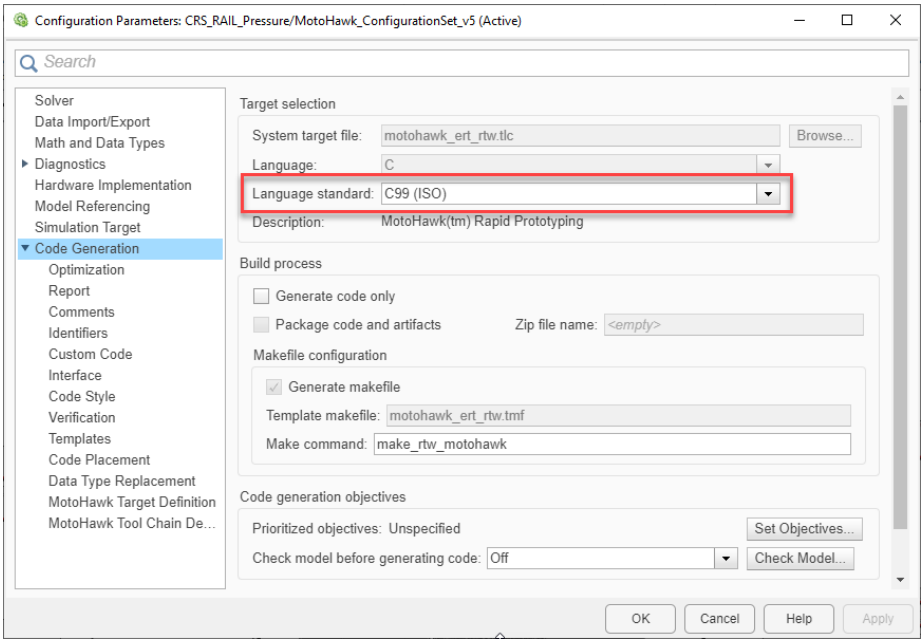


Figure 1-10. ‘C’ Language Standard Setting (Simulink)

In addition, it is important to set the sample_time for the Model Reference blocks to the correct trigger times where they will be triggered in the application model itself.

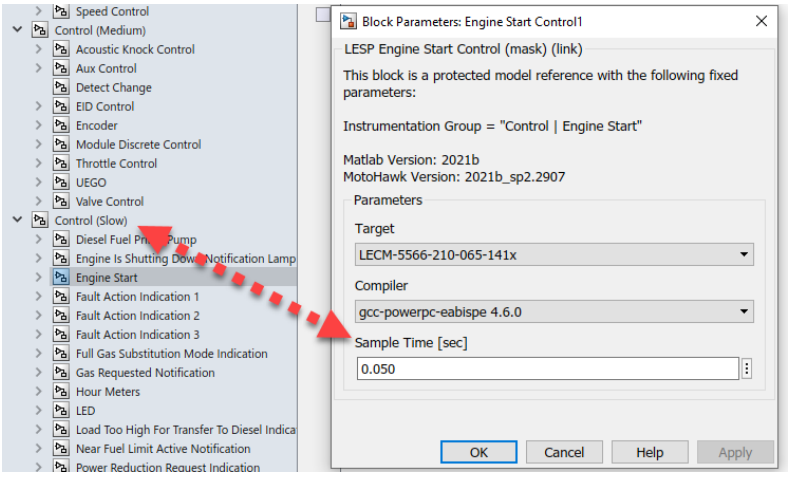


Figure 1-11. Note ‘Sample Time Settings’ to Match Trigger Rate of Block

Chapter 2.
Control - CRS Rail Pressure Control

Introduction

The ‘Control - LESP CRS Rail Pressure Control’ blockset is used to perform rail pressure control and diagnostics monitoring. Inside the blockset, three blocks are available as can be seen below:

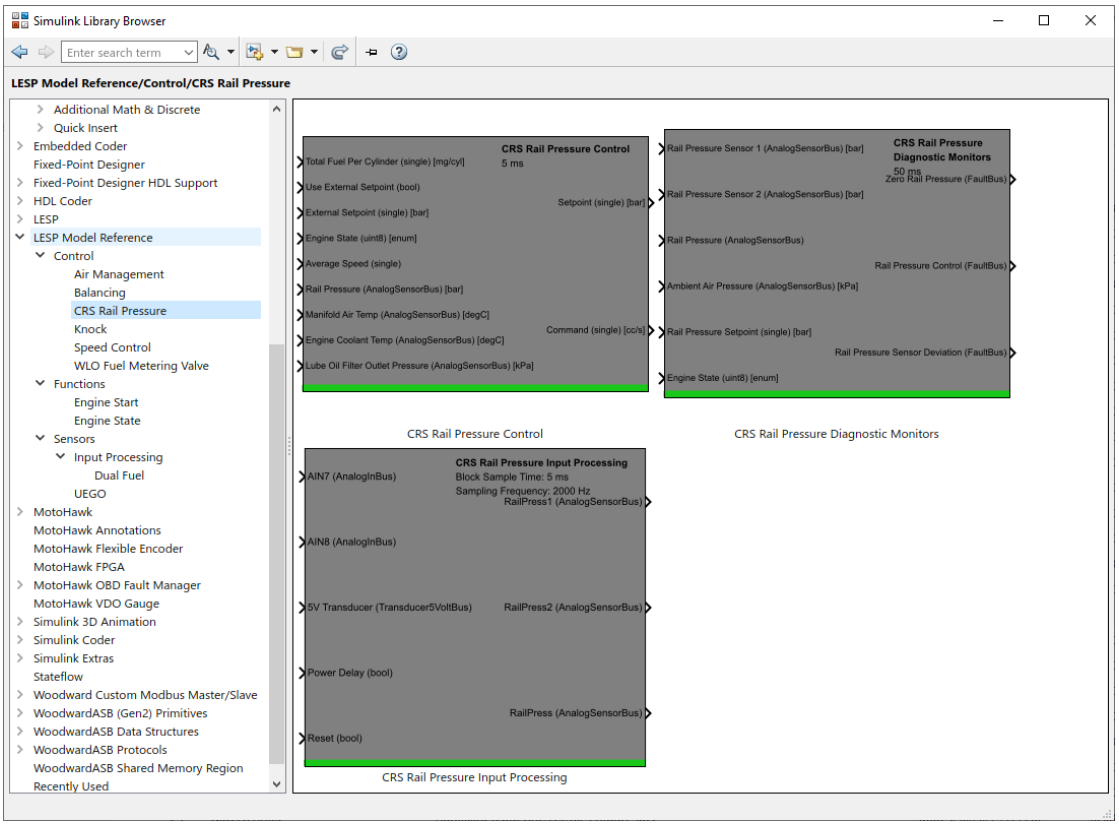


Figure 2-1. LESP Model Reference CRS Rail Pressure Blocks

- CRS Rail Pressure Control
  This block provides the Rail Pressure Control logic, based on the measured rail pressure, fuel per cylinder, and actual speed. The output of the block gives the command signal, which can be used to drive a fuel metering valve (FMV) on a high pressure fuel pump.

- CRS Rail Pressure Input Processing
  This block provides the logic to process the Rail Pressure sensor info. It defines the inputs, including voltage compensation, and performs the logic to combine the two sensors into a signal value which can be used in the application. The logic to be used can be selected from this block and the outputs can be used to feed the other inputs of the blocks in this blockset.

- CRS Rail Pressure Diagnostic Monitors
  This block provides the Diagnostic monitors for the Rail Pressure Control block. Based on several input signals, the block generates additional CRS Rail Pressure based diagnostics output signals, like Rail Pressure Sensor Deviation, zero Rail Pressure detection and more, which can be very useful for the application to know. The outputs will be set as FaultBus outputs, containing as many details as needed in the application that is created.

CRS Rail Pressure Control Block Interface

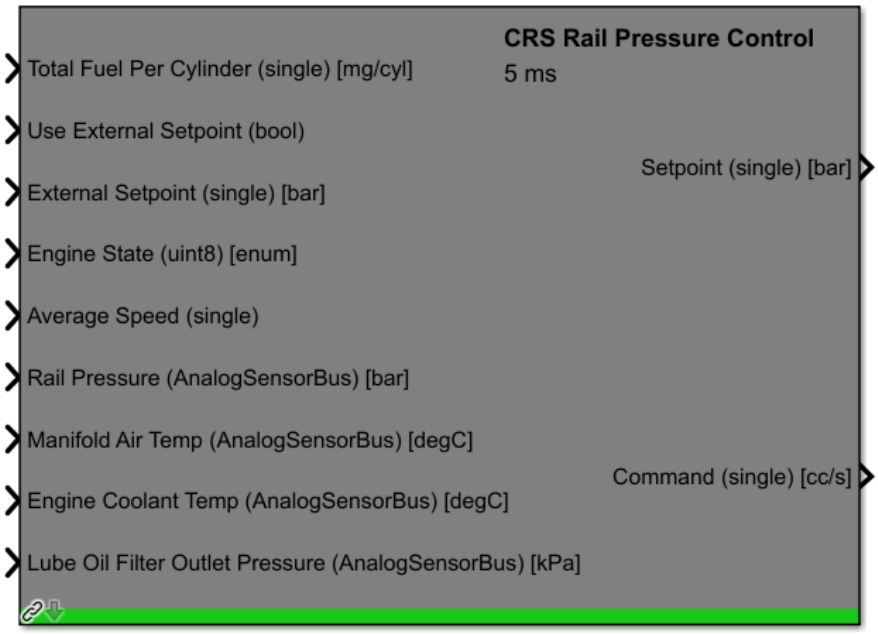


Figure 2-2. 'CRS Rail Pressure Control' Block

Table 2-1. 'CRS Rail Pressure Control' Block

Port	DataType	Description
[In] Total Fuel Per Cylinder [mg/cyl]	single	The total fuel per cylinder that is commanded from the speed control logic to maintain stable speed control.
[In] Use external Setpoint	boolean	If an external source for Rail Pressure setpoint is required, this parameter will force it to be used. When set to TRUE, the 'External Setpoint' value will be used as reference. When set to FALSE, the internal Rail Pressure setpoint will be used.
[In] External Setpoint [bar]	single	The external Rail Pressure setpoint value that will be used in case the Boolean input 'Use External Setpoint' is set to TRUE.
[In] Engine State [enum]	uint8	The actual engine state the engine is operating in at that particular moment. These states (lesp_engine_state_enum) are defined as below: 0 – Stopped 1 – Prestart 2 – Starting 3 – Warmup 4 – Running 5 – Cooldown 6 – Stopping 7 – Postrun
[In] Average Speed	single	The average engine speed as sensed in the application.

[In] Rail Pressure [bar]	AnalogSensorBus	Actual Rail Pressure as sensed. Bus Details: AnalogSensorBus: - FilteredValue - MeasurementStatus_enum MeasurementStatus_enum: (lesp_measurement_status_enum) 0 – Valid 1 – Disabled 2 – SignalLow 3 – SignalHigh 4 – SensorSupplyFailed 5 – ReferenceSourceFailed 6 – OpenCircuit
[In] Manifold Air Temp [degC]	AnalogSensorBus	Actual Manifold Air Temperature (MAT). Bus Details: AnalogSensorBus: - FilteredValue - MeasurementStatus_enum MeasurementStatus_enum: (lesp_measurement_status_enum) 0 – Valid 1 – Disabled 2 – SignalLow 3 – SignalHigh 4 – SensorSupplyFailed 5 – ReferenceSourceFailed 6 – OpenCircuit
[In] Engine Coolant Temp [degC]	AnalogSensorBus	Actual Engine Coolant Temperature (ECT). Bus Details: AnalogSensorBus: - FilteredValue - MeasurementStatus_enum MeasurementStatus_enum: (lesp_measurement_status_enum) 0 – Valid 1 – Disabled 2 – SignalLow 3 – SignalHigh 4 – SensorSupplyFailed 5 – ReferenceSourceFailed 6 – OpenCircuit
[In] Lube Oil Filter Outlet Pressure [kPa]	AnalogSensorBus	Actual Lube Oil Filter Outlet Pressure Bus Details: AnalogSensorBus: - FilteredValue - MeasurementStatus_enum MeasurementStatus_enum: (lesp_measurement_status_enum) 0 – Valid 1 – Disabled 2 – SignalLow 3 – SignalHigh 4 – SensorSupplyFailed 5 – ReferenceSourceFailed 6 – OpenCircuit
[Out] Setpoint [bar]	single	Actual active Rail Pressure Setpoint in [bar].

[Out] Command [cc/s]	single	The command output in cc/seconds which can be used to control the fuel metering valve (FMV) on the high pressure CRS pump.

CRS Rail Pressure Diagnostic Monitors Block Interface

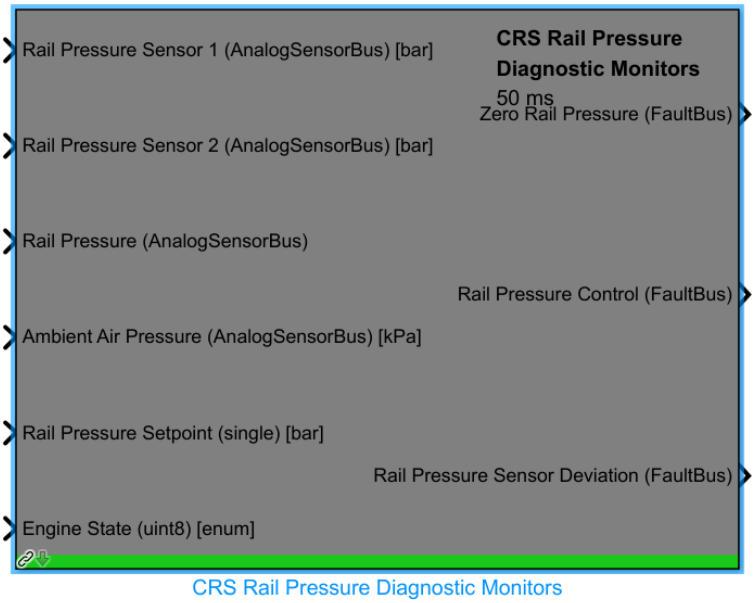


Figure 2-3. ‘CRS Rail Pressure Diagnostic Monitors’ Block

Table 2-2. ‘CRS Rail Pressure Diagnostic Monitors’ Block

Port	DataType	Description
[In] Rail Pressure Sensor 1 [bar]	AnalogSensorBus	The Rail Pressure as sensed by Rail Pressure Sensor #1. Bus Details: AnalogSensorBus: - FilteredValue - MeasurementStatus_enum MeasurementStatus_enum: (lesp_measurement_status_enum) 0 – Valid 1 – Disabled 2 – SignalLow 3 – SignalHigh 4 – SensorSupplyFailed 5 – ReferenceSourceFailed 6 – OpenCircuit
[In] Rail Pressure Sensor 2 [bar]	AnalogSensorBus	The Rail Pressure as sensed by Rail Pressure Sensor #2. Bus Details: AnalogSensorBus: - FilteredValue - MeasurementStatus_enum MeasurementStatus_enum: (lesp_measurement_status_enum) 0 – Valid 1 – Disabled 2 – SignalLow

		3 – SignalHigh 4 – SensorSupplyFailed 5 – ReferenceSourceFailed 6 – OpenCircuit
[In] Rail Pressure	AnalogSensorBus	The Rail Pressure as used for control purposes, which can be HSS, LSS, or average (depending on selection in the application. Bus Details: AnalogSensorBus: - FilteredValue - MeasurementStatus_enum MeasurementStatus_enum: (lesp_measurement_status_enum) 0 – Valid 1 – Disabled 2 – SignalLow 3 – SignalHigh 4 – SensorSupplyFailed 5 – ReferenceSourceFailed 6 – OpenCircuit
[In] Ambient Air Pressure [kPa]	AnalogSensorBus	The Actual Ambient Air Pressure. Bus Details: AnalogSensorBus: - FilteredValue - MeasurementStatus_enum MeasurementStatus_enum: (lesp_measurement_status_enum) 0 – Valid 1 – Disabled 2 – SignalLow 3 – SignalHigh 4 – SensorSupplyFailed 5 – ReferenceSourceFailed 6 – OpenCircuit
[In] Rail Pressure Setpoint [bar]	single	The actual Rail Pressure setpoint as defined in the application
[In] Engine State [enum]	uint8	The actual engine state the engine is operating in at that particular moment. These states (lesp_engine_state_enum) are defined as below: 0 – Stopped 1 – Prestart 2 – Starting 3 – Warmup 4 – Running 5 – Cooldown 6 – Stopping 7 – Postrun
[Out] Zero Rail Pressure	FaultBus	Indication if zero Rail Pressure has been sensed at a moment where rail pressure is expected.
[Out] Rail Pressure Control	FaultBus	Indication of faults during Rail Pressure Control.
[Out] Rail Pressure Sensor Deviation	FaultBus	Indication of faults when Rail Pressure Sensors are deviating from each other

CRS Rail Pressure Input Processing

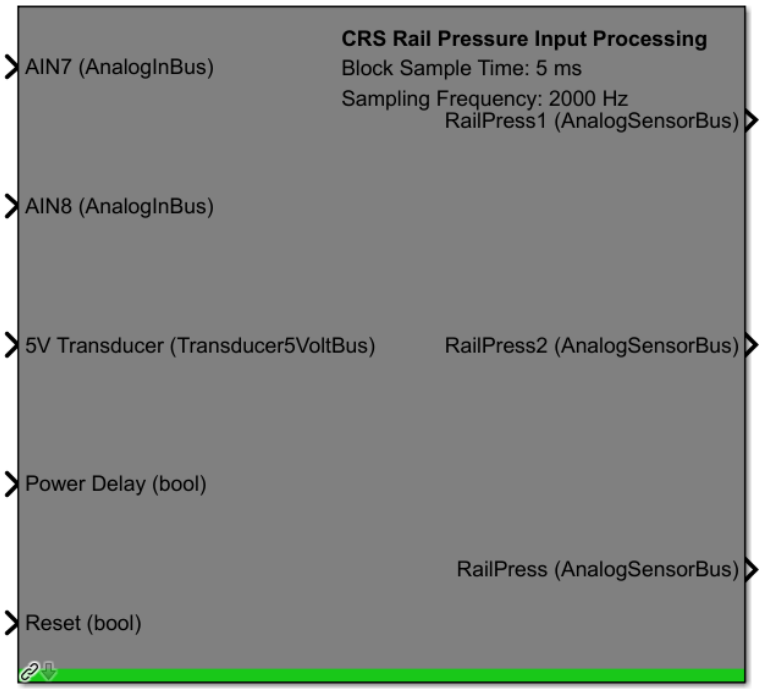


Figure 2-4. 'CRS Rail Pressure Input Processing' Block

Table 2-3. 'CRS Rail Pressure Input Processing' Block

Port	DataType	Description
[In] AIN7	AnalogSensorBus	Analog Input Channel definition where the high frequency sampling is used to sample/measure the rail pressure as sensed by Rail Pressure Sensor #1. Bus Details: AnalogSensorBus: - FilteredValue - MeasurementStatus_enum MeasurementStatus_enum: (lesp_measurement_status_enum) 0 – Valid 1 – Disabled 2 – SignalLow 3 – SignalHigh 4 – SensorSupplyFailed 5 – ReferenceSourceFailed 6 – OpenCircuit
[In] AIN8	AnalogSensorBus	Analog Input Channel definition where the high frequency sampling is used to sample/measure the Rail Pressure as sensed by Rail Pressure Sensor #2. Bus Details: AnalogSensorBus: - FilteredValue - MeasurementStatus_enum MeasurementStatus_enum: (lesp_measurement_status_enum) 0 – Valid 1 – Disabled

		2 – SignalLow 3 – SignalHigh 4 – SensorSupplyFailed 5 – ReferenceSourceFailed 6 – OpenCircuit
[In] 5V Transducer	Transducer5VoltBus	The actual voltage of the 5V transducer, used for ratiometric compensation of the input signals.
[In] Power Delay	boolean	The delay in power up of the 5 V transducer voltage after reboot of the ECU. This is required to prevent false fault detection.
[In] Reset	boolean	Reset input to reset the analog input faults, in case they are in failure mode.
[Out] RailPress1	AnalogSensorBus	The actual RailPressure #1 signal as measured and processed by the logic. Bus Details: AnalogSensorBus: - FilteredValue - MeasurementStatus_enum MeasurementStatus_enum: (lesp_measurement_status_enum) 0 – Valid 1 – Disabled 2 – SignalLow 3 – SignalHigh 4 – SensorSupplyFailed 5 – ReferenceSourceFailed 6 – OpenCircuit
[Out] RailPress2	AnalogSensorBus	The actual RailPressure #2 signal as measured and processed by the logic. Bus Details: AnalogSensorBus: - FilteredValue - MeasurementStatus_enum MeasurementStatus_enum: (lesp_measurement_status_enum) 0 – Valid 1 – Disabled 2 – SignalLow 3 – SignalHigh 4 – SensorSupplyFailed 5 – ReferenceSourceFailed 6 – OpenCircuit
[Out] RailPress	AnalogSensorBus	The actual RailPressure signal that is the result of the combination of the 2 RailPressure sensors and selected logic inside this blockset. Bus Details: AnalogSensorBus: - FilteredValue - MeasurementStatus_enum MeasurementStatus_enum: (lesp_measurement_status_enum) 0 – Valid 1 – Disabled 2 – SignalLow 3 – SignalHigh 4 – SensorSupplyFailed

		5 – ReferenceSourceFailed 6 – OpenCircuit

Example Model Using the Blocks

Below is a very simple setup using MotoHawk Calibration and MotoHawk Probes blocks to make the model suitable for building.

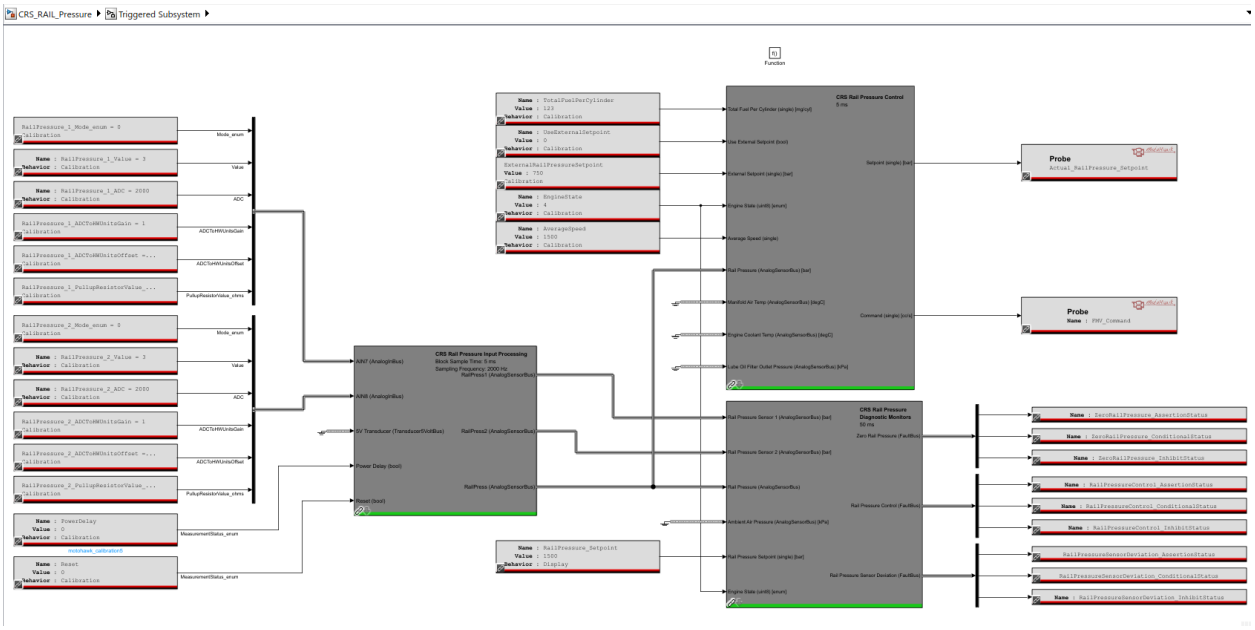


Figure 2-5. Simple Example Model Using the CRS RailPressure Control Blocks

When compiling, it will generate the required files. If the Toolkit required blocks are also added to the application model, it will generate the required SID files for the Toolkit HMI tool creation.

Default ToolKit Pages

Opening a new Toolkit application and selecting the correctly generated .sid file will look like the screenshot below:

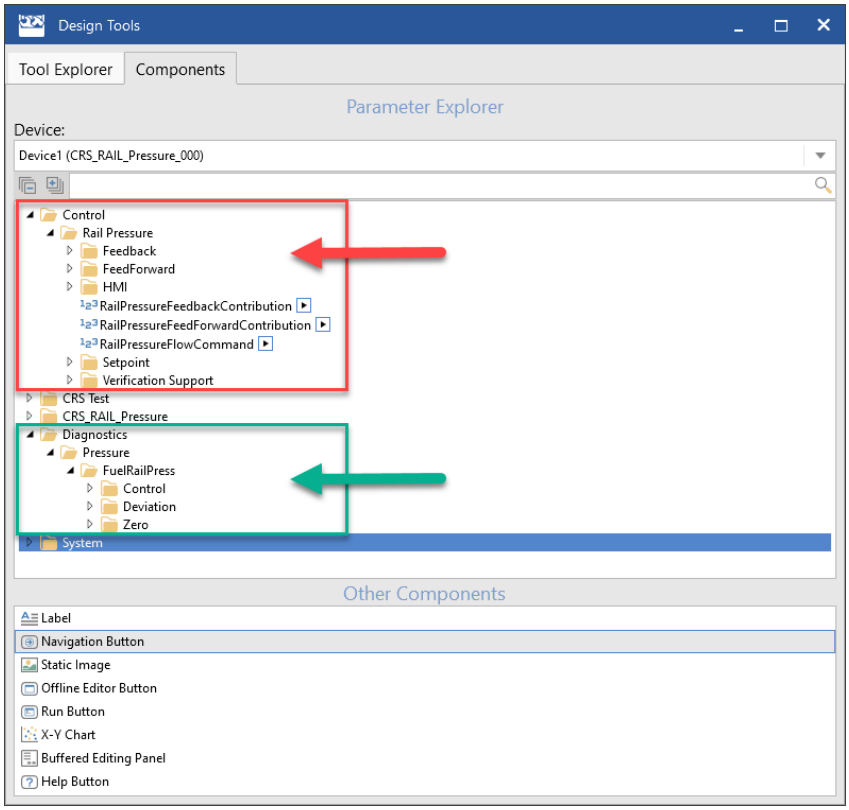


Figure 2-6. SID File Items Included in the Model Reference Blocks

As shown, two major parts can be recognized as coming from the Model Reference blocks:

1) Control
2) Diagnostics

Each of the items have sub-systems containing more details.
Default pages are created in an example Toolkit, which can be used as a starting point for the application specific Toolkit HMI tool.

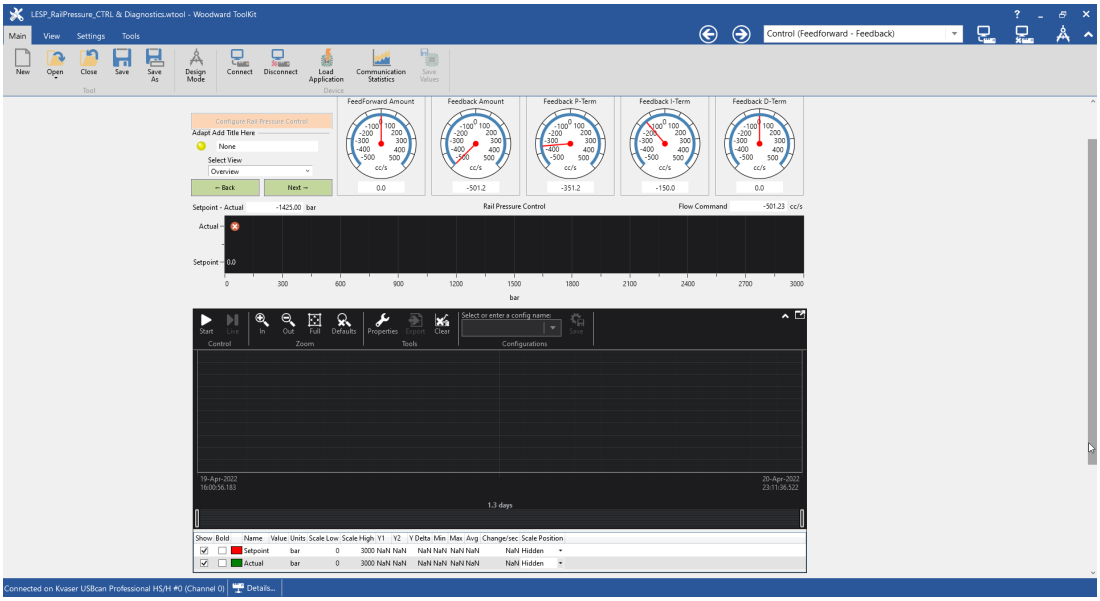


Figure 2-7. Overview Page Rail Pressure Control

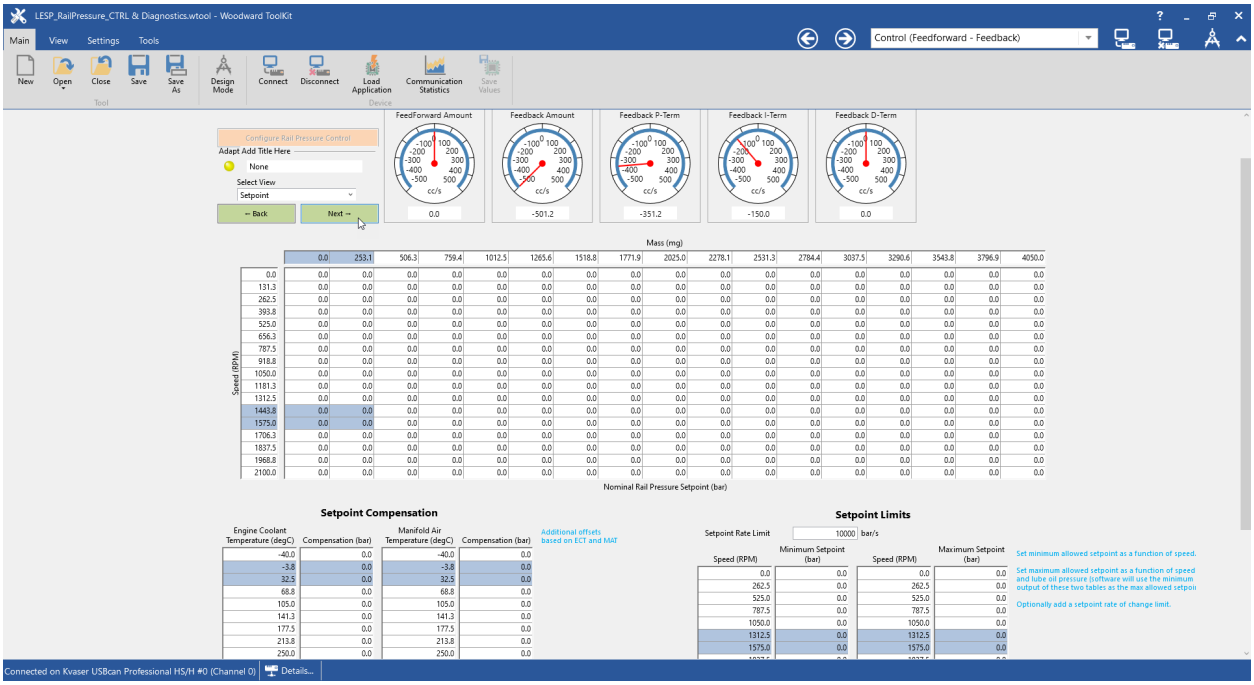


Figure 2-8. Setpoint Page Rail Pressure Control

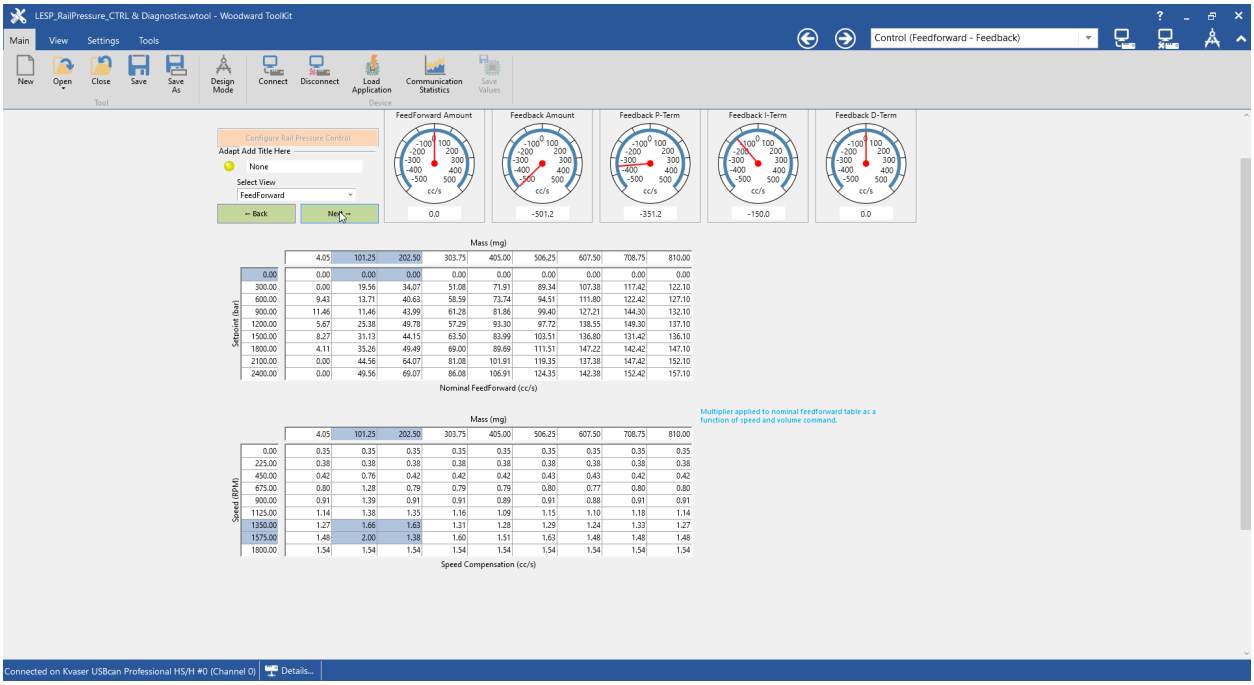


Figure 2-9. FeedForward Page Rail Pressure Control

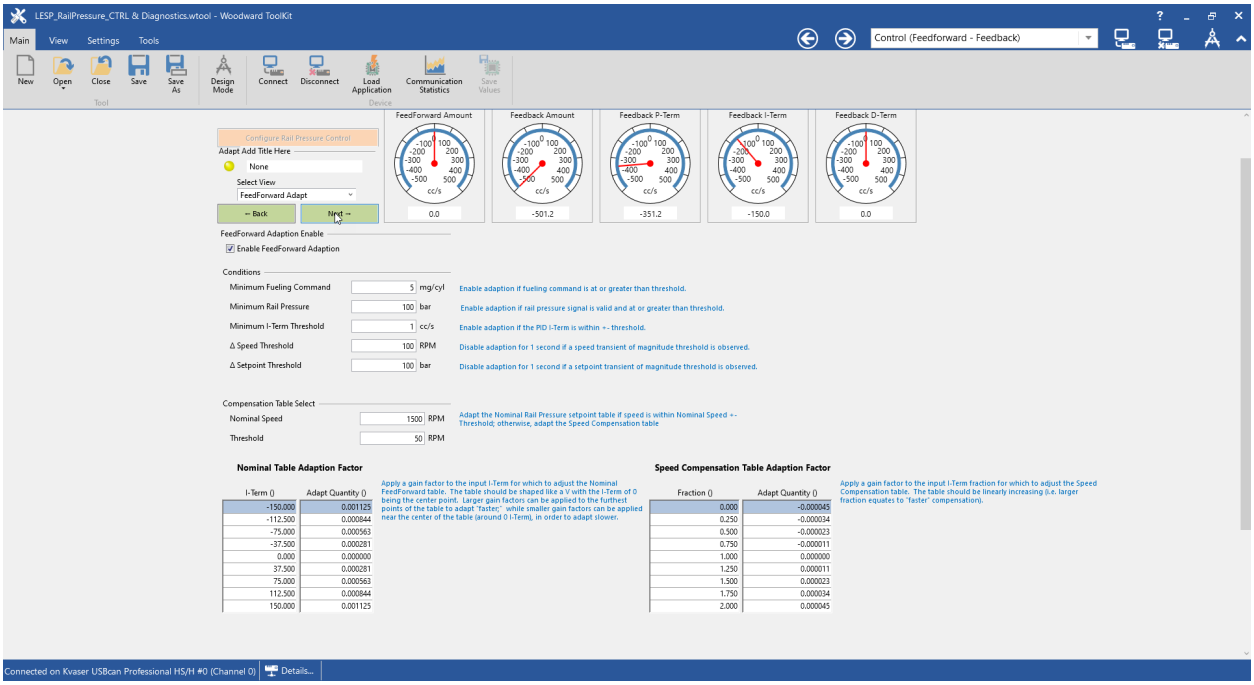


Figure 2-10. FeedForward Adapt Page Rail Pressure Control

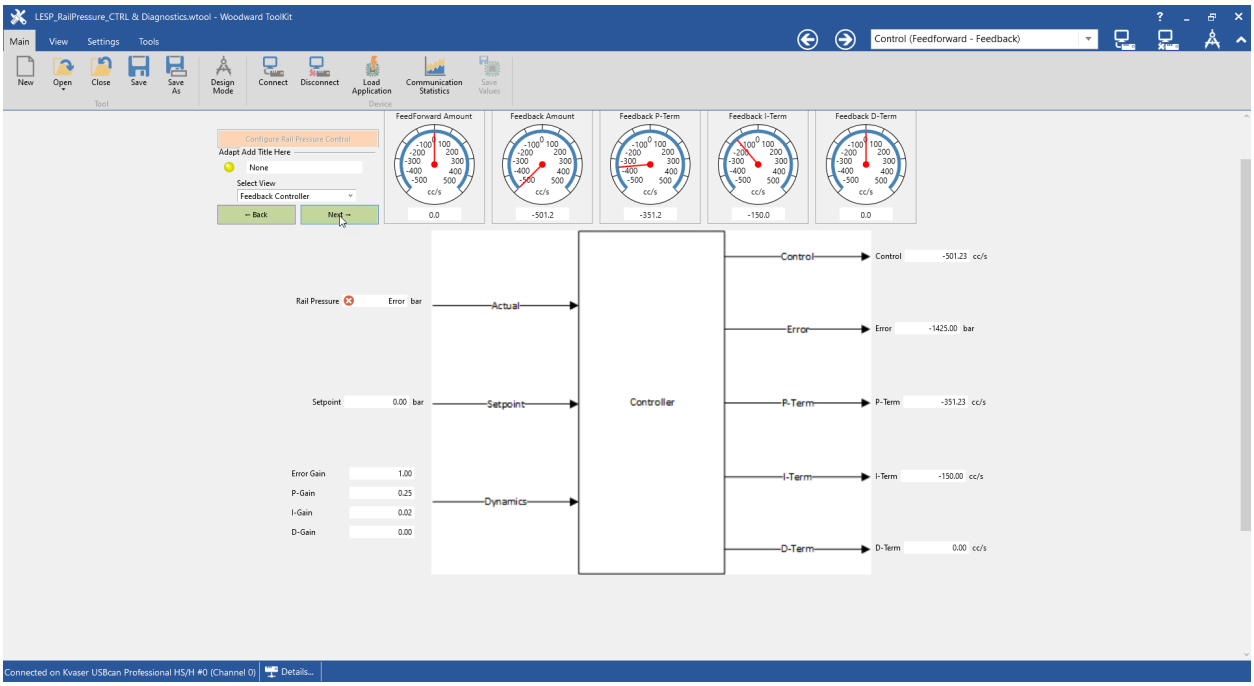


Figure 2-11. FeedBack Controller Page Rail Pressure Control

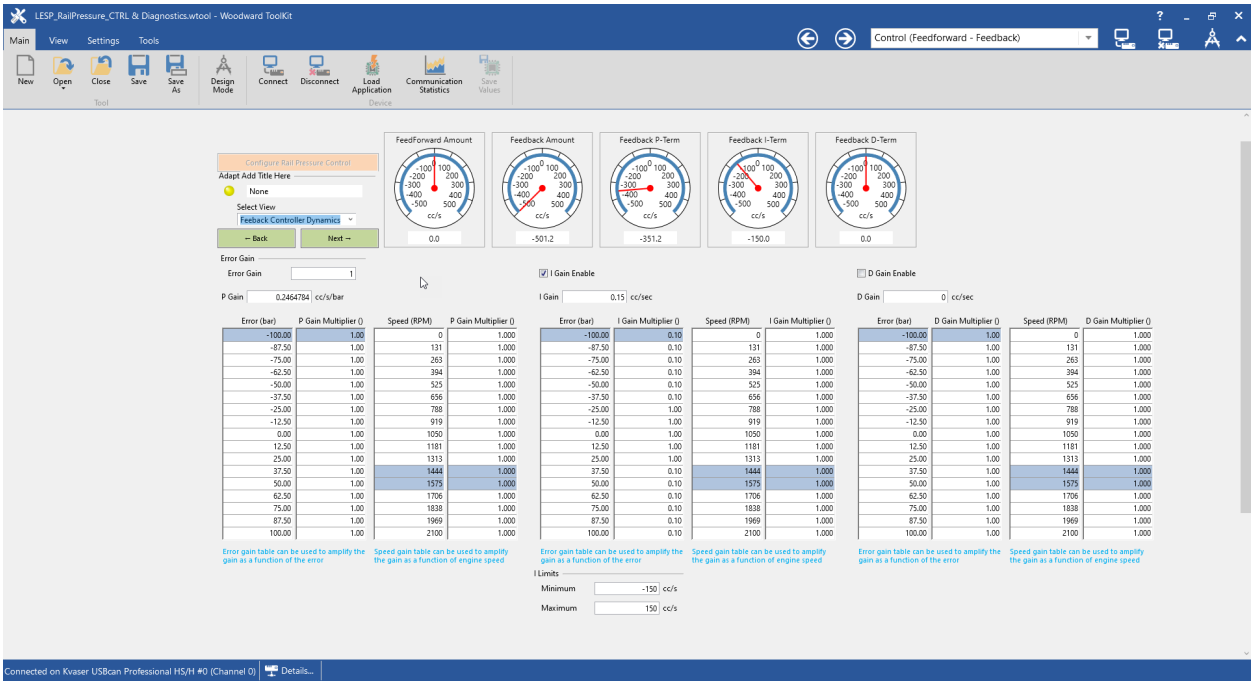


Figure 2-12. FeedBack Controller Dynamics Page Rail Pressure Control

There is also a default page for the diagnostics as shown below. Enabling, thresholds, and time delays can be set from this page. The LEDs are an example of the outputs.

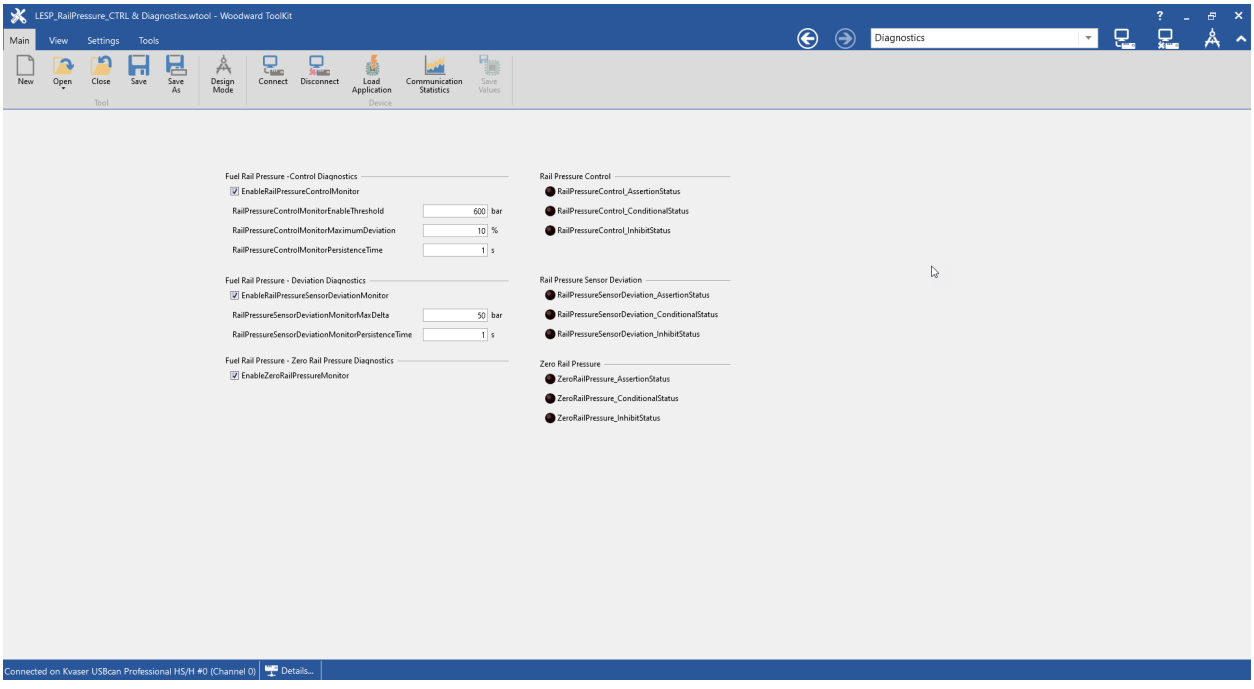


Figure 2-13. Example of the Toolkit Pages for Rail Pressure Diagnostics

Functional Description - CRS Rail Pressure Control Logic

The following describes Large Engine Control Module Common Rail System (LECM CRS) platform software rail pressure sensing and control strategies.

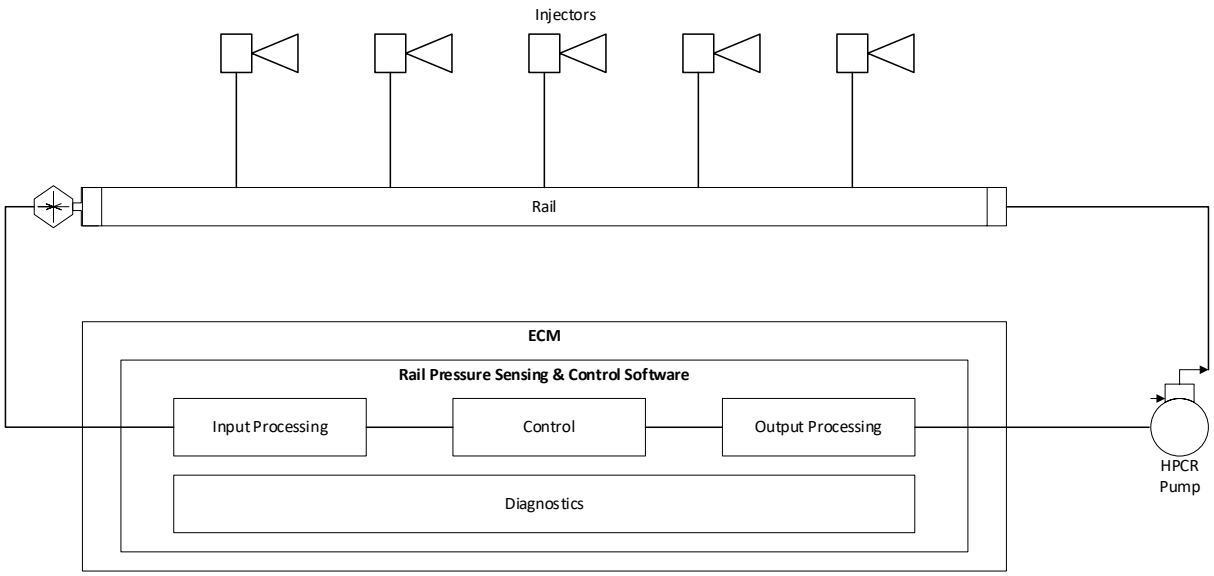


Figure 2-14. Rail Pressure Sensing & Control Overview

Rail Pressure Sensing

Introduction

The standard LECM CRS software implements redundant fast sampling of rail pressure sensing. Each sensor samples the hardware input at 2 kHz and implements a configurable N^{th} order low-pass Butterworth filter. Sensor arbitration is used to select the downstream signal used for rail pressure control. In order to support fast sampling, the rail pressure sensors support fixed analog inputs as outlined in the table below.

Table 2-4. Rail Pressure Sensor IO Mappings

Rail Pressure Sensor	IO Mapping
Sensor #1	Not Used = Sensor is not available AIN07 = Sensor utilizes AIN07 which can be configured for 0 – 5V or 4 – 20 mA sensor type.
Sensor #2	Not Used = Sensor is not available AIN08 = Sensor utilizes AIN08 which can be configured for 0 – 5V or 4 – 20 mA sensor type.

IMPORTANT

Rail pressure signals are sampled at a rate greater than or equal to 1.8 kHz to have the latest sample of rail pressure prior to an injection event and filter noise from rail pressure dynamics.

Example:
1800 RPM, 20 cylinder engine with a single shot equates to 300 firing events per second. Minimum Nyquist 2x sample rate equates to 300 x 2 = 600 Hz; however, a > 6x factor is used instead.

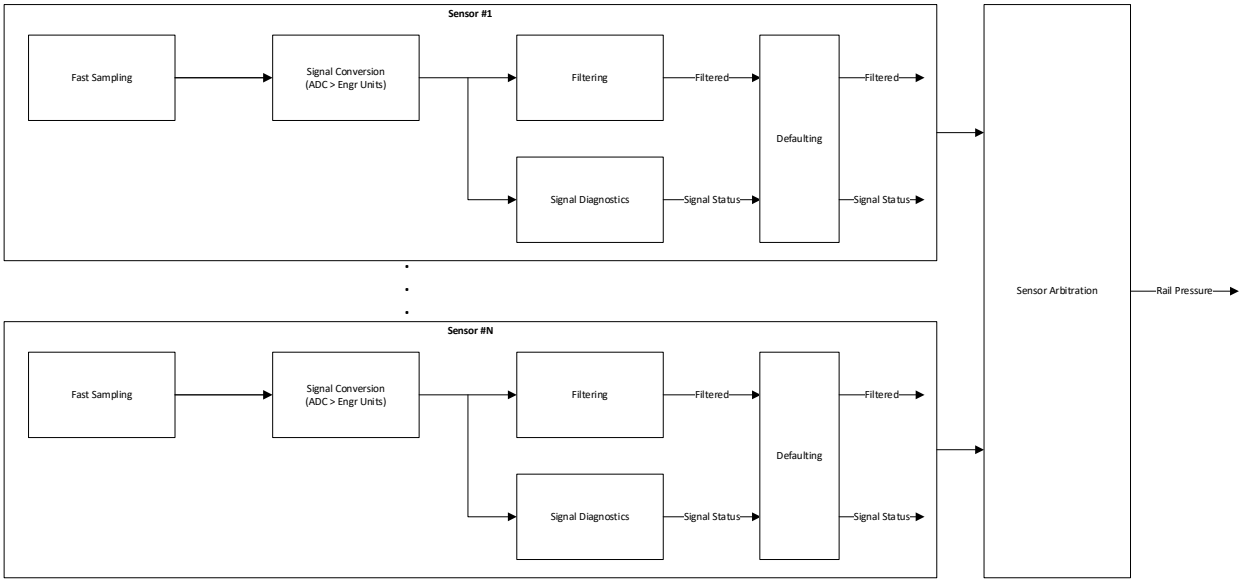


Figure 2-15. Rail Pressure Sensing Strategy

Conversion

The conversion logic implements a linear *mx+b* conversion and uses standard units of bar. The gain and offset values are tunable and HW units are based on the IO input value.

$$Pressure\ [bar] = \left(Gain\ \left[\frac{bar}{HW\ Units}\right]\ x\ IO\ Value\ [HW\ Units]\right) + Offset\ [bar]$$

The conversion logic also supports an option to enable ratiometric scaling by means of the 5V transducer that powers the sensor. When this feature is enabled, the Gain is adjusted by the power supply reference correction factor. The ratiometric correction factor is limited to a range between 0.95 and 1.05.

$$Pressure\ [bar] = \left((Gain\ x\ Correction_{Ratiometric})x\ \left[\frac{bar}{HW\ Units}\right]\ x\ IO\ Value\ [HW\ Units]\right) + Offset\ [bar]$$

Figure 2-16. Rail Pressure Sensor Configuration

Filtering

The software implements a configurable N^{th} order Butterworth Low-Pass filter with global settings applied to all enabled rail pressure sensors. They can optionally be disabled (test purposes), the cutoff frequency adjusted, and the filter order adjusted between 1 and 3.

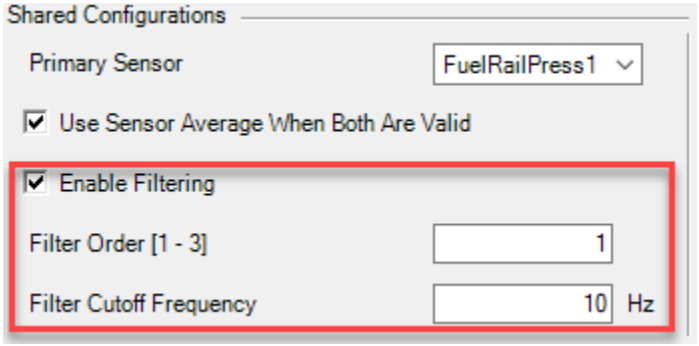


Figure 2-17. Rail Pressure Filter Configurations

IMPORTANT **It is highly recommended to utilize the default filtering options of filter order = 1 and cutoff frequency = 10 Hz; however, adjustments may be required for different systems.**

Signal Diagnostics

The signal diagnostics determine the validity of the input signal and support the following statuses:

- **Disabled**
  - A disabled status is reported when the sensor is not mapped to IO.
- **SignalLow**
  - A signal low status is reported when the input IO value is below a user-defined value.
- **SignalHigh**
  - A signal high status is reported when the input IO value is above a user-defined value.
- **SensorSupplyFailed**
  - A sensor supply failed status is reported when the pressure sensor is supplied by a transducer supply and the transducer supply status is not valid.
- **Valid**
  - A valid status is reported otherwise.

Defaulting

The defaulting logic supports three modes of defaulting the downstream signal (engineering units):

- **None**
  - The downstream signal will report the converted input signal regardless of the input (i.e., no defaulting).
- **Use Constant (NonLatching)**
  - The downstream signal will use a use-defined constant value while the status is not valid. The default override will be disabled once the status returns to valid.
- **Use Last Valid Value (NonLatching)**
  - The downstream signal will use the last signal valid value while the status is not valid. The default override will be disabled once the status returns to valid.

- **Use Last Valid Value (Latching)**
  - The downstream signal will use the last signal valid value while the status is not valid. The default override will be disabled once the status returns to valid AND only after a Reset All Active Faults event is requested.

Sensor Arbitration

The software supports a standard sensor arbiter that selects the rail pressure signal used in downstream control logic of all enabled sensors based on the signal status illustrated in the figure below. The arbiter also supports an option to override the downstream filtered signal value with an average of all valid sensors.

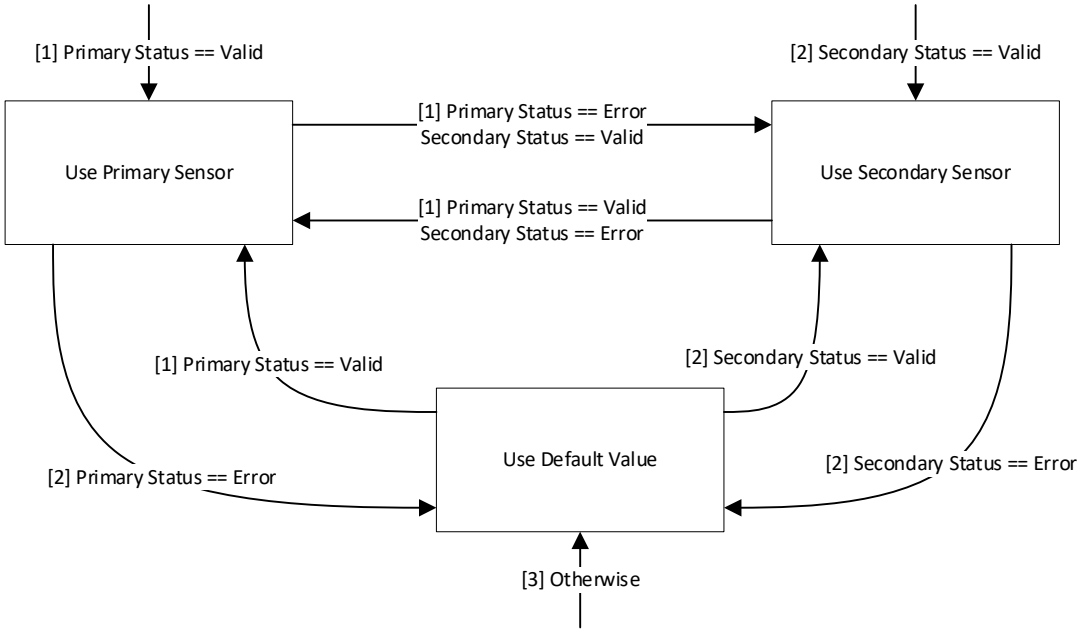


Figure 2-18. Sensor Arbitration Overview

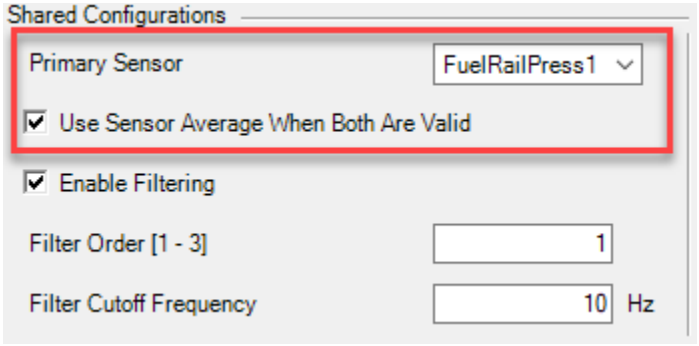


Figure 2-19. Sensor Arbitration Configurations

Functional Description - Rail Pressure Control

The following describes the adaptive table-based feedforward + PID feedback rail pressure control strategy utilized in the LECM CRS software.

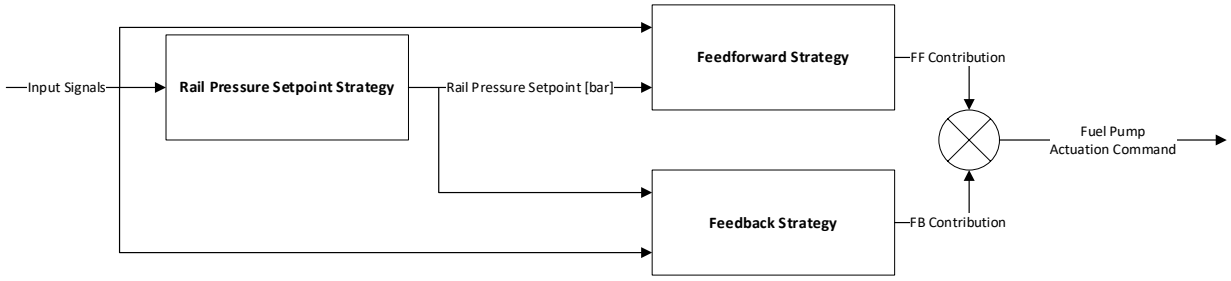


Figure 2-20. Rail Pressure Control Strategy Overview

Setpoint Strategy

The software supports a nominal 17x17 2-D lookup table to command a base rail pressure setpoint as a function of engine speed [RPM] and fuel per cylinder demand [mm^3]. The base setpoint is compensated by engine coolant temperature and manifold air temperature biases. A rate limit can also be applied to the setpoint to avoid large step changes in order to provide for a more stable control. Independent minimum and maximum setpoint tables as a function of engine speed are then applied to the final commanded rail pressure setpoint.

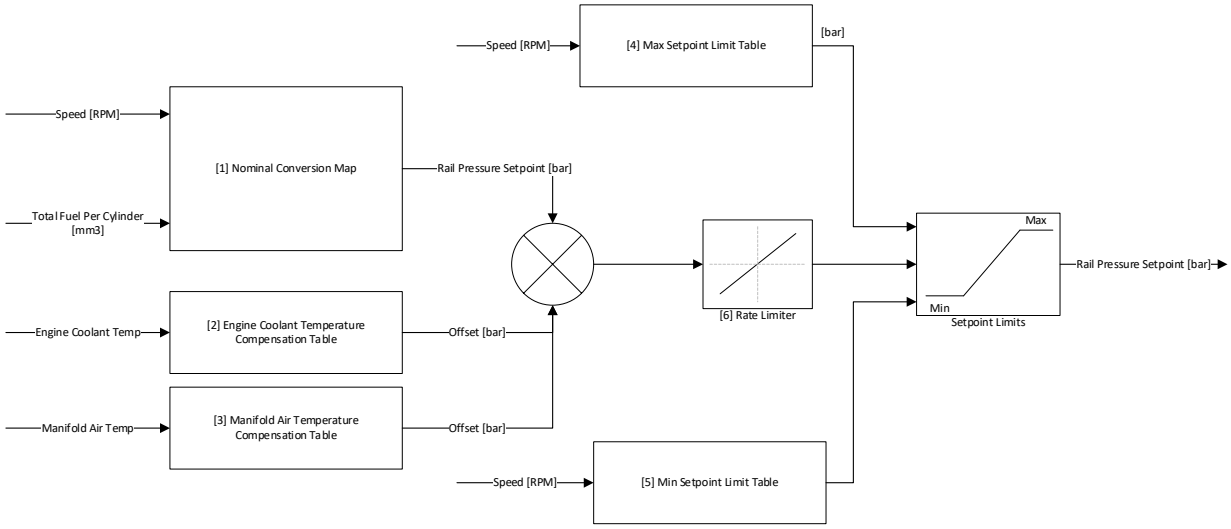


Figure 2-21. Setpoint Strategy

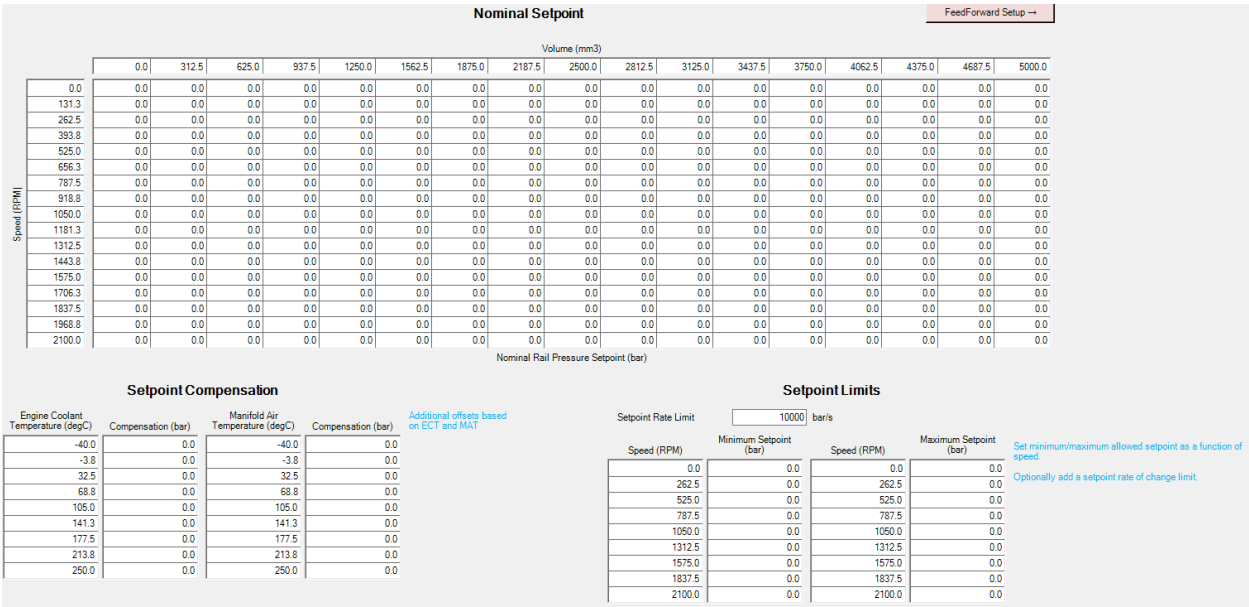


Figure 2-22. Setpoint Configurations

Feedforward Strategy

The software feedforward strategy consists of two 2-D tables that utilize engine speed, actual rail pressure, and fuel demand to command a fuel flow [cc/s]. The first table (a.k.a. Nominal) provides a base fuel flow command as a function of rail pressure and fuel demand, and the second table (a.k.a. Speed Compensation) provides a gain factor based on engine speed and fuel demand. Both tables can be dynamically adapted by the control at runtime to automatically “learn” and adapt to the system and reduce the feedback contribution.

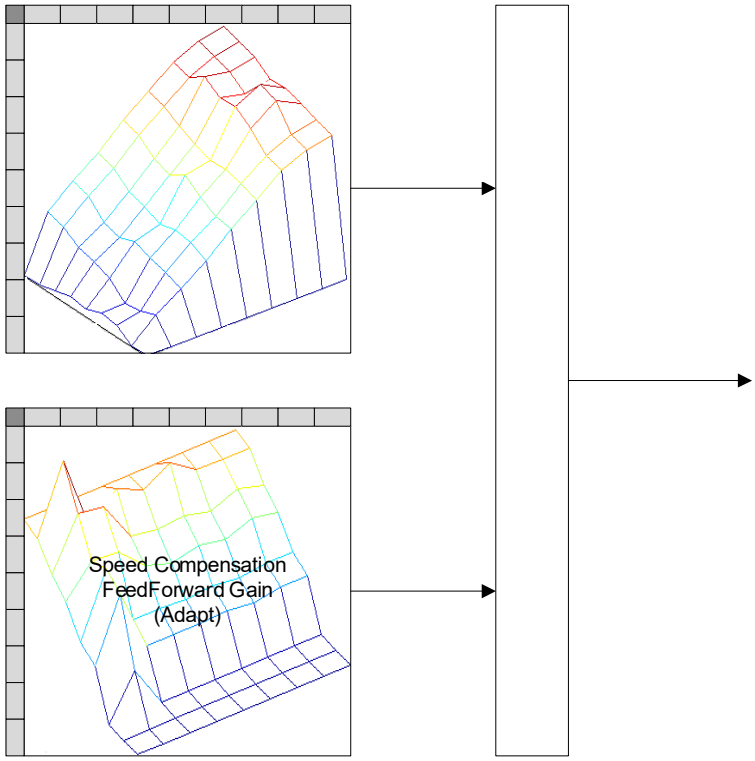


Figure 2-23. Feedforward Strategy

The speed compensation table can be thought of “gaining” or “sliding” the nominal table linearly up and down along a path for a given volume.

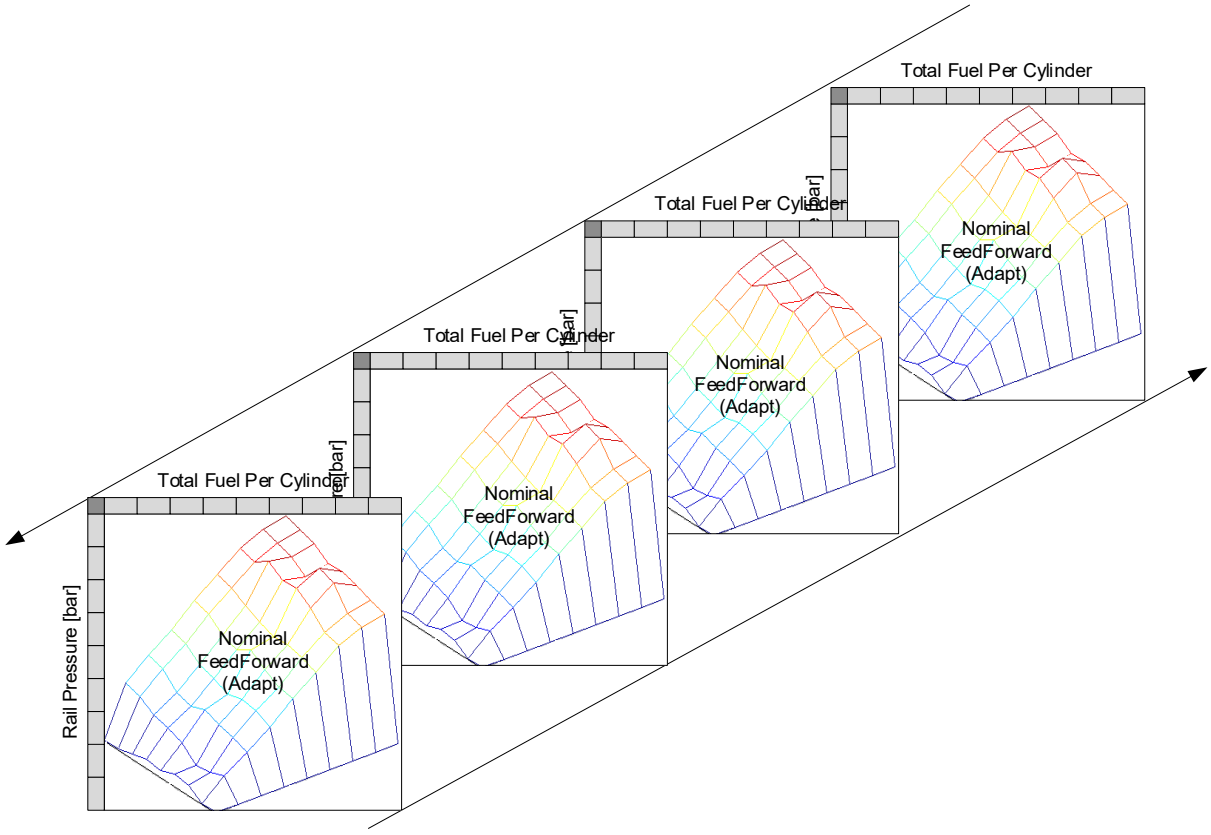


Figure 2-24. Feedforward Speed Compensation

← Setpoint

FeedForward Adapt →

Volume (mm3)

Setpoint [bar]	5.00	125.00	250.00	375.00	500.00	625.00	750.00	875.00	1000.00
0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
300.00	0.00	19.56	34.07	51.08	71.91	89.34	107.38	117.42	122.10
600.00	9.43	13.71	40.63	58.59	73.74	94.51	111.80	122.42	127.10
900.00	11.46	11.46	43.99	61.28	81.86	99.40	127.21	144.30	132.10
1200.00	5.67	25.38	49.78	57.29	93.30	97.72	138.55	149.30	137.10
1500.00	8.27	31.13	44.15	63.50	83.99	103.51	136.80	131.42	136.10
1800.00	4.11	35.26	49.49	69.00	89.69	111.51	147.22	142.42	147.10
2100.00	0.00	44.56	64.07	81.08	101.91	119.35	137.38	147.42	152.10
2400.00	0.00	49.56	69.07	86.08	106.91	124.35	142.38	152.42	157.10

Nominal FeedForward (cc/s)

Multiplier applied to nominal feedforward table as a function of speed and volume command.

Volume (mm3)

Speed [RPM]	5.00	125.00	250.00	375.00	500.00	625.00	750.00	875.00	1000.00
0.00	0.35	0.35	0.35	0.35	0.35	0.35	0.35	0.35	0.35
225.00	0.38	0.38	0.38	0.38	0.38	0.38	0.38	0.38	0.38
450.00	0.42	0.76	0.42	0.42	0.42	0.43	0.43	0.42	0.42
675.00	0.80	1.28	0.79	0.79	0.79	0.80	0.77	0.80	0.80
900.00	0.91	1.39	0.91	0.91	0.89	0.91	0.88	0.91	0.91
1125.00	1.14	1.38	1.35	1.16	1.09	1.15	1.10	1.18	1.14
1350.00	1.27	1.66	1.63	1.31	1.28	1.29	1.24	1.33	1.27
1575.00	1.48	2.00	1.38	1.60	1.51	1.63	1.48	1.48	1.48
1800.00	1.54	1.54	1.54	1.54	1.54	1.54	1.54	1.54	1.54

Speed Compensation (cc/s)

Figure 2-25. Feedforward Table Configurations

When feedforward adaption/learning is enabled, the control will only adapt the nominal table when speed is within a small window around normal operating speed; otherwise, the speed compensation table will be adapted.

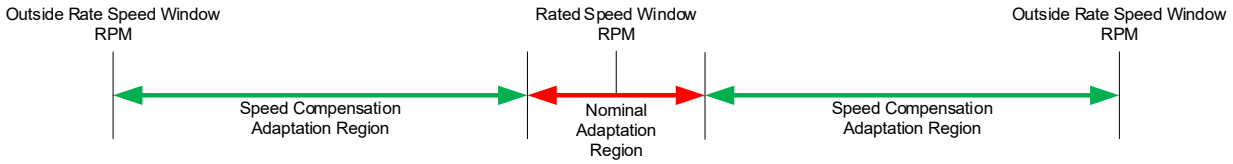


Figure 2-26. Feedforward Adaption Regions

The nominal feed forward table's adaption is controlled by the I-Term (integrator contribution) of the feedback PID controller multiplied by a gain. The direction of the adaption is the same as the I-Term, which drives it to a value of zero, and the rate of change is controlled by a V-shaped table that is indexed by the I-Term. The V-shaped table causes faster adaption the further the I-Term is away from zero and slower adaptation the closer the I-Term is to zero.

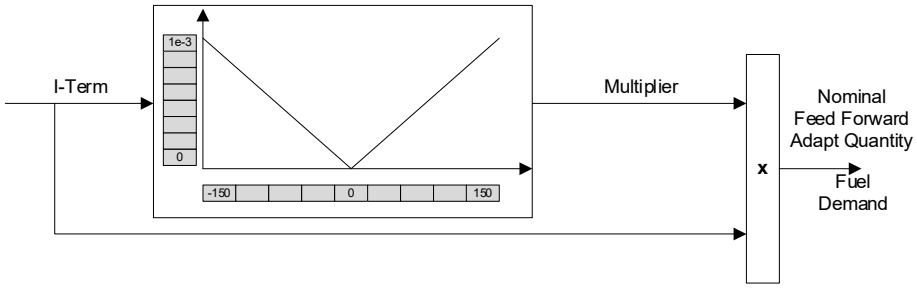


Figure 2-27. Feedforward Nominal Adaption

Nominal Table Adaption Factor

I-Term ()	Adapt Quantity ()
-150.000	0.001
-112.500	0.001
-75.000	0.001
-37.500	0.000
0.000	0.000
37.500	0.000
75.000	0.001
112.500	0.001
150.000	0.001

Apply a gain factor to the input I-Term for which to adjust the Nominal FeedForward table. The table should be shaped like a V with the I-Term of 0 being the center point. Larger gain factors can be applied to the furthest points of the table to adapt "faster;" while smaller gain factors can be applied near the center of the table (around 0 I-Term), in order to adapt slower.

Figure 2-28. Feedforward Nominal Adaption Configuration

The active position in the nominal feedforward table will be biased by the adaption quantity as illustrated in the figure below. For example, if the I-Term is zero, then no adjustment of the cell value is required; however, if the I-Term is -10 cc/s, the feedback controller is attempting to remove 10 cc/s flow command from the total, so the adaption quantity will be negative, and the feed forward table's cell value will decrease respectively (causing the I-Term to approach zero). Visually, this is equivalent to taking the point on the table and pulling it down vertically (reduce the feed forward table value so the I-Term approaches zero).

Note, the I-Term gain table's values illustrated in Figure 2-23 are generally small causing adaption to occur over long periods of time; however, during initial calibration, the gain values can be increased to obtain an initial coarse feed forward table and then reduced again to allow finer-grained changes to occur slowly over time.

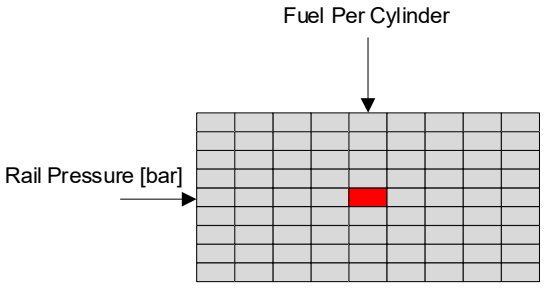


Figure 2-29. Nominal Feedforward Adaption Quantity

The following conditions must all be valid for the system to dynamically adapt the nominal feedforward table:

1. Adaption must be enabled.
2. Magnitude of feedback I-Term must be above a threshold.
3. Rail pressure must be above a threshold.
4. Fuel demand must be above a threshold.
5. Once nominal adaption window is *entered*, it must remain within the window for a period of time.
6. Rail pressure setpoint must be stable (within a +- window) for a period of time.

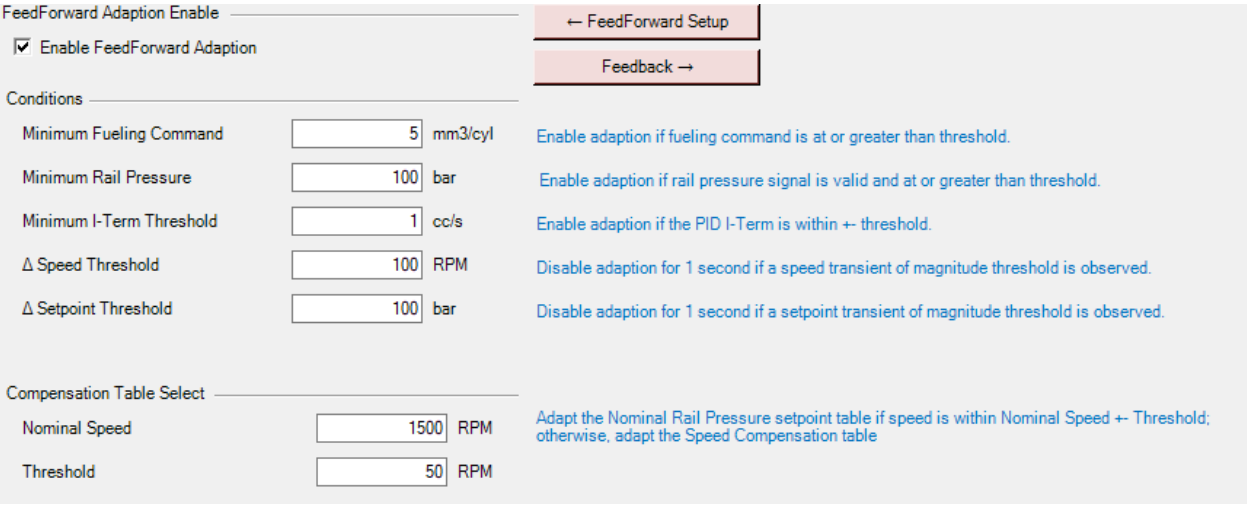


Figure 2-30. Feedforward Adaption Enable Condition Configurations

The second feed forward table is the 2-D "speed compensation" gain table, which applies a multiplier to the nominal feed forward gain table. Recall from above that for a given volume, this is logically moving the nominal feed forward table up and down along a path to compensate for engine speed (or pump speed). The input to the speed compensation table implements the following:

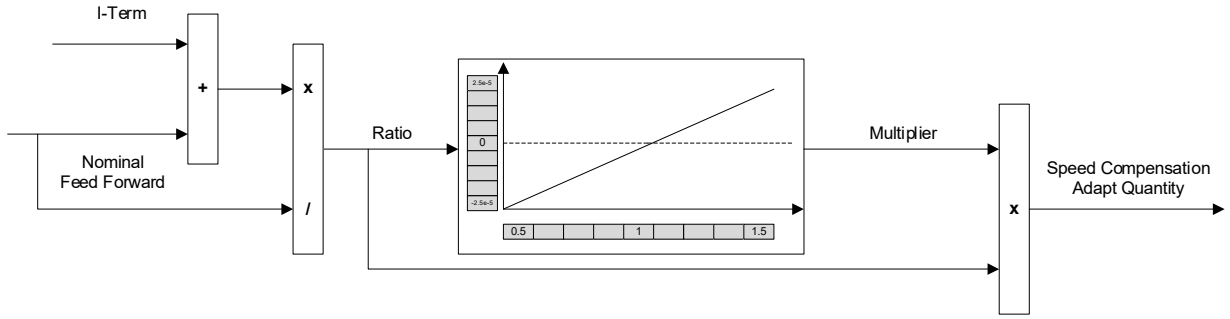


Figure 2-31. Speed Compensation Adaption

Speed Compensation Table Adaption Factor

Apply a gain factor to the input I-Term fraction for which to adjust the Speed Compensation table. The table should be linearly increasing (i.e. larger fraction equates to "faster" compensation).

Fraction ()	Adapt Quantity ()
0.000	0.000
0.250	0.000
0.500	0.000
0.750	0.000
1.000	0.000
1.250	0.000
1.500	0.000
1.750	0.000
2.000	0.000

Figure 2-32. Speed Compensation Adaption Configuration

This equation causes the speed compensation table gain to reduce when the I-Term is negative (reduction in flow required) and increase when the I-Term is positive (additional flow required).

The following conditions must all be valid for the system to dynamically adapt the speed compensation feedforward table:

1. Adaption must be enabled.
2. Magnitude of feedback I-Term must be above a threshold.
3. Rail pressure must be above a threshold.
4. Fuel demand must be above a threshold.
5. Once nominal adaption window is *exited*, it must remain outside the window for a period of time.
6. Rail pressure setpoint must be stable (within a +- window) for a period of time.

Note—the enable conditions are the same as for the nominal enable conditions; however, the engine speed condition is inverted (i.e., speed is outside the nominal window as illustrated in Figure 2-26).

Calibration Guide

A. Coarse Nominal Feed Forward Adaptation

- For the nominal feed forward table, run the engine at nominal speed and enable all conditions for adaption.
- Increase the rate of adaption by gaining the values of the adapt multiplier tables (this is not required but suggested for faster adaptation).
- For rail pressures [300 600 900 1200 1500 1800] (commonly used rows of the table), adjust total volume commanded per cylinder from the lowest column to the highest column and let the feed forward map learn until the I-Term is within +- 3 cc/s (coarse mapping).

B. Coarse Speed Compensation Feed Forward Adaption

- Repeat the procedure listed above for the speed compensation map by adjusting speed outside of the nominal speed range to fill in the table at different volume commands.

C. PID Tuning

- Perform any necessary PID tuning to maximize and balance control performance and robustness either by a structured or "by feel" tuning method.

Note—minor iterations of steps A-C may be required.

Feedback Strategy

The software feedback strategy utilizes a basic PID controller with gain scheduling for the P, I, and D-terms as a function of PID controller error as illustrated in the figure below. The gain schedules include 1-D lookup tables as a function of error and engine speed.

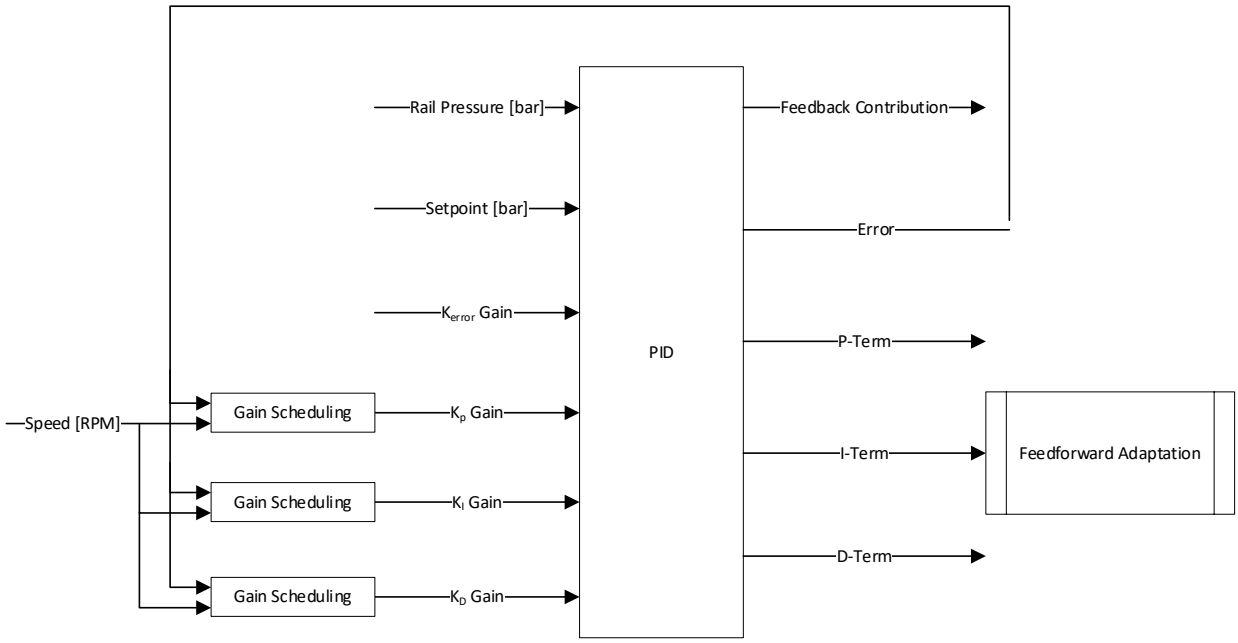


Figure 2-33. Feedback Strategy

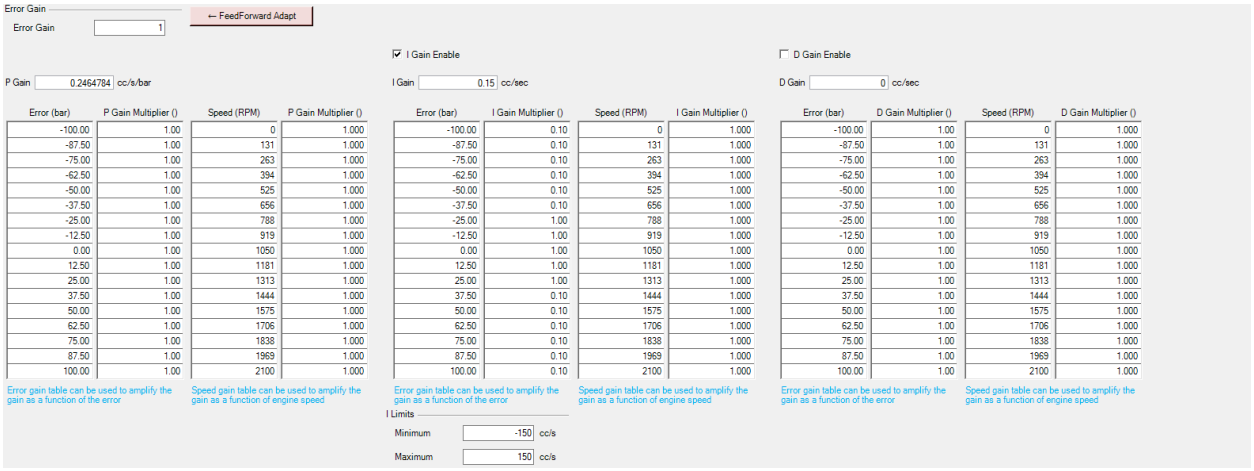


Figure 2-34. Feedback Configurations

$$P_{Gain} = P_{base} x\ P_{error} x\ P_{speed}$$

$$I_{Gain} = 0\ if\ enabled; otherwise, \{I_{base} x\ I_{error} x\ I_{speed}\}_{limited}$$

$$D_{Gain} = 0\ if\ enabled; otherwise\ (D_{base} x\ D_{error} x\ D_{speed})$$

Functional Description - Rail Pressure Diagnostics

The following describes the rail pressure diagnostics and monitors supported in the LECM CRS software.

Range Diagnostics & Monitors

For each rail pressure sensor, the software supports range low/high sensor diagnostics by monitoring the sensor hardware units (V or mA) and detecting if the sensor signal is continuously below or above a threshold for a fixed period of time. The software also supports range low/high monitors for the engineering units (bar) of the signal when the signal's corresponding diagnostic is considered valid (i.e., sensor is in range). Rail pressure range diagnostics and monitors utilize the same range diagnostics strategy as described in LECM CRS Diagnostics & Monitoring document.

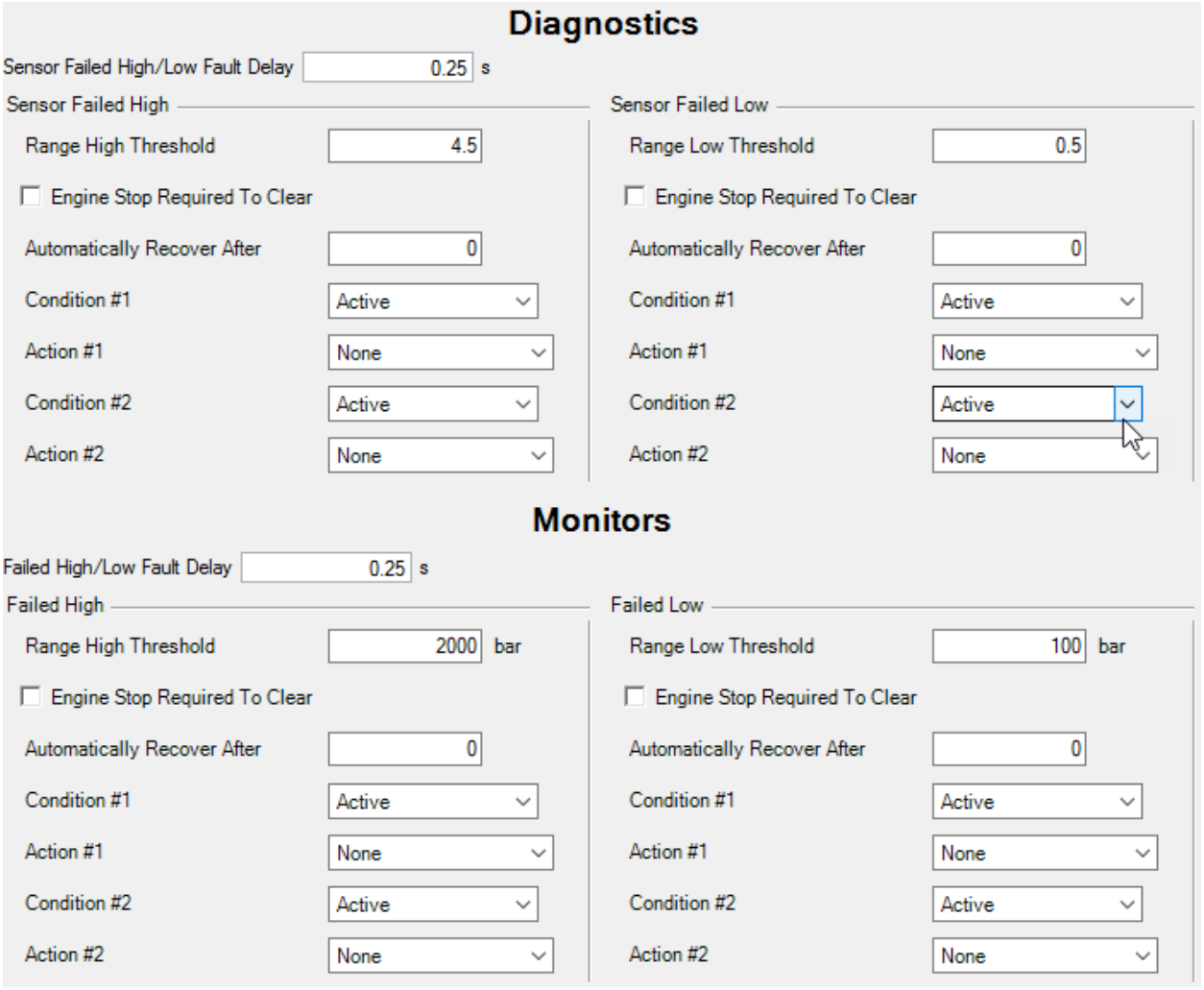


Figure 2-35. Rail Pressure Range Diagnostic & Monitor Configurations

Zero Rail Pressure Monitor

The software supports an optional zero rail pressure monitor that asserts a fault if the rail pressure is at or below the ambient air pressure while the engine is in a run state. This fault can be used to drive fault actions if rail pressure sensing is inoperable.

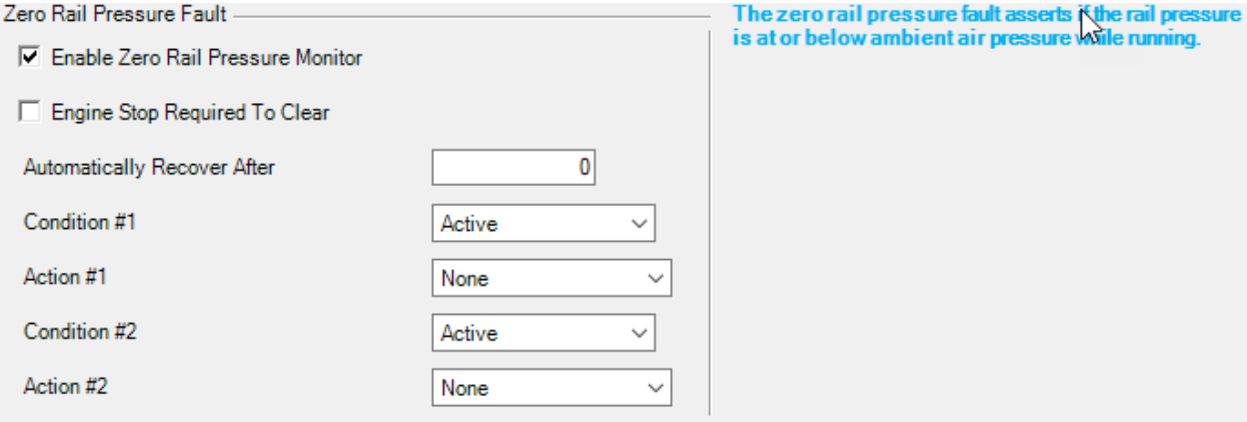


Figure 2-36. Zero Rail Pressure Monitor Configurations

Rail Pressure Control Monitor

The software supports an optional rail pressure control monitor that asserts a fault if a deviation in the actual rail pressure as a percentage of the setpoint continuously exceeds a limit for a fixed period of time. The monitor is only active when enabled and when rail pressure is above a threshold. This fault is used to confirm the rail pressure control is stable within user-defined expectations.

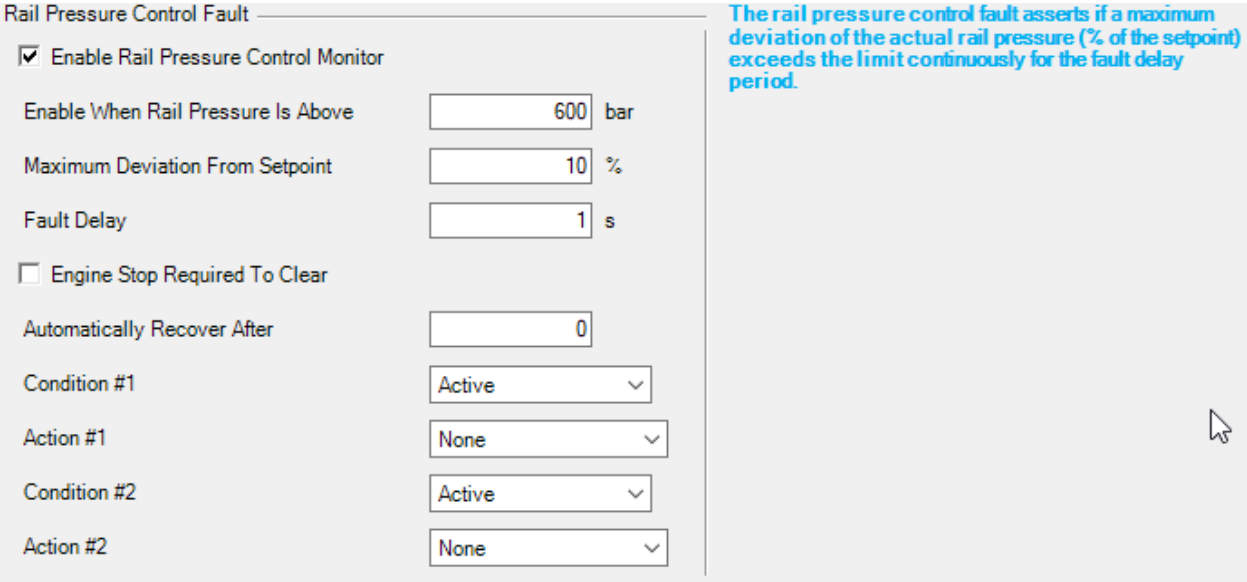


Figure 2-37. Rail Pressure Control Monitor Configurations

Rail Pressure Sensor Deviation Monitor

For systems with redundant rail pressure sensors, the software supports an optional rail pressure sensor deviation monitor that asserts if there is a maximum pressure differential between the two sensors for a fixed period of time. The monitor is only active when enabled and when multiple rail pressure sensors are available. This fault is used to confirm the rail pressure readings are consistent within a window between the two sensors.

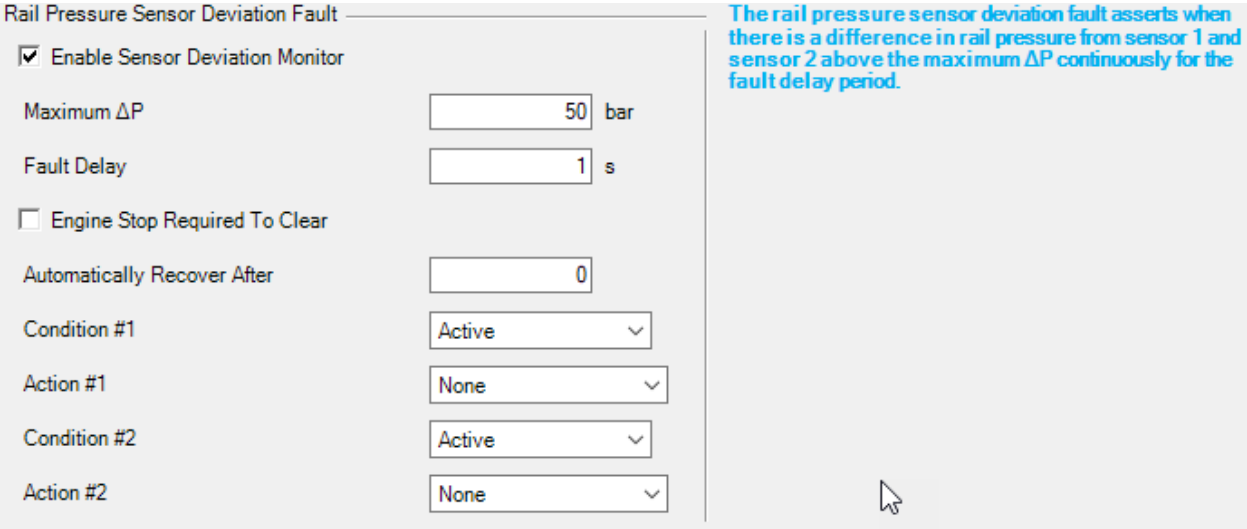


Figure 2-38. Rail Pressure Sensor Deviation Monitor Configurations

Chapter 3.
Control - WLO Fuel Metering Valve

Introduction

The ‘Control - WLO Fuel Metering Valve’ blockset is used to control the Woodward L’Orange Fuel Metering valves (FMV’s) and the corresponding Diagnostics Monitoring. A Dual Pump Fuel Distributor blockset is available when an application requires 2 pumps to supply enough flow for the engine to reach full load. Inside the blockset, four blocks are available as shown below.

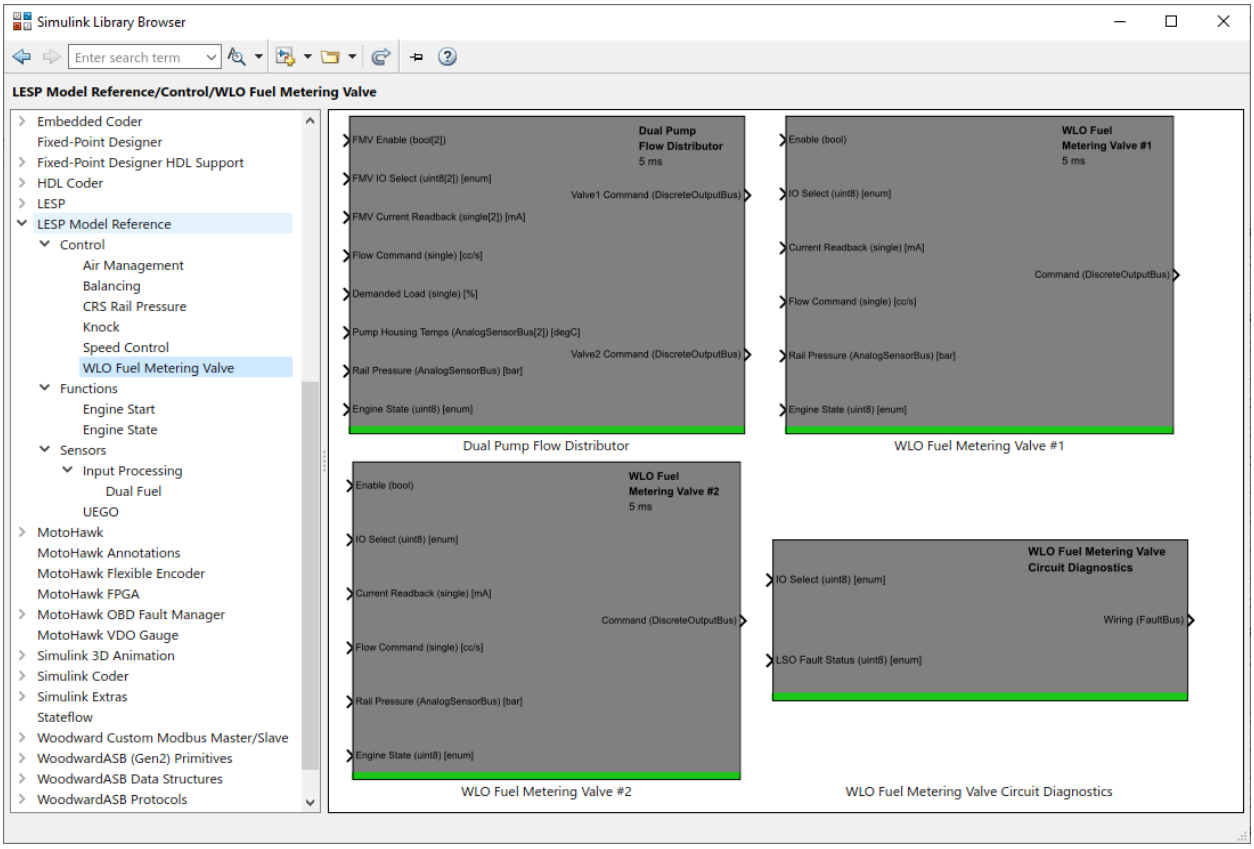


Figure 3-1. LESP Model Reference ‘WLO Fuel Metering Valve’ Blocks

- Dual Pump Flow Distributor
  This blockset is used when two High Pressure Fuel Pumps are used in the application. It provides certain strategies to share between both pumps the equal amount of flow.

- WLO Fuel Metering Valve #1
  This blockset is used to control the fuel metering valve #1 inside the application.

- WLO Fuel Metering Valve #2
  This blockset is used to control the fuel metering valve #2 inside the application.

- WLO Fuel Metering Valve Circuit Diagnostics
  This blockset is used to provide diagnostic information on the outputs used to drive the fuel metering valves of the high pressure common rail pumps.

Dual Pump Flow Distributor Block Interface

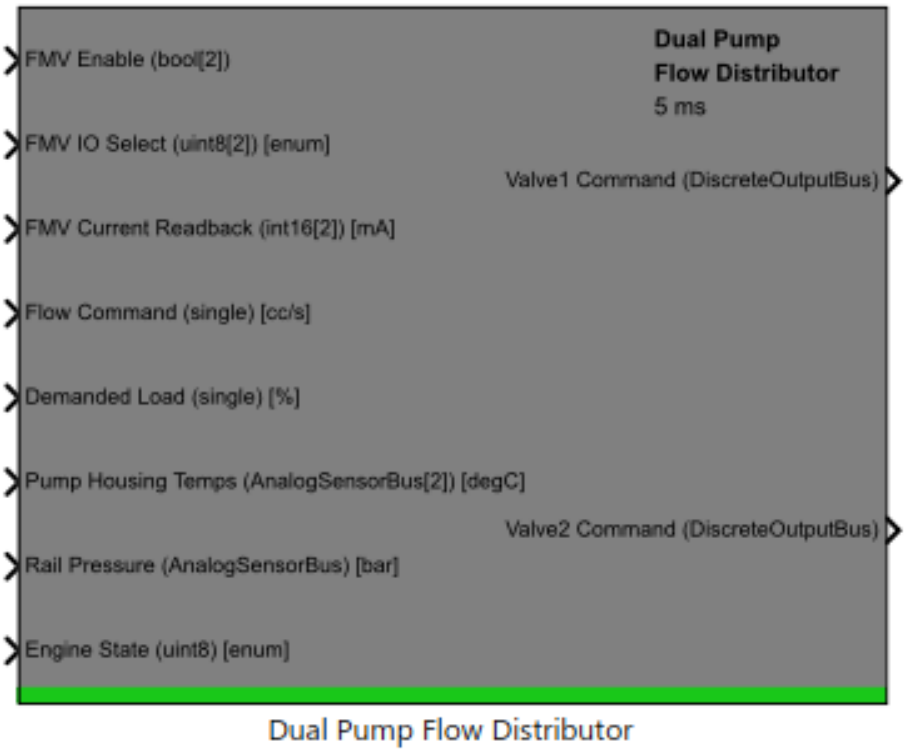


Figure 3-2. ‘Dual Pump Flow Distributor’ Block

Table 3-1. ‘CRS Rail Pressure Diagnostic Monitors’ Block

Port	DataType	Description
[In] FMV Enable [2]	boolean	These are the enable inputs of both the pumps, so it requires double values
[In] FMV IO Select [enum] [2]	uint8	This field is used to select the correct IO on the hardware of the selected target. It requires 2 x the inputs as for pump 1 and 2. IOSelect_enum (lecm_discrete_out_io_select_enum) 0 – Not Used 1 – LSO1 2 – LSO2 3 – LSO3 4 – LSO4 5 – LSO5 6 – LSO6 7 – LSO7 8 – LSO8 9 – LSO9 10 – LSO10 11 – LSO11 12 – LSO12 13 – HSO1 14 – HSO2 15 – HSO3 16 – HSO4 17 – HSO5

		18 – HSO6 19 – HSO7 20 – HSO8 21 – IgnTrig1 22 – IgnTrig2 23 – IgnTrig3 24 – IgnTrig4
[In] FMV Current Readback [mA] [2]	Int16	The Fuel Metering Valve current readback in mA as read from the hardware. These are 2 x the current readback values as for Pump 1 and 2.
[In] Flow Command [cc/s]	single	The required flow as commanded from the application. This is total flow, so in case of 2 pumps, it is for both.
[In] Demanded Load [%]	single	The actual demanded load in the application at that moment.
[In] Pump Housing Temps [degC] [2]	AnalogSensorBus	The actual temperatures of the Pump Housings (in degrees Celsius), which can be used for flow distribution logic to split flow for 2 pumps. These inputs require 2 x the values as for pump 1 and for pump 2. Bus Details: AnalogSensorBus: - FilteredValue - MeasurementStatus_enum MeasurementStatus_enum: (lesp_measurement_status_enum) 0 – Valid 1 – Disabled 2 – SignalLow 3 – SignalHigh 4 – SensorSupplyFailed 5 – ReferenceSourceFailed 6 – OpenCircuit
[In] Rail Pressure [bar]	AnalogSensorBus	The actual Rail Pressure (in bar) that the application has measured. Bus Details: AnalogSensorBus: - FilteredValue - MeasurementStatus_enum MeasurementStatus_enum: (lesp_measurement_status_enum) 0 – Valid 1 – Disabled 2 – SignalLow 3 – SignalHigh 4 – SensorSupplyFailed 5 – ReferenceSourceFailed 6 – OpenCircuit
[In] Engine State [enum]	uint8	The actual engine state the engine is operating in at that particular moment. These states (lesp_engine_state_enum) are defined as below: 0 – Stopped 1 – Prestart 2 – Starting 3 – Warmup

		4 – Running 5 – Cooldown 6 – Stopping 7 – Postrun
[Out] Valve1 Command	DiscreteOutputBus	The Fuel Metering Valve #1 command output, which can be used to the correct output FMV logic. DiscreteOutputBus: - IOSelect_enum - BehaviorSelect_enum - State_enum - DutyCycle_Pct - Frequency_Hz IOSelect_enum (lecm_discrete_out_io_select_enum) 0 – Not Used 1 – LSO1 2 – LSO2 3 – LSO3 4 – LSO4 5 – LSO5 6 – LSO6 7 – LSO7 8 – LSO8 9 – LSO9 10 – LSO10 11 – LSO11 12 – LSO12 13 – HSO1 14 – HSO2 15 – HSO3 16 – HSO4 17 – HSO5 18 – HSO6 19 – HSO7 20 – HSO8 21 – IgnTrig1 22 – IgnTrig2 23 – IgnTrig3 24 – IgnTrig4 BehaviorSelect_enum (lesp_discrete_out_behavior_select_enum): 0 – Discrete 1 - PWM State_enum (lesp_discrete_out_enum): 0 – Not Used 1 – Inactive 2 – Active
[Out] Valve2 Command	DiscreteOutputBus	The Fuel Metering Valve #2 command output, which can be used to the correct output FMV logic. DiscreteOutputBus: - IOSelect_enum - BehaviorSelect_enum

		- State_enum - DutyCycle_Pct - Frequency_Hz IOSelect_enum (lecm_discrete_out_io_select_enum) 0 – Not Used 1 – LSO1 2 – LSO2 3 – LSO3 4 – LSO4 5 – LSO5 6 – LSO6 7 – LSO7 8 – LSO8 9 – LSO9 10 – LSO10 11 – LSO11 12 – LSO12 13 – HSO1 14 – HSO2 15 – HSO3 16 – HSO4 17 – HSO5 18 – HSO6 19 – HSO7 20 – HSO8 21 – IgnTrig1 22 – IgnTrig2 23 – IgnTrig3 24 – IgnTrig4 BehaviorSelect_enum ((lesp_discrete_out_behavior_select_enum): 0 – Discrete 1 - PWM State_enum (lesp_discrete_out_enum): 0 – Not Used 1 – Inactive 2 – Active

WLO Fuel Metering Valve #1 Block Interface

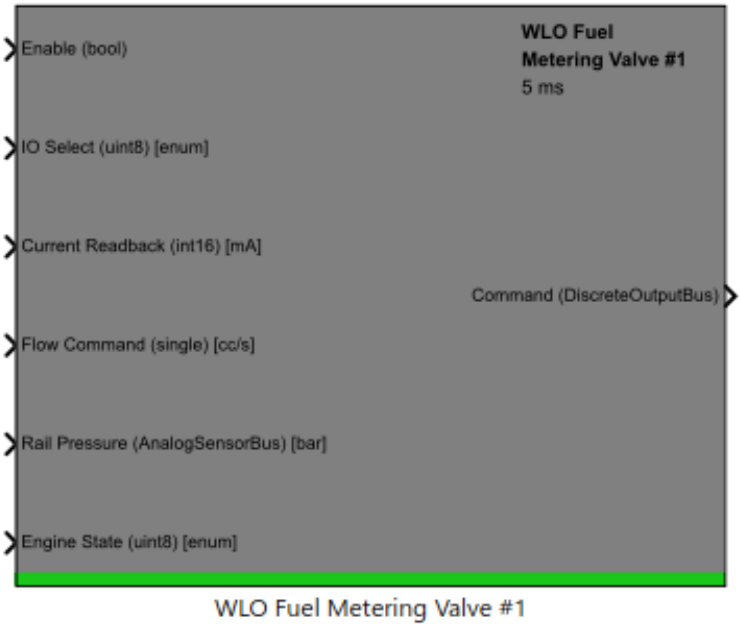


Figure 3-3. 'WLO Fuel Metering Valve #1' Block

Table 3-2. 'WLO Fuel Metering Valve #1' Block

Port	DataType	Description
[In] Enable	boolean	The enable input of the FMV of the Pump #1.
[In] IO Select [enum]	uint8	This field is used to select the correct IO on the hardware of the selected target for the FMV of pump #1. IOSelect_enum (lecm_discrete_out_io_select_enum) 0 – Not Used 1 – LSO1 2 – LSO2 3 – LSO3 4 – LSO4 5 – LSO5 6 – LSO6 7 – LSO7 8 – LSO8 9 – LSO9 10 – LSO10 11 – LSO11 12 – LSO12 13 – HSO1 14 – HSO2 15 – HSO3 16 – HSO4 17 – HSO5 18 – HSO6 19 – HSO7 20 – HSO8 21 – IgnTrig1

		22 – IgnTrig2 23 – IgnTrig3 24 – IgnTrig4
[In] Current Readback [mA]	Int16	The Fuel Metering Valve current readback in mA as read from the hardware for FMV of pump #1.
[In] Flow Command [cc/s]	single	The required flow as commanded from the application for pump #1.
[In] Rail Pressure [bar]	AnalogSensorBus	The actual Rail Pressure (in bar) that the application has measured. Bus Details: AnalogSensorBus: - FilteredValue - MeasurementStatus_enum MeasurementStatus_enum: (lesp_measurement_status_enum) 0 – Valid 1 – Disabled 2 – SignalLow 3 – SignalHigh 4 – SensorSupplyFailed 5 – ReferenceSourceFailed 6 – OpenCircuit
[In] Engine State [enum]	uint8	The actual engine state the engine is operating in on that particular moment. These states (lesp_engine_state_enum) are defined as below: 0 – Stopped 1 – Prestart 2 – Starting 3 – Warmup 4 – Running 5 – Cooldown 6 – Stopping 7 – Postrun
[Out] Command	DiscreteOutputBus	The actual Command Output signal going to the Fuel Control Valve. DiscreteOutputBus: - IOSelect_enum - BehaviorSelect_enum - State_enum - DutyCycle_Pct - Frequency_Hz IOSelect_enum (lecm_discrete_out_io_select_enum) 0 – Not Used 1 – LSO1 2 – LSO2 3 – LSO3 4 – LSO4 5 – LSO5 6 – LSO6 7 – LSO7 8 – LSO8 9 – LSO9

		10 – LSO10 11 – LSO11 12 – LSO12 13 – HSO1 14 – HSO2 15 – HSO3 16 – HSO4 17 – HSO5 18 – HSO6 19 – HSO7 20 – HSO8 21 – IgnTrig1 22 – IgnTrig2 23 – IgnTrig3 24 – IgnTrig4 BehaviorSelect_enum ((lesp_discrete_out_behavior_select_enum): 0 – Discrete 1 - PWM State_enum (lesp_discrete_out_enum): 0 – Not Used 1 – Inactive 2 – Active

WLO Fuel Metering Valve #2 Block Interface

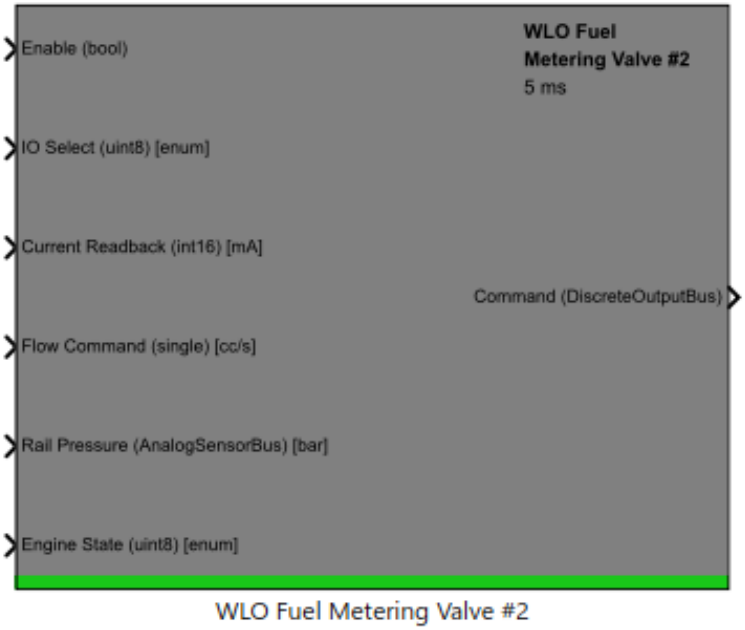


Figure 3-4. 'WLO Fuel Metering Valve #2' Block

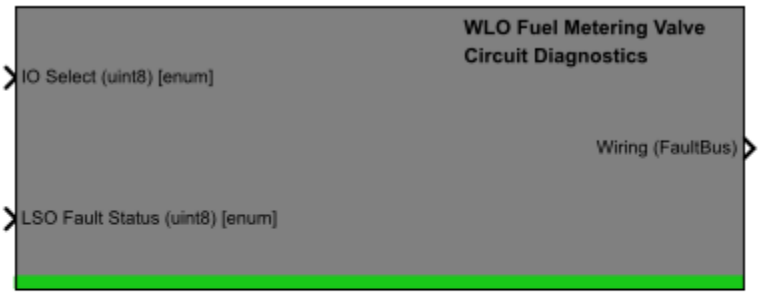
Table 3-3. 'WLO Fuel Metering Valve #2' Block

Port	DataType	Description
[In] Enable	boolean	The enable input of the FMV of the Pump #2.
[In] IO Select [enum]	uint8	This field is used to select the correct IO on the hardware of the selected target for the FMV of pump #2. IOSelect_enum (lecm_discrete_out_io_select_enum) 0 – Not Used 1 – LSO1 2 – LSO2 3 – LSO3 4 – LSO4 5 – LSO5 6 – LSO6 7 – LSO7 8 – LSO8 9 – LSO9 10 – LSO10 11 – LSO11 12 – LSO12 13 – HSO1 14 – HSO2 15 – HSO3 16 – HSO4 17 – HSO5 18 – HSO6 19 – HSO7 20 – HSO8 21 – IgnTrig1

		22 – IgnTrig2 23 – IgnTrig3 24 – IgnTrig4
[In] Current Readback [mA]	Int16	The Fuel Metering Valve current readback in mA as read from the hardware for FMV of pump #2.
[In] Flow Command [cc/s]	single	The required flow as commanded from the application for pump #2.
[In] Rail Pressure [bar]	AnalogSensorBus	The actual Rail Pressure (in bar) that the application has measured. Bus Details: AnalogSensorBus: - FilteredValue - MeasurementStatus_enum MeasurementStatus_enum: (lesp_measurement_status_enum) 0 – Valid 1 – Disabled 2 – SignalLow 3 – SignalHigh 4 – SensorSupplyFailed 5 – ReferenceSourceFailed 6 – OpenCircuit
[In] Engine State [enum]	uint8	The actual engine state the engine is operating in at that particular moment. These states (lesp_engine_state_enum) are defined as below: 0 – Stopped 1 – Prestart 2 – Starting 3 – Warmup 4 – Running 5 – Cooldown 6 – Stopping 7 – Postrun
[Out] Command	DiscreteOutputBus	The actual Command Output signal going to the Fuel Control Valve. DiscreteOutputBus: - IOSelect_enum - BehaviorSelect_enum - State_enum - DutyCycle_Pct - Frequency_Hz IOSelect_enum (lecm_discrete_out_io_select_enum) 0 – Not Used 1 – LSO1 2 – LSO2 3 – LSO3 4 – LSO4 5 – LSO5 6 – LSO6 7 – LSO7 8 – LSO8 9 – LSO9

		10 – LSO10 11 – LSO11 12 – LSO12 13 – HSO1 14 – HSO2 15 – HSO3 16 – HSO4 17 – HSO5 18 – HSO6 19 – HSO7 20 – HSO8 21 – IgnTrig1 22 – IgnTrig2 23 – IgnTrig3 24 – IgnTrig4 BehaviorSelect_enum ((lesp_discrete_out_behavior_select_enum): 0 – Discrete 1 - PWM State_enum (lesp_discrete_out_enum): 0 – Not Used 1 – Inactive 2 – Active

WLO Fuel Metering Valve Circuit Diagnostics Block Interface



WLO Fuel Metering Valve Circuit Diagnostics

Figure 3-5. 'WLO Fuel Metering Valve Circuit Diagnostics' Block

Table 3-4. 'WLO Fuel Metering Valve #2' Block

Port	DataType	Description
[In] IO Select [enum]	uint8	This field is used to select the correct IO on the hardware of the selected target for the FMV pump. IO Select should match the corresponding IO mapping for the respective valve. IOSelect_enum (lecm_discrete_out_io_select_enum) 0 – Not Used 1 – LSO1 2 – LSO2 3 – LSO3 4 – LSO4 5 – LSO5 6 – LSO6 7 – LSO7 8 – LSO8 9 – LSO9 10 – LSO10 11 – LSO11 12 – LSO12 13 – HSO1 14 – HSO2 15 – HSO3 16 – HSO4 17 – HSO5 18 – HSO6 19 – HSO7 20 – HSO8 21 – IgnTrig1 22 – IgnTrig2 23 – IgnTrig3 24 – IgnTrig4

[In] LSO Fault Status [enum]	Uint8	The actual status input of the LSO output used to drive the FMV for the pump. fault_status_enum (lesp_discrete_out_fault_status_enum): 0 – Valid 1 – ShortCircuitToBattery 2 – OpenOrShortToGround 3 – OpenOrShortToBattery 4 – PWMDutyCycleForcedToLimit 5 – CurrentMeasurementOverrun 6 – CurrentMeasurementSaturated 7 - OverCurrent
[Out] Wiring	FaultBus	Diagnostics info in form of FaultBus FaultBus: - AssertionStatus - ConditionStatus - InhibitStatus

Example Model Using the Blocks

The following is a very simple setup using MotoHawk Calibration and MotoHawk Probes blocks to make the model suitable for building.

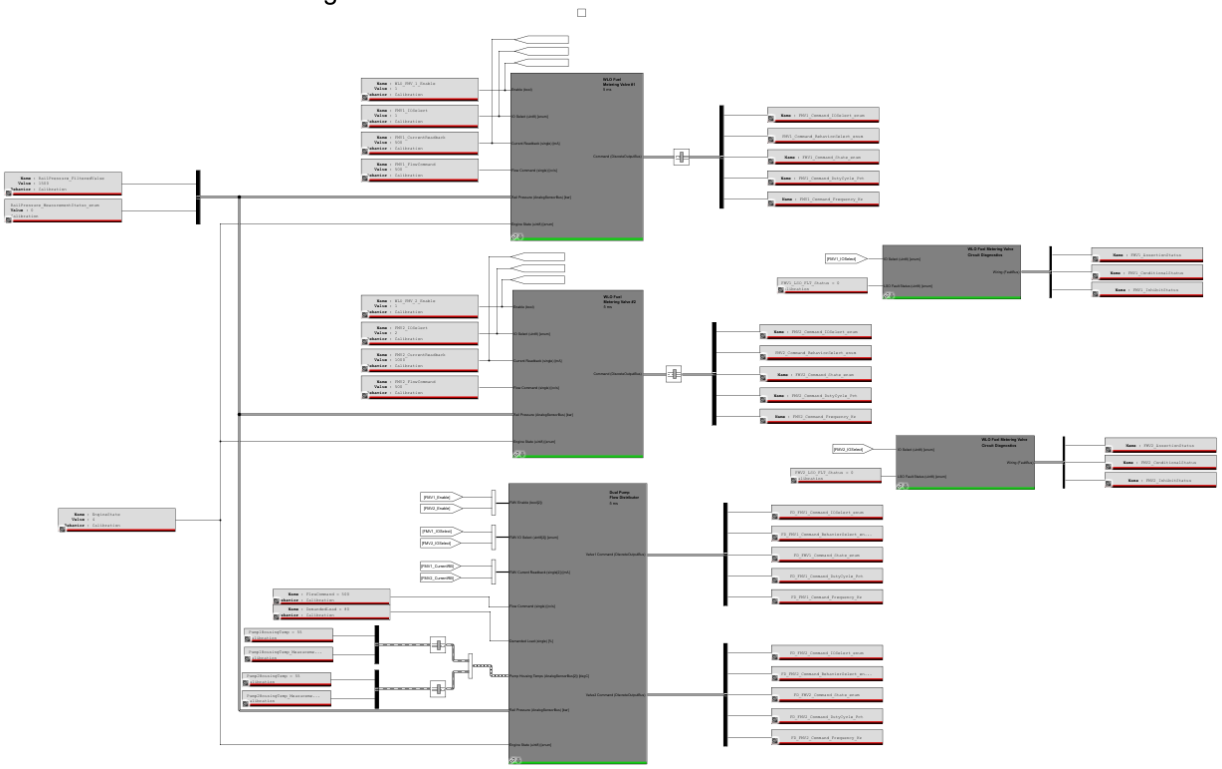


Figure 3-6. Simple Example Model, Using the Control- WLO Fuel Metering Valves Blocks

When compiling, it will generate the required files, and if the Toolkit required blocks are also added to the application model, it will generate the required SID files for the Toolkit HMI tool creation.

Because the example model contains all the blocks for the fuel metering valves, a decision needs to be made whether to use the “Flow Distributor” block or the individual “Fuel Metering Valves” blocks. This can

be done easily by the ‘Comment out’ function in Simulink. When leaving all blocks in, it will fail to create a successful build. See the figure below for possible options.

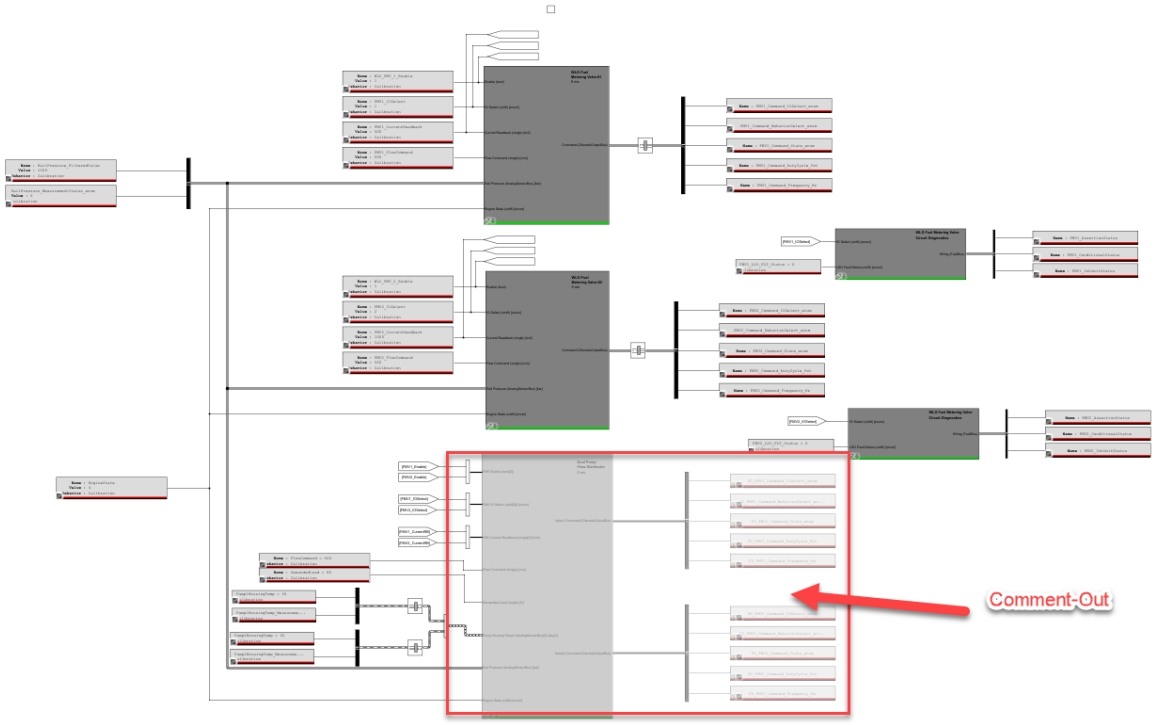


Figure 3-7. Individual FMV’s with ‘Flow Distributor Blocks’ Commented Out

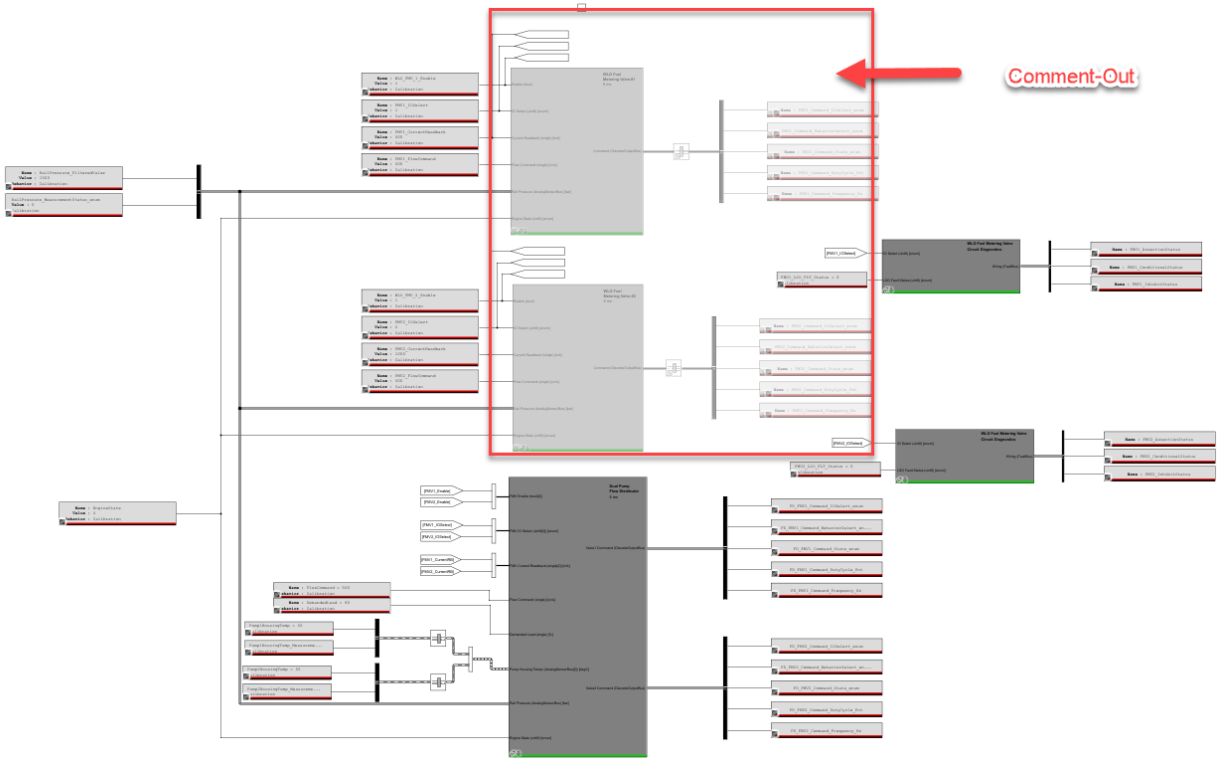


Figure 3-8. ‘Flow Distributor Block’ Active with Individual ‘FMV Blocks’ Commented Out

Default ToolKit Pages

When opening a new Toolkit application and selecting the correctly generated .sid file, the page will look like the screenshot below.

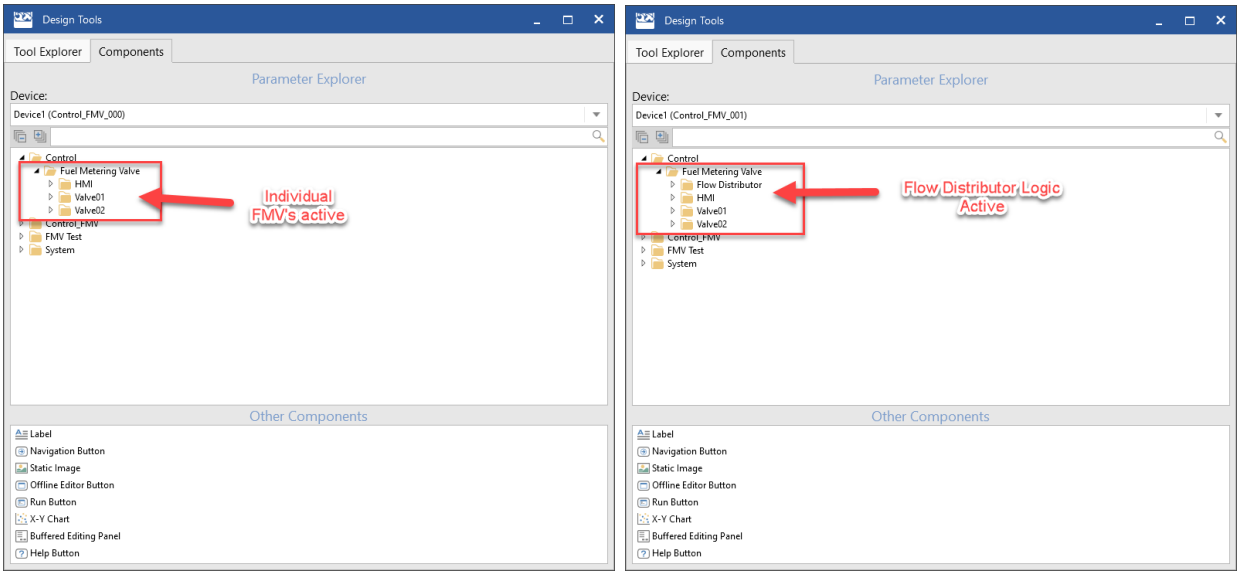
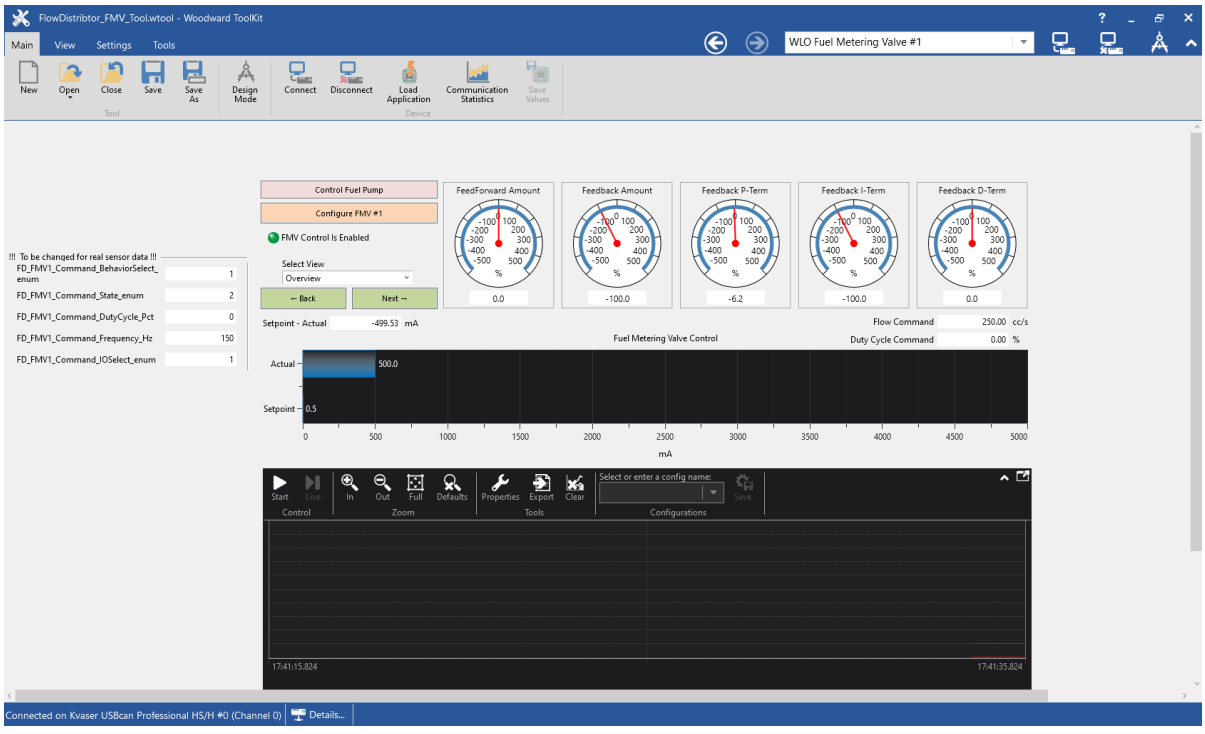
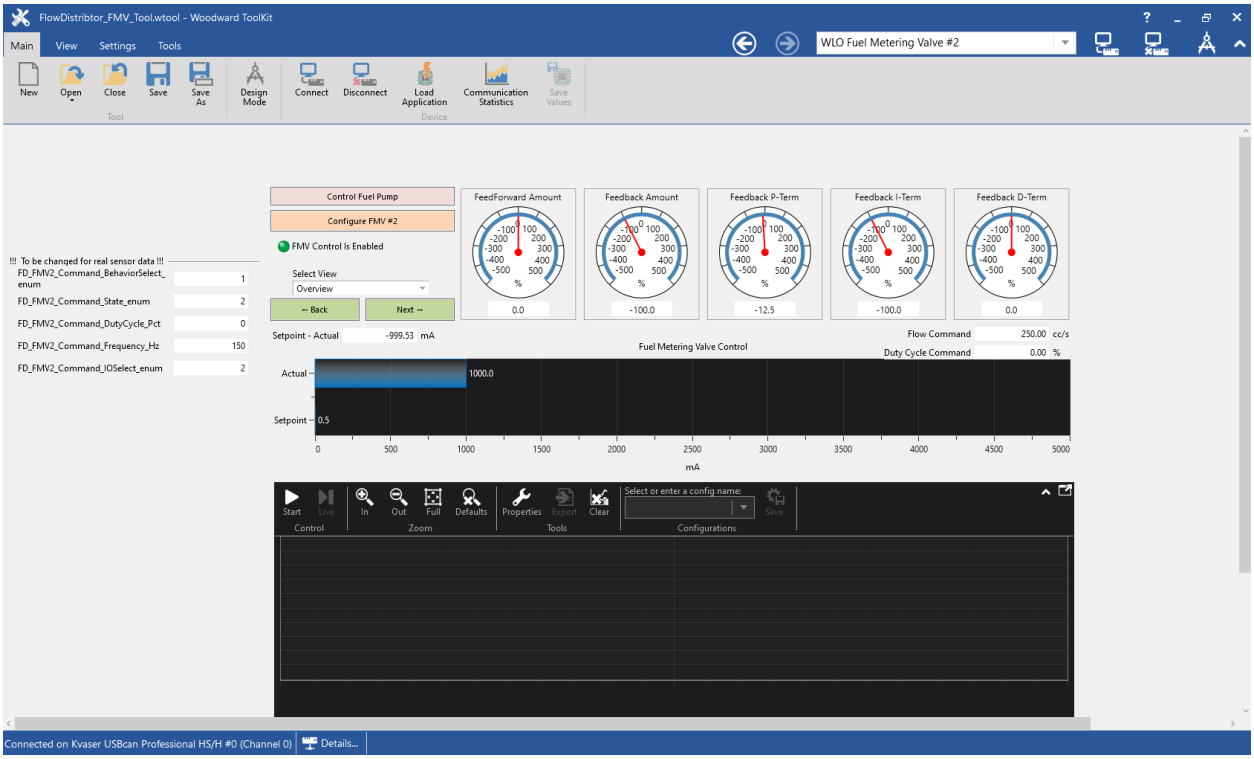


Figure 3-9. SID File Selection and Layout for Both Options

The next description will focus on having double FMV's defined, as this will be the most extensive use, having most options included.

Each FMV has one page, including offline pages, to setup the FMV according to factory specification.





The Pump Flow Distributor logic requires real sensor feedback to work properly. Options for balancing the flow can be selected from this page.

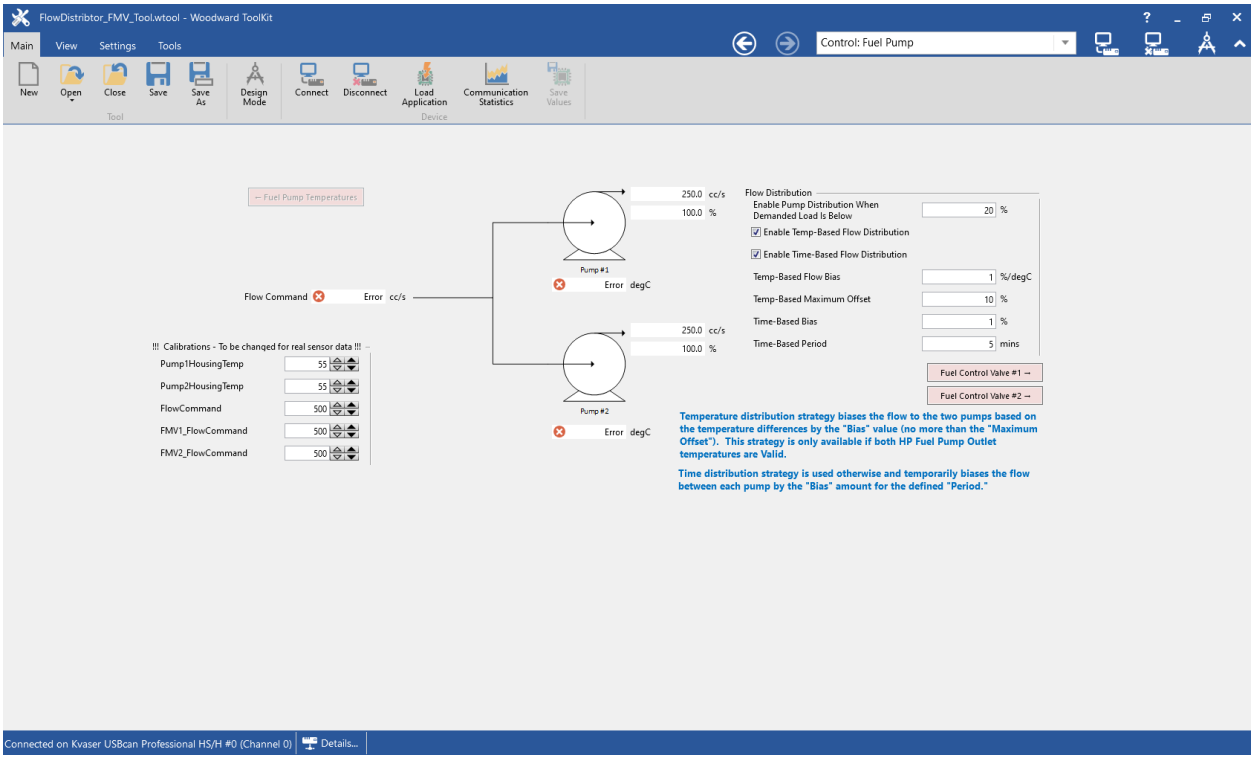


Figure 3-10. Examples of the Toolkit Pages for FMV Control

This example Toolkit file is in the ‘Tools’ directory of the “Control_FMV” example model and can be used as a base setup for the application specific service tool.

Functional Description - Fuel Control Valve

The following describes the fuel control valve (FCV, AKA fuel metering valve) strategy. The FCV control strategy is intended for a Woodward L'Orange current controlled high pressure fuel metering valve. The software supports up to two control valves for dual pump systems commanding a common rail with a flow distribution algorithm to ensure fuel pumps maintain and operate with similar temperature conditions.

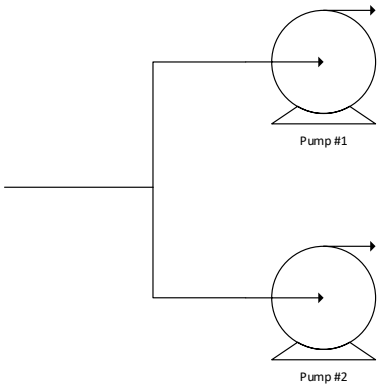


Figure 3-11. Dual Fuel Pumps

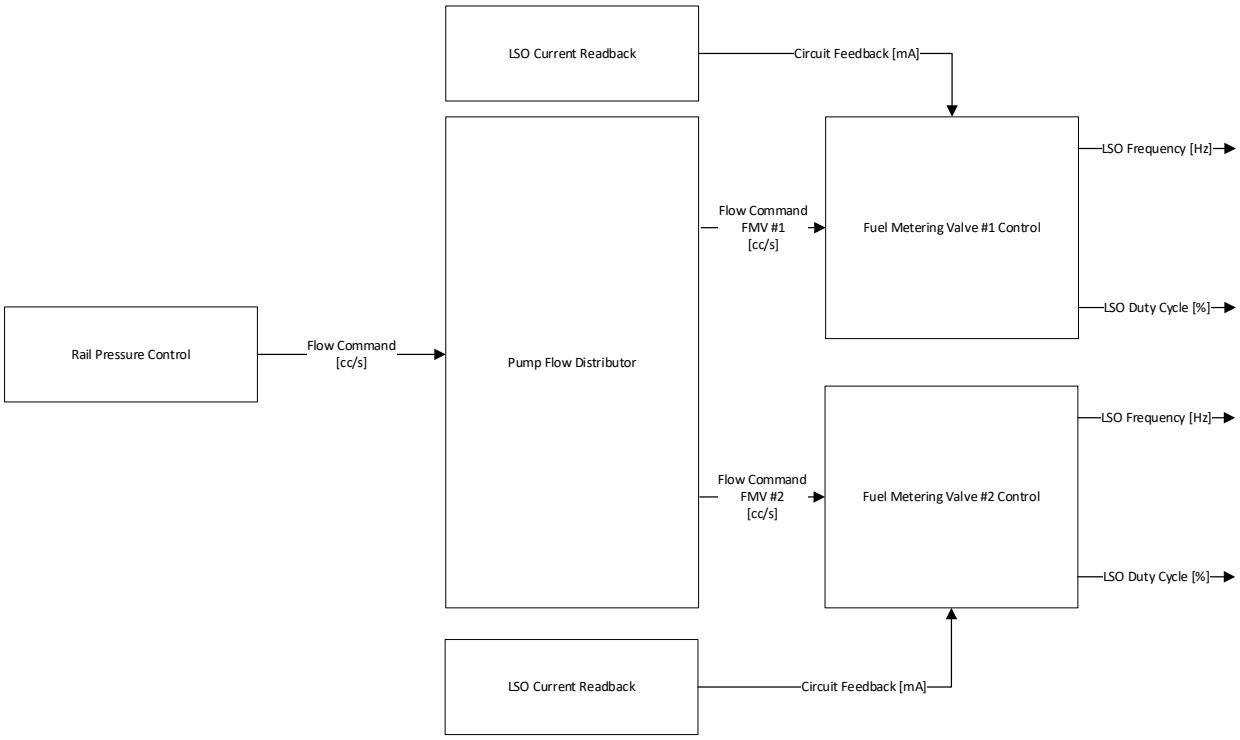


Figure 3-12. Fuel Control Valve

Pump Flow Distributor

The pump flow distribution logic is used to direct and distribute the rail pressure flow command to one or more of the available fuel pumps (control valves) using pump temperatures and/or a time-based strategy to modify the flow to each pump in order to keep the pump temperatures balanced. When a single pump is used, the rail pressure command is fed directly to the respective pump flow command (i.e., no distribution).

Temperature-Based Strategy

The temperature-based strategy utilizes pump housing temperatures and biases the flow of pump 1 and pump 2 by integrating the difference in housing temperatures relative to the average temperatures multiplied by a gain. This strategy will effectively distribute more flow to the hotter pump relative to the cooler pump.

Note: More flow is distributed to the hotter pump as high flow results in a larger cooling effect.

Time-Based Strategy

If pump housing temperatures are not available, a time-based strategy can be enabled to continuously distribute flow between two pumps, cycling the flow amount linearly over time. This strategy is illustrated in the figure below.

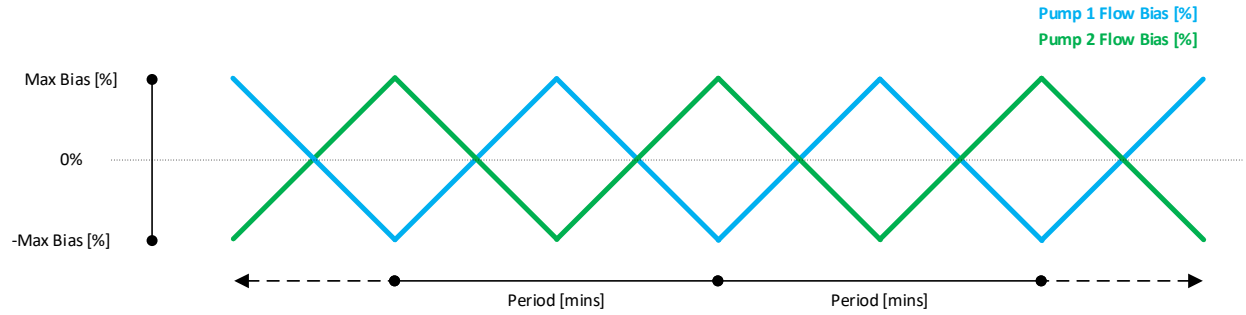


Figure 3-13. Time-Based Pump Flow Distribution Strategy

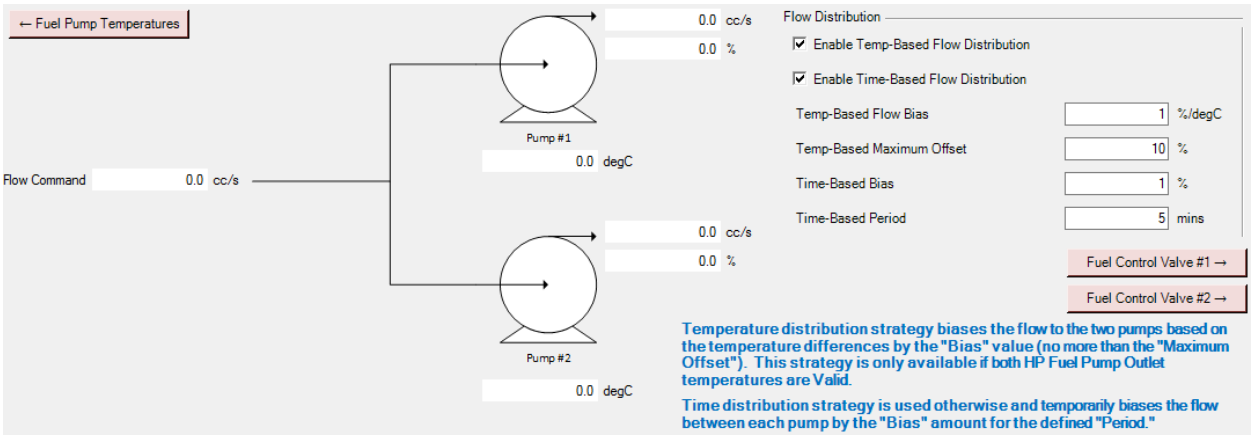


Figure 3-14. Fuel Pump Distribution Configurations

Fuel Control Valve

Each fuel metering valve indirectly controls the fuel flow via current by adjusting the duty cycle of a low-side drive at a fixed frequency. The control strategy consists of a fixed feedforward table and PID feedback to close the loop on current.

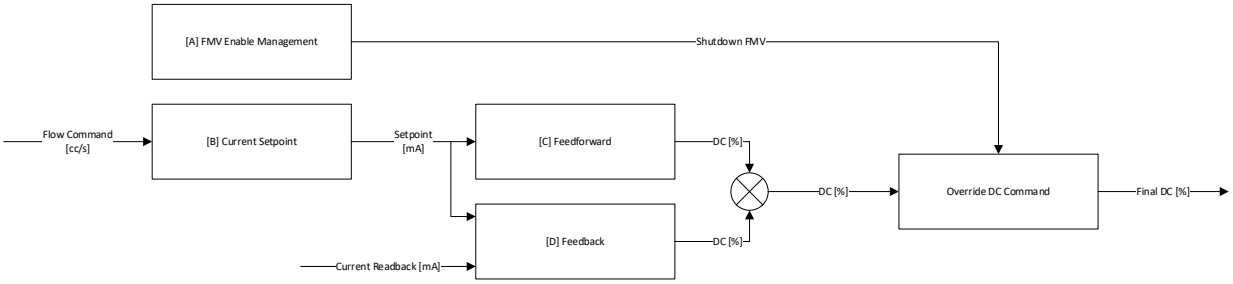


Figure 3-15. Fuel Control Valve Strategy

IMPORTANT

The LECM hardware only supports current readback on LSO1 – LSO4, requiring only these output drives to be used for the pump control valves.

Enable Management and Setup

The fuel control valve can be mapped to any LSO with current feedback. Current readback must be enabled for the corresponding LSO and the current measurement average periods should be set to a value that supports updates at a minimum rate of 2x the driving frequency. For example, if the drive frequency is 150 Hz, the PWM average period should be set to a value of 2 or less.

The ‘Apply Duty Cycle Correction’ checkbox must stay unchecked as the FCV is a highly inductive load. It also requires a diode to be placed in the wiring harness, preferably as close as possible to the FCV.

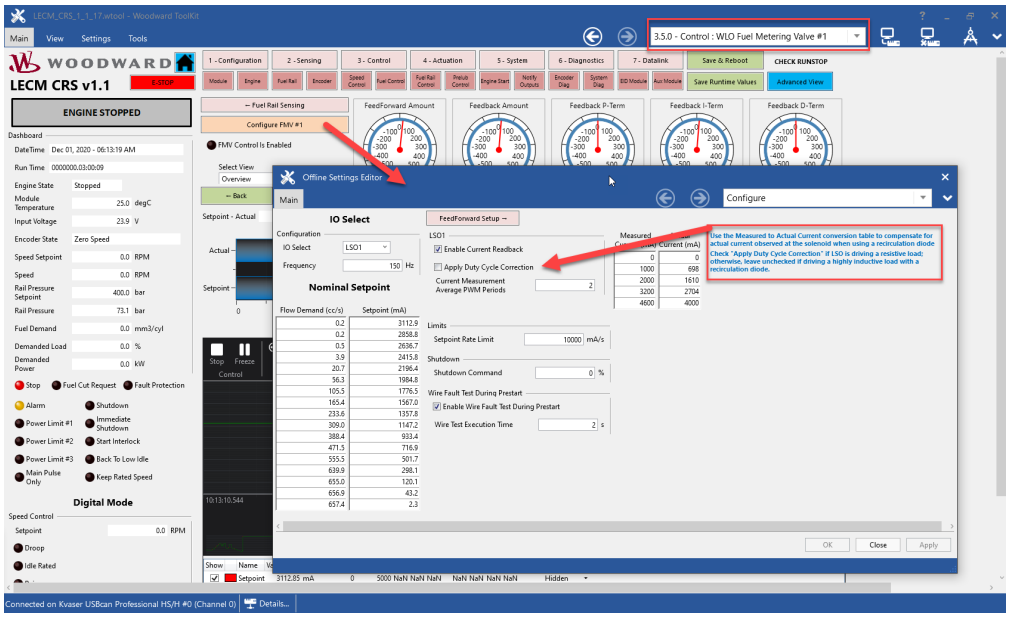
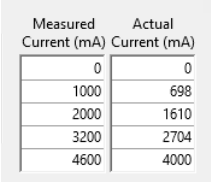


Figure 3-16. Fuel Control Valve Configuration

There is also a conversion table available to make corrections for the Measured to Actual Current. This table can be used to compensate for the actual current observed at the solenoid when using a recirculation diode. Typical setup of this table is similar to the example below.



Measured Current (mA)	Actual Current (mA)
0	0
1000	698
2000	1610
3200	2704
4600	4000

Figure 3-17. Fuel Control Valve Configuration

The fuel control valve is forcibly disabled when no speed is sensed by the LESP control system. The user has the option to select a fixed LSO duty cycle command when the fuel control valve is shutdown.

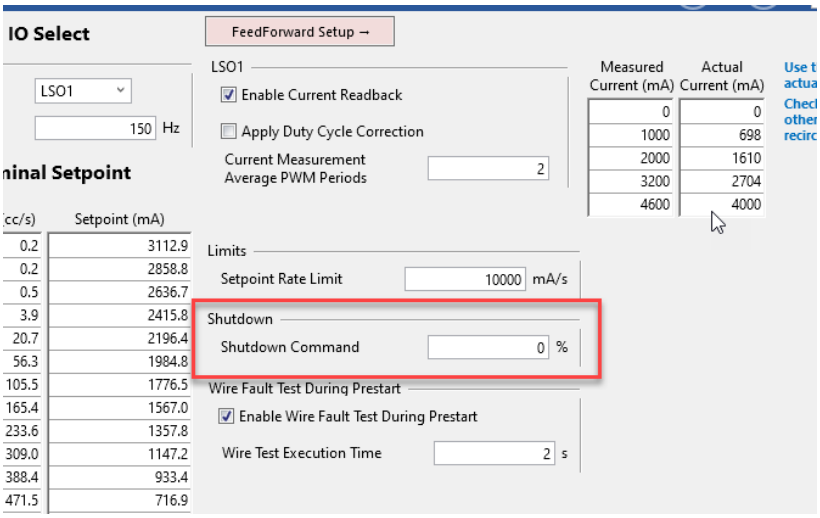
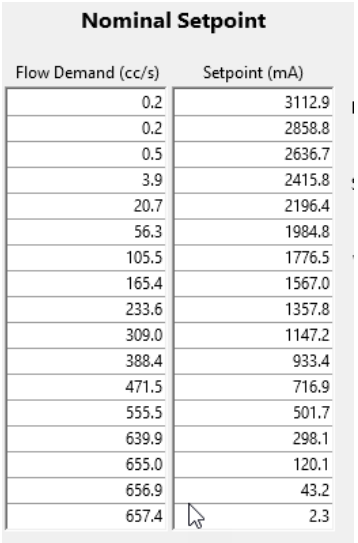


Figure 3-18. Fuel Control Valve Configuration

Setpoint Strategy

The fuel control valve setpoint is based on a 1-D linear lookup table that converts a fuel demand [cc/s] to a current command [mA]. This can be forward or reverse acting, depending on the FCV used for the application. An optional setpoint rate limit can also be enabled to avoid large step changes in setpoints to allow for more stable control.



Nominal Setpoint

Flow Demand (cc/s)	Setpoint (mA)
0.2	3112.9
0.2	2858.8
0.5	2636.7
3.9	2415.8
20.7	2196.4
56.3	1984.8
105.5	1776.5
165.4	1567.0
233.6	1357.8
309.0	1147.2
388.4	933.4
471.5	716.9
555.5	501.7
639.9	298.1
655.0	120.1
656.9	43.2
657.4	2.3

Figure 3-19. Fuel Control Valve Setpoint Configurations

Feedforward Strategy

The fuel control feedforward strategy supports a fixed 1-D lookup table as a function of setpoint [mA] to command a given LSO duty cycle [%]. Using the feedforward strategy can improve the response of the current control loop in total. Regardless, the PID must be setup correctly also.

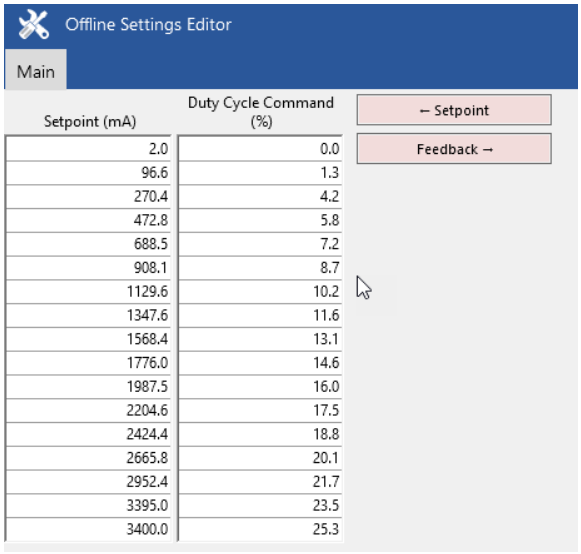


Figure 3-20. Example for Fuel Control Valve Feedforward Configuration

Feedback Strategy

The software feedback strategy utilizes a basic PID controller with gain scheduling for the P, I, and D-terms as a function of PID controller error as illustrated in the figure below. The gain schedules include 1-D lookup tables as a function of error and rail pressure.

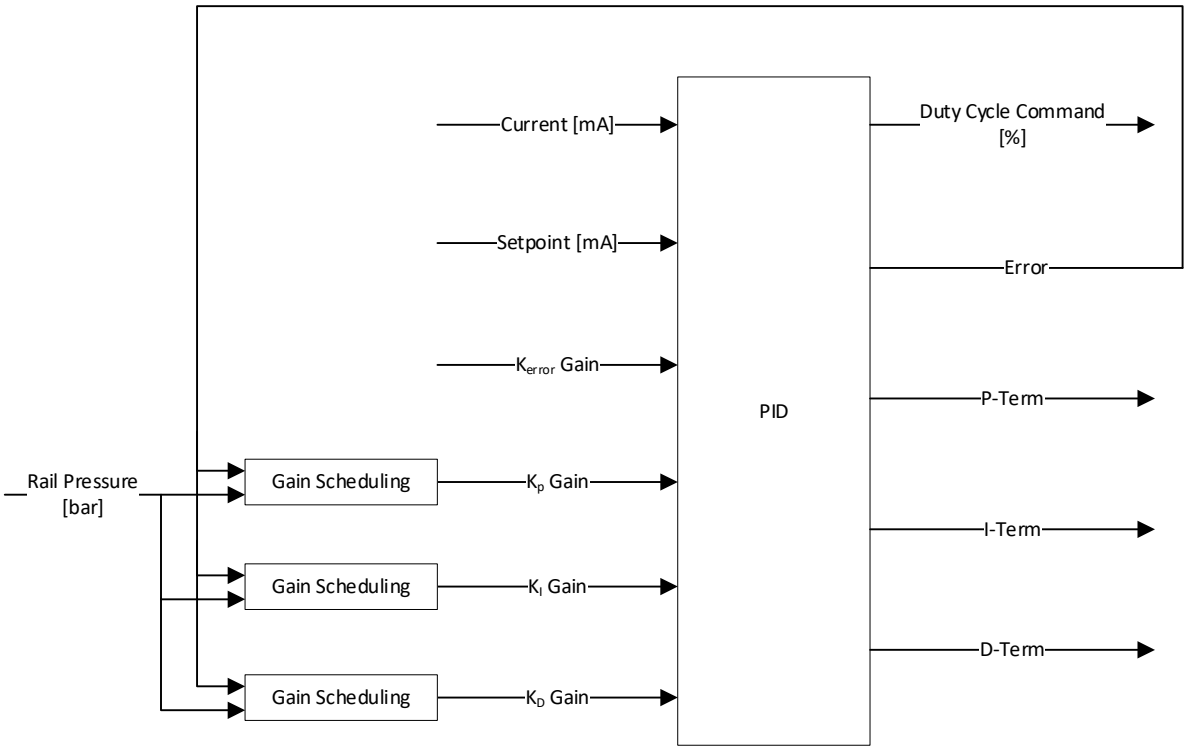


Figure 3-21. Feedback Strategy

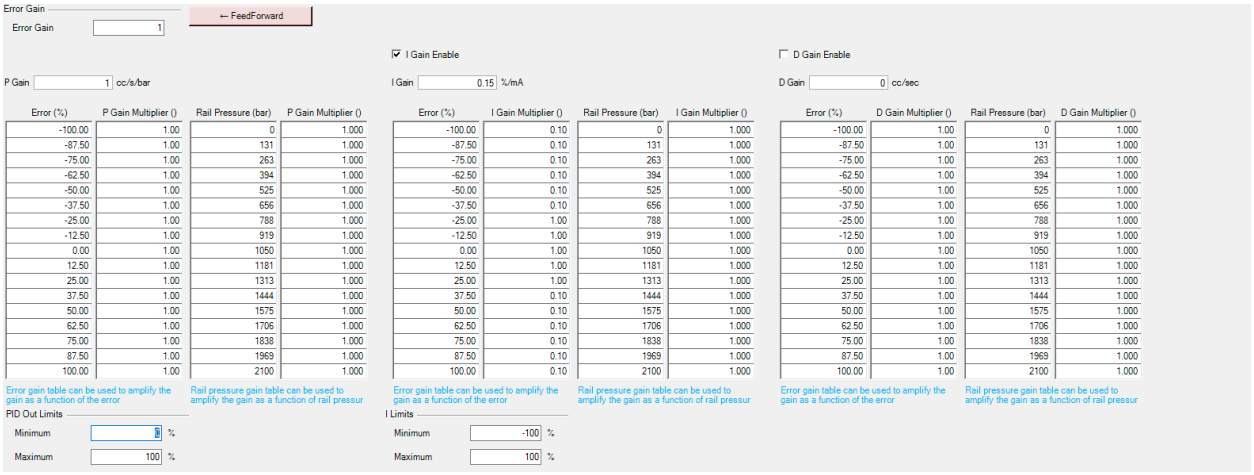


Figure 3-22. Feedback Configurations

$$P_{Gain} = P_{base} x\ P_{error} x\ P_{rail\ presusre}$$

$$I_{Gain} = 0\ if\ enabled; otherwise, \{I_{base} x\ I_{error} x\ I_{rail\ pressure}\}_{limited}$$

$$D_{Gain} = 0\ if\ enabled; otherwise\ (D_{base} x\ D_{error} x\ D_{rail\ pressure})$$

Chapter 4.
Control - AUX Acoustic Knock Mitigation

Introduction

The ‘Control - AUX Acoustic Knock Mitigation’ blockset is used to control the mitigation logic for the acoustic knock that is processed by the LECM Aux board and transmitted on the internal bus to the LECM main board. Inside the blockset, one block is available as shown below.

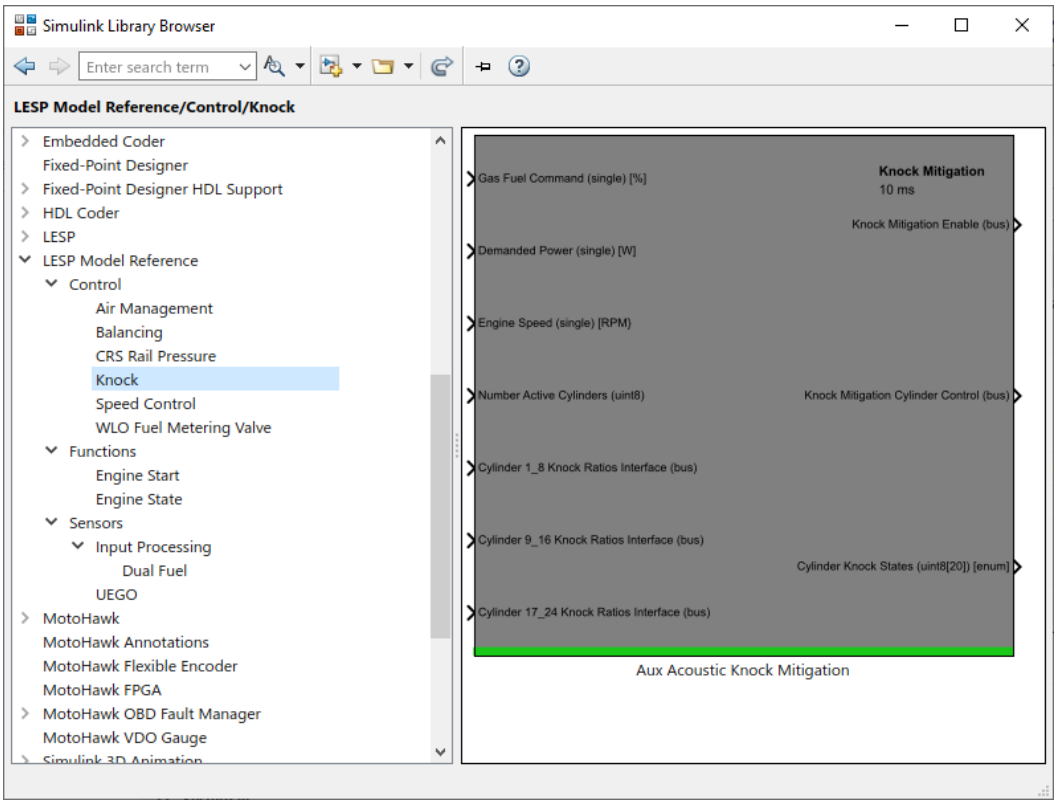


Figure 4-1. LESP Model Reference ‘AUX Acoustic Knock Mitigation’ Block

- Aux Acoustic Knock Mitigation
  This blockset is used to read and process the knock specific data from LECM Aux module and provides the ability to setup different mitigation logics that can be used in the application. The mitigation logic inside this blockset is freely configurable/selectable providing the detailed logic for retarding timing and/or load derating of the engine load, to solve potential knock issues.

Table 4-1. ‘Aux Acoustic Knock Mitigation’ Block

Port	DataType	Description
[In] Gas Fuel Command [%]	Single	The gas % substitution rate that is active at that moment in the application. These fields are used to enable/disable the mitigation logic.
[In] Demanded Power [W]	Single	The demanded power that is active at that moment in the application. These fields are used to enable/disable the mitigation logic.
[In] Engine Speed [rpm]	Single	The actual engine speed at that moment in the application. These fields are used to enable/disable the mitigation logic.

[In] Number Active Cylinders	Uint8	The number of active cylinders in the application, where the knock detection is active. These fields are used to enable/disable the mitigation logic.
[In] Cylinder 1_8 Knock Ratios Interface	bus	The cylinder knock ratios #1 - #8 interface reports the cylinder knock ratios values for cylinder #1 - #8 from AUX module to the main module. AuxKnockRatioInterfaceBus: - Status(AuxRxInterfaceStatusBus) - Data(AuxKnockRatioInterfaceDataBus) Status(AuxRxInterfaceStatusBus): - IsEnabled - Timeout_ms - RegionAge_ms - ConfigurationFault - TimeoutFault - DatalinkSelect_enum Data(AuxKnockRatioInterfaceDataBus): - Statuses_enum - NewSamples - KnockRatios_pct - KnockSaturations_pct - ReferenceSaturations_pct DatalinkSelect_enum (lesp_datalink_select_enum): 0 – None 1 – CAN_1 2 – CAN_2 3 – CAN_3 4 – CAN_4 5 – CAN_5 6 – CAN_6 7 – CAN_7 8 – CAN_8 9 – CAN_9 10 – CAN_10 11 – SPI_1 12 – SPI_2 13 – SPI_3 14 – SPI_4 15 – RS232_1 16 – RS485_1 17 – Ethernet_1 Statuses_enum (lesp_aux_knock_status_enum): 0 – Valid 1 – Disabled 2 – Oversampled 3 – Undersampled 4 – Window Outside Boundary 5 – Resource Error 6 – Encoder Schedule Error 7 – Channel Overlap 8 – System Not Initialized 9 – Shared Buffer Unavailable 10 – Sensor Failed

		11 – No Sync Detected 12 – Interface Error/Not Available
[In] Cylinder 9_16 Knock Ratios Interface	bus	The cylinder knock ratios #9 - #16 interface reports the cylinder knock ratios values for cylinder #9 - #16 from the AUX module to the main module AuxKnockRatioInterfaceBus: - Status(AuxRxInterfaceStatusBus) - Data(AuxKnockRatioInterfaceDataBus) Status(AuxRxInterfaceStatusBus): - IsEnabled - Timeout_ms - RegionAge_ms - ConfigurationFault - TimeoutFault - DatalinkSelect_enum Data(AuxKnockRatioInterfaceDataBus): - Statuses_enum - NewSamples - KnockRatios_pct - KnockSaturations_pct - ReferenceSaturations_pct DatalinkSelect_enum (lesp_datalink_select_enum): 0 – None 1 – CAN_1 2 – CAN_2 3 – CAN_3 4 – CAN_4 5 – CAN_5 6 – CAN_6 7 – CAN_7 8 – CAN_8 9 – CAN_9 10 – CAN_10 11 – SPI_1 12 – SPI_2 13 – SPI_3 14 – SPI_4 15 – RS232_1 16 – RS485_1 17 – Ethernet_1 Statuses_enum (lesp_aux_knock_status_enum): 0 – Valid 1 – Disabled 2 – Oversampled 3 – Undersampled 4 – Window Outside Boundary 5 – Resource Error 6 – Encoder Schedule Error 7 – Channel Overlap 8 – System Not Initialized 9 – Shared Buffer Unavailable 10 – Sensor Failed

		11 – No Sync Detected 12 – Interface Error/Not Available
[In] Cylinder 17_24 Knock Ratios Interface	bus	The cylinder knock ratios #17 - #24 interface reports the cylinder knock ratios values for cylinder #17 - #24 from the AUX module to the main module. AuxKnockRatioInterfaceBus: - Status(AuxRxInterfaceStatusBus) - Data(AuxKnockRatioInterfaceDataBus) Status(AuxRxInterfaceStatusBus): - IsEnabled - Timeout_ms - RegionAge_ms - ConfigurationFault - TimeoutFault - DatalinkSelect_enum Data(AuxKnockRatioInterfaceDataBus): - Statuses_enum - NewSamples - KnockRatios_pct - KnockSaturations_pct - ReferenceSaturations_pct DatalinkSelect_enum (lesp_datalink_select_enum): 0 – None 1 – CAN_1 2 – CAN_2 3 – CAN_3 4 – CAN_4 5 – CAN_5 6 – CAN_6 7 – CAN_7 8 – CAN_8 9 – CAN_9 10 – CAN_10 11 – SPI_1 12 – SPI_2 13 – SPI_3 14 – SPI_4 15 – RS232_1 16 – RS485_1 17 – Ethernet_1 Statuses_enum (lesp_aux_knock_status_enum): 0 – Valid 1 – Disabled 2 – Oversampled 3 – Undersampled 4 – Window Outside Boundary 5 – Resource Error 6 – Encoder Schedule Error 7 – Channel Overlap 8 – System Not Initialized 9 – Shared Buffer Unavailable 10 – Sensor Failed

		11 – No Sync Detected 12 – Interface Error/Not Available
[Out] Knock Mitigation Enable	bus	This field provides the feedback if the Knock Mitigation is active / enabled. EngineKnock MitigationEnableBus: - DetectionEnable - MitigationEnable - MitigationDiagnosticsEnable - MitigationRetardEnable_enum - MitigationDerateEnable_enum MitigationRetardEnable_enum: (lesp_engine_knock_timing_retard_enable_enum) 0 – Disabled 1 – Light Knock Only 2 – Heavy Knock Only 3 – Light Or Heavy Knock MitigationDerateEnable_enum: (lesp_engine_knock_derate_enable_enum) 0 – Disabled 1 – Light Knock Only 2 – Heavy Knock Only 3 – Light Or Heavy Knock
[Out] Knock Mitigation Cylinder Control	bus	This field provides the cylinder individual knock mitigation control signals which can be used for timing / duration corrections. EngineKnockMitigationCylinderControlBus This provides: - Cylinder Timing Retard_x8degATDC - Cylinder Derate Request_pct - MaxCylinderDerateRequest_pct - MaxDerateCylinder
[Out] Cylinder knock States [enum] [20]	bus	This field provides the actual status of the individual cylinders concerning knock states. Statuses_enum (lesp_aux_knock_status_enum): 0 – Valid 1 – Disabled 2 – Oversampled 3 – Undersampled 4 – Window Outside Boundary 5 – Resource Error 6 – Encoder Schedule Error 7 – Channel Overlap 8 – System Not Initialized 9 – Shared Buffer Unavailable 10 – Sensor Failed 11 – No Sync Detected 12 – Interface Error/Not Available

Example Model Using the Blocks

Below is a very simple setup using MotoHawk Calibration and MotoHawk Probes blocks to make the model suitable for building.

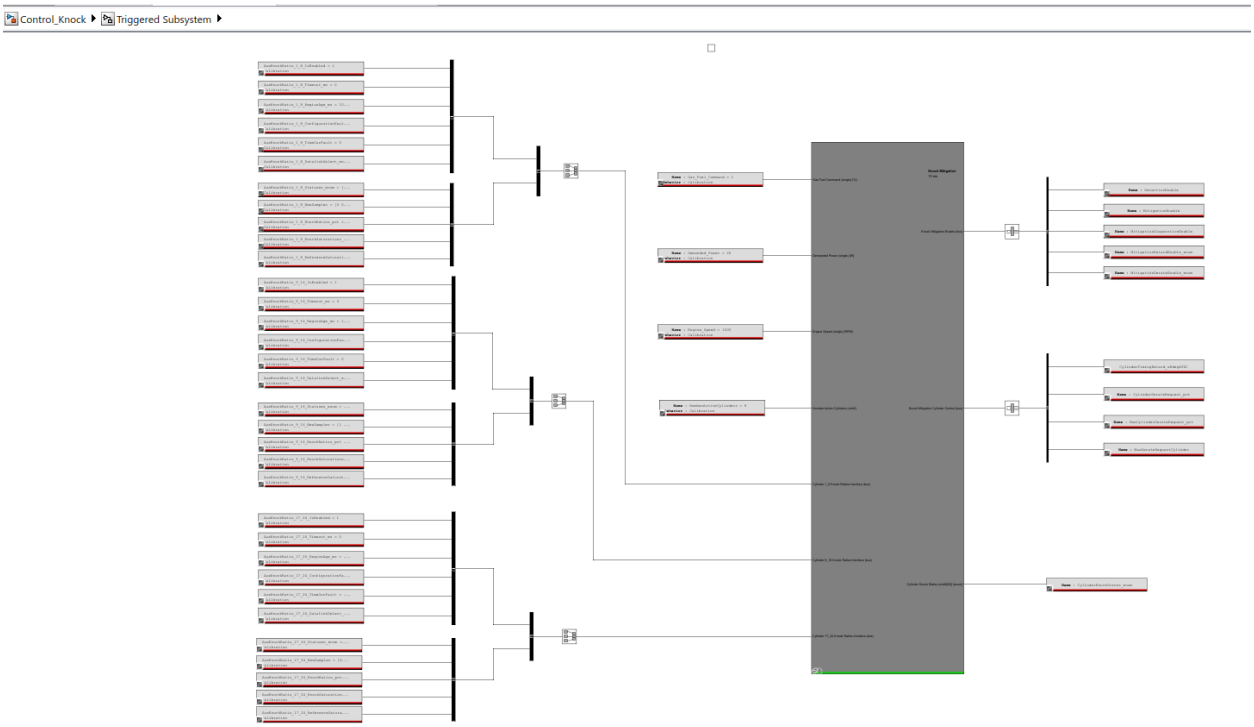


Figure 4-2. Simple Example Model, Using the Control - Knock Mitigation Block

When compiling, the required files will generate and if the Toolkit required blocks are also added to the application model, the required SID files will also generate for the Toolkit HMI tool creation.

Default ToolKit Pages

Open a new Toolkit application and select the correctly generated .sid file, as shown below.

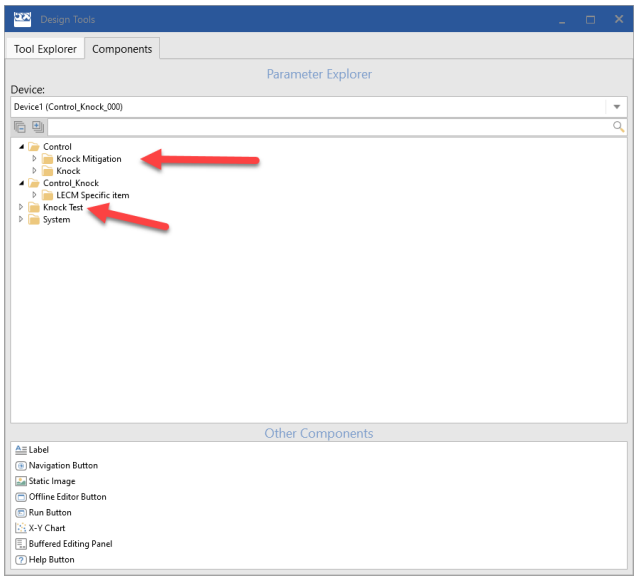
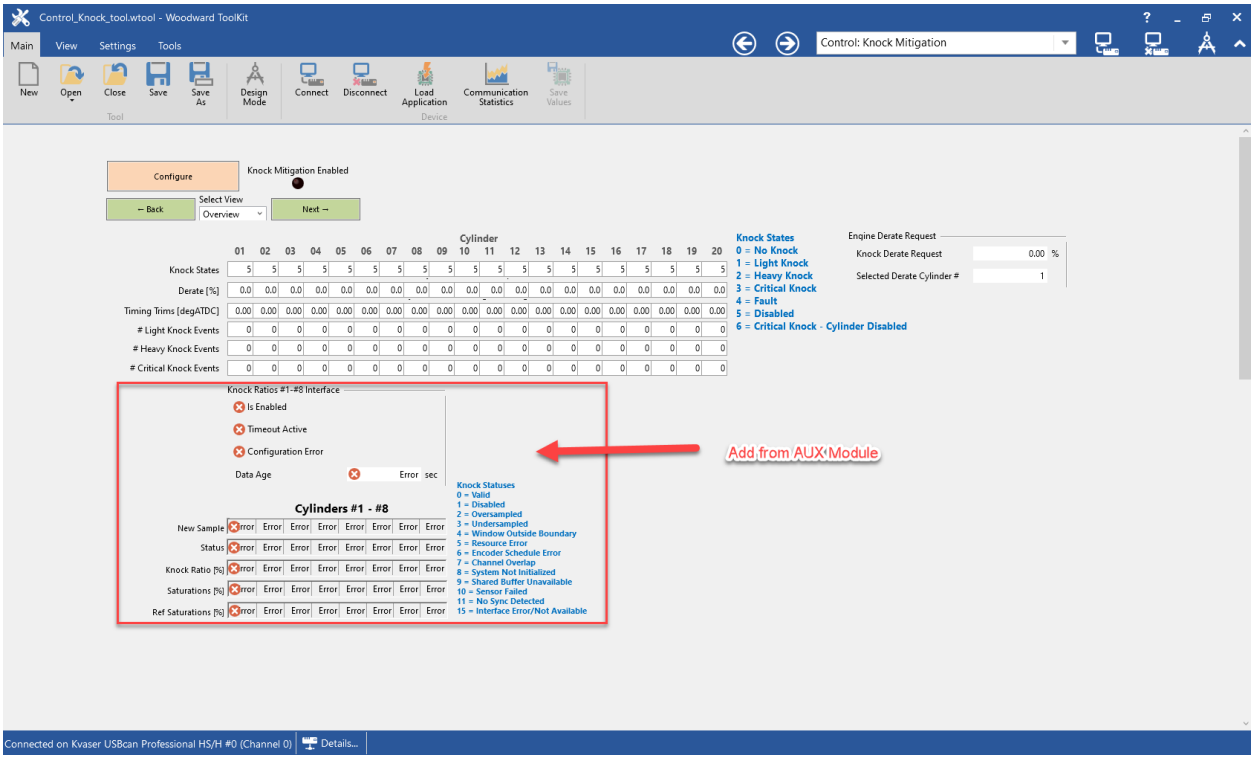
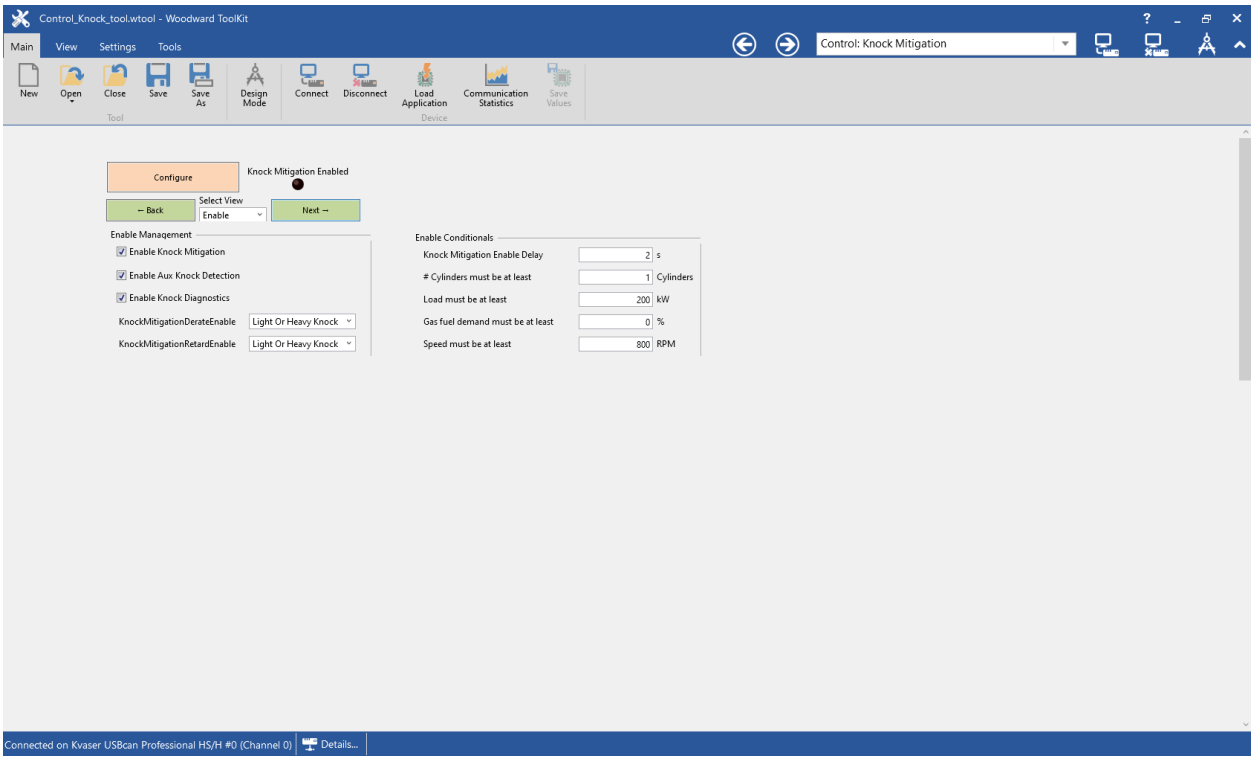


Figure 4-3. SID File Selection Knock Logic

Below are a few of the Toolkit pages containing most of the defined parameters in the example model. Of course, the real sensors and data must be attached to complete it for the application that is being worked on. For that reason, some red X's appear.





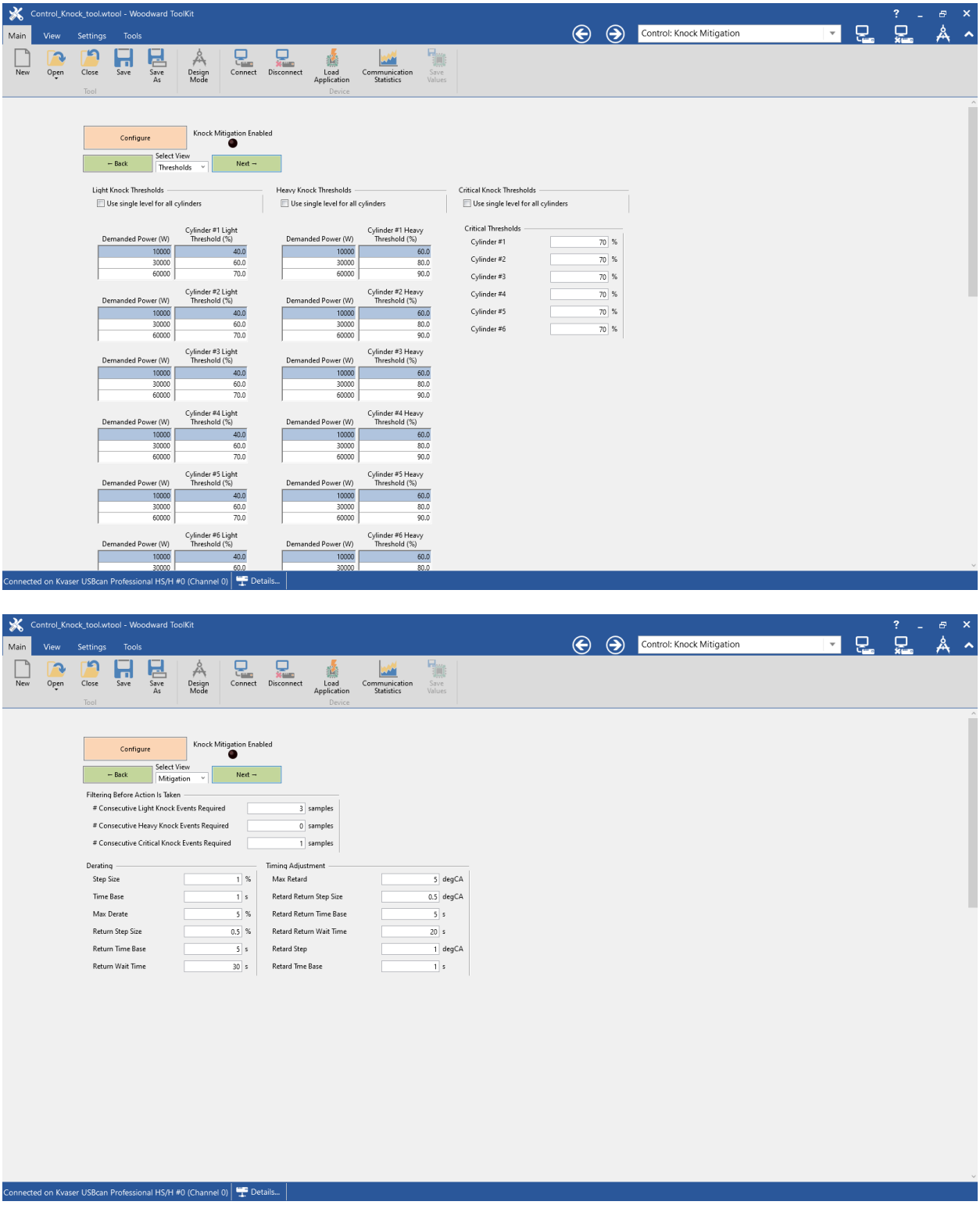


Figure 4-4. Examples of Toolkit Pages

In the Toolkit, there are also some diagnostics available, but they are not included in the example model. The model will need to be updated with the diagnostics as set in the application where they are used.

Functional Description - Knock Detection and Mitigation Logic

The following describes the logic that has been setup for knock detection and mitigation. The knock measurement will be made in the LECM Knock Module. See manual B35138 LECM Aux Platform Knock Installation and Operation Manual for details on calibration and configuration of that module. The knock mitigation is accomplished by adjusting Microplot timing and SOGAV injection duration on the LECM EID Module. See manual B35175 Large Engine Control Module (LECM) Platform EID Injection Driver Large Engine Injection Driver Application Manual for details on configuring the module.

Micropilot Timing Adjustment

When light and heavy knock is detected, the control can retard the Microplot injection timing to delay combustion and mitigate knock. The timing control retards the timing when light knock is detected at the specified retard step once every retard time base until the max retard limit is reached. When heavy knock is detected, the timing retards to the max retard. For the timing to advance, the control cannot detect light or heavy knock for the retard wait time, then the timing will advance at the specified Retard Return Step Size once every Retard Return Time Base.

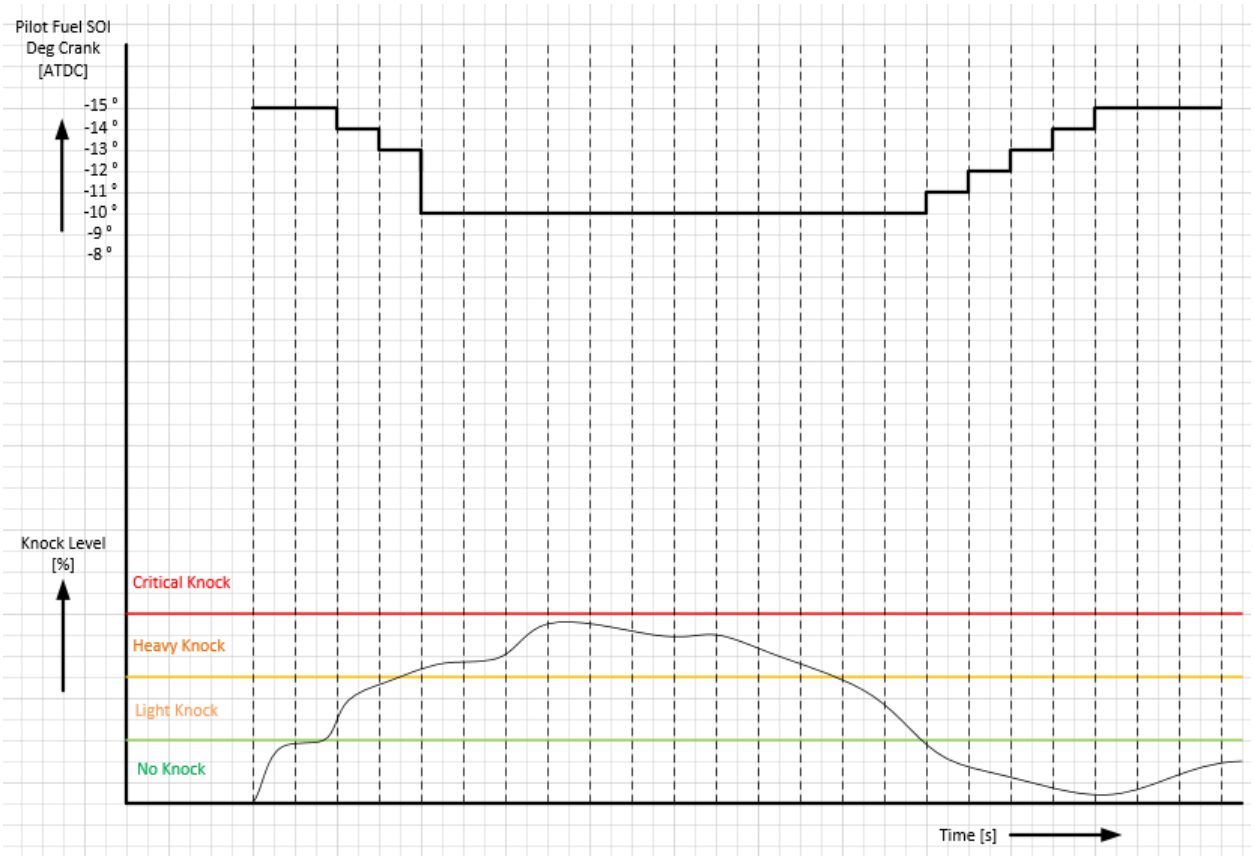


Figure 4-5. Example of the Micropilot Mitigation Logic

Derate Limiting

When light and heavy knock is detected, the control can reduce the fuel limit to the Fuel Limiter Control to mitigate knock. The derate control reduces the fuel limit when light knock is detected at the specified step size once every time base until the max derate limit is reached. When heavy knock is detected, the timing retards to the max retard. For the fuel limit to increase, the control cannot detect light or heavy knock for the Return Wait Time, then the timing will increase at the specified Return Step Size once every Return Time Base.

See Toolkit Page Control: Knock Mitigation Knock for control and calibration of the main module.

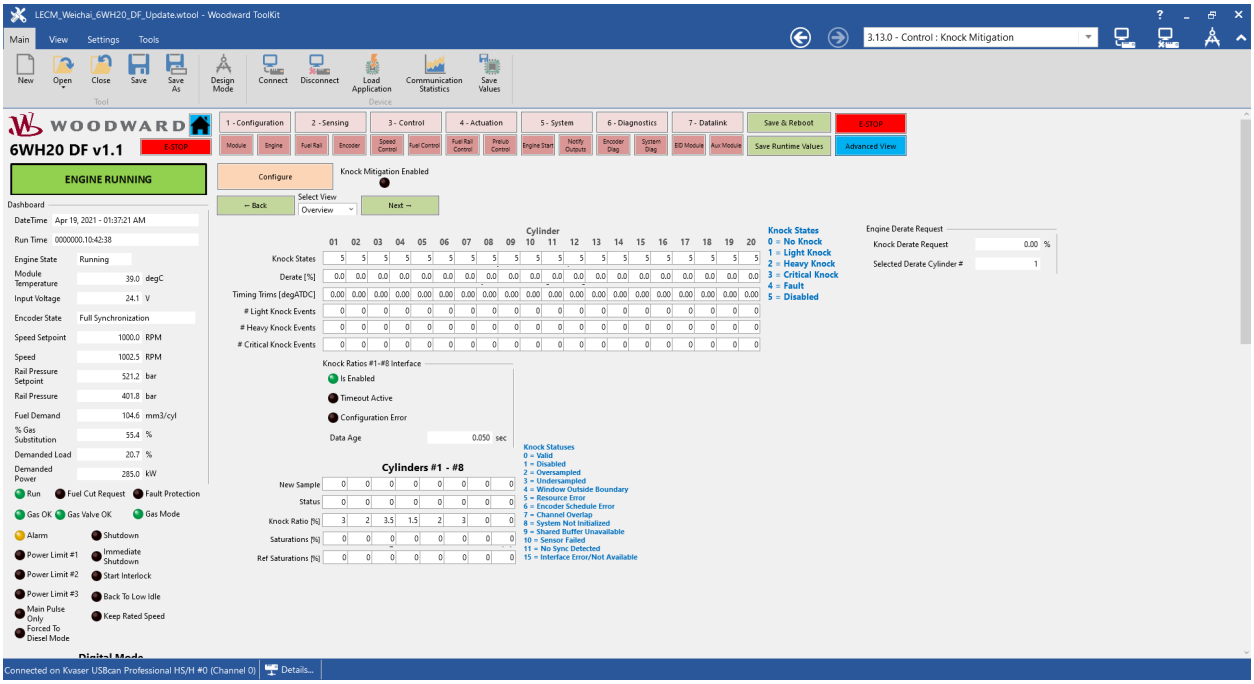


Figure 4-6. Toolkit Page – Control: Knock Mitigation

Configuration

The configuration can be accessed by selecting the configuration button on the Toolkit page 3.13.0 – Control: Knock Mitigation. Toolkit will open an offline editor window.

Enable Management

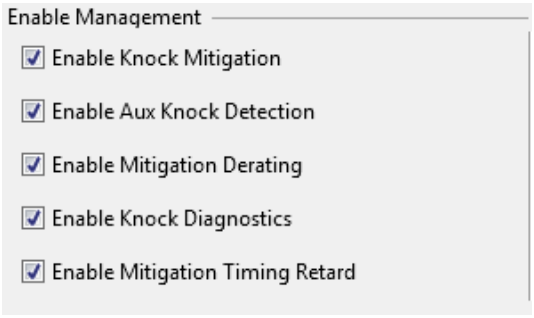


Figure 4-7. Knock Mitigation Enable Management

Enable Knock Mitigation: Check the box to enable knock mitigation.
Enable Aux Knock Detection: Check the box to enable Aux knock.

Enable Mitigation Derating: Check the box to enable knock mitigation mode that will generate a fuel limiter derate command used by the Speed Control - Fuel Demand Limiting system. The configuration of this function can be found on the Mitigation page.
Enable Knock Diagnostics: Check this box to enable knock diagnostics. The diagnostics configuration can be found on the Diagnostics page.
Enable Mitigation Timing Retard: Check the box to enable the knock mitigation timing adjustments to the Microplot timing. The configuration of this function can be found on the Mitigation page.

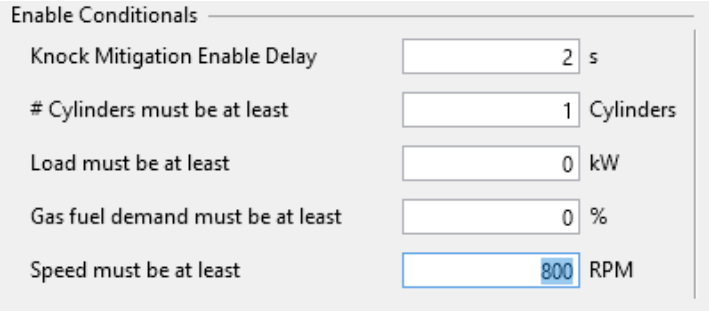


Figure 4-8. Knock Mitigation Enable Conditions

The knock mitigation control requires several thresholds to be true prior to enabling the mitigation control and diagnostics.

Knock Mitigation Enable Delay: Delay time that must expire before knock mitigation actions will be taken.

 # Cylinders Must be at Least: Minimum number of valid knock signals must be greater than this threshold for the knock migration actions will be taken.

Load Must be at Least: The demanded load must be greater than this threshold for the knock migration actions will be taken.

Gas Fuel Demand Must be at Least: The % gas substitution must be greater than this threshold for the knock migration actions to be taken.

Speed Must be at Least: The speed must be greater than this threshold for the knock migration actions to be taken.

Thresholds

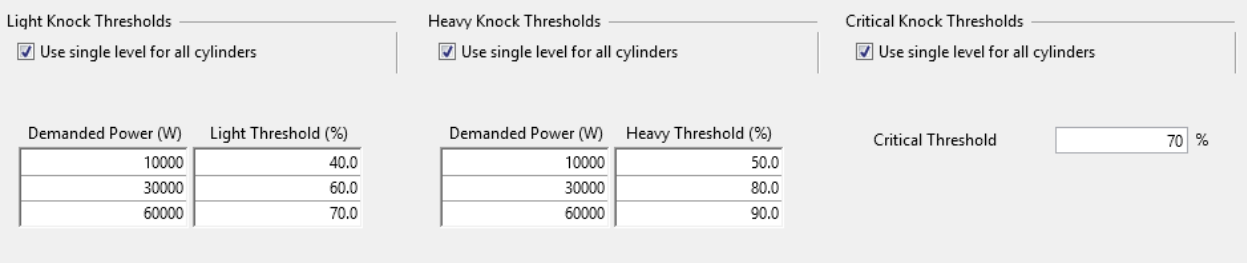


Figure 4-9. Knock Thresholds Setup

Use Single Level for All Cylinders: Check the box to enable a single value for all cylinders to be used for the knock threshold. Unchecking this box will expose thresholds for each cylinder.

Light Knock

Light knock is defined by many OEMs as a condition where ripples on the combustion pressure trace is visible and the maximum ripple magnitude is less than the maximum in cylinder pressure.

Heavy Knock

Heavy knock is defined by many OEMs as a condition where ripples on the combustion pressure trace is visible and the maximum ripple magnitude is the maximum in cylinder pressure, approaching the OEM defined limit (possibly the limit of the cylinder).

When a knock intensity value is above the Heavy threshold value, the timing will go to the “Maximum Retard Bias”, if enabled. Separate alarm and shutdown delay timers are configurable.

Critical Knock

Critical knock is defined by many OEMs as a condition where ripples on a combustion pressure trace are visible, and the maximum ripple magnitude is the maximum in cylinder pressure, exceeding the mechanical limit of the engine. As a vibration-based signal, it could also be a large vibration event like a broken valve rattling around inside a cylinder.

Critical knock is determined by any single measurement of knock exceeding the defined threshold. If any single measurement exceeds the critical knock threshold, the control will generate a “Shutdown” if enabled and attempt to stop the unit using the faulted shutdown sequence.

Mitigation

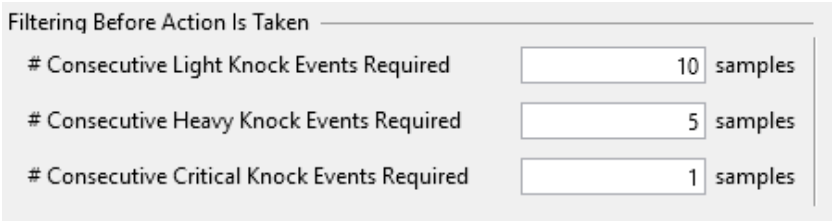


Figure 4-10. Knock Mitigation Filtering Setup

 # Consecutive Light Knock Events Required: The minimum number of consecutive knock values exceeding the light knock threshold for the control to increment a knock count, and if configured, indicate a fault and take Derate or Timing Adjustment actions.
 # Consecutive Heavy Knock Events Required: The minimum number of consecutive knock values exceeding the heavy knock threshold for the control to increment a knock count, and if configured, indicate a fault and take Derate or Timing Adjustment actions.
 # Consecutive Critical Knock Events Required: The minimum number of consecutive knock values exceeding the critical knock threshold for the control to increment a knock count, and if configured, take engine shutdown fault action.

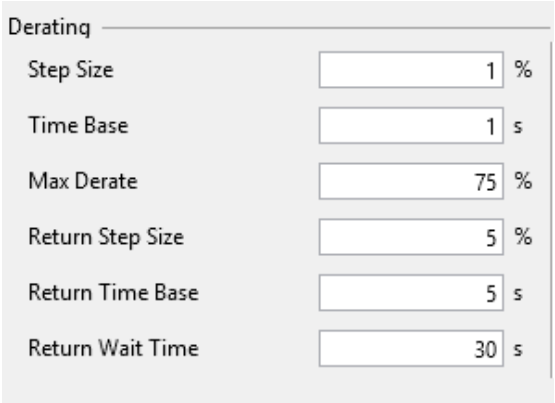


Figure 4-11. Knock Mitigation Settings

Maximum Derate: The maximum derate % that can be applied.

Derate Step: Derate step taken when the knock intensity exceeds the “Light Knock Threshold”.

Time Base: Time period between derate steps. Allows time for derate to have effect.

Return Time Base: After the initial “Return Wait Time” expires, this value becomes the interval between derate return steps. Typically, longer than “Time Base” to create an asymmetric mitigation that reduces cycling for sustained knocking conditions.

Return Wait Time: Once knock has stopped; the control waits for the “Return Wait Time” minus the “Return Time Base” before the control starts to return. Typically, 5 to 10 times longer than “Return Time Base”.

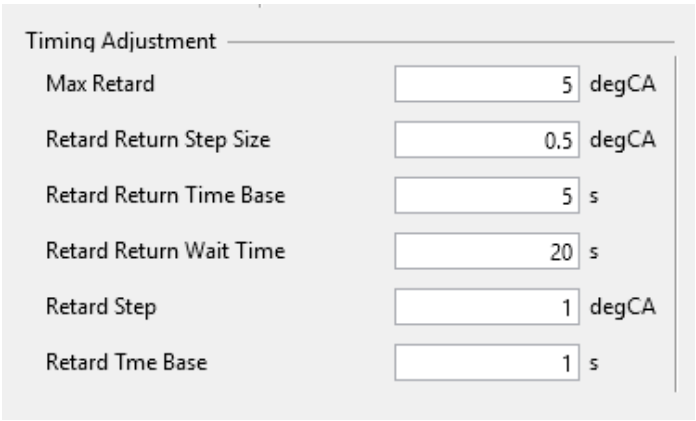


Figure 4-12. Knock Mitigation Timing Retard Setup

Max Retard: The maximum timing retard bias that will be applied to the ignition timing.

Retard Step: Timing bias step taken when the knock intensity exceeds the “Light Knock Threshold”.

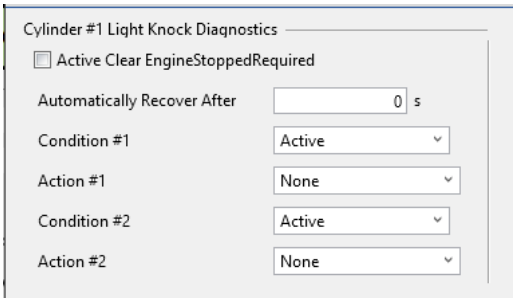
Time Base: Time period between retard steps. Allows time for retard to have affect.

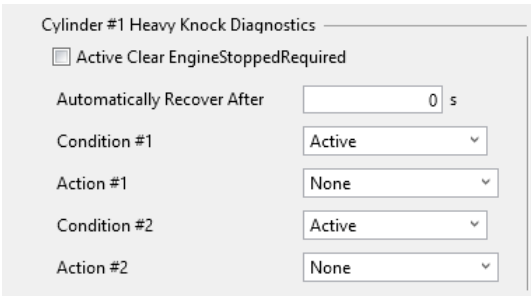
Return Time Base: After the initial “Return Wait Time” expires, this value becomes the interval between retard return steps. Typically, longer than “Time Base” to create an asymmetric mitigation reducing cycling for sustained knocking conditions.

Return Wait Time: Once knock has stopped; the control waits for the “Return Wait Time” minus the “Return Time Base” before the control starts to return. Typically, 5 to 10 times longer than “Return Time Base”.

Diagnostics

The diagnostics configuration allows the fault condition and action to be configured in addition to Derate and Timing Adjustment actions.





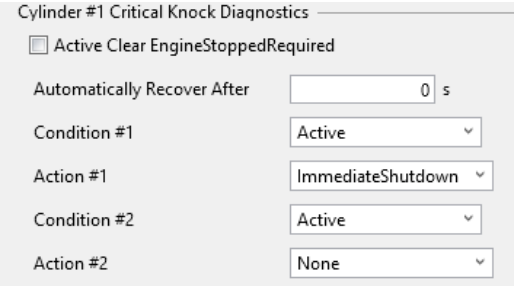


Figure 4-13. Knock Diagnostics Setup

Chapter 5.
Control – Speed Control

Introduction

The ‘Control – Speed Control’ blockset is used to define the engine speed control logic which can be used in various applications like Genset, Marine Propulsion, Pump control, etc. Inside the blockset, two blocks are available as shown below.

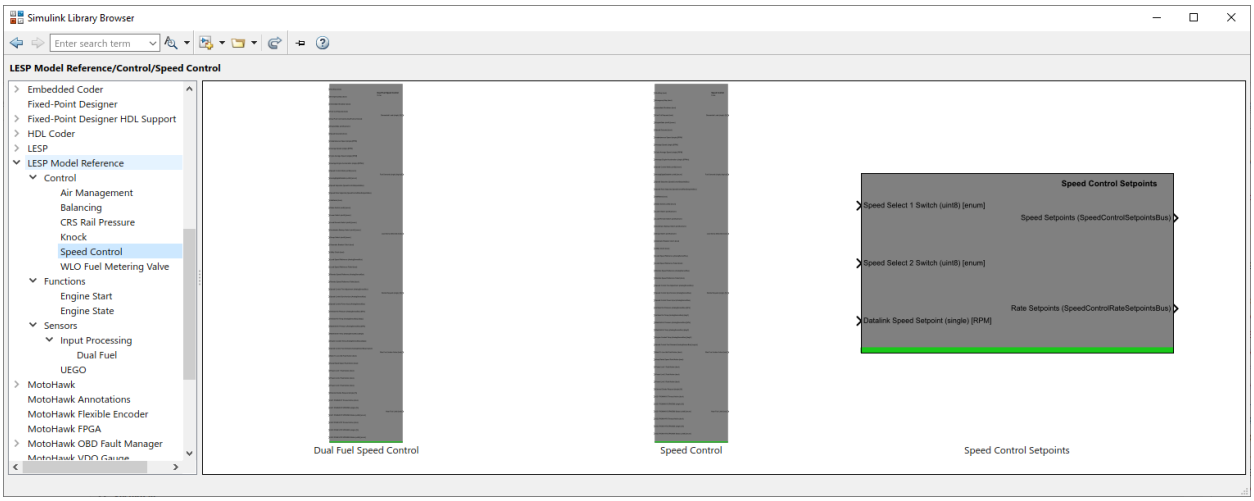


Figure 5-1. LESP Model Reference ‘Control – Speed Control’ Blockset

- Speed Control
  This block contains the base input and output signals required for the speed control logic.
  The J1939 interface signals are included, even as the PID with the correct gain scheduling tables for each different mode. The base speed control can be used for Genset applications where normally the discrete inputs, such as Raise, Lower, Idle/Rated, etc. are used. Analog Speed control mode will be possible as it has two Analog Speed reference inputs available, which normally can be used for ship propulsion systems. Operating pump drives or fracking trucks will also be possible with the base Speed Control logic. Switching between the different modes is included in the logic, even as fuel limiters, power limiters, etc., which are all belonging to the speed control setup.

- Dual Fuel Speed Control
  This block is specific for dual fuel speed control but contains mainly the same setup as the base speed control block. The dual fuel versions have the more extensive dynamics selections that will be available with for each mode their own gain scheduling. The fuel split logic is a separate item in this block, which can be set depending on the required substitution rates that will be used in the application.

- Speed Control Setpoints
  The Speed Control Setpoints block is a block that can be used to determine rated speed levels and/or datalink speed setpoints (J1939…). More details on the blocks can be found in the next chapters.

Speed Control Block Interface

Figure 5-2. ‘Speed Control’ Block

Table 5-1. 'Speed Control' Block

Port	DataType	Description
[In] RunStop	boolean	Input that requires the RunStop status feedback in the application. Run – Engine is ready to run and SpeedControl will be active. Stop – Engine will shut down and go into Stop state.
[In] EmergencyStop	boolean	Input that requires the EmergencyStop status feedback in the application.
[In] Shutdown	boolean	Input that requires the Shutdown status feedback in the application.
[In] Fuel Cut Request	boolean	Input that requires the FuelCut Request status feedback in the application. FuelCut is mostly used during Load Rejections to "Fast" drop the fuel amount to zero level.
[In] Engine State [enum]	uint8	The actual engine state the engine is operating in at that particular moment. These states (lesp_engine_state_enum) are defined as below: 0 – Stopped 1 – Prestart 2 – Starting 3 – Warmup 4 – Running 5 – Cooldown 6 – Stopping 7 – Postrun
[In] Speed Detected	boolean	Input that is required when Speed is detected in the application.
[In] Instantaneous Speed [RPM]	single	Instant Speed Sensing (fast) from Encoder logic inside the application.
[In] Average Speed [RPM]	single	Average Speed Sensing (medium fast) from Encoder logic inside the application. Some average calculation is done on this value.
[In] Cycle Average Speed [RPM]	single	Cycle Average Speed Sensing (slow) from Encoder logic inside the application. The Cycle Average Speed is the speed that is updated every complete engine cycle.
[In] Average Engine Acceleration [RPM/s]	single	The Average Engine acceleration that is sensed by the Speed / Encoder logic in the application. This value is used to trigger the Fuel Cut request during load rejections.
[In] Speed Control Mode [enum]	Uint8	The actual Speed control mode. This can be internal Speed control mode or external Speed control mode. This can be found in the Enumeration feedback when using the: lesp_speed_ctrl_mode_select_enum 0 – Internal 1 – External Fuel Demand
[In] AnalogDigitalDatalink [enum]	Uint8	The actual AnalogDigital Datalink mode. This can be found in the Enumeration feedback when using the: lesp_speed_ctrl_analog_digital_datalink_select_enum 0 – Analog 1 – Digital

		2 – Datalink
[In] Speed Setpoints	SpeedControlSetpointsBus	The actual Speed Setpoints. This can be found in the Enumeration feedback when using the: SpeedControlSetpointsBus -IdleSpeedSetpoint_RPM -RatedSpeedSetpoint_RPM -MaximumSpeedSetpoint_RPM -MinimumAnalogSpeedSetpoint_RPM -MaximumAnalogSpeedSetpoint_RPM -DatalinkSpeedSetpoint_RPM
[In] Speed Rate Setpoints	SpeedControlRateSetpoints Bus	The actual Speed Rate Setpoints. This can be found in the Enumeration feedback when using the: SpeedControlRateSetpointsBus -InstantRate_rpmpers -AccelerationRate_rpmpers -DecelerationRate_rpmpers -DigitalRaiseRate_rpmpers -DigitalLowerRate_rpmpers -AnalogFastRate_rpmpers -AnalogNormalRate_rpmpers -DatalinkRate_rpmpers
[In] IdleRated	boolean	Input that requires the Idle / Rated Selection status feedback in the application.
[In] Raise Switch [enum]	Uint8	This is the Raise Speed input when in Digital Speed control mode. The enum to look for is the: lesp_discrete_in_state_enum 0 – Not Used 1 – Inactive 2 – Active
[In] Lower Switch [enum]	Uint8	This is the Lower Speed input when in Digital Speed control mode. The enum to look for is the: lesp_discrete_in_state_enum 0 – Not Used 1 – Inactive 2 – Active
[In] Local Remote Switch [enum]	Uint8	This is the Local / Remote input when in Analog Speed control mode. The enum to look for is the: lesp_discrete_in_state_enum 0 – Not Used 1 – Inactive 2 – Active
[In] Accelerator Backup Switch [enum]	Uint8	This is the Accelerator Backup input when in Digital Speed control mode. It will activate analog mode even when in digital speed control mode. The enum to look for is the: lesp_discrete_in_state_enum 0 – Not Used 1 – Inactive 2 – Active
[In] Droop Switch [enum]	Uint8	This is the Droop select input when in Analog or Digital Speed control mode. The enum to look for is the: lesp_discrete_in_state_enum

		0 – Not Used 1 – Inactive 2 – Active
[In] Generator Breaker Clutch	boolean	Generator Breaker / Clutch position feedback. This is used for e.g. Dynamics selection.
[In] Utility Clutch	boolean	Utility Breaker / Clutch position feedback. This is used for e.g. Dynamics selection.
[In] Local Speed Reference	AnalogSensorBus	For analog Speed Setting mode, this is the Local Speed Reference selection signal. This signal is used to set the actual RPM setpoint when in Analog Speed Control Mode, when Local signal is selected. Bus Details: AnalogSensorBus: - FilteredValue - MeasurementStatus_enum MeasurementStatus_enum: (lesp_measurement_status_enum) 0 – Valid 1 – Disabled 2 – SignalLow 3 – SignalHigh 4 – SensorSupplyFailed 5 – ReferenceSourceFailed 6 – OpenCircuit
[In] Local Speed Reference Failed	boolean	When the Local Speed Reference Signals is failing, this input must be used to report this.
[In] Remote Speed Reference	AnalogSensorBus	For analog Speed Setting mode, this is the Remote Speed Reference selection signal. This signal is used to set the actual RPM setpoint when in Analog Speed Control Mode, when Remote signal is selected. Bus Details: AnalogSensorBus: - FilteredValue - MeasurementStatus_enum MeasurementStatus_enum: (lesp_measurement_status_enum) 0 – Valid 1 – Disabled 2 – SignalLow 3 – SignalHigh 4 – SensorSupplyFailed 5 – ReferenceSourceFailed 6 – OpenCircuit
[In] Remote Speed Reference Failed	boolean	When the Remote Speed Reference Signals is failing, this input must be used to report this.
[In] Speed Control Fine Adjustment	AnalogSensorBus	For analog Speed Setting mode, this is the Speed Control Fine Adjustment selection signal. This signal is used to set the actual RPM setpoint when in Analog Speed Control Mode, when Speed Control Fine Adjustment signal is selected. Bus Details: AnalogSensorBus: - FilteredValue - MeasurementStatus_enum MeasurementStatus_enum: (lesp_measurement_status_enum)

		0 – Valid 1 – Disabled 2 – SignalLow 3 – SignalHigh 4 – SensorSupplyFailed 5 – ReferenceSourceFailed 6 – OpenCircuit
[In] Speed Control Synchronizer	AnalogSensorBus	For all Speed Setting modes, this is the Speed Control Synchronizer. This signal is used to bias (+/-) the actual RPM setpoint when in Analog or Digital Speed Control Mode. It is mostly used for synchronizing the genset to the local bus or grid. Bus Details: AnalogSensorBus: - FilteredValue - MeasurementStatus_enum MeasurementStatus_enum: (lesp_measurement_status_enum) 0 – Valid 1 – Disabled 2 – SignalLow 3 – SignalHigh 4 – SensorSupplyFailed 5 – ReferenceSourceFailed 6 – OpenCircuit
[In] Speed Control Power Input	AnalogSensorBus	For all Speed Setting mode, this is the Power Input (kW Transducer input). This signal is used to provide the actual Power delivered, when in Speed Control Mode. Bus Details: AnalogSensorBus: - FilteredValue - MeasurementStatus_enum MeasurementStatus_enum: (lesp_measurement_status_enum) 0 – Valid 1 – Disabled 2 – SignalLow 3 – SignalHigh 4 – SensorSupplyFailed 5 – ReferenceSourceFailed 6 – OpenCircuit
[In] Ambient Air Pressure [kPa]	AnalogSensorBus	Requires the actual Ambient Air Pressure signal value to be used for internal logic for Fuel Limiting. Bus Details: AnalogSensorBus: - FilteredValue - MeasurementStatus_enum MeasurementStatus_enum: (lesp_measurement_status_enum) 0 – Valid 1 – Disabled 2 – SignalLow 3 – SignalHigh 4 – SensorSupplyFailed 5 – ReferenceSourceFailed 6 – OpenCircuit

[In] Ambient Air Temperature [degC]	AnalogSensorBus	Requires the actual Ambient Air Temperature signal value to be used for internal logic for Fuel Limiting. Bus Details: AnalogSensorBus: - FilteredValue - MeasurementStatus_enum MeasurementStatus_enum: (lesp_measurement_status_enum) 0 – Valid 1 – Disabled 2 – SignalLow 3 – SignalHigh 4 – SensorSupplyFailed 5 – ReferenceSourceFailed 6 – OpenCircuit
[In] Manifold Air Pressure [kPa]	AnalogSensorBus	Requires the actual Manifold Air Pressure signal value to be used for internal logic for Fuel Limiting. Bus Details: AnalogSensorBus: - FilteredValue - MeasurementStatus_enum MeasurementStatus_enum: (lesp_measurement_status_enum) 0 – Valid 1 – Disabled 2 – SignalLow 3 – SignalHigh 4 – SensorSupplyFailed 5 – ReferenceSourceFailed 6 – OpenCircuit
[In] Manifold Air Temperature [degC]	AnalogSensorBus	Requires the actual Manifold Air Temperature signal value to be used for internal logic for Fuel Limiting. Bus Details: AnalogSensorBus: - FilteredValue - MeasurementStatus_enum MeasurementStatus_enum: (lesp_measurement_status_enum) 0 – Valid 1 – Disabled 2 – SignalLow 3 – SignalHigh 4 – SensorSupplyFailed 5 – ReferenceSourceFailed 6 – OpenCircuit
[In] Engine Coolant Temperature [degC]	AnalogSensorBus	Requires the actual Engine Coolant Temperature signal value to be used for internal logic for Fuel Limiting. Bus Details: AnalogSensorBus: - FilteredValue - MeasurementStatus_enum MeasurementStatus_enum: (lesp_measurement_status_enum) 0 – Valid 1 – Disabled

		2 – SignalLow 3 – SignalHigh 4 – SensorSupplyFailed 5 – ReferenceSourceFailed 6 – OpenCircuit
[In] Speed Control Fuel Demand [mg/cyl]	AnalogSensorBus	Requires the actual Speed Control Fuel Demand when an external Speed control is used. (Internal Speed control is off) Bus Details: AnalogSensorBus: - FilteredValue - MeasurementStatus_enum MeasurementStatus_enum: (lesp_measurement_status_enum) 0 – Valid 1 – Disabled 2 – SignalLow 3 – SignalHigh 4 – SensorSupplyFailed 5 – ReferenceSourceFailed 6 – OpenCircuit
[In] Back to Low Idle Fault Action	boolean	This input must be used when a Back to Low Idle command action is required. When TRUE, it will activate the Low Idle Speed Setpoint and is meant to be coming from any of the diagnostics that needs to drive this action.
[In] Keep Rated Speed Fault Action	boolean	This input must be used when a Keep Rated Speed command action is required. When TRUE, it will keep the Rated Speed Setpoint and is meant to be coming from any of the diagnostics that needs to drive this action (e.g. Breaker Closed, connected to net/grid).
[In] Power limit 1 Fault Action	boolean	This input must be used when a Power Limit 1 command action is required. When TRUE, it will activate a power limit function belonging to the power limit 1 settings from any of the diagnostics that needs to drive this action.
[In] Power limit 2 Fault Action	boolean	This input must be used when a Power Limit 2 command action is required. When TRUE, it will activate a power limit function belonging to the power limit 2 settings from any of the diagnostics that needs to drive this action.
[In] Power limit 3 Fault Action	boolean	This input must be used when a Power Limit 3 command action is required. When TRUE, it will activate a power limit function belonging to the power limit 3 settings from any of the diagnostics that needs to drive this action.
[In] External Derate Request [%]	single	This input must be used when an external Derate Request comes from an external device.
[In] GC1 PGN64915 Timeout Active	boolean	This input must be used to report any issue when the GC1 Message (PGN64915) is failing or is having a timeout.
[In] GC1 PGN64915 SPN5568 (%)	single	This input must be used from a CAN J1939 signal reporting the PGN64915 SPN 5568 (GC1) to set the Droop Percentage for the speed control logic.
[In] GC1 PGN64915 SPN5568 Status [enum]	Uint8	This input must be used to report the actual status of the CAN J1939 PGN64915 SPN 5568

		(GC1) message. The Enum: lesp_j1939_spn_status_enum: 0 - Valid 1 - Error 2 - Reserved 3 - Not Available
[In] GC2 PGN61470 Timeout Active	boolean	This input must be used to report any issue when the GC2 Message (PGN61470) is failing or is having a timeout.
[In] GC2 PGN61470 SPN3938 (%)	single	This input must be used from a CAN J1939 signal reporting the PGN61470 SPN 3938 (GC2) to set the Generator Governing Bias Signal (+/-) which can be used for synchronizing, load sharing, or other bias purposes.
[In] GC2 PGN61470 SPN3938 Status [enum]	Uint8	This input must be used to report the actual status of the CAN J1939 PGN61470 SPN 3938 (GC2) message. The Enum: lesp_j1939_spn_status_enum: 0 - Valid 1 - Error 2 - Reserved 3 - Not Available
[Out] Demanded load [%]	single	This output defines the required load demand to keep the speed control under control. It is a percentage that can be transferred into fuel amount in the application it is used in.
[Out] Fuel Demand [mg/cyl]	single	This output defines the required fuel demand to keep the speed control under control. It is in mg/cyl. and can be used as actual fuel amount in the application it is used in.
[Out] Load Dump Detected	boolean	This output defines if a Load Drop / Dump is detected. This can be used in the application as feedforward signal to set engine devices faster in a correct position instead of waiting for the control logic to act.
[Out] Derate Request [%]	single	This output defines the required derate as commanded from the speed control logic. Not under all circumstances and automatic derate can take place as it could result in engine shutdown / power cut. For those reasons, this output can be sent to the load management system to remove load or trip non-essential users.
[Out] Max Fuel Limiter Active	boolean	This output confirms if the maximum fuel limiter is active. Under normal circumstances this should never happen, but it can indicate that there is a leak somewhere or another issue is ongoing on the engine side.
[Out] Near Fuel Limiter Active	boolean	This output defines if the actual amount of requested fuel is close to a fuel limiter. This can be used to indicate Changeable Pitch Propellor systems, to reduce pitch and prevent the engine from going down in overload conditions.

Dual Fuel Speed Control Block Interface

Figure 5-3. ‘Dual Fuel Speed Control’ Block

Table 5-2. 'Dual Fuel Speed Control' Block

Port	DataType	Description
[In] RunStop	boolean	Input that requires the RunStop status feedback in the application. Run – Engine is ready to run and SpeedControl is active. Stop – Engine will shut down and go into Stop state.
[In] EmergencyStop	boolean	Input that requires the EmergencyStop status feedback in the application.
[In] Shutdown	boolean	Input that requires the Shutdown status feedback in the application.
[In] Fuel Cut Request	boolean	Input that requires the FuelCut Request status feedback in the application. FuelCut is mostly used during Load Rejections to "Fast" drop the fuel amount to zero level.
[In] Dual Fuel Commands	DualFuelCommand	Input that requires the Dual Fuel Command input for the application. Definition: DualFuelCommandBus: -DieselFuelCommand_mgpercyl -GasFuelCommand_mgpercyl -DieselFuelCommand_Pct -GasFuelCommand_Pct -LoadTooHighToTransferToDieselMode
[In] Engine State [enum]	uint8	The actual engine state the engine is operating in at that particular moment. These states (lesp_engine_state_enum) are defined as below: 0 – Stopped 1 – Prestart 2 – Starting 3 – Warmup 4 – Running 5 – Cooldown 6 – Stopping 7 – Postrun
[In] Speed Detected	boolean	Input that is required when Speed is detected in the application.
[In] Instantaneous Speed [RPM]	single	Instant Speed Sensing (fast) from Encoder logic inside the application.
[In] Average Speed [RPM]	single	Average Speed Sensing (medium fast) from Encoder logic inside the application. Some average calculation is done on this value.
[In] Cycle Average Speed [RPM]	single	Cycle Average Speed Sensing (slow) from Encoder logic inside the application. The Cycle Average Speed is the speed that is updated every complete engine cycle.
[In] Average Engine Acceleration [RPM/s]	single	The Average Engine acceleration that is sensed by the Speed / Encoder logic in the application. This value is used to trigger the Fuel Cut request during load rejections.
[In] Speed Control Mode [enum]	Uint8	The actual Speed Control mode. This can be internal Speed Control mode or external Speed Control mode. This can be found in the Enumeration feedback when using the: lesp_speed_ctrl_mode_select_enum 0 – Internal

		1 – External Fuel Demand
[In] AnalogDigitalDatalink [enum]	Uint8	The actual AnalogDigital Datalink mode. This can be found in the Enumeration feedback when using the: lesp_speed_ctrl_analog_digital_datalink_select_enum 0 – Analog 1 – Digital 2 – Datalink
[In] Speed Setpoints	SpeedControlSetpointsBus	The actual Speed Setpoints. This can be found in the Enumeration feedback when using the: SpeedControlSetpointsBus -IdleSpeedSetpoint_RPM -RatedSpeedSetpoint_RPM -MaximumSpeedSetpoint_RPM -MinimumAnalogSpeedSetpoint_RPM -MaximumAnalogSpeedSetpoint_RPM -DatalinkSpeedSetpoint_RPM
[In] Speed Rate Setpoints	SpeedControlRateSetpointsBus	The actual Speed Rate Setpoints. This can be found in the Enumeration feedback when using the: SpeedControlRateSetpointsBus -InstantRate_rpmpers -AccelerationRate_rpmpers -DecelerationRate_rpmpers -DigitalRaiseRate_rpmpers -DigitalLowerRate_rpmpers -AnalogFastRate_rpmpers -AnalogNormalRate_rpmpers -DatalinkRate_rpmpers
[In] IdleRated	boolean	Input that requires the Idle / Rated Selection status feedback in the application.
[In] Raise Switch [enum]	Uint8	This is the Raise Speed input when in Digital Speed control mode. The enum to look for is the: lesp_discrete_in_state_enum 0 – Not Used 1 – Inactive 2 – Active
[In] Lower Switch [enum]	Uint8	This is the Lower Speed input when in Digital Speed control mode. The enum to look for is the: lesp_discrete_in_state_enum 0 – Not Used 1 – Inactive 2 – Active
[In] Local Remote Switch [enum]	Uint8	This is the Local / Remote input when in Analog Speed control mode. The enum to look for is the: lesp_discrete_in_state_enum 0 – Not Used 1 – Inactive 2 – Active
[In] Accelerator Backup Switch [enum]	Uint8	This is the Accelerator Backup input when in Digital Speed control mode. It will activate Analog mode even when in Digital Speed control mode. The enum to look for is the: lesp_discrete_in_state_enum

		0 – Not Used 1 – Inactive 2 – Active
[In] Droop Switch [enum]	Uint8	This is the Droop select input when in Analog or Digital Speed control mode. The enum to look for is the: lesp_discrete_in_state_enum 0 – Not Used 1 – Inactive 2 – Active
[In] Generator Breaker Clutch	boolean	Generator Breaker / Clutch position feedback. This is used for Dynamics selection.
[In] Utility Clutch	boolean	Utility Breaker / Clutch position feedback. This is used for Dynamics selection.
[In] Local Speed Reference	AnalogSensorBus	For analog Speed Setting mode, this is the Local Speed Reference selection signal. This signal is used to set the actual RPM setpoint when in Analog Speed Control Mode and Local signal is selected. Bus Details: AnalogSensorBus: - FilteredValue - MeasurementStatus_enum MeasurementStatus_enum: (lesp_measurement_status_enum) 0 – Valid 1 – Disabled 2 – SignalLow 3 – SignalHigh 4 – SensorSupplyFailed 5 – ReferenceSourceFailed 6 – OpenCircuit
[In] Local Speed Reference Failed	boolean	When the Local Speed Reference signals is failing, this input must be used to report this.
[In] Remote Speed Reference	AnalogSensorBus	For analog Speed Setting mode, this is the Remote Speed Reference selection signal. This signal is used to set the actual RPM setpoint when in Analog Speed Control Mode and Remote signal is selected. Bus Details: AnalogSensorBus: - FilteredValue - MeasurementStatus_enum MeasurementStatus_enum: (lesp_measurement_status_enum) 0 – Valid 1 – Disabled 2 – SignalLow 3 – SignalHigh 4 – SensorSupplyFailed 5 – ReferenceSourceFailed 6 – OpenCircuit
[In] Remote Speed Reference Failed	boolean	When the Remote Speed Reference Signals is failing, this input must be used to report this.
[In] Speed Control Fine Adjustment	AnalogSensorBus	For analog Speed Setting mode, this is the Speed Control Fine Adjustment selection signal. This signal is used to set the actual RPM setpoint when in Analog Speed Control Mode,

		and Speed Control Fine Adjustment signal is selected. Bus Details: AnalogSensorBus: - FilteredValue - MeasurementStatus_enum MeasurementStatus_enum: (lesp_measurement_status_enum) 0 – Valid 1 – Disabled 2 – SignalLow 3 – SignalHigh 4 – SensorSupplyFailed 5 – ReferenceSourceFailed 6 – OpenCircuit
[In] Speed Control Synchronizer	AnalogSensorBus	For all Speed Setting modes, this is the Speed Control Synchronizer. This signal is used to bias (+/-) the actual RPM setpoint when in Analog or Digital Speed Control Mode. It is mostly used for synchronizing the genset to the local bus or grid. Bus Details: AnalogSensorBus: - FilteredValue - MeasurementStatus_enum MeasurementStatus_enum: (lesp_measurement_status_enum) 0 – Valid 1 – Disabled 2 – SignalLow 3 – SignalHigh 4 – SensorSupplyFailed 5 – ReferenceSourceFailed 6 – OpenCircuit
[In] Speed Control Power Input	AnalogSensorBus	For all Speed Setting modes, this is the Power Input (kW Transducer input). This signal is used to provide the actual power delivered when in Speed Control Mode. Bus Details: AnalogSensorBus: - FilteredValue - MeasurementStatus_enum MeasurementStatus_enum: (lesp_measurement_status_enum) 0 – Valid 1 – Disabled 2 – SignalLow 3 – SignalHigh 4 – SensorSupplyFailed 5 – ReferenceSourceFailed 6 – OpenCircuit
[In] Ambient Air Pressure [kPa]	AnalogSensorBus	Requires the actual Ambient Air Pressure signal value to be used for internal logic for Fuel Limiting. Bus Details: AnalogSensorBus: - FilteredValue - MeasurementStatus_enum MeasurementStatus_enum:

		(lesp_measurement_status_enum) 0 – Valid 1 – Disabled 2 – SignalLow 3 – SignalHigh 4 – SensorSupplyFailed 5 – ReferenceSourceFailed 6 – OpenCircuit
[In] Ambient Air Temperature [degC]	AnalogSensorBus	Requires the actual Ambient Air Temperature signal value to be used for internal logic for Fuel Limiting. Bus Details: AnalogSensorBus: - FilteredValue - MeasurementStatus_enum MeasurementStatus_enum: (lesp_measurement_status_enum) 0 – Valid 1 – Disabled 2 – SignalLow 3 – SignalHigh 4 – SensorSupplyFailed 5 – ReferenceSourceFailed 6 – OpenCircuit
[In] Manifold Air Pressure [kPa]	AnalogSensorBus	Requires the actual Manifold Air Pressure signal value to be used for internal logic for Fuel Limiting. Bus Details: AnalogSensorBus: - FilteredValue - MeasurementStatus_enum MeasurementStatus_enum: (lesp_measurement_status_enum) 0 – Valid 1 – Disabled 2 – SignalLow 3 – SignalHigh 4 – SensorSupplyFailed 5 – ReferenceSourceFailed 6 – OpenCircuit
[In] Manifold Air Temperature [degC]	AnalogSensorBus	Requires the actual Manifold Air Temperature signal value to be used for internal logic for Fuel Limiting. Bus Details: AnalogSensorBus: - FilteredValue - MeasurementStatus_enum MeasurementStatus_enum: (lesp_measurement_status_enum) 0 – Valid 1 – Disabled 2 – SignalLow 3 – SignalHigh 4 – SensorSupplyFailed 5 – ReferenceSourceFailed 6 – OpenCircuit
[In] Engine Coolant Temperature [degC]	AnalogSensorBus	Requires the actual Engine Coolant Temperature signal value to be used for internal logic for Fuel Limiting.

		Bus Details: AnalogSensorBus: - FilteredValue - MeasurementStatus_enum MeasurementStatus_enum: (lesp_measurement_status_enum) 0 – Valid 1 – Disabled 2 – SignalLow 3 – SignalHigh 4 – SensorSupplyFailed 5 – ReferenceSourceFailed 6 – OpenCircuit
[In] Speed Control Fuel Demand [mg/cyl]	AnalogSensorBus	Requires the actual Speed Control Fuel Demand when an external speed control is used (Internal Speed control is off). Bus Details: AnalogSensorBus: - FilteredValue - MeasurementStatus_enum MeasurementStatus_enum: (lesp_measurement_status_enum) 0 – Valid 1 – Disabled 2 – SignalLow 3 – SignalHigh 4 – SensorSupplyFailed 5 – ReferenceSourceFailed 6 – OpenCircuit
[In] Back to Low Idle Fault Action	boolean	This input must be used when a Back to Low Idle command action is required. When TRUE, it will activate the Low Idle Speed Setpoint and is meant to come from any of the diagnostics that need to drive this action.
[In] Keep Rated Speed Fault Action	boolean	This input must be used when a Keep Rated Speed command action is required. When TRUE, it will keep the Rated Speed Setpoint and is meant to come from any of the diagnostics that need to drive this action (e.g., breaker closed, connected to net/grid).
[In] Power limit 1 Fault Action	boolean	This input must be used when a Power Limit 1 command action is required. When TRUE, it will activate a power limit function belonging to the power limit 1 settings and come from any of the diagnostics that needs to drive this action.
[In] Power limit 2 Fault Action	boolean	This input must be used when a Power Limit 2 command action is required. When TRUE, it will activate a power limit function belonging to the power limit 2 settings and comes from any of the diagnostics that need to drive this action.
[In] Power limit 3 Fault Action	boolean	This input must be used when a Power Limit 3 command action is required. When TRUE, it will activate a power limit function belonging to the power limit 3 settings and comes from any of the diagnostics that needs to drive this action.
[In] External Derate Request [%]	single	This input must be used when an external Derate Request comes in from an external device.

[In] GC1 PGN64915 Timeout Active	boolean	This input must be used to report any issue when the GC1 Message (PGN64915) is failing or is having a timeout.
[In] GC1 PGN64915 SPN5568 (%)	single	This input comes from a CAN J1939 signal reporting the PGN64915 SPN 5568 (GC1) to set the Droop Percentage for the speed control logic.
[In] GC1 PGN64915 SPN5568 Status [enum]	Uint8	This input must be used to report the actual status of the CAN J1939 PGN64915 SPN 5568 (GC1) message. The Enum: lesp_j1939_spn_status_enum: 0 - Valid 1 - Error 2 - Reserved 3 - Not Available
[In] GC2 PGN61470 Timeout Active	boolean	This input must be used to report any issue when the GC2 Message (PGN61470) is failing or is having a timeout.
[In] GC2 PGN61470 SPN3938 (%)	single	This input must be used from a CAN J1939 signal reporting the PGN61470 SPN 3938 (GC2) to set the Generator Governing Bias Signal (+/-) which can be used for synchronizing, load sharing, or other bias purposes.
[In] GC2 PGN61470 SPN3938 Status [enum]	Uint8	This input must be used to report the actual status of the CAN J1939 PGN61470 SPN 3938 (GC2) message. The Enum: lesp_j1939_spn_status_enum: 0 - Valid 1 - Error 2 - Reserved 3 - Not Available
[Out] Demanded load [%]	single	This output defines the required load demand to keep the speed control under control. It is a percentage and can be transferred into fuel amount for the application it is used in.
[Out] Fuel Demand [mg/cyl]	single	This output defines the required fuel demand to keep the speed control under control. It is in mg/cyl. and can be used as actual fuel amount in the application it is used in.
[Out] Load Dump Detected	boolean	This output defines if a Load Drop / Dump is detected. This can be used in the application as feedforward signal to set engine devices faster in a correct position instead of waiting for the control logic to act.
[Out] Derate Request [%]	single	This output defines the required derate as commanded from the speed control logic. Under no circumstances can an automatic derate take place as it could result in engine shutdown / power cut. For those reasons, this output can be sent to the load management system to remove load or trip non-essential users.
[Out] Max Fuel Limiter Active	boolean	This output confirms if the maximum fuel limiter is active. This should never happen under normal circumstances, but it can indicate that there is a leak somewhere or another issue is ongoing on the engine side.
[Out] Near Fuel Limiter Active	boolean	This output confirms if the actual amount of requested fuel is close to a fuel limiter. This can

		be used as indication for Changeable Pitch Propellor systems, to reduce pitch to prevent the engine going down under overload conditions.

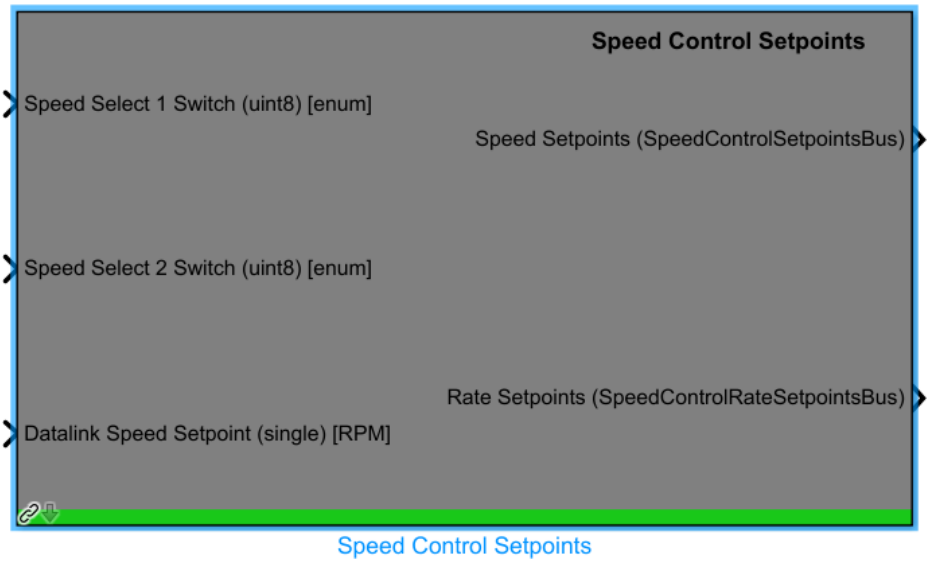


Figure 5-4. 'Speed Control Setpoints' Block

Table 5-3. 'Speed Control Setpoints' Block

Port	DataType	Description
[In] Speed Select 1 Switch [enum]	uint8	This input requires the actual status of the Speed Select 1 Switch input. This is used to determine the actual rated speed setpoint. The enum to look for is the: lesp_discrete_in_state_enum 0 – Not Used 1 – Inactive 2 – Active
[In] Speed Select 2 Switch [enum]	uint8	This input requires the actual status of the Speed Select 2 Switch input. This is used to determine the actual rated speed setpoint. The enum to look for is the: lesp_discrete_in_state_enum 0 – Not Used 1 – Inactive 2 – Active
[In] Datalink Speed Setpoint [RPM]	Single	This input requires the actual Datalink Speed Setpoint command. It can be summed from many other Datalink setpoints where the application can do some arbitration.
[Out] Speed Setpoints	SpeedControlSetpointBus	This output provides the actual Speed Setpoint active at that moment. This can be found in the Enumeration feedback when using the: SpeedControlSetpointsBus -IdleSpeedSetpoint_RPM

		-RatedSpeedSetpoint_RPM -MaximumSpeedSetpoint_RPM -MinimumAnalogSpeedSetpoint_RPM -MaximumAnalogSpeedSetpoint_RPM -DatalinkSpeedSetpoint_RPM
[Out] Rate Setpoints	SpeedControlRateSetpointBus	This output provides the actual Rate Setpoint active at that moment. This can be found in the Enumeration feedback when using the: SpeedControlRateSetpointsBus -InstantRate_rpmpers -AccelerationRate_rpmpers -DecelerationRate_rpmpers -DigitalRaiseRate_rpmpers -DigitalLowerRate_rpmpers -AnalogFastRate_rpmpers -AnalogNormalRate_rpmpers -DatalinkRate_rpmpers

Example Model Using the Blocks

The following is a very simple setup using MotoHawk Calibration and MotoHawk Probes blocks to make the model suitable for building.

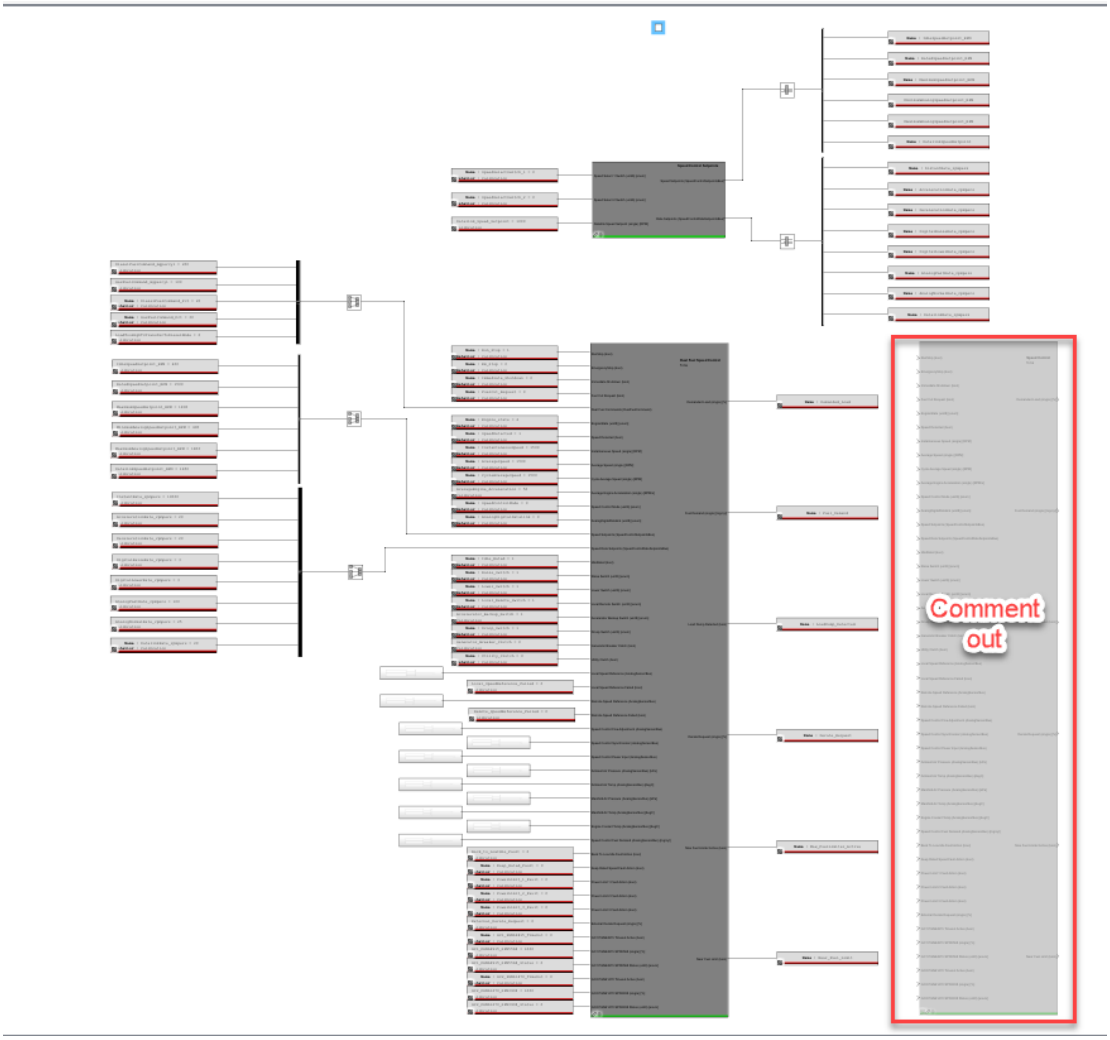


Figure 5-5. Simple Example Model, Using the Control – Speed Control Blocks

In the above example, only the Dual Fuel Speed Control block is used. The normal Speed Control block is commented-out in grey but could easily replace the Dual Fuel Speed control block.

When compiling, it will generate the required files and if the Toolkit required blocks are also added to the application model, it will generate the required SID files for the Toolkit HMI tool creation.

Default ToolKit Pages

Open a new Toolkit application and select the correctly generated .sid file, as shown in the screenshot below.

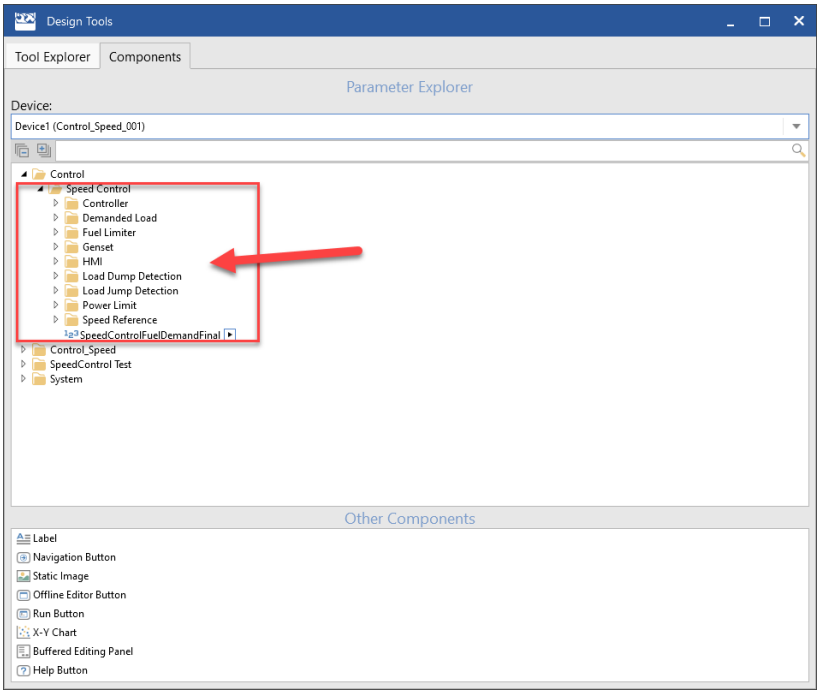
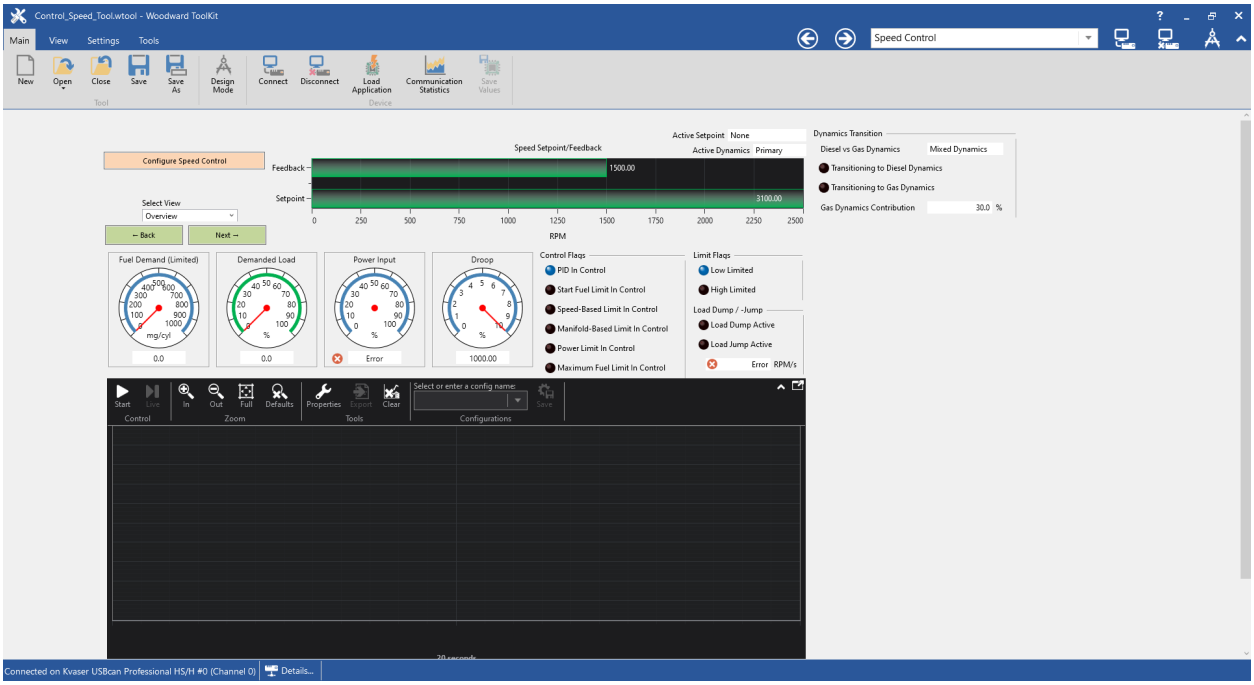
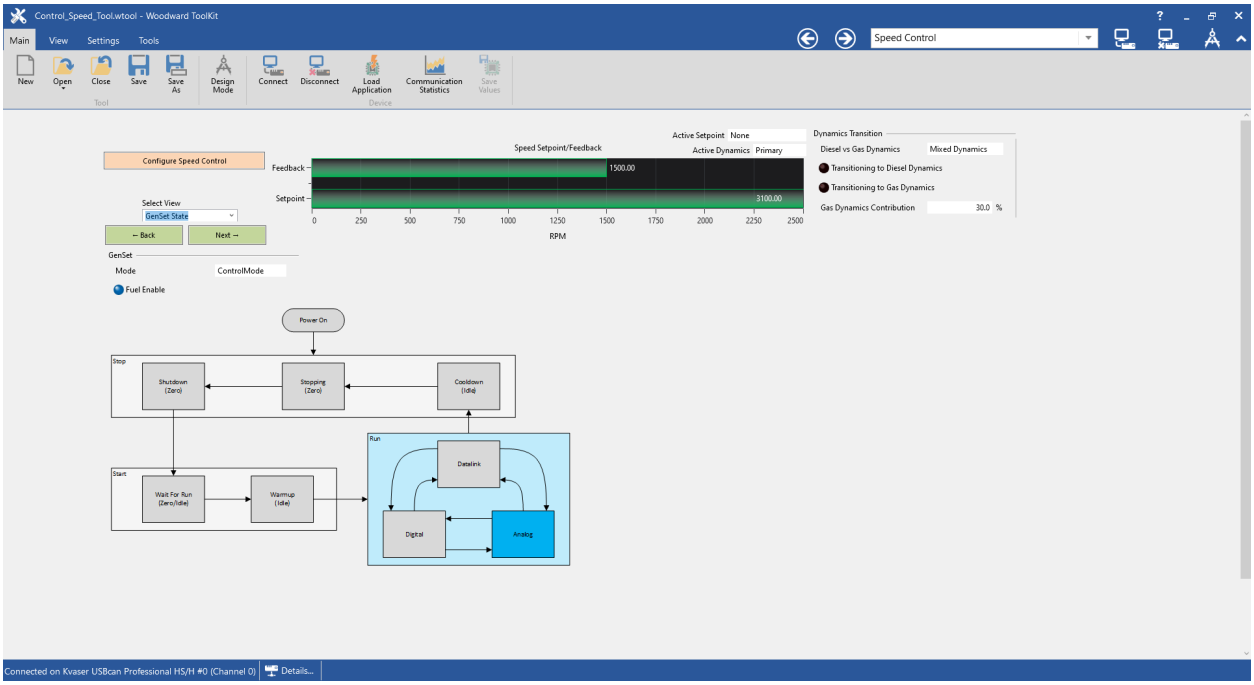
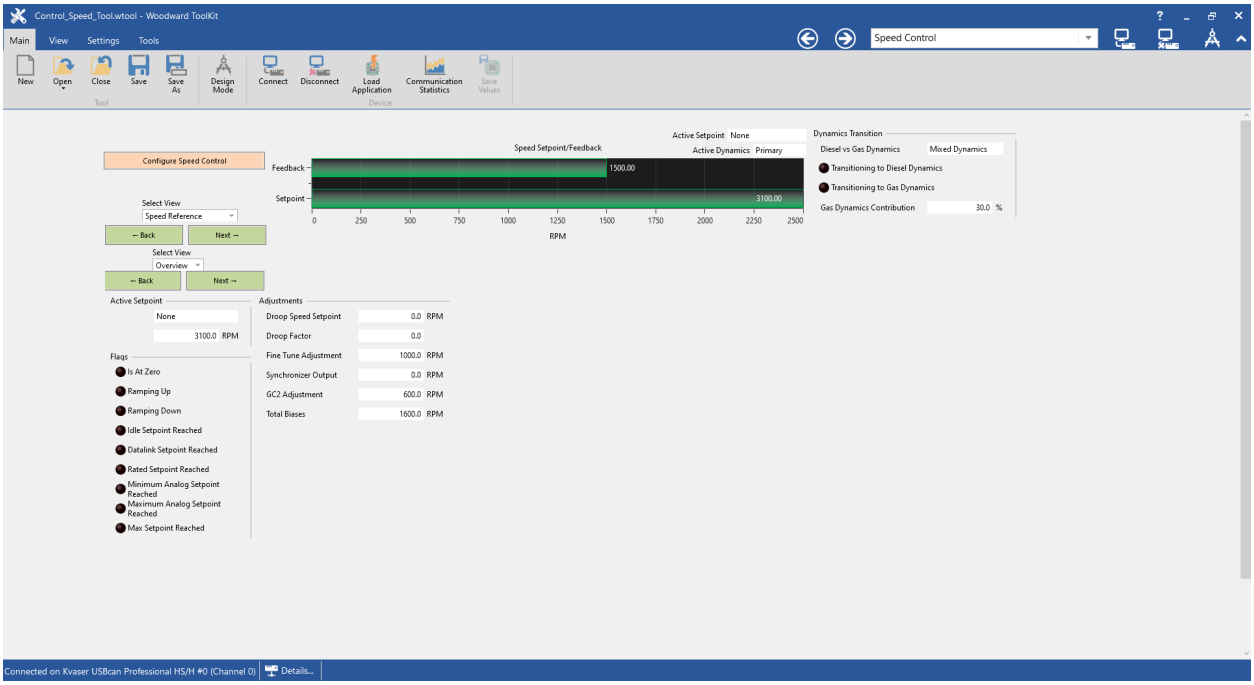


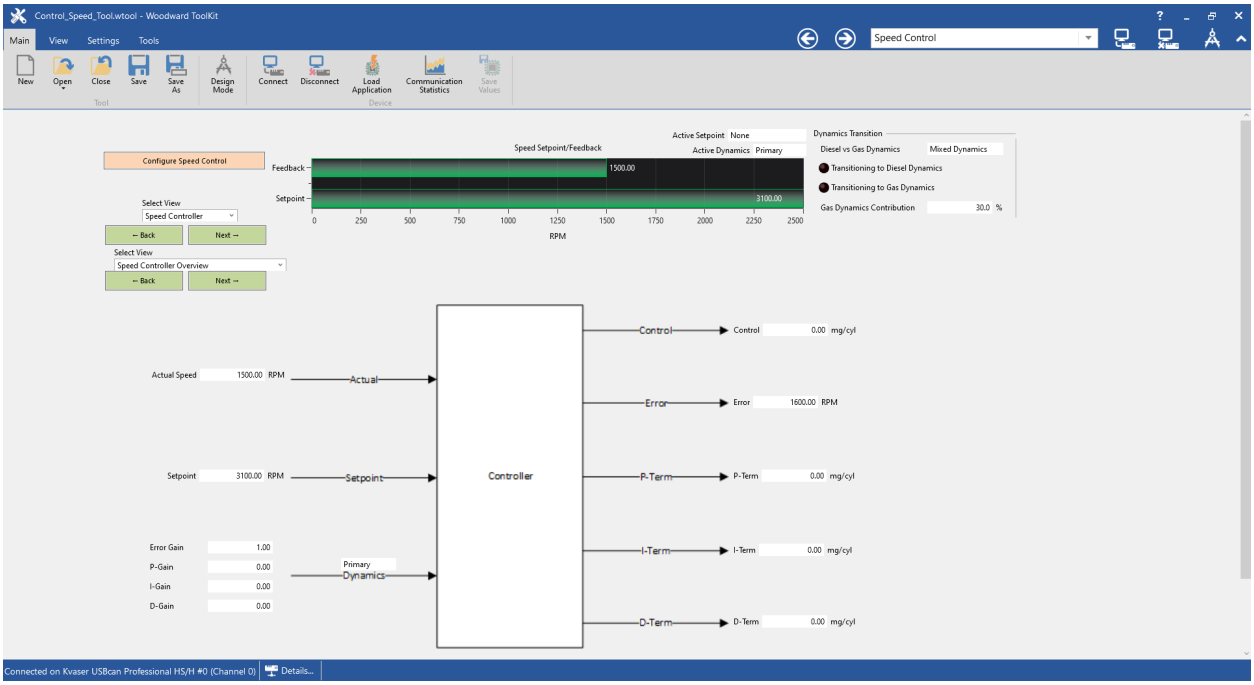
Figure 5-6. SID File Selection Control - Speed Logic

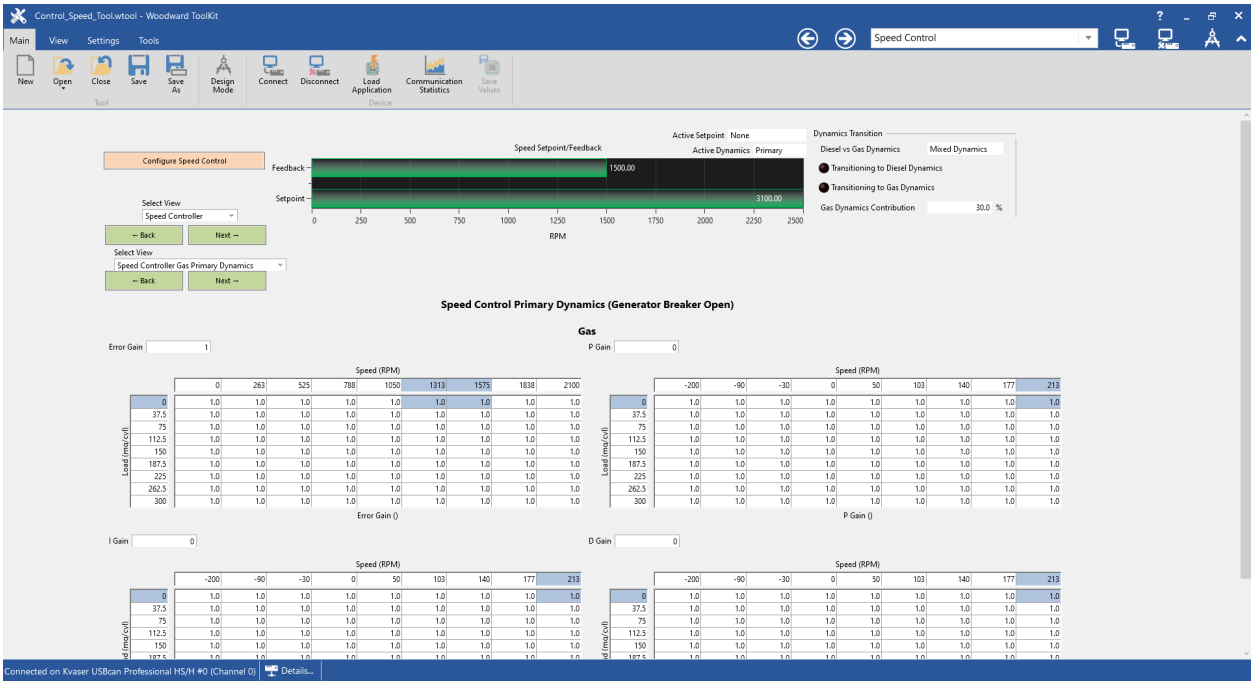
The following are a few of the Toolkit pages that will contain most of the defined parameters in the example model. Of course, the real sensors and data must be attached to complete it for the application that is being worked on. For that reason, some red X's appear.

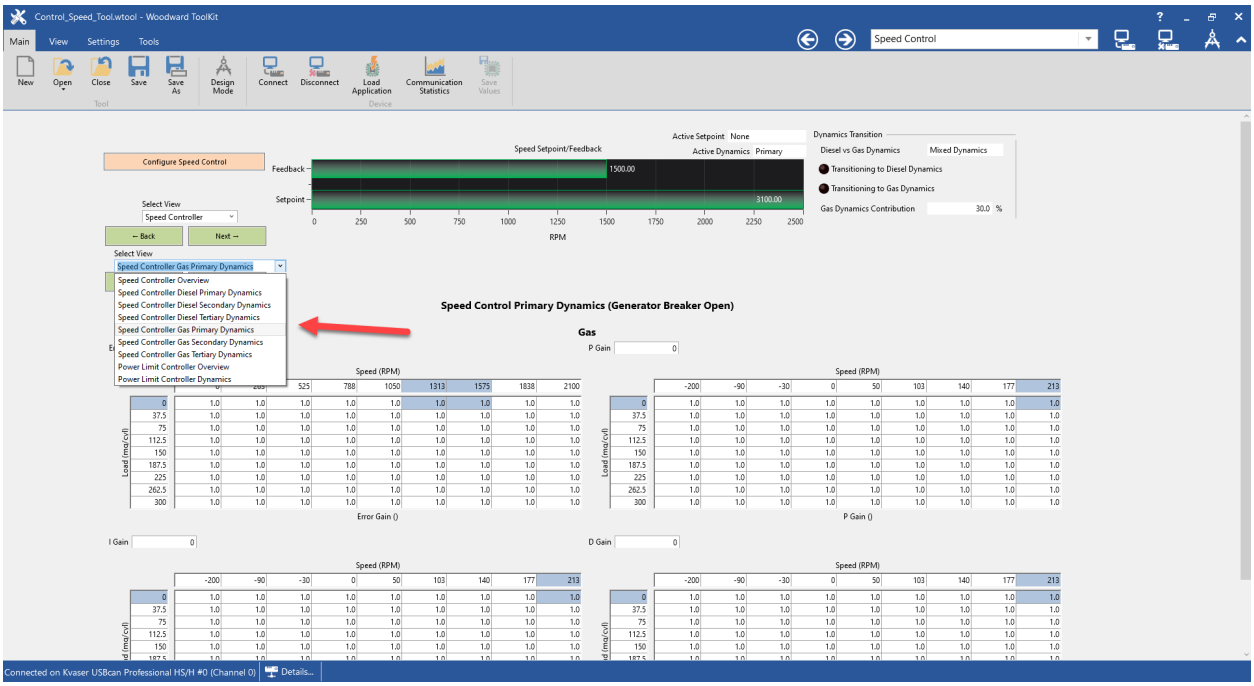

Control_Speed_Tool.wtool - Woodward ToolKit
Main
View
Settings
Tools
Speed Control
New
Open
Close
Save
Save As
Design Mode
Connect
Disconnect
Load Application
Communication Statistics
Save Values
Configure Speed Control
Speed Setpoint/Feedback
Feedback
Setpoint
1500.00
3100.00
RPM
Active Setpoint None
Active Dynamics Primary
Dynamics Transition
Diesel vs Gas Dynamics
Mixed Dynamics
Transitioning to Diesel Dynamics
Transitioning to Gas Dynamics
Gas Dynamics Contribution
30.0 %
Select View
Overview
Back
Next
Fuel Demand (Limited)
Demanded Load
Power Input
Droop
mg/cyl
Error
1000.00
Control Flags
PID In Control
Start Fuel Limit In Control
Speed-Based Limit In Control
Manifold-Based Limit In Control
Power Limit In Control
Maximum Fuel Limit In Control
Limit Flags
Low Limited
High Limited
Load Dump / -Jump
Load Dump Active
Load Jump Active
Error RPM/s
Start
Stop
In
Out
Full
Defaults
Properties
Export
Clear
Select or enter a config name:
Save
Control
Zoom
Tools
Configurations
Connected on Kvaser USBcan Professional HS/H #0 (Channel 0)
Details...


Control_Speed_Tool.wtool - Woodward ToolKit
Main
View
Settings
Tools
Speed Control
New
Open
Close
Save
Save As
Design Mode
Connect
Disconnect
Load Application
Communication Statistics
Save Values
Configure Speed Control
Speed Setpoint/Feedback
Feedback
Setpoint
1500.00
3100.00
RPM
Active Setpoint None
Active Dynamics Primary
Dynamics Transition
Diesel vs Gas Dynamics
Mixed Dynamics
Transitioning to Diesel Dynamics
Transitioning to Gas Dynamics
Gas Dynamics Contribution
30.0 %
Select View
GenSet State
Back
Next
GenSet
Mode
ControlMode
Fuel Enable
Power On
Stop
Shutdown (Zero)
Stopping (Zero)
Cooldown (Idle)
Start
Wait For Run (Zero/Idle)
Warmup (Idle)
Run
Datalink
Digital
Analog
Connected on Kvaser USBcan Professional HS/H #0 (Channel 0)
Details...


Control_Speed_Tool.wtool - Woodward ToolKit
Main
View
Settings
Tools
Speed Control
New
Open
Close
Save
Save As
Design Mode
Connect
Disconnect
Load Application
Communication Statistics
Save Values
Configure Speed Control
Speed Setpoint/Feedback
Active Setpoint None
Active Dynamics Primary
Feedback
Setpoint
1500.00
3100.00
0 250 500 750 1000 1250 1500 1750 2000 2250 2500
RPM
Dynamics Transition
Diesel vs Gas Dynamics
Mixed Dynamics
Transitioning to Diesel Dynamics
Transitioning to Gas Dynamics
Gas Dynamics Contribution
30.0 %
Select View
Speed Reference
← Back
Next →
Select View
Overview
← Back
Next →
Active Setpoint
None
3100.0 RPM
Adjustments
Droop Speed Setpoint 0.0 RPM
Droop Factor 0.0
Fine Tune Adjustment 1000.0 RPM
Synchronizer Output 0.0 RPM
GC2 Adjustment 600.0 RPM
Total Biases 1600.0 RPM
Flags
Is At Zero
Ramping Up
Ramping Down
Idle Setpoint Reached
Datalink Setpoint Reached
Rated Setpoint Reached
Minimum Analog Setpoint Reached
Maximum Analog Setpoint Reached
Max Setpoint Reached
Connected on Kvaser USBcan Professional HS/H #0 (Channel 0)
Details...


Control_Speed_Tool.wtool - Woodward ToolKit
Main
View
Settings
Tools
Speed Control
Configure Speed Control
Speed Setpoint/Feedback
Active Setpoint None
Active Dynamics Primary
1500.00
3100.00
RPM
Dynamics Transition
Diesel vs Gas Dynamics
Mixed Dynamics
Transitioning to Diesel Dynamics
Transitioning to Gas Dynamics
Gas Dynamics Contribution
30.0 %
Select View
Speed Controller
Select View
Speed Controller Overview
Actual Speed 1500.00 RPM
Setpoint 3100.00 RPM
Error Gain 1.00
P-Gain 0.00
I-Gain 0.00
D-Gain 0.00
Primary Dynamics
Actual
Setpoint
Controller
Control
Error
P-Term
I-Term
D-Term
Control 0.00 mg/cyl
Error 1600.00 RPM
P-Term 0.00 mg/cyl
I-Term 0.00 mg/cyl
D-Term 0.00 mg/cyl
Connected on Kvaser USBcan Professional HS/H #0 (Channel 0)
Details...


Control_Speed_Tool.wtool - Woodward ToolKit
Main
View
Settings
Tools
Speed Control
New
Open
Close
Save
Save As
Design Mode
Connect
Disconnect
Load Application
Communication Statistics
Save Values
Configure Speed Control
Speed Setpoint/Feedback
Feedback
Setpoint
1500.00
3100.00
RPM
Active Setpoint None
Active Dynamics Primary
Dynamics Transition
Diesel vs Gas Dynamics
Mixed Dynamics
Transitioning to Diesel Dynamics
Transitioning to Gas Dynamics
Gas Dynamics Contribution
30.0 %
Select View
Speed Controller
← Back
Next →
Select View
Speed Controller Gas Primary Dynamics
← Back
Next →
Speed Control Primary Dynamics (Generator Breaker Open)
Gas
Error Gain
P Gain
I Gain
D Gain
Speed (RPM)
Load (mg/cyl)
Error Gain ()
P Gain ()
Connected on Kvaser USBcan Professional HS/H #0 (Channel 0)
Details...


Control_Speed_Tool.wtool - Woodward ToolKit
Main
View
Settings
Tools
Speed Control
Configure Speed Control
Speed Setpoint/Feedback
Feedback
Setpoint
1500.00
3100.00
RPM
Active Setpoint None
Active Dynamics Primary
Dynamics Transition
Diesel vs Gas Dynamics
Mixed Dynamics
Transitioning to Diesel Dynamics
Transitioning to Gas Dynamics
Gas Dynamics Contribution
30.0 %
Select View
Speed Controller
← Back
Next →
Select View
Speed Controller Gas Primary Dynamics
Speed Controller Overview
Speed Controller Diesel Primary Dynamics
Speed Controller Diesel Secondary Dynamics
Speed Controller Diesel Tertiary Dynamics
Speed Controller Gas Primary Dynamics
Speed Controller Gas Secondary Dynamics
Speed Controller Gas Tertiary Dynamics
Power Limit Controller Overview
Power Limit Controller Dynamics
Speed Control Primary Dynamics (Generator Breaker Open)
Gas
P Gain
I Gain
D Gain
Speed (RPM)
Load (mg/cyl)
Error Gain ()
P Gain ()
Connected on Kvaser USBcan Professional HS/H #0 (Channel 0)
Details...

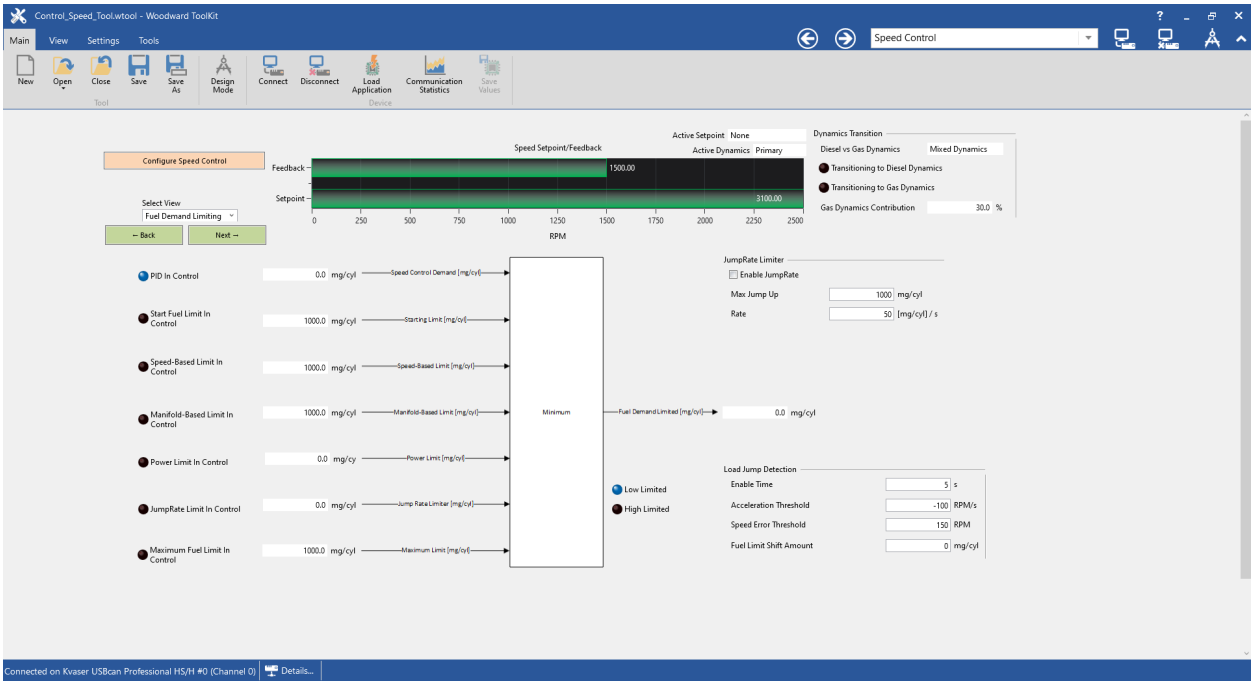


Figure 5-7. Example Toolkit Pages

Functional Description - Speed Control

The following describes the implementation and supported features of the Speed Control logic in the standard LECM CRS software. There is a single all-speed governor included in the software that has been designed to support all speed control functionality for either variable / all speed applications or stationary applications.

The design of the governor utilizes three key logical sections – a generator state machine, speed setpoint manager, and a closed loop proportional-integral-derivation (PID) with gain scheduling controller. In addition to the speed error, the PID strategy also utilizes information from the governor state machine in order to either hold or reset the P, I, and D terms when commanded. The P, I, and D gains in the PID controller also contain gain scheduling tables to allow for non-linear responses to speed errors.

Engine Speed Control Overview

The LECM Control System contains all speed governing functionality for engine speed control and engine power control. Speed settings support two different modes:

1) Analog Speed Setting Mode
   For marine use, compressor drive, or other variable speed control applications, engine speed depends on a target engine speed signal from another device. This can be a combination of a bridge and/or engine control room, potentiometer at engine side, PLC system, or other analog speed setting device.

2) Digital Speed Setting Mode
   For stationary/genset use, engine speed is normally operated from a fixed working speed setpoint, denoted as “rated” speed. Digital speed setting inputs, including Idle/ Rated, Raise / Lower contact inputs, etc., are used to operate the engine in this mode.

Some applications require a combination of both above mentioned speed setting modes. For example, engines used in a propulsion application where a shaft generator is connected to the gearbox.
For such applications, a mode selection switch can be used to switch between the analog or digital speed setting modes.

Selection between Isochronous and droop operation is available in both speed setting modes. *Isochronous* means that speed is fixed at rated speed level. An additional load sharing device is required in multiple engine applications if the load needs to be shared between engines. *Droop* means the engine speed setpoint will drop x% when load is increasing. Isochronous and droop operations are used in parallel to grid applications and/or load sharing between engines, for which speed drops are tolerated. Often these applications have a load management system above the speed control to correct for speed / frequency deviations (raise/lower commands). Switching between the different modes is “bump-less,” allowing a smooth transition between speed setpoint changes.

The engine speed control function determines the “required fuel injection quantity” in order to maintain target engine speed. Maximum engine speed is limited by the speed control function to prevent engine overspeed. The overspeed detection function is independent from speed control algorithm.

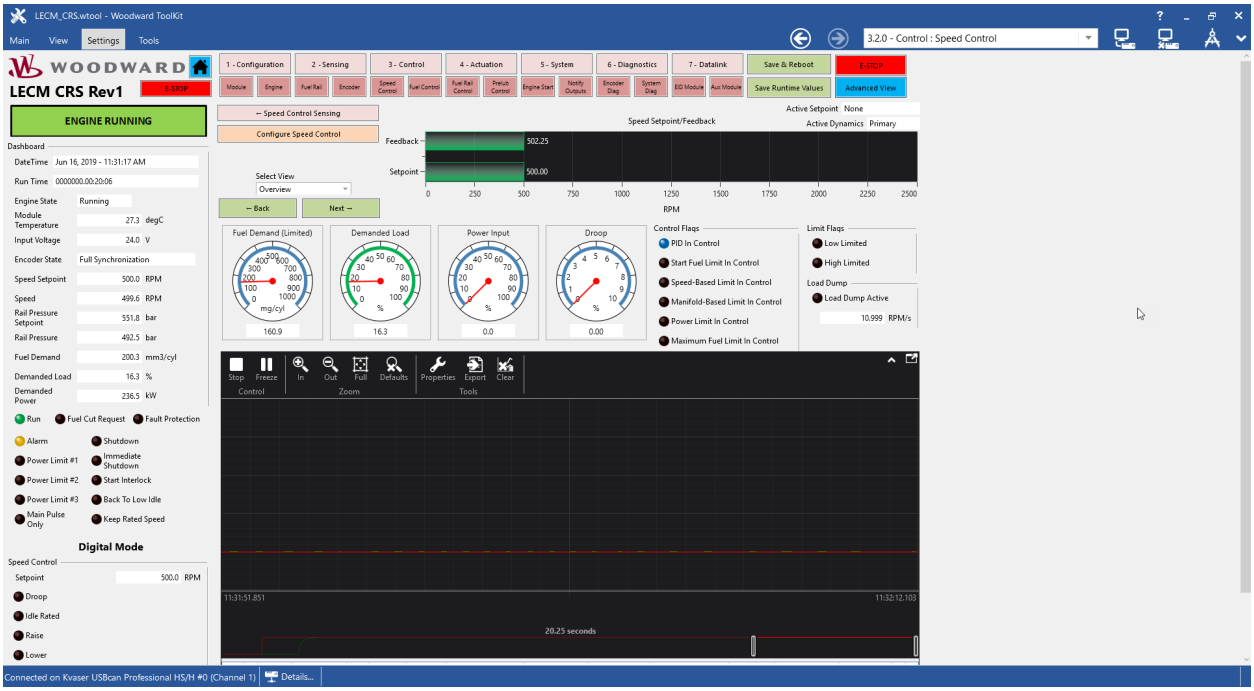


Figure 5-8. Speed Control Overview

Analog Speed Setting Mode

When the engine is started in analog speed setting mode, the speed reference will be set to idle speed setpoint. After the warmup phase is completed, the speed reference will be ramped and follow the setpoint as set by the analog input. The speed setpoint is set by the local/remote speed reference analog inputs. The local/remote discrete input is used to select which local/remote speed reference actively controls the speed setpoint. Changes to setpoints will always ramp to avoid step changes.

When in analog speed setting mode, the idle speed setpoint and maximum speed setpoint are the lower and upper speed limits. A tunable exists for both acceleration and deceleration rates to prevent sudden speed setpoint variations.

The analog speed setting is controlled with a deadband controller. The deadband controller utilizes two deadbands:

a) DB#1--> Inside this range, the deadband is satisfied and no raise/lower commands will be given.
b) DB#2 --> Outside this deadband, an additional output is active (DB2 Exceeded). This is used to ramp at a faster rate to react more quickly when outside the defined window.

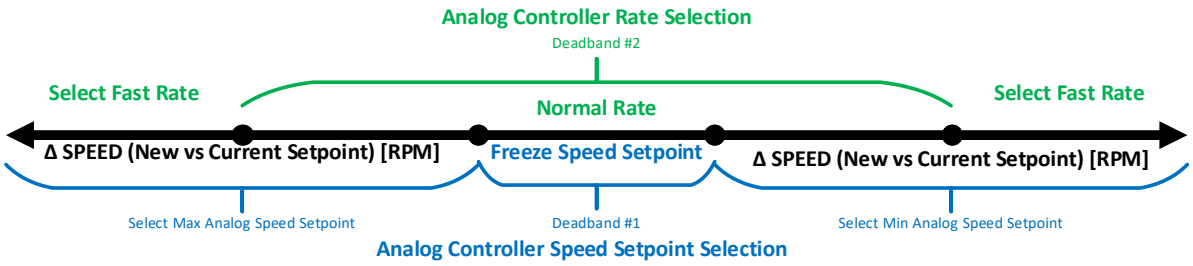


Figure 5-9. Deadband Overview

With errors between DB#1 and DB#2, the 'Analog Speed Change Rate' will be used. Outside DB#2, the 'Analog Speed Change Fast Rate' will be used. The following figure shows the different settings that are

available when in Analog Speed Setting mode.

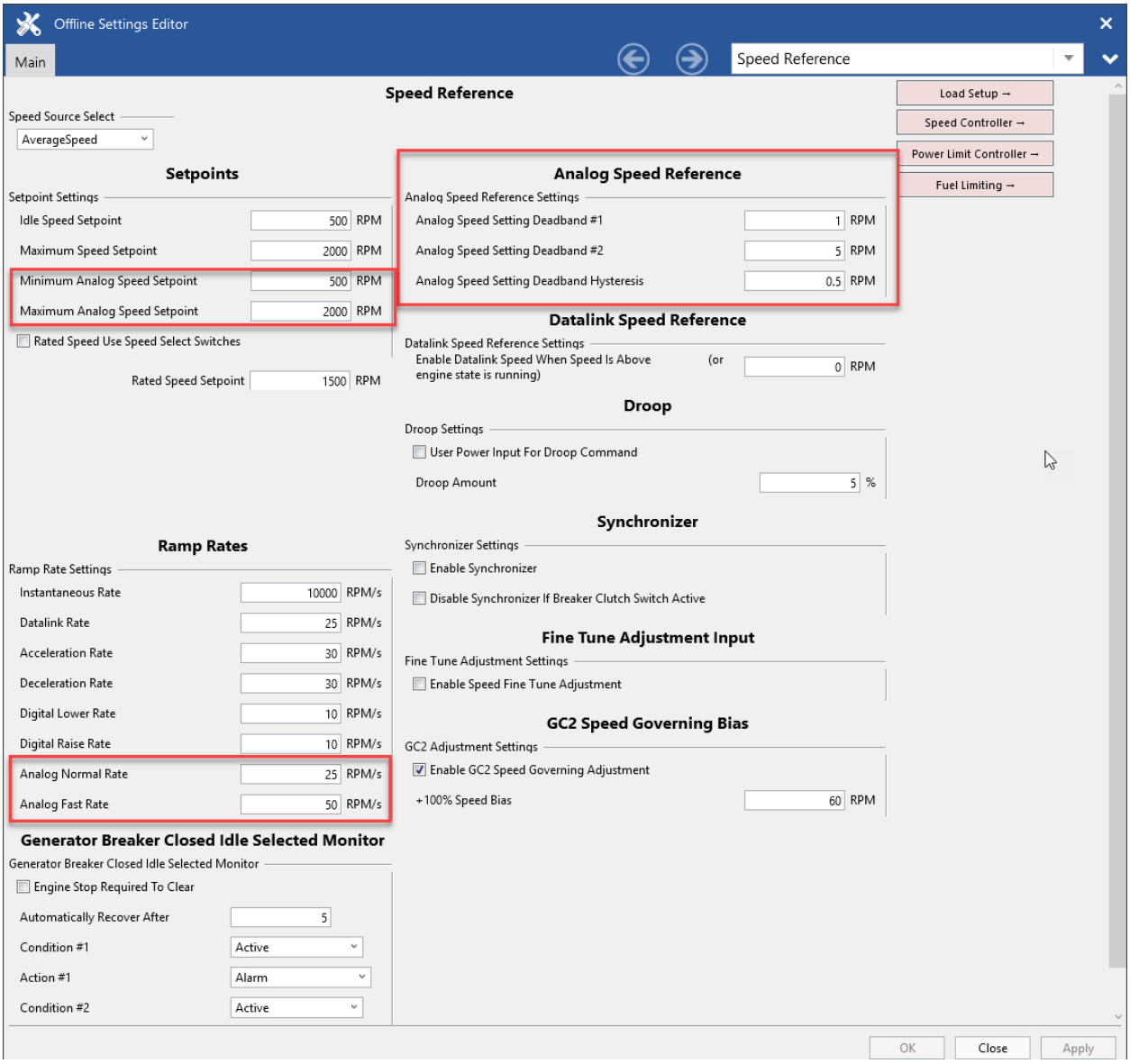


Figure 5-10. Analog Speed Setting Calibration Values

When digital control is required, even when in analog speed setting mode, the accelerator backup contact input can be activated, causing the speed reference to freeze its actual setpoint. Once frozen, the raise and lower contact inputs can be used to modify the speed setpoint. As the accelerator backup contact input de-activates, the speed reference will be ramped to the actual analog speed reference setpoint that will be active and commanded. This can be either higher or lower than the actual speed setpoint that is active at switch-over.

If a failure of the Local/Remote speed reference setpoint is detected while active, the speed setpoint will freeze at its last known healthy position. By means of the accelerator backup contact input switch, the raise lower contact inputs can be activated to control the engine again. The setpoint freeze action normally stays active, until the signal is healthy again and an operator reset command has been issued to the control. Setup of this behavior can be changed, depending on the action chosen on the failure mode.

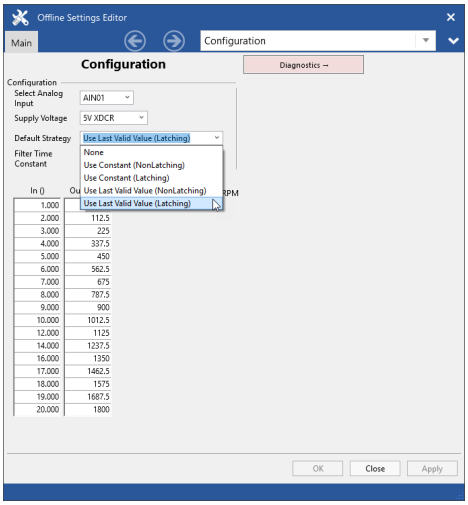


Figure 5-11. Strategy Selection

Analog Speed Setting Mode Calibration Guide

Start the calibration by connecting the input to either a potentiometer, 4–20 mA source or 0–5 VDC source according to the control wiring diagram as applicable to your application.

Calibrate the range where the analog speed setting device will be used. Use the table setup to set the range correctly (0–100%), so that a fault can also be detected.

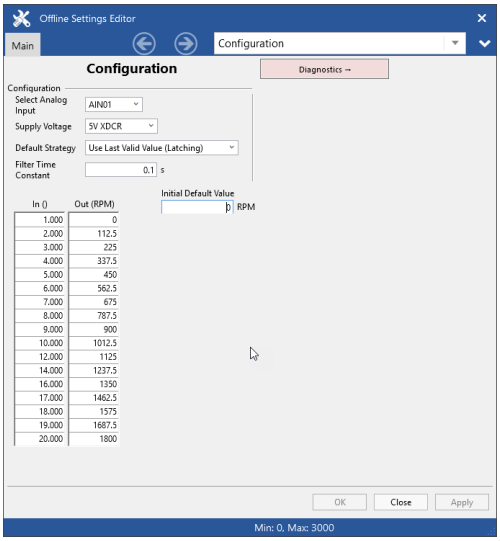


Figure 5-12. Engineering Unit Setup

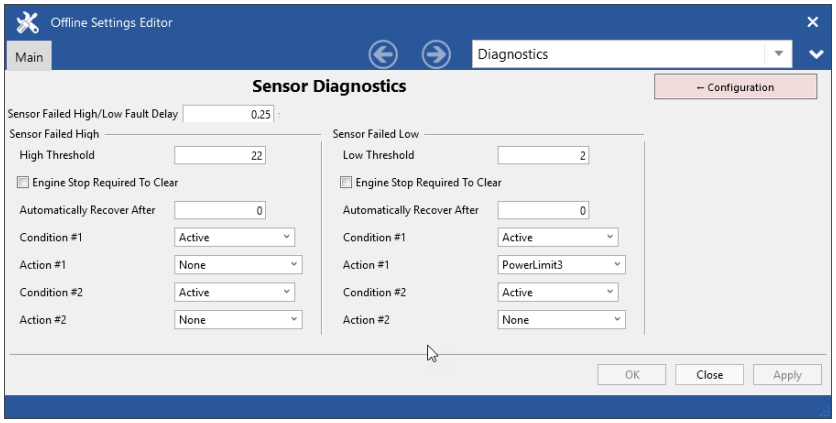


Figure 5-13. Diagnostic Setup

Digital Speed Setting Mode

When the engine is started in digital speed setting mode, the speed reference is set to the idle speed setpoint. After the warmup phase is complete, the idle/rated contact input can be activated (idle/rated select = 1), the speed reference will accelerate with the acceleration rate to the rated speed reference setpoint.

When at the rated speed setpoint and idle/rated contact input is deactivated (idle/rated select = 0), the speed reference will decelerate with the deceleration rate to the idle speed reference setpoint. If a raise and/or lower command is received during the ramping of the speed reference from idle to rated, or rated to idle, the automatic ramping with acceleration or deceleration rate is stopped. The commands from the raise and/or lower contact inputs will be followed with the corresponding raise and lower rates, depending on which input is active. Toggling the idle/rated contact input again will start the automatic ramping again (idle/rated select = 0 → idle, idle/rated select = 1 → rated).

When in digital speed setting mode, if the raise contact input is active, the actual speed reference is increased with the raise rate as long as the contact input remains closed. When in digital speed setting mode, if the lower contact input is active, the actual speed reference is decreased with the lower rate as long as the contact input remains closed. In the case where both the raise and lower contact inputs are set at the same time, only the lower command will be active.

The idle speed is the lowest speed reference setpoint. The rated speed is the setpoint for which the stationary engine operates. The maximum speed setpoint is an absolute maximum setpoint. This setpoint must be set lower than the overspeed trip setpoint level and above a level for which full load operation is possible in case droop is activated.

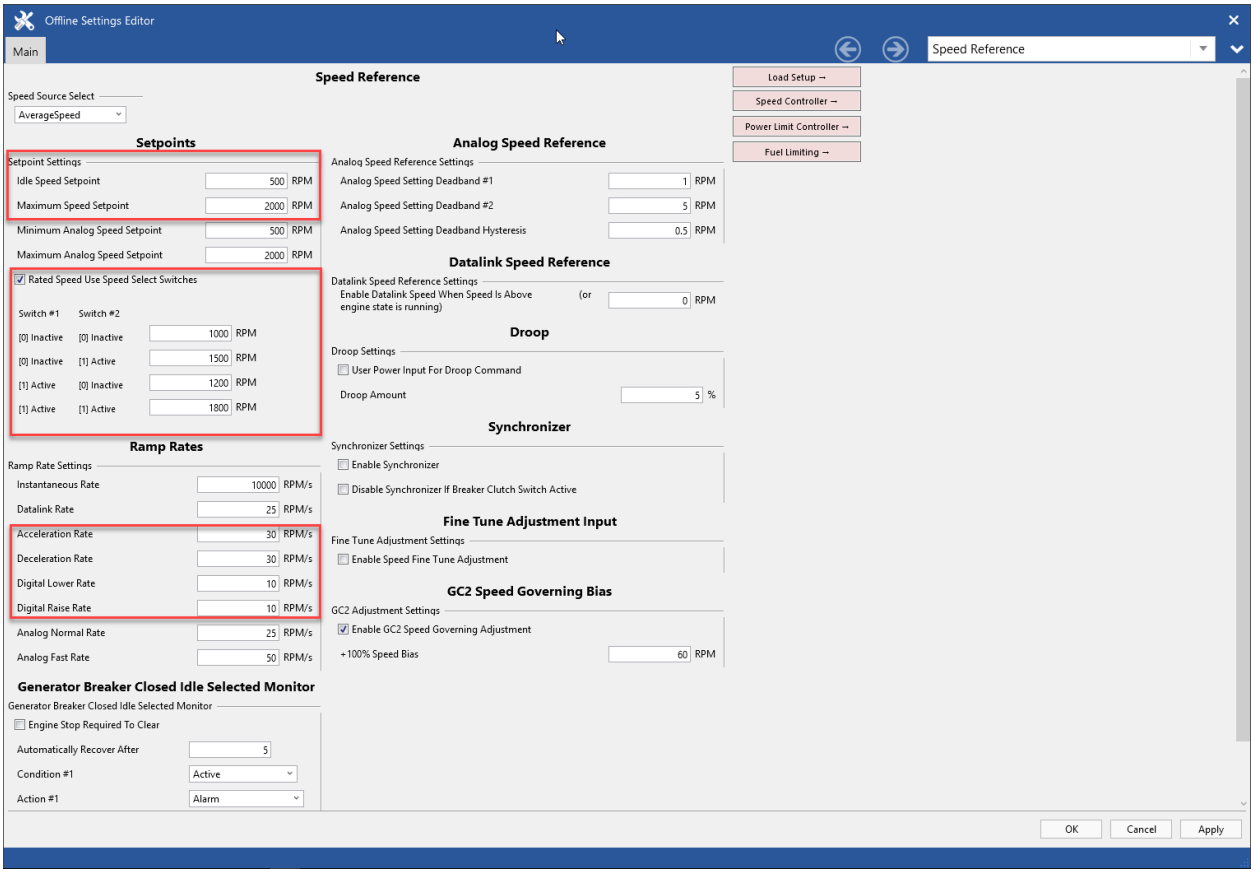


Figure 5-14. Digital Speed Setting Calibration Parameters

Digital Speed Setting Mode Calibration Guide

Set the speed setpoints correctly prior to starting the engine. During commissioning, set the acceleration and deceleration rates— the engine can follow without less overshoot.

The Rated Speed Reference setpoint can either be set by a separate parameter or can be selected to be dependent on the status of two speed select switches. The speed select switches could be used as wiring harness jumpers for engines with different rated speed level applications (50Hz/60Hz usage).

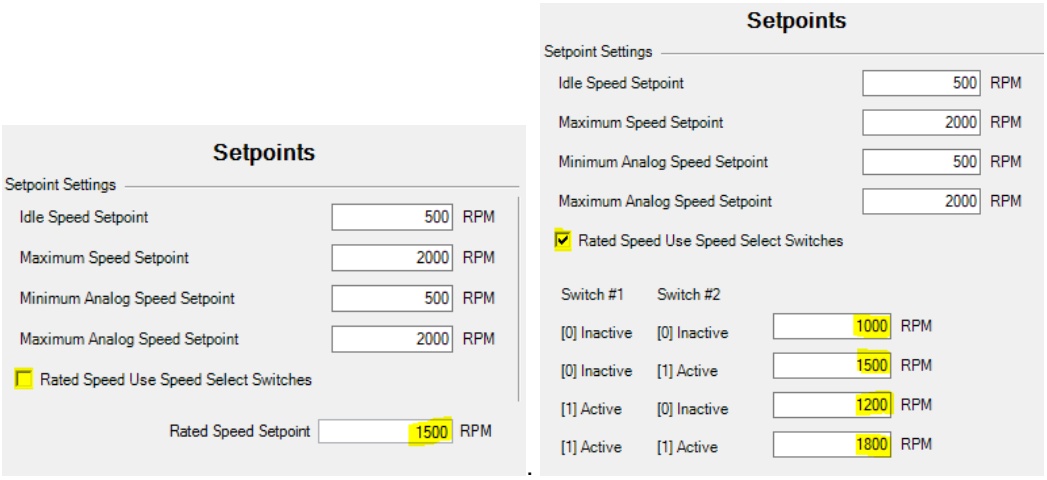


Figure 5-15. Different Rated Speed Setpoint Selection Options

Speed Control Logic

This chapter describes the speed control logic as used in the all-speed governor setup of the LECM CRS control.

Speed Sensing

The speed sensing logic for the LECM CRS logic is setup with the encoder definitions as is required to determine the crank angular domain for the fuel injection logic. The encoder definition logic is described in a separate application note. For the speed control, the actual speed sensing logic is derived from the encoder signals and is utilized by the speed control logic with three selectable speed sources.

1) **Instantaneous Speed**
   Instantaneous speed is an unfiltered measure of engine speed. It is calculated using the most recent high resolution measure of the time taken to traverse two encoder teeth. The result is often noisy as it calculates speed based on short-term data. It is most useful for slow speed conditions or very rapidly changing speed conditions (like engine startup). At rated speed, this measurement is often too noisy for speed control purposes and filtering it results in a slower response time than using AvgRPM or CycleAvgRPM.

2) **Average Speed**
   Average speed is calculated based upon the time taken to traverse adjacent RPM sample points (TDC angles). It essentially averages the equivalent of the instantaneous speeds reported throughout the period between the most recent two sample points (TDC angles).

3) **Cycle Average Speed**
   Cycle average speed is calculated using a measured cycle period (accumulation of prior speed sample point timespans) and updates on each sample point that has been defined for the active encoder source. It essentially averages all average speed calculations over a full engine cycle and updates the result to the application at each sample point (TDC angle).

IMPORTANT

All speed calculations have a resolution of 1/256 revolutions per minute.
Speed calculations may not report correctly while the encoder is not fully synchronized.
All RPM sample points in the software are equivalent to the user-defined TDC angles.
All RPM sample points resolve to the closest critical tooth edge. For example, if the actual TDC angle lies in-between the observed tooth edges, the next tooth edge will be used as the sample point.

Each of the above options can be selected as active speed input for the speed control logic. Depending on the application needs, a faster or slower update may be required. Most applications should utilize the average speed input, as this is a compromise between stability and fast updates.

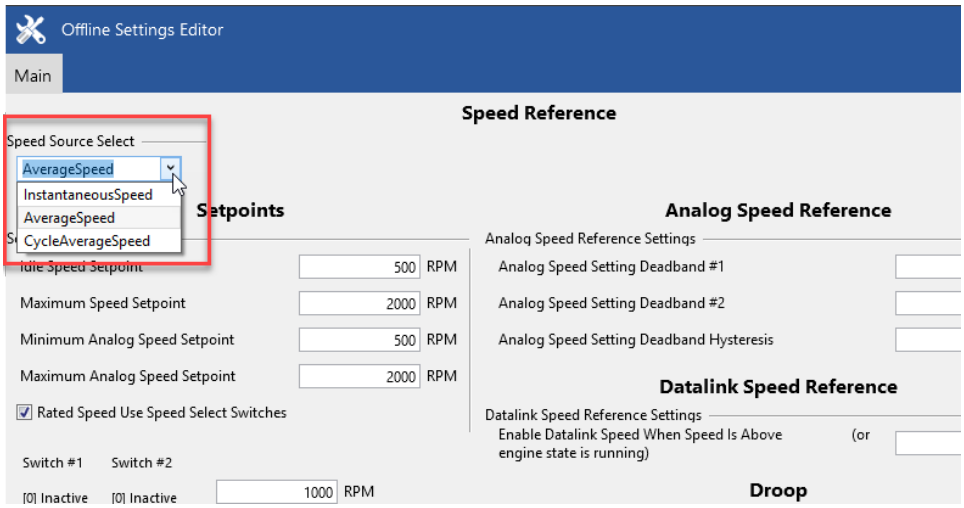


Figure 5-16. Speed Source Select Options

The encoder setup has the option to use redundancy. In that way, the speed updates have redundancy available. In case of an issue with the encoder (signal lost or wrong teeth count), an alarm will be flagged, and if configured, switching over to the backup encoder will become active. If that is not possible or if both encoder signals are not healthy, fuel injection will be stopped in the worst case scenario. Actual speed updates will report as zero since no fuel will be injected in the engine. Good diagnostic setup of the encoder faults will trigger an immediate shutdown in that case, and the engine will be stopped.

The speed control logic uses the encoder speed feedback as the actual speed signal. Together with the speed reference setpoint coming either from the digital and/or the analog speed reference logic, these signals are used on a PID input, as actual and reference input. The output of the PID will be used to determine the amount of fuel to be injected for each cylinder.

Background:

The PID is a proportional integral derivative control loop.

In general, three things take place in a closed loop control system— measurement, decision, and action—whenever the system is in automatic.

The measurement section is connected directly to the process (actual speed in this case) and transmits the variable to be controlled to the control station, usually some distance away. The control station, which technically includes the controller, allows comparison between the desired control point (set point) and the measured control point. The amount of the deviation, the rate of the deviation and the duration of the deviation all predict controller response according to the controller's gain, derivative, and reset settings.

Proportional control eliminates the cyclic characteristics of the on-off mode. Integral and derivative are refinements of, and work in conjunction with, proportional control.

Proportional Control:
Setting hand throttle to keep constant speed on straight and level. Proportional control results in a certain speed as long as the car is not subjected to any load change such as a hill.
If a hand throttle is set to any particular setting, the speed of the car will remain constant as long as the car remains straight and level. If the car goes up a hill, it will slow down. Of course, going down a hill the car would gain speed. Proportional response is directly proportional to a process change.

Integral Control:
Cruise control maintains constant speed regardless of hills. Integral (sometimes called reset) provides additional action to the original proportional response as long as the process variable remains away from the set point. Integral is a function of the magnitude and duration of the deviation. In this analogy, the reset response would keep the car speed constant regardless of the terrain. Integral compensates for process and set point load changes.

Derivative Control:
Accelerating into a high speed lane with merging traffic. Derivative is sometimes called "pre-act" or "rate" and is very difficult to draw an accurate analogy to because the action takes place only when the process changes and is directly related to the speed at which the process changes.
Merging into high speed traffic of a freeway from an "on" ramp is no easy task and requires accelerated correction (temporary overcorrection) in both increasing and decreasing directions. The application of brakes to fall behind the car in the first continuous lane or passing gear to get ahead of the car in the first continuous lane is a derivative action.

Proportional response improves the cyclic characteristics of on-off by providing a throttling action or some portion of the maximum available. It is proportional to the process variable change only.
Integral compensates for load changes either process initiated or set point initiated by adding additional proportional action until the measured variable and the desired control point (set point) are the same. Integral action takes place only when a difference (deviation) exists between the set point and the process measurement. Derivative accelerates (advances) the proportional action based on the speed of the process change only. It compensates for long transfer lags and reduces overshoot resulting from large process disturbances. Combinations of proportional, integral, and derivative will provide the type of process control required.

The LECM CRS-based application software also uses a PID control in order to act on different engine situations, like load changes, speed jumps, misfires, etc. Actual speed input will be compared with the speed reference setpoint. The output of the PID is determined as the fuel amount per cylinder. If speed drops and the error between actual speed and speed reference setpoint increases, the PID output will start to increase and vice versa. Setting the P, I, and D gains will provide for fast control causing the error between actual speed and reference to be recovered quickly in a stable manner.

Implementation:
The PID-based speed control logic is fully integrated in the LECM CRS software. The following screenshots provide an overview of the software implementation.

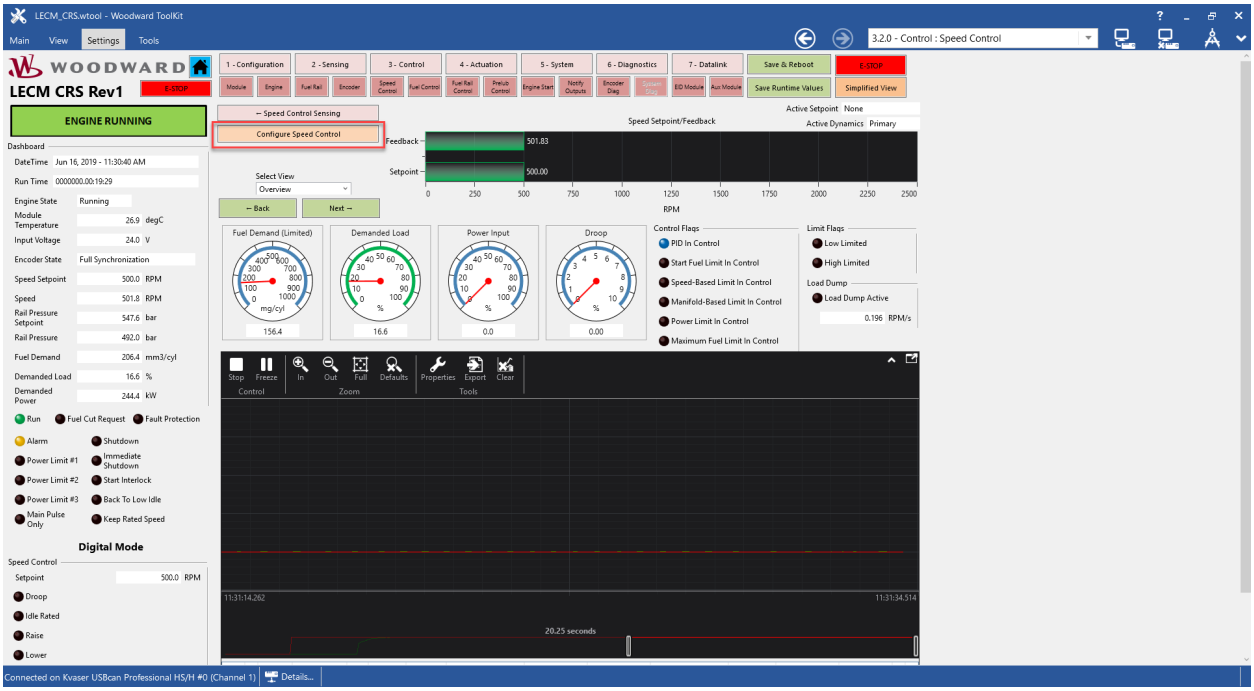


Figure 5-17. Speed Control Main Page

Clicking on the above red square button will bring you into the Configuration page of the speed control.

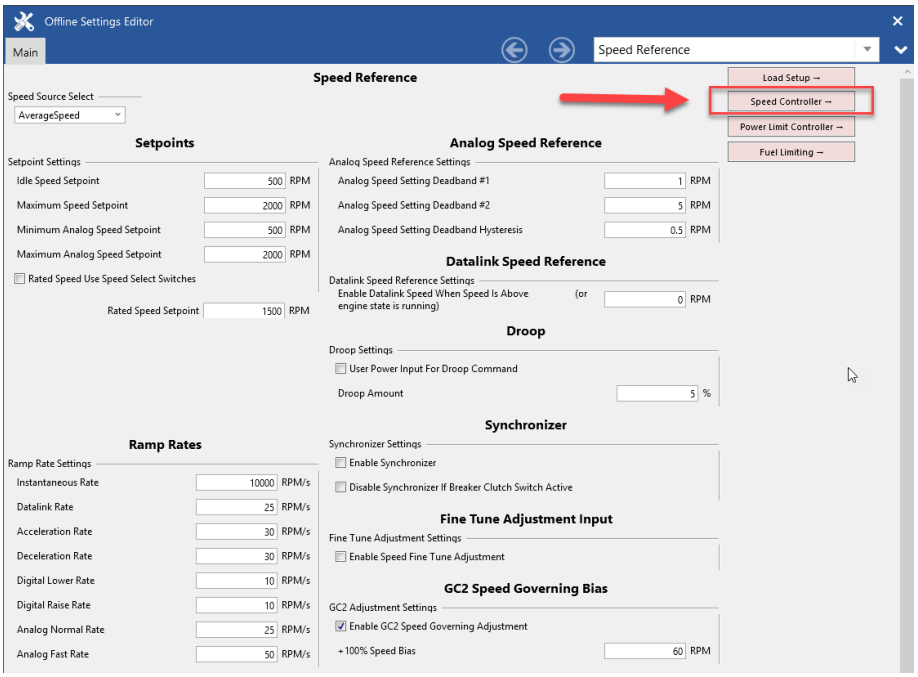


Figure 5-18. Speed Control Configuration

Click “Speed Controller” and a window will open where you can setup the PID configuration related items. The following is an overview of items that configures parts of the controller:

a) Min and max PID output values → This corresponds with min/max fuel amounts per cylinder
b) I-Term min and max values
c) I-Term hold function when PID is not in control
d) Speed Source Selection
e) Speed Control Dynamics set selections (max of 3 sets can be setup)

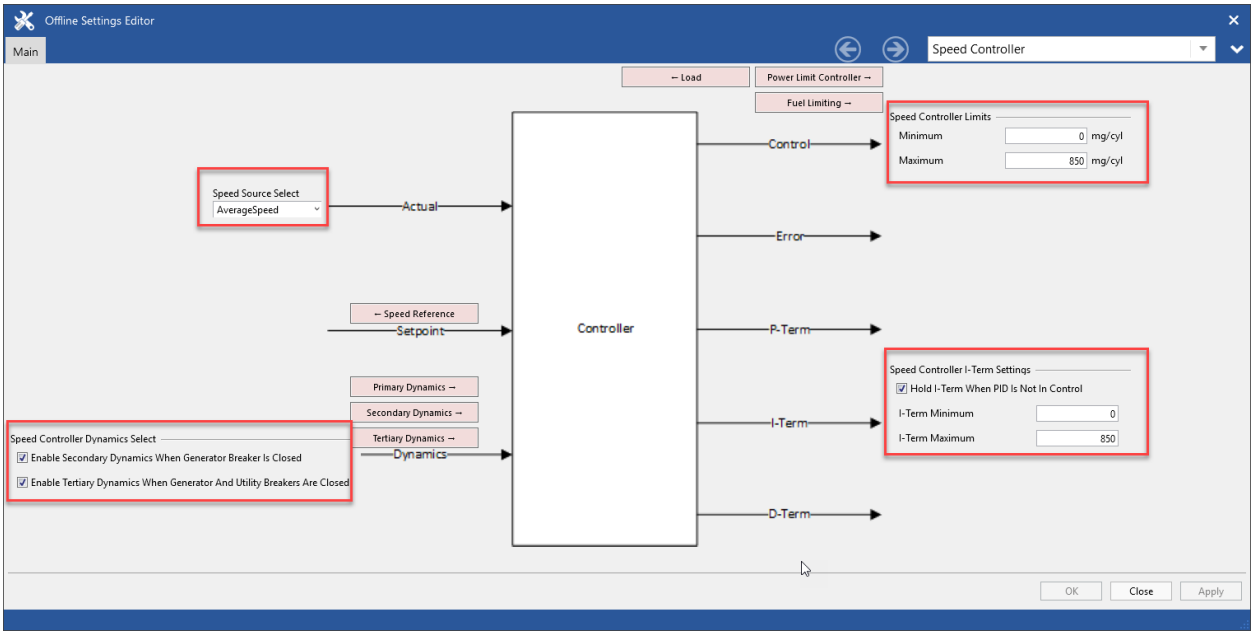


Figure 5-19. Main Speed Control Items to Setup

All of these features will determine the behavior of the PID.

On the main page of the Speed Control Toolkit page, you can select the page view. The available options are shown in the following screenshot.

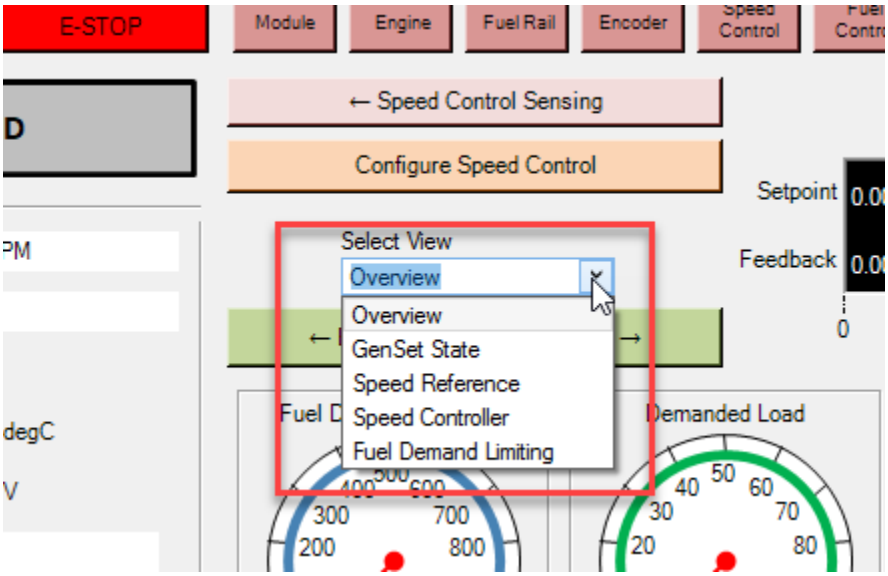


Figure 5-20. Select View – Speed Control Page

Each view will display the corresponding selected pages with possible settings to tune while running the engine.

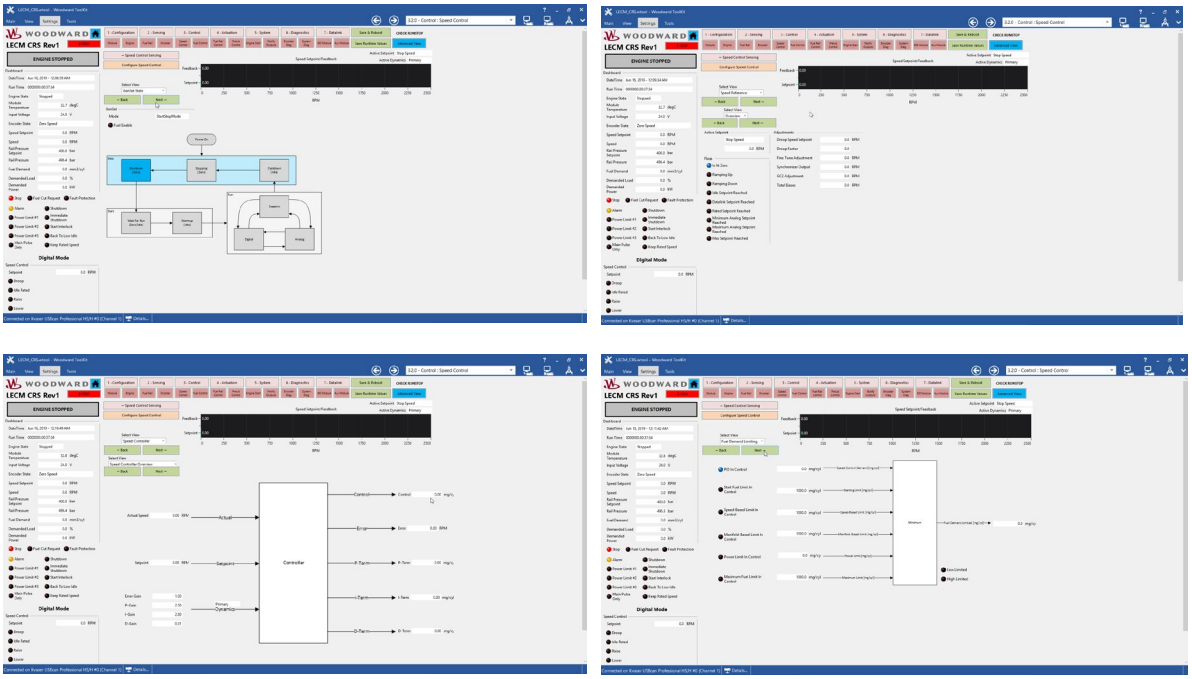


Figure 5-21. Overview of Possible Pages

For this section, the focus will be on the Speed Controller page. This specific page also has multiple selections, which will all have their dedicated tunable values.

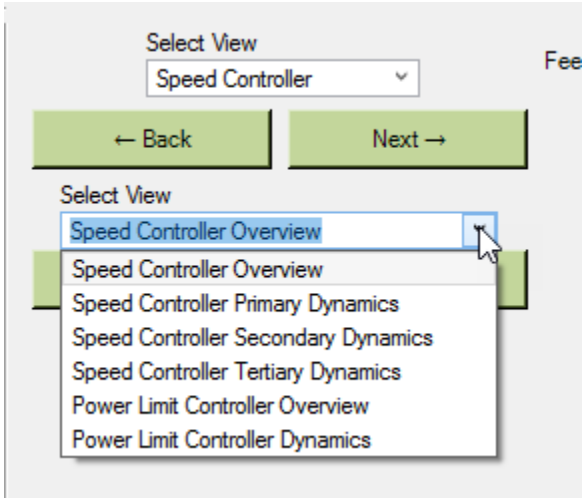


Figure 5-22. Speed Controller Pages

The Speed Controller Overview page is indicated in the figure above.

The next three pages display the different dynamics settings, providing the tuning options for the gain scheduling in that particular mode.

Primary dynamics are active when Breaker/Clutch feedback discrete input is open.
Secondary dynamics are active when Breaker/Clutch feedback discrete input is closed.
Tertiary dynamics are active when Breaker/Clutch feedback AND Utility Breaker feedback are closed.

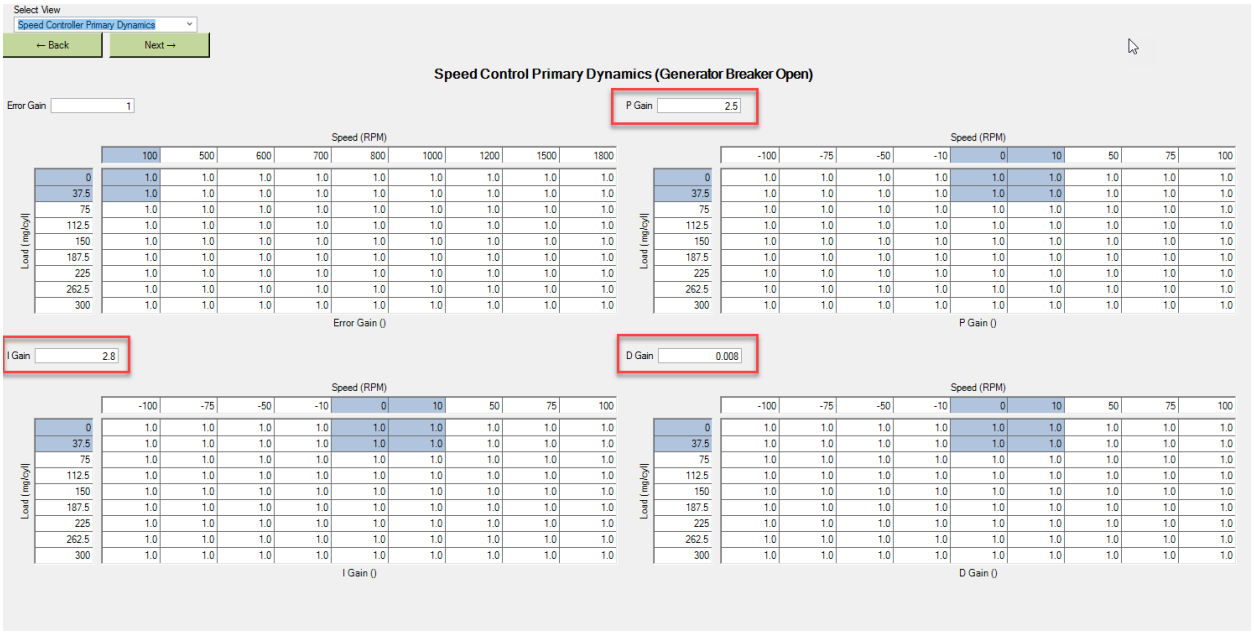


Figure 5-23. Example of Gain Scheduling Options

There is one generic P, I, and D value available. Each P, I, and D value includes a load and speed error multiplier factor to gain the value over the entire load and error range. These options make it possible to increase gains at higher loads, make the gains faster when the error is increasing, and vice versa. All these values are run time tunable values and must be calibrated during engine commissioning when used in a certain application. The optimum is reached when the engine response is fast enough, but engine speed stays stable under all engine operating conditions.

To improve engine performance even further, some more options are available.

Speed Control Load Dump / Load Jump Detection

Some unique features to the speed control strategy are the load dump / load jump detection scheme, which allows for faster governor fueling response when a load dump is detected (RPM increases rapidly).

Load Dump Detection

A strategy to detect a load dump is employed to help prevent RPM flares once the load from the electric generator is released from the engine. This strategy essentially works where acceleration of the engine must be larger than a threshold, and the RPM must be above the RPM setpoint by a threshold. To quickly remove fuel when a load dump is detected, the PID terms of the speed control PID controller are reset to zero. Once the load dump detection completes, the speed control PID will take control again and maintain actual speed at the speed reference setpoint active.

The load dump parameters can be setup in the offline configuration menu of the speed controller, going to the load selection.

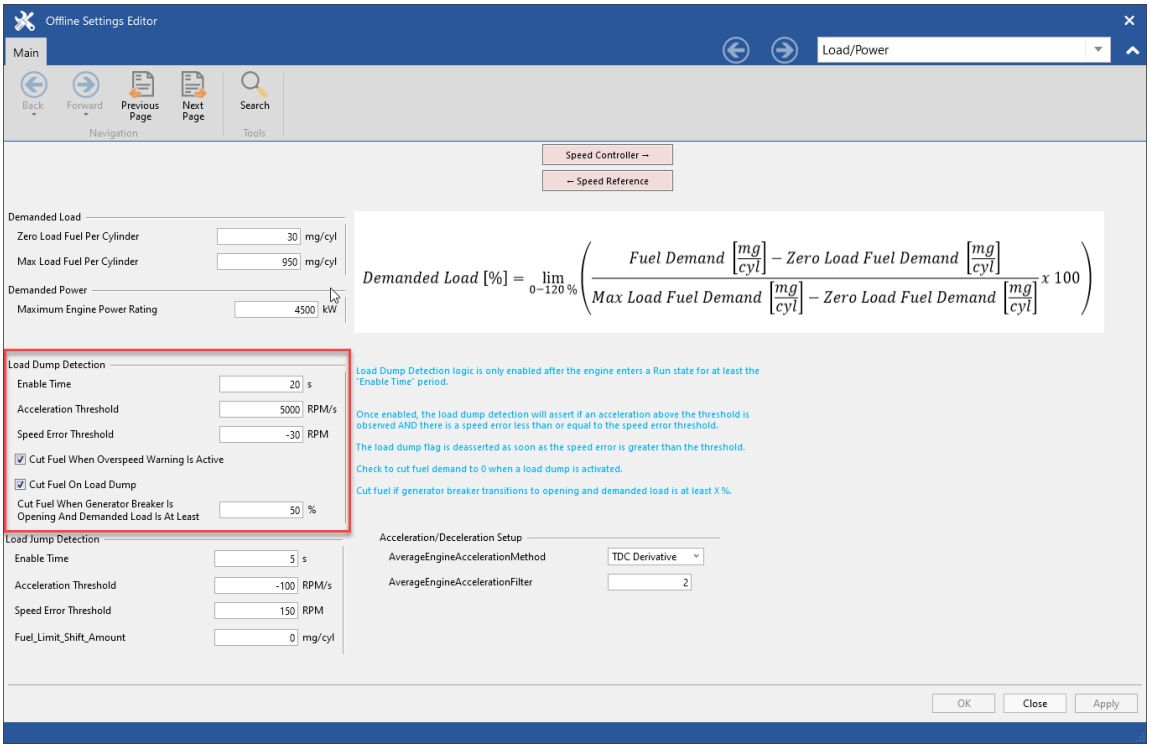


Figure 5-24. Load Dump Dependent Parameters

Load Jump Detection

Next to the load dump, there is also a load jump detection. This works similar to the load dump, but in the opposite way. This feature uses the deceleration of the engine in combination with the error between actual speed and speed reference setpoint. Once these thresholds are recognized, the fuel limiter (manifold air pressure and torque fuel limiter) can be shifted temporary to allows better acceptance of the load, without being immediately blocked by the fuel limiter. If this feature is not required, let the Fuel_Limit_Shift_Amount be on 0 mg/cyl.

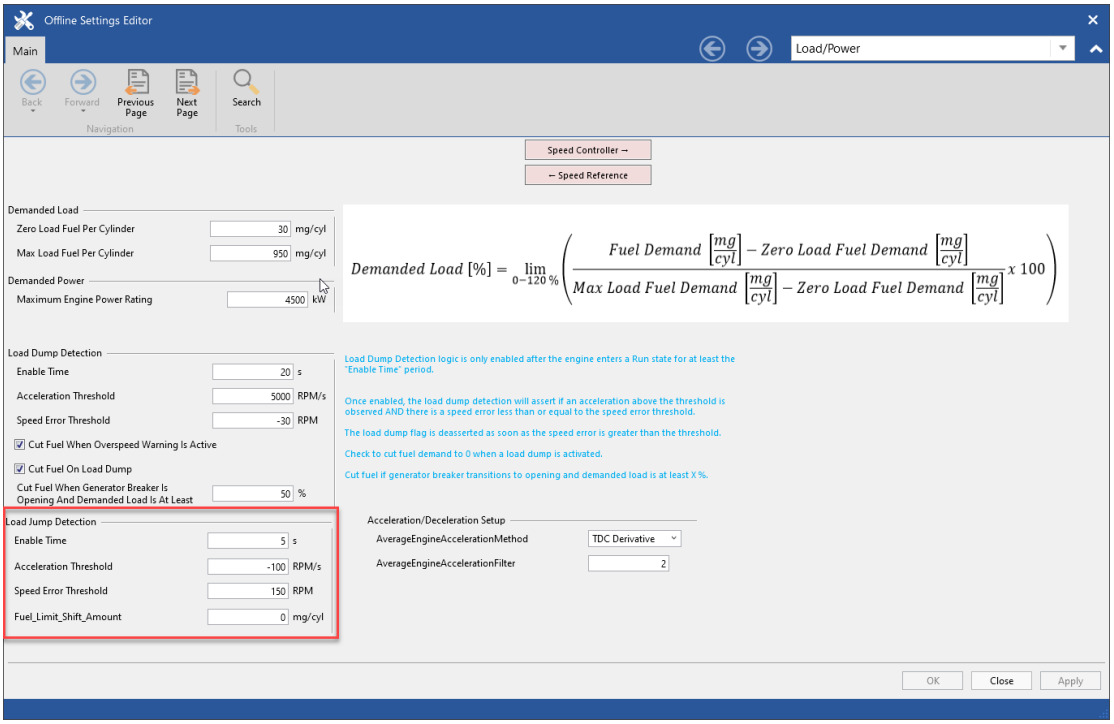


Figure 5-25. Load Jump Dependent Parameters

Engine Acceleration / Deceleration Setup

Both Load Dump and Load Jump are using the engine acceleration / deceleration detection. The software contains a few parameters to find the optimum setting for detecting these. These settings can be found in the Toolkit tool as indicated in the figure below.

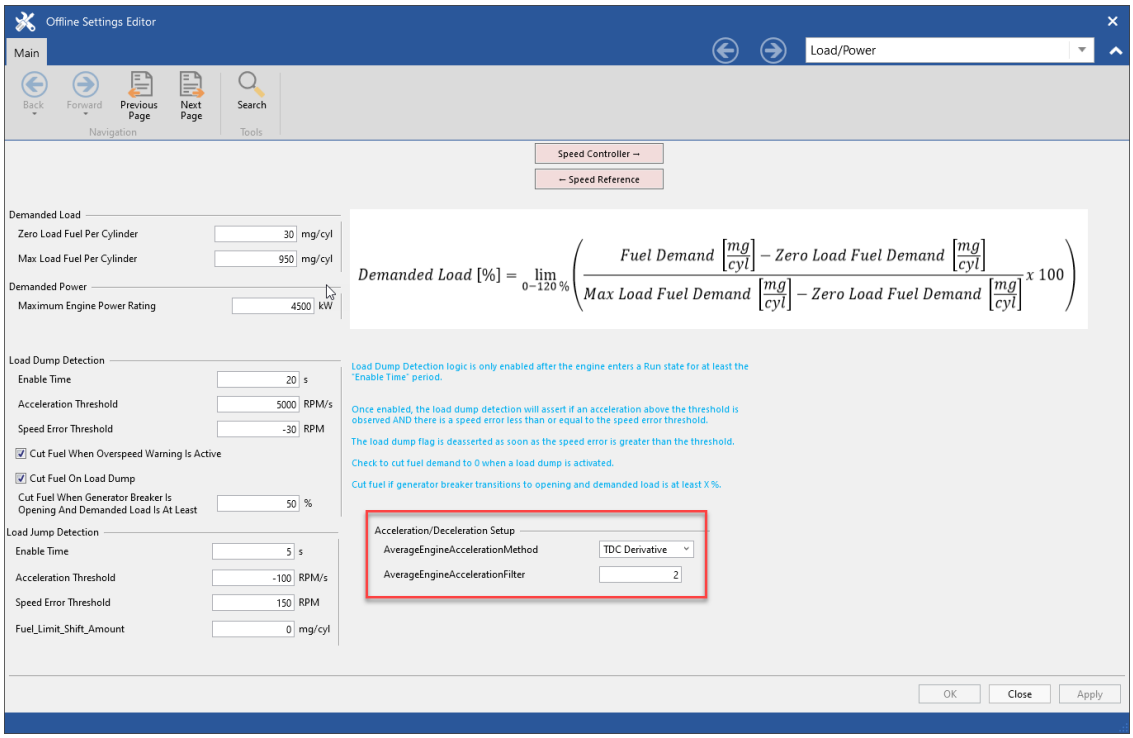


Figure 5-26. Acceleration / Deceleration Setup

There are two different selections that can be made to setup the acceleration / deceleration measurement: (1) TDC Derivate and (2) TDC1 Derivative.

The TDC derivative is measured by the dRPM/dT high resolution calculation, where the AverageEngineAccelerationFilter constant will help to eliminate too much noise for the signal. More filtering means slower updates, so an optimum for tuning must be found.

The TDC1 derivative is calculated from TDC to TDC of each cylinder event. Also, the above mentioned filter settings can be used here to filter out the noise from the signal.

Functional Description - Fuel Limiters

The following describes the fuel limiter options that are included in the LECM CRS application software. Fuel limiters are used to protect the engine against over fueling and to prevent black smoke during several engine operating tasks.

The following fuel limiters are available in the LECM CRS application software:

1) Start Fuel Limiter
2) Torque Fuel Limiter (a.k.a. speed based fuel limiter)
3) MAP Limiter (a.k.a. Manifold Pressure Based Fuel Limiter)
4) Power Limiter
5) Jump Rate Limiter
6) Maximum Fuel Limiter

All fuel limiters are coupled with the PID output value to a so-called “LSS bus”. This is a low signal select bus that only allows the lowest values feeding that bus to be used as output.

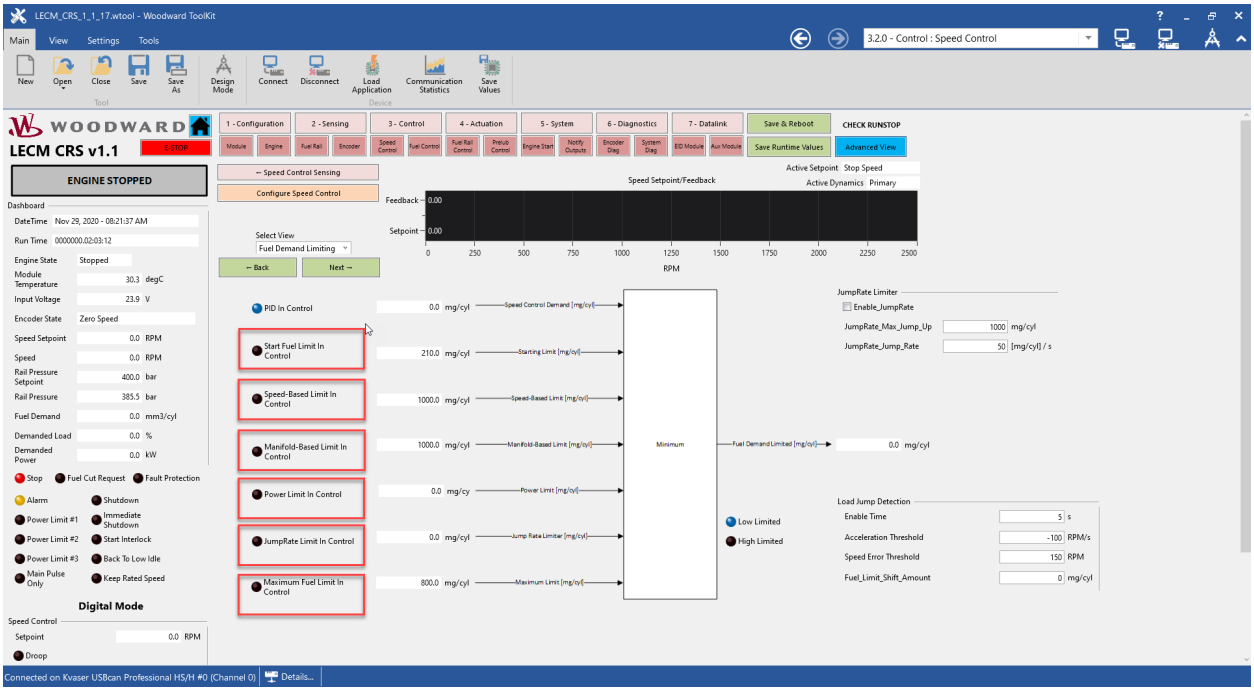


Figure 5-27. Overview of Fuel Limiters

Start Fuel Limiter

When the engine is started, the available air through the engine is limited, so fuel must also be limited to prevent black smoke and improve engine starting behavior.

Before the start fuel limiter is active, during the starting phase an additional “fuel limiter” is active. This fuel volume is setup as a base fuel per cylinder, depending on speed and manifold air temperature. It can be compensated with an ECT correction. This base fuel setup during starting can be found in the Fuel Control menu of the application toolkit page.

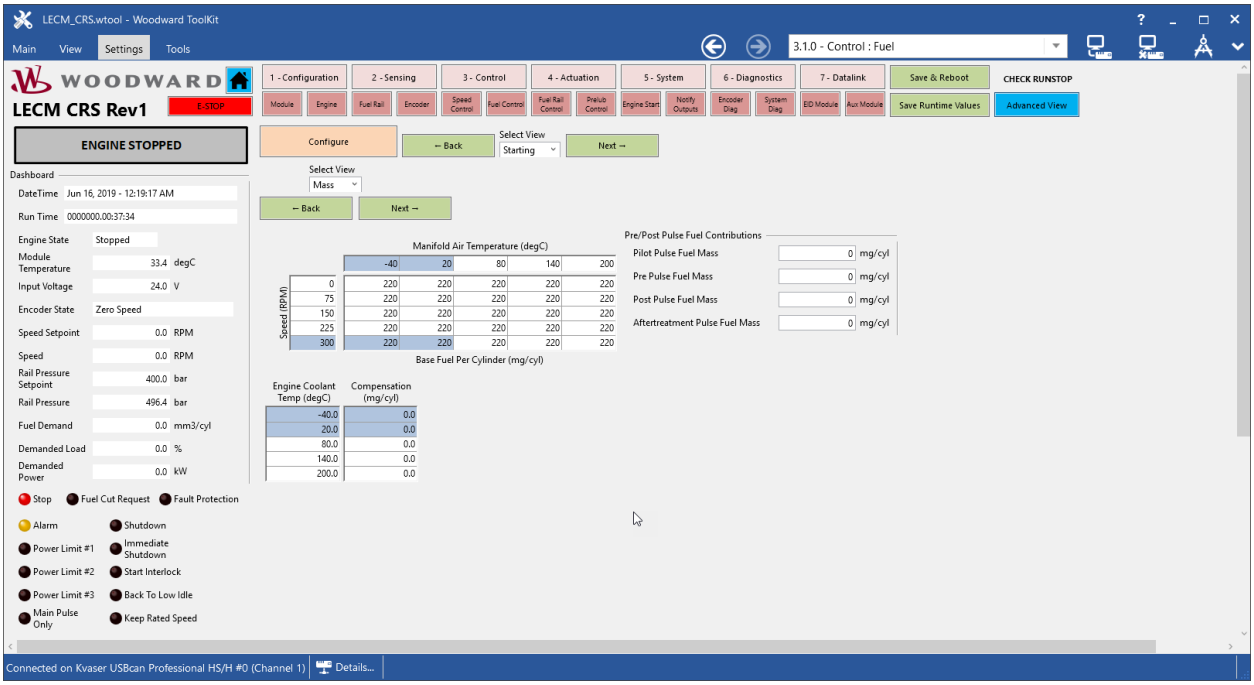


Figure 5-28. Starting Phase Fuel Demand

When the engine mode transitions from starting mode into running mode, the real start fuel limiter will be active. The parameters to setup the start fuel limiter can be found in the speed controller page, configuration menu, fuel limiting part.

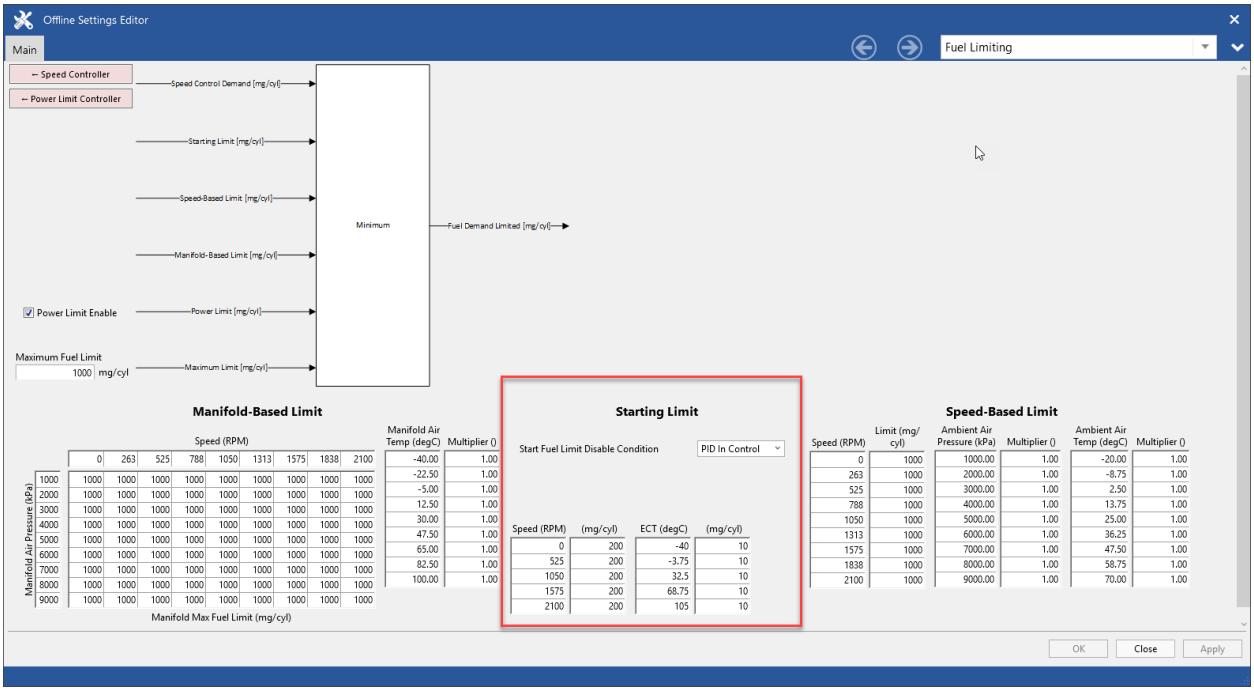


Figure 5-29. Fuel Limiters Setup – Start Fuel Limit

The Start Fuel Limiter is a speed based max fuel amount limit, which is a 5 point table with a 5 point engine coolant temperature correction. The start fuel limiter will stay active until one of the selectable ‘disable modes’ has been reached.

a) PID in Control
b) When Speed Setpoint has been reached

Either option can be selected from the same Start Fuel Limit configuration page. When the PID in Control option is selected, the start fuel limiter will be switched off when the PID control has full control of the engine speed and the fuel amount commanded to the engine is below the start fuel limit value.

If the Speed Setpoint option is selected to disable the start fuel limit, another option will become available.

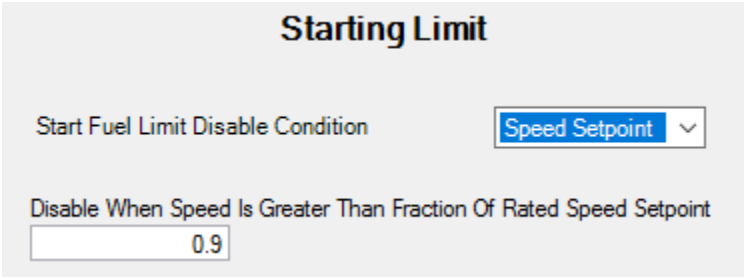


Figure 5-30. Speed Setpoint Dependent Switch Off

The Speed Setpoint is related to the active Rated Speed Setpoint. If, for example, rated speed is 1500 rpm, but we want the start fuel limit to be switched off at 350 rpm, we must set the multiplier to 350/1500 = 23.33%. When the actual speed reaches this value, the start fuel limit will be switched off and the PID must take over from that point.

Torque Fuel Limit / Speed Based Fuel Limiter

This fuel limiter is a speed based fuel limiter and can be set from the same menu as the start fuel limiter.

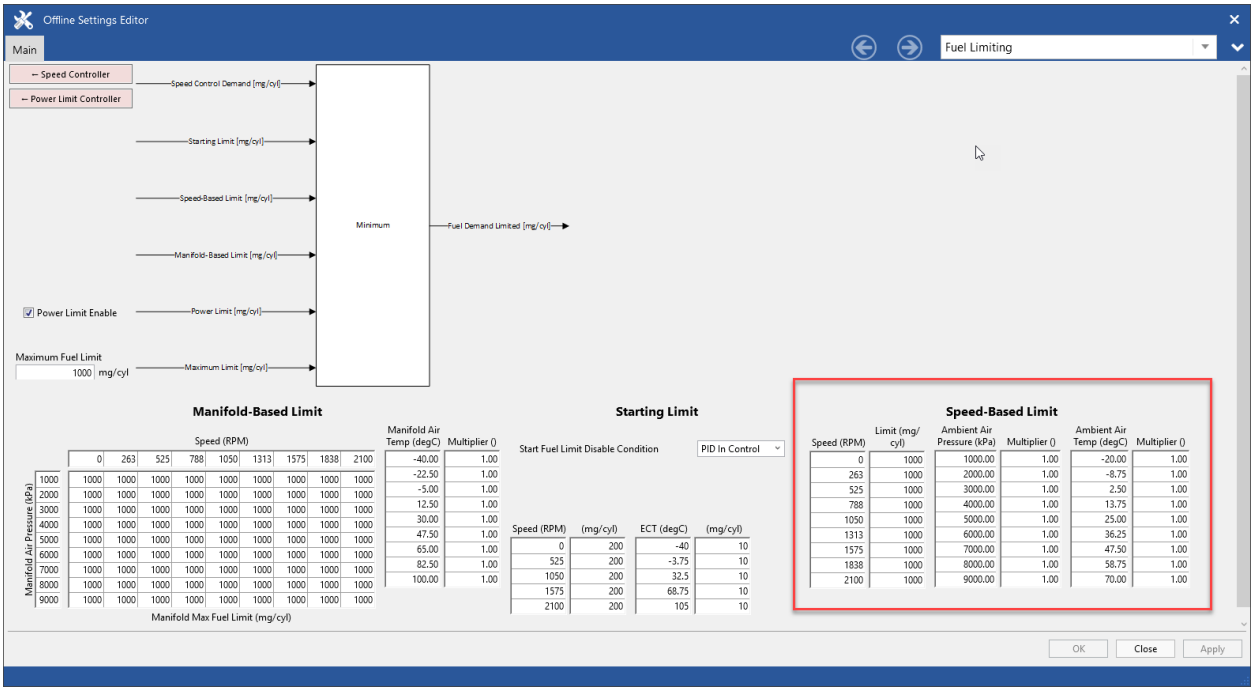


Figure 5-31. Torque Fuel Limiter

If an engine is used in a variable speed application, like ship propulsion or compressor, this limiter is used to protect the engine against over torque situations. An engine is designed to deliver the max torque / power at a certain speed. Deviating from that speed will have an impact on the max torque/power the engine can deliver. This torque fuel limiter is meant to protect the engine against over-torque in this situation.

The torque fuel limiter has ambient air and ambient temperature as compensation factors, so the torque limit line can be reduced for higher altitudes or when the environment the engine operates in is cold. This

is a multiplier factor for the defined torque curve, so leaving this multiplier to 1 disables the ambient related factors. The torque fuel limit will be active during engine running state. In addition, this limiter feeds into the previously mentioned LSS bus, where all limiters are coupled together.

MAP Fuel Limit / Manifold Air Pressure Fuel Limiter

This fuel limiter is a manifold air pressure based fuel limiter and can be set from the same menu as the previously mentioned fuel limiters.

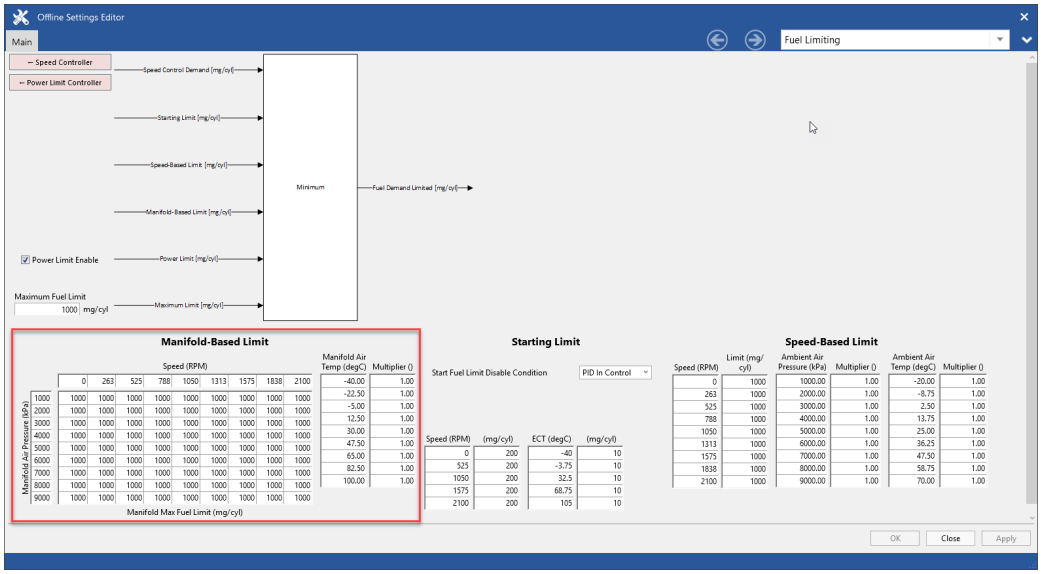


Figure 5-32. Manifold Air Pressure Fuel Limiter

This limiter is meant to limit the maximum fuel amount at a given manifold air pressure value. It can be used on stationary applications and/or variable speed applications. In variable speed applications, this limiter can cause issues as the manifold air pressure build up will take time while load is increasing. The limiter is meant to be used to prevent over-fueling of the engine, causing black smoke and/or bad engine response. The MAP limiter can be compensated by the Manifold Air Temperature. This is a multiplication factor for the limiter based on the actual manifold air temperature. In addition, this limiter is coupled to the LSS bus as previously mentioned.

Maximum Fuel Limit

This fuel limiter is an absolute maximum allowed fuel limiter. No more fuel than this limiter allows will be injected into the engine. It can be setup from the same configuration page as the previously mentioned fuel limiters. The maximum fuel limiter is the absolute maximum fuel amount allowed to the engine and also feeds the same LSS bus as previously mentioned.

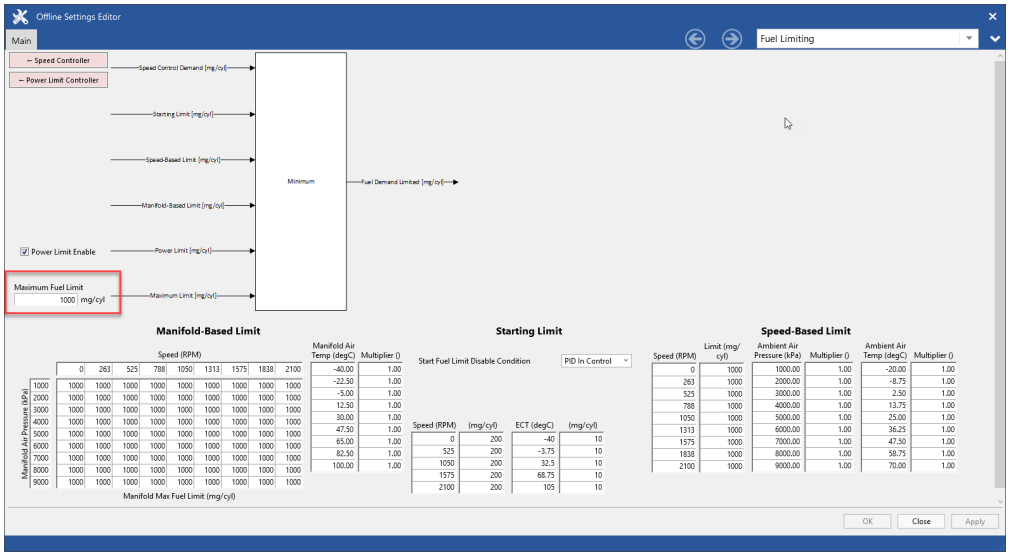


Figure 5-33. Maximum Fuel Limiter

Power Limiter

The Power Limit Controller reduces engine power to various calibrate able levels when critical faults are active. Which faults engage the Power Limit Controller is set via calibration in the Fault Manager by tying a fault to a Power Limit Level fault action. There are four power limit levels. The actual power limit corresponding to these levels is set by calibration. When engaged, the controller reduces PowerLimitFPC until the current operating power level of the engine is equal to or less than the calibrated threshold. This reduction in power is achieved by a feedback PID controller, where the output FPC is the minimum of either FPC or Power Limiter FPC. It is critical to calibrate the FPC to the torque conversion table such that engine power can be estimated for the Power Limiter.

The Power Limiter can be setup to either be active internally in the software or externally. There are situations where you do not want to power limit the engine automatically, such as a genset running in island mode for example. Derate would result in lowering the frequency, which is undesired. In those cases, the external selection of the power limiter must be activated, so a power management system can decide to trip non preferred consumers before lowering the load. An analogue output is available to indicate the derate level in that case.

The Power Limit can be setup in the Speed Controller configuration page, under the Power Limit Controller section.

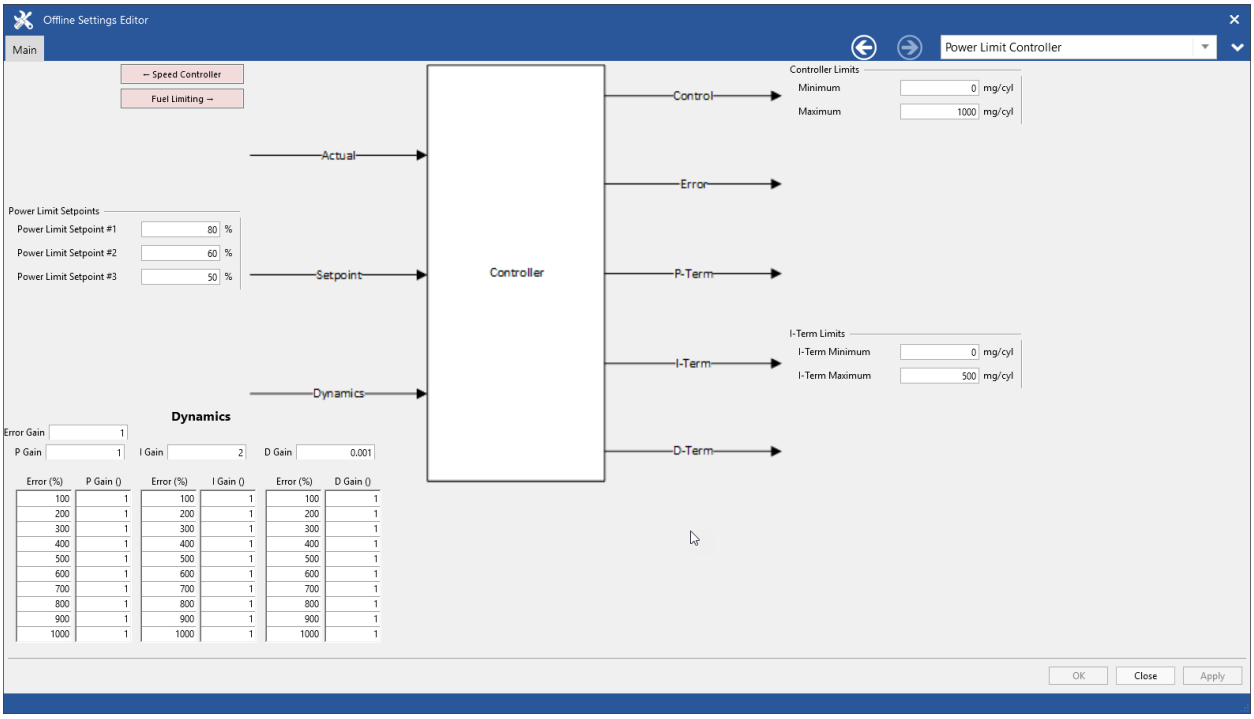


Figure 5-34. Power Limit Parameter Setup

Three different power limits can be setup and the dynamic behavior of this must be tuned with another set of P, I, and D values. These P, I, and D values are gain values which are multiplied again with an error dependent correction factor. Tuning these values can be done in the Speed Controller main page, selecting the Power Limit Controller Dynamics.

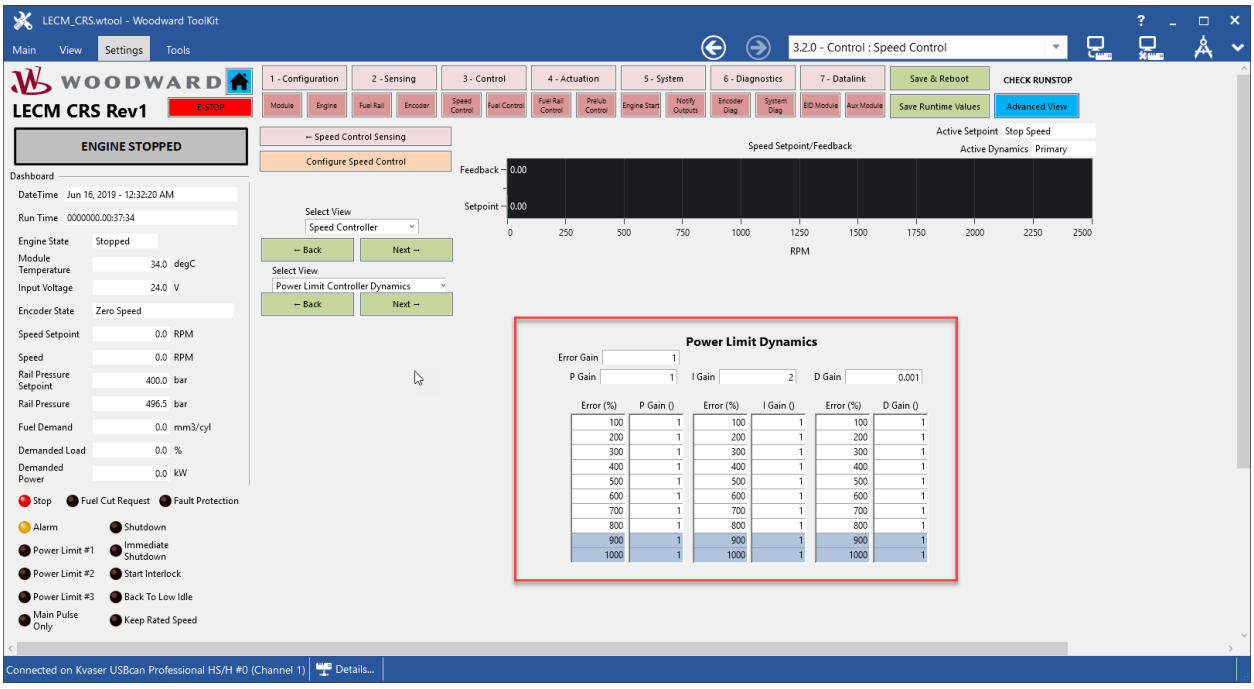


Figure 5-35. Power Limit Dynamics Tuning Option

Jump Rate Limiter

The Jump Rate limiter is a feature to prevent worsening engine performance due to overfuelling of the engine (shortage of air due to turbo lag). The Jump Rate limiter allows an instant jump in fuel demand

increase, whereafter it starts ramping up till engine speed gets in control again. In this way, more overfuelling is prevented and engine recovery will be improved. Settings for this Jump Rate limiter can be set in Toolkit from the Speed Control Page. Leaving the Enable_JumpRate selection unchecked will disable this feature.

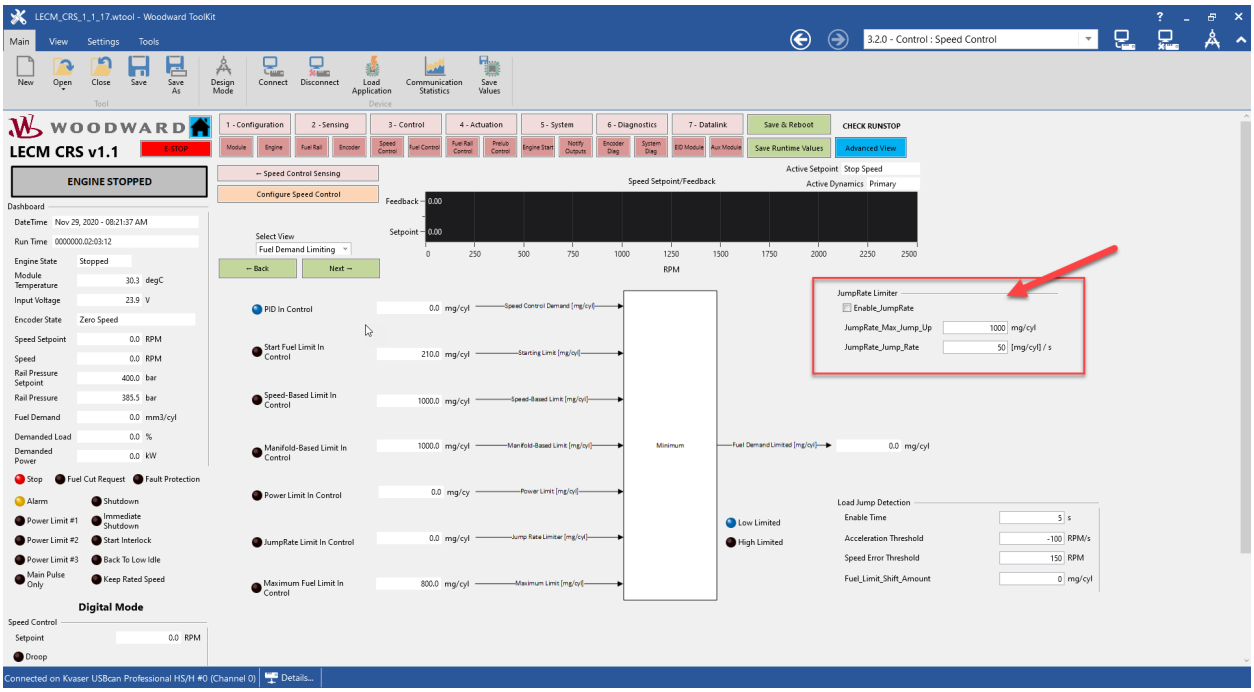


Figure 5-36. Jump Rate Limiter Settings

If multiple fuel limiters will be active at the same time, the following priority will be active, starting with the highest priority:

1) Start Fuel Limiter
2) Torque Fuel Limiter (a.k.a. Speed Based Fuel Limiter)
3) MAP Limiter (a.k.a. Manifold Pressure Based Fuel Limiter)
4) Power Limiter
5) Jump Rate Limiter
6) Maximum Fuel Limiter

Near Fuel Limit Notification

When the fuel demand is getting closer to a fuel limit value, there is a Near Fuel Limit Notification that can be setup to indicate that the engine is close to its maximum load at that certain engine performance point. This could be used as feedback to a load controller, or on board of a ship to the ship propellor pitch adjustment system to indicate maximum load reached position.

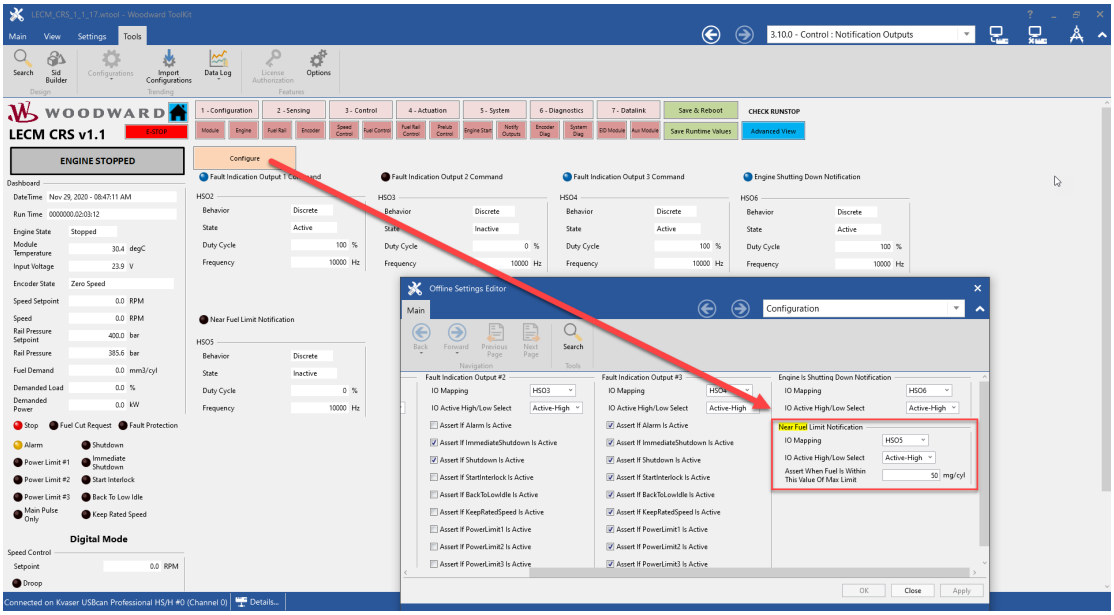


Figure 5-37. Near Fuel limit Notification

The notification can be configured to any of the available outputs of the LECM hardware and will energize when the conditions are met.

How fast this indication is set can be adjusted with the parameter: Assert When Fuel Is Within This Value Of Max Limit [mg/cyl]. Making this bigger will turn on the notification output earlier than when this value is smaller. The Notification Output stays activated until the fuel demand is lower than the above mentioned parameter from the lowest fuel limit value.

J1939 / Modbus Speed Control Options

The following describes the implementation and features of the speed control options that are supported on J1939 and Modbus communication. The following chapters will describe each protocol in detail.

ModBus Speed Control Options

The LECM CRS application software does support Modbus communication, which is available on the Ethernet TCP/IP port or on serial RS-485 port (optional). The ModBus list is customer specific for each application. The ModBus writes need to be activated before they will be active. Activating them and allowing to have action is done on the Toolkit Datalink Page.

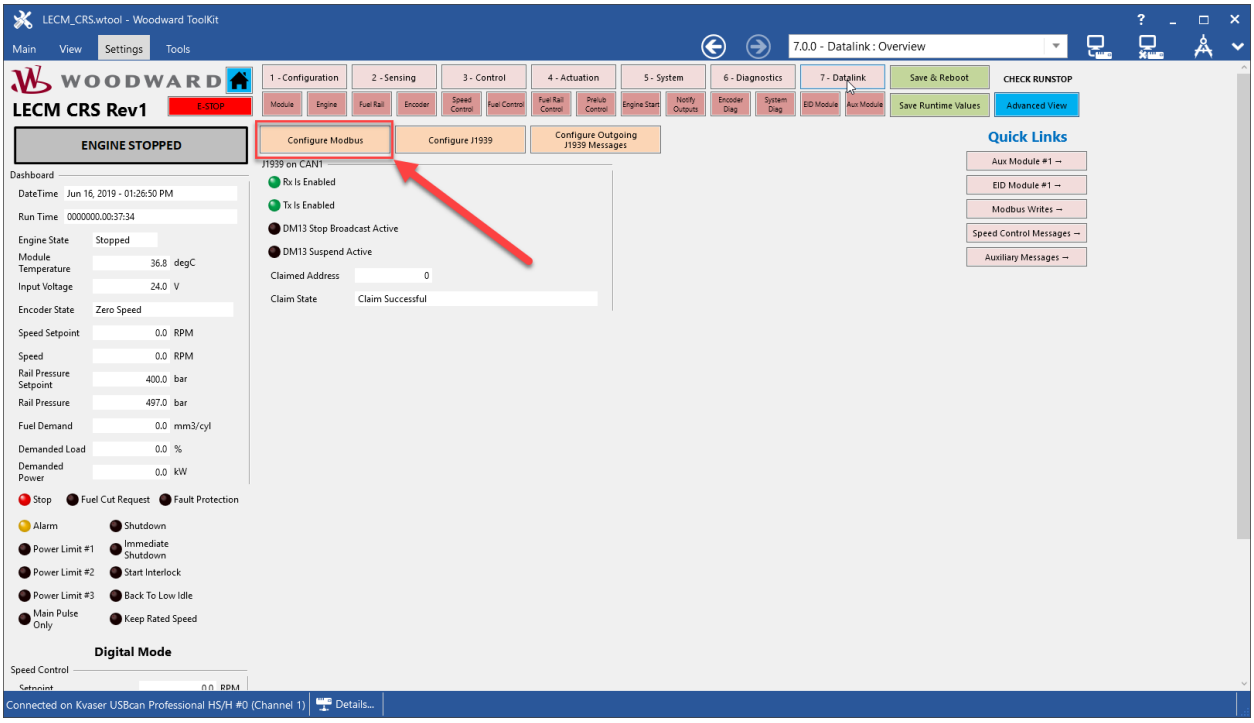


Figure 5-38. ModBus Configuration

Pushing the button will open an offline configuration page, where the ModBus writes can be activated / allowed. Not enabling will block the write commands, to prevent unwanted actions taken by ModBus commands.

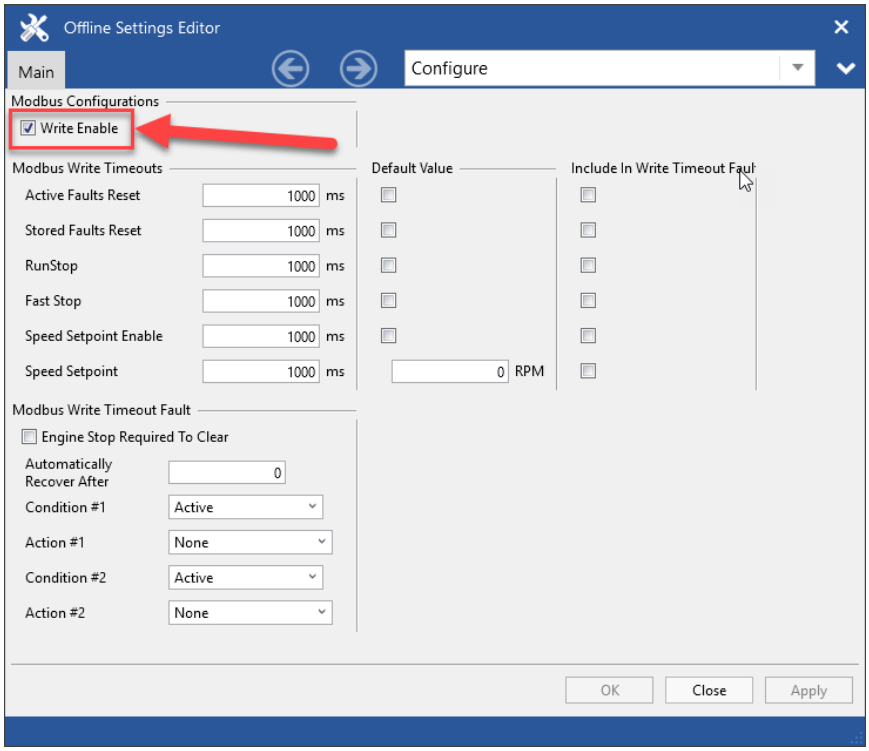


Figure 5-39. ModBus Configuration

If the timeouts exceed the specified timeouts times, the Modbus writes will no longer be active and a Fault will be reported.

J1939 Speed Control Options

The LECM CRS application software supports J1939 CAN communication, which is available on the CAN port#1 of the LECM. All transmission and reception of messages can be globally disabled to deactivate the protocol. Differences may exist in application-specific variants.

In this chapter, only the J1939 messages related to speed control will be described.

The following messages are supported in the LECM CRS application software:

- TSC1 ➔ The TSC1 (Torque/Speed Control 1) message is used to control the speed reference setpoint via J1939.
- GC1 ➔ The GC1 (Generator Control 1) message is used to control the speed reference droop % and idle/rated select via J1939.
- GC2 ➔ The GC2 (Generator Control 2) message is used to control the speed reference bias via J1939.
- ACS ➔ The ACS (AC Switching Device Status) message is used to communicate various status of breakers (Generator- / Utility Breaker)
- GTACP ➔ The GTACP (Generator Total AC Power) message is used to communicate generator power readings.

To activate / inactivate these messages, the offline settings editor on the Datalink page of toolkit can be used.

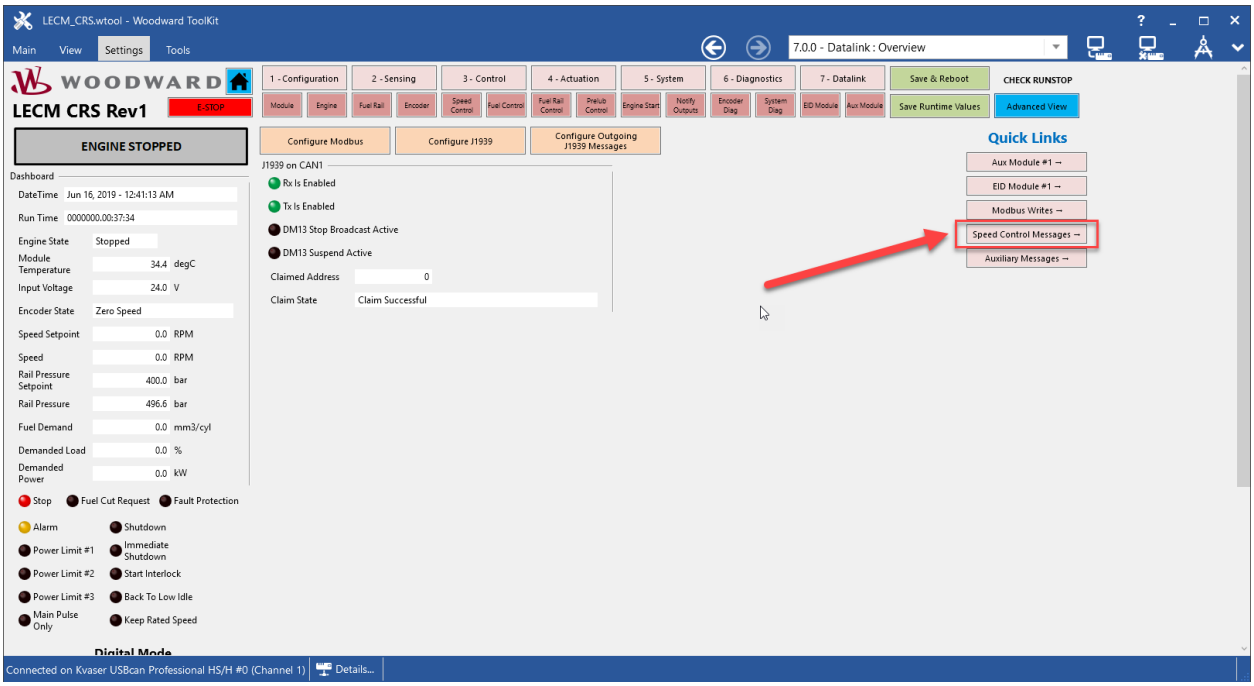


Figure 5-40. Speed Control Messages

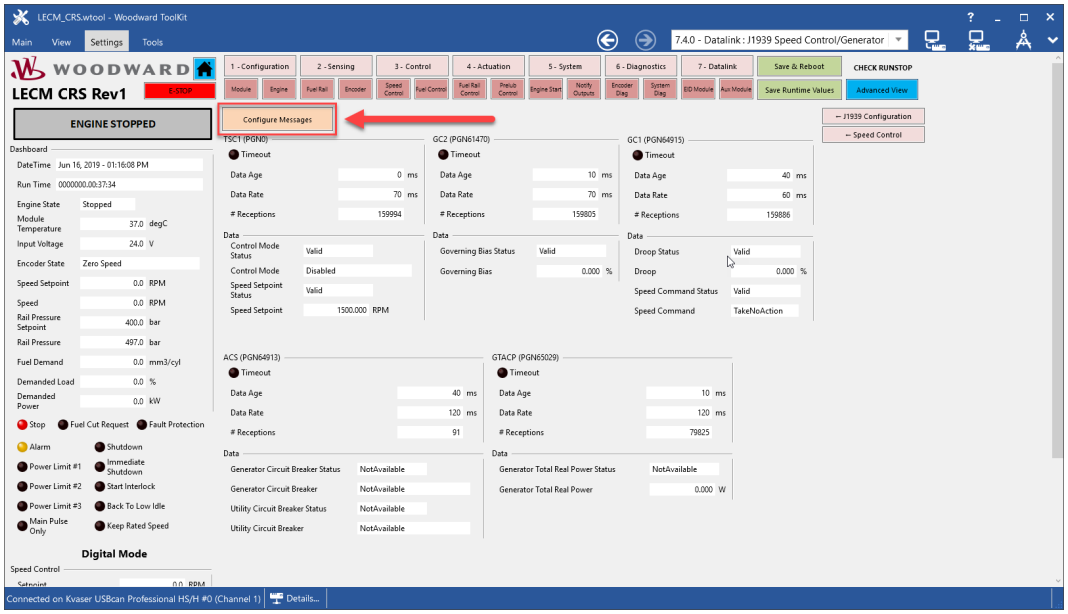


Figure 5-41. Speed Control Messages Configuration

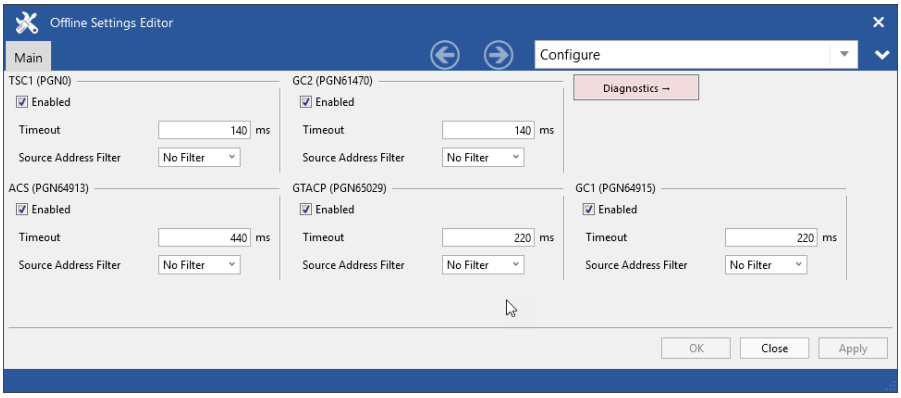


Figure 5-42. Enable/Disable Messages

If the messages are enabled, the diagnostics will also be active. The diagnostics can be setup in the configure page also.

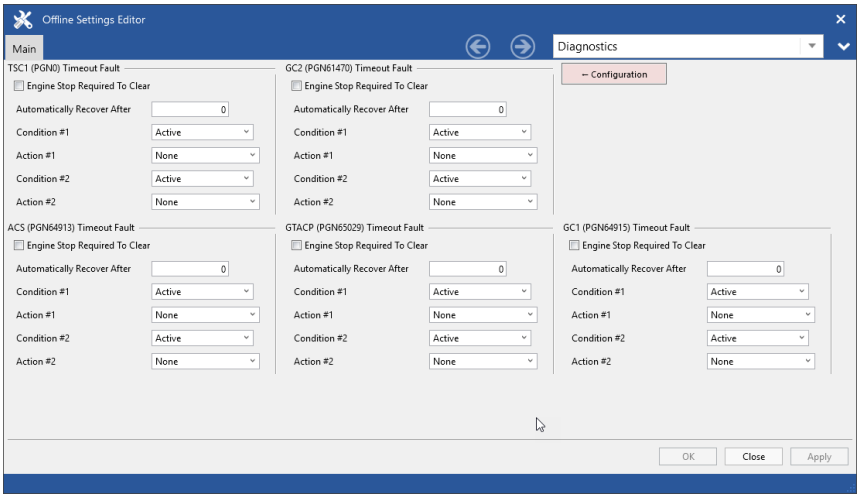


Figure 5-43. Speed Control Messages Diagnostics Setup

Each fault can be set to its own action. In addition, the option to clear the faults at stopped engine only can be set.
The details on the messages are defined as:

- TSC1

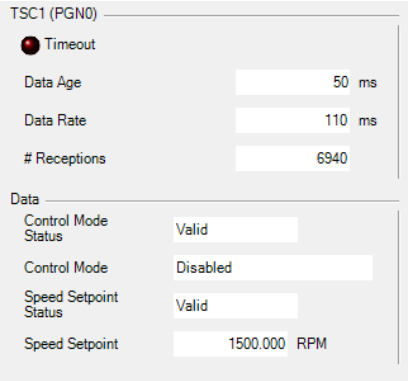


Figure 5-44. TSC1 Details

- GC1

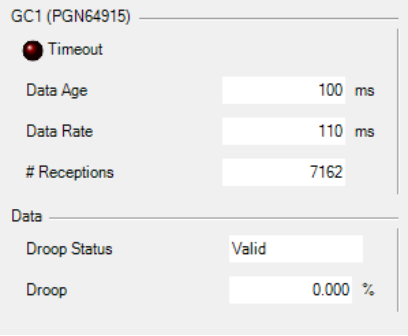


Figure 5-45. GC1 Details

- GC2

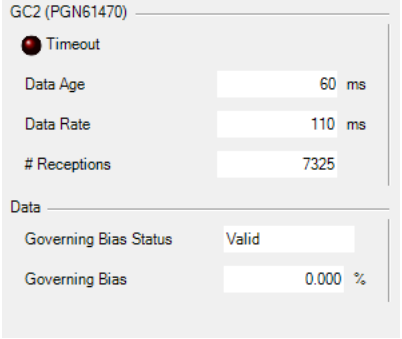


Figure 5-46. GC2 Details

- ACS

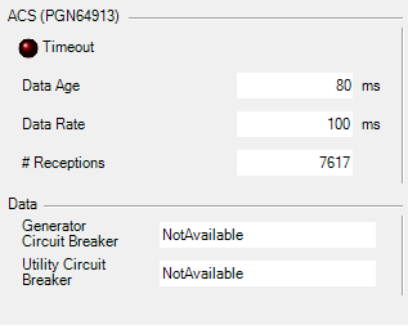


Figure 5-47. ACS Details

- GTACP

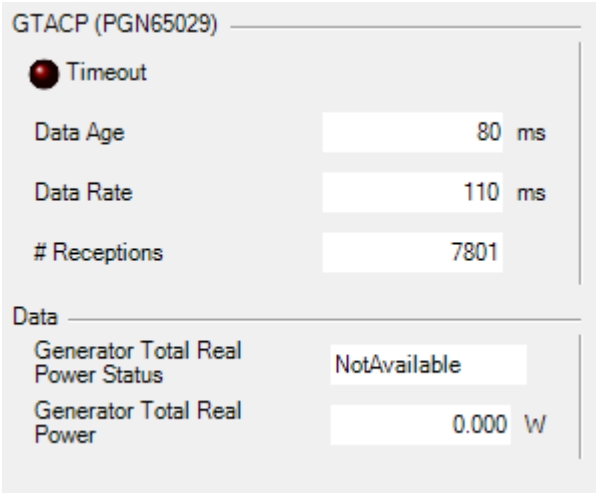


Figure 5-48. GTACP Details

External Speed Control Option

The following describes the setup of the LECM CRS software to utilize an external speed control instead of the build-in all-speed governor. Reasons for using external speed control can be an extension to an already installed engine base, where the same speed control must be used to match the existing engines.

Configuration

To setup the LECM CRS software to external speed control mode, the other speed control must be able to be configured to output either a 4-20 mA or 0-5V Fuel Demand Signal. Most Woodward speed controls support a 0-200 mA actuator output, which normally is used as the actuator output signal. This command normally sets the fuel rack to the requested fuel amount position. When using this signal as fuel demand input for the LECM CRS application software, a 25 Ohm resistor can be used to convert the signal to 0-5V. Note that the resistor power rating must be sufficient.

On the Toolkit Engine Configuration page, select the Configure mode button.

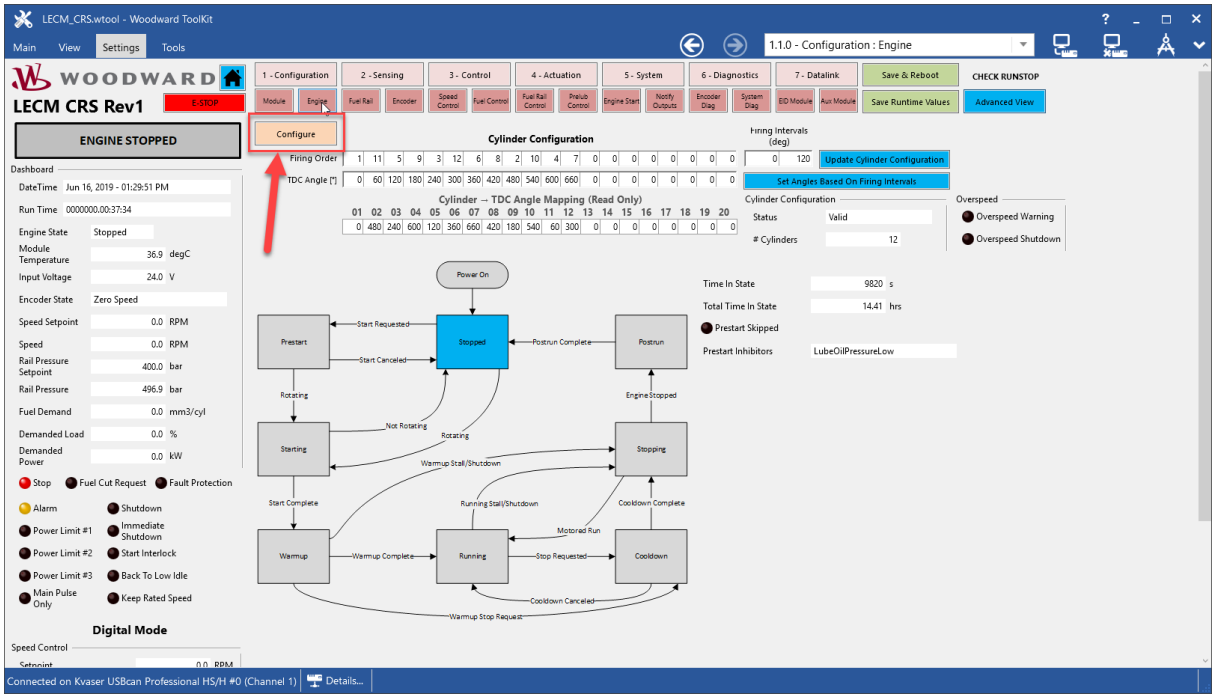


Figure 5-49. Engine Configuration Mode

This opens a page where you can select between internal and external speed control.
Once the selection is made, a save and reboot of the LECM is required as this is a configuration parameter that will be initialized on reboot only.

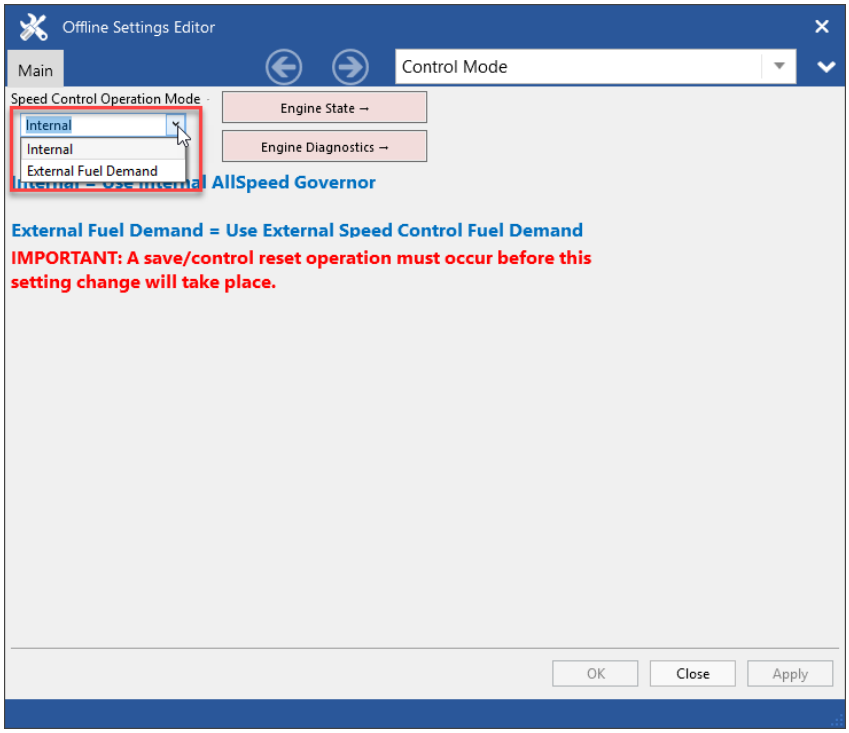


Figure 5-50. Engine Control Mode Selection

Once the save and reboot is completed, the fuel demand signal needs to be defined. On the Sensing Overview page, there is a selection button named Fuel Demand.

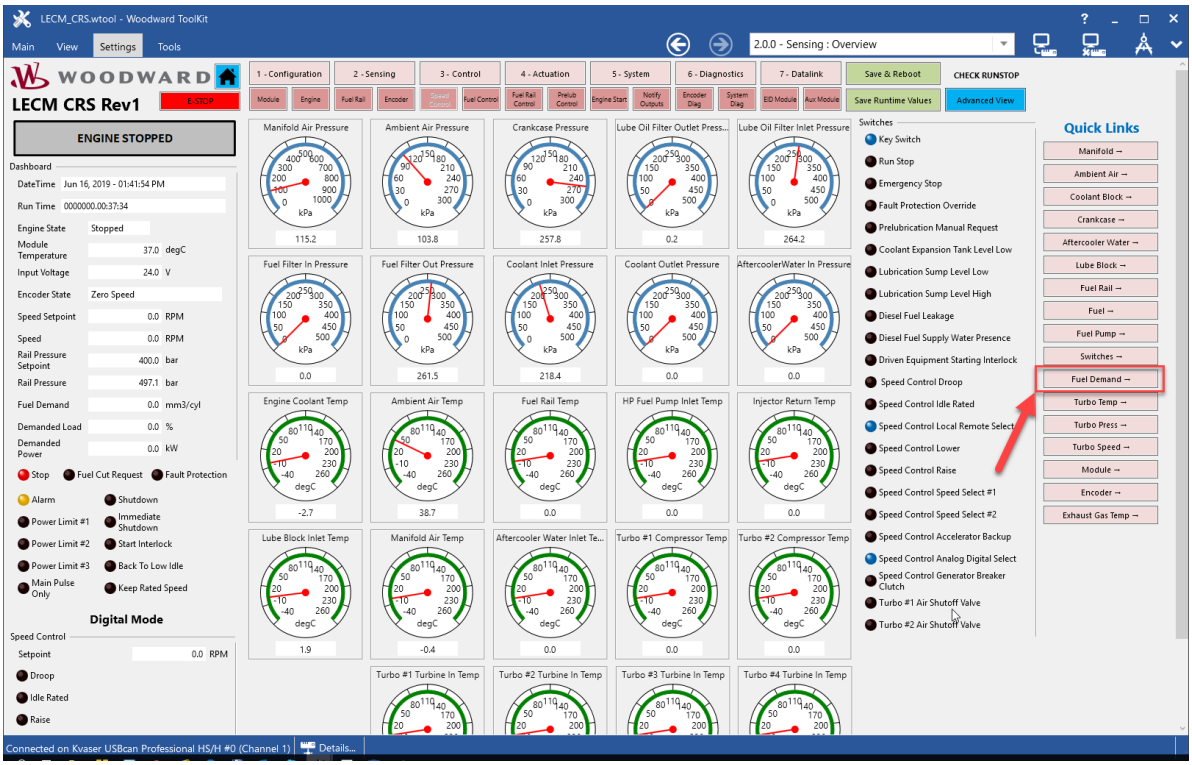


Figure 5-51. Fuel Demand Selection

Pushing this button will open another page called Fuel Demand. On this page, you find another Configuration button, which allows you to open the offline editor and setup the fuel demand signal details.

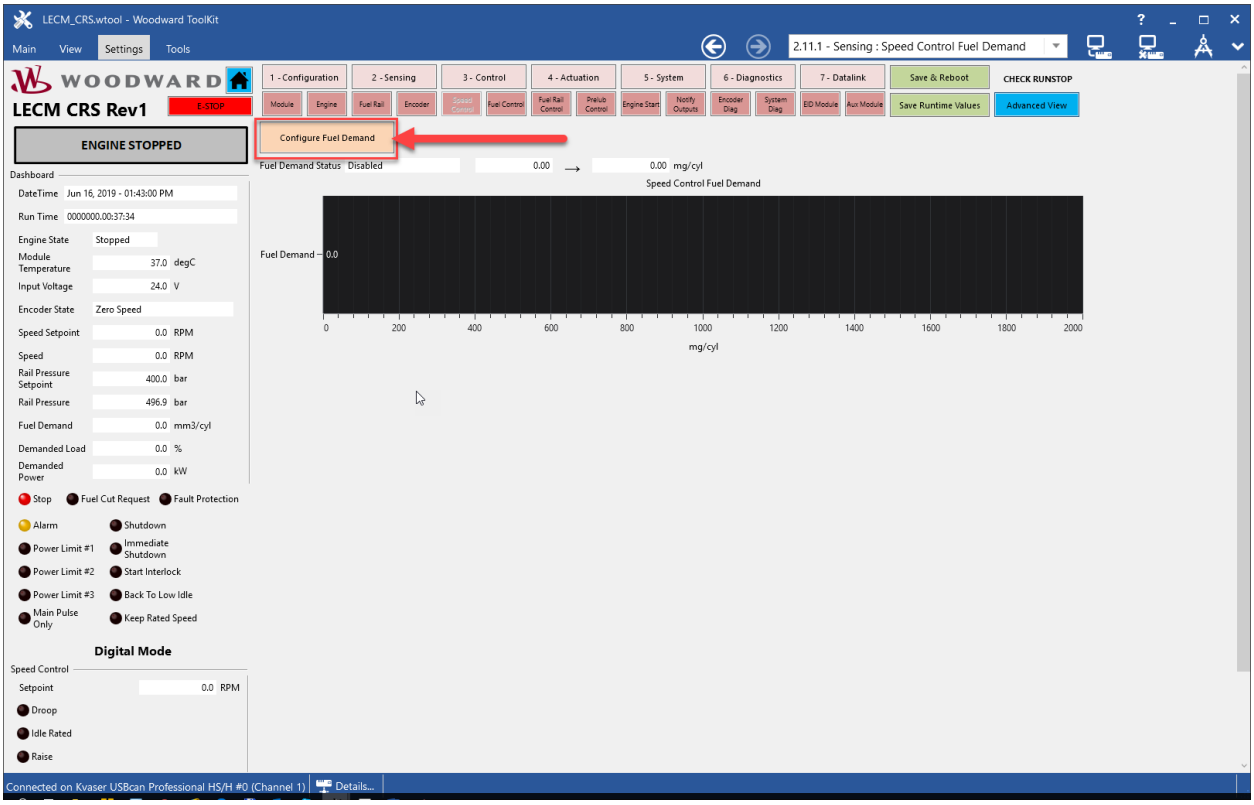


Figure 5-52. Fuel Demand Overview

The Fuel Demand signal can be hooked up to any of the free analog input channels. In addition, the default strategy to follow if the signal is failing can be set from this screen.

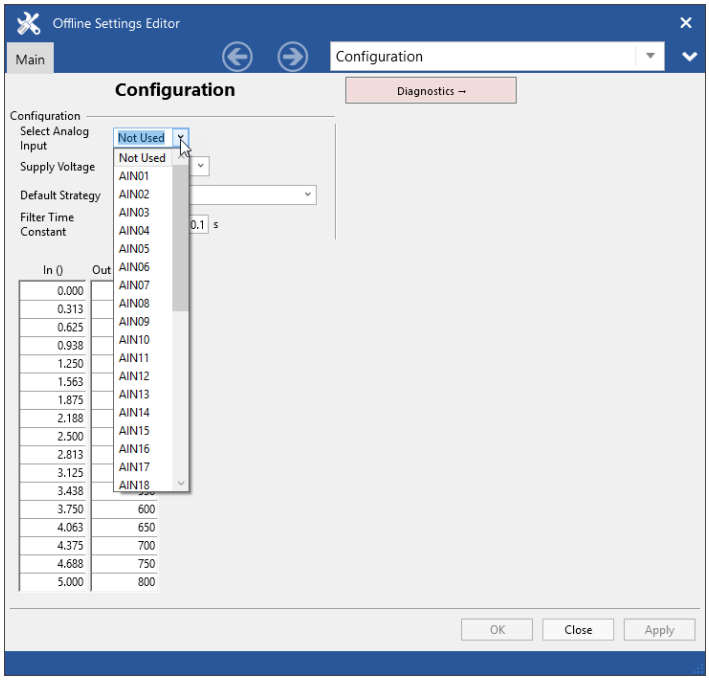


Figure 5-53. Fuel Demand Configuration

The selection of the analog input is either 4-20 mA or 0-5 V and needs to be set on the Configuration Module Page, selection Configure Module Hardware.

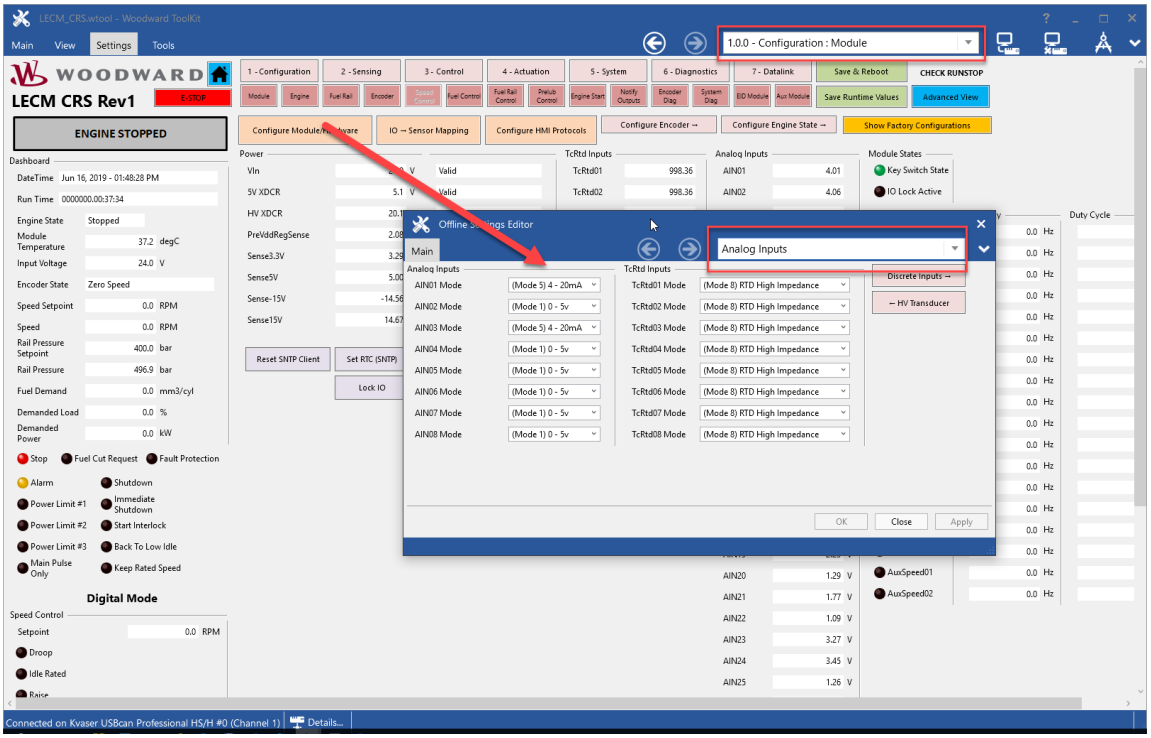


Figure 5-54. Analog Input Type Selection

Once the selection is made, the mapping of the analog signal to the fuel amount can be done to setup the table in the fuel demand configuration screen. Below is an example setup where the fuel amount is linear.

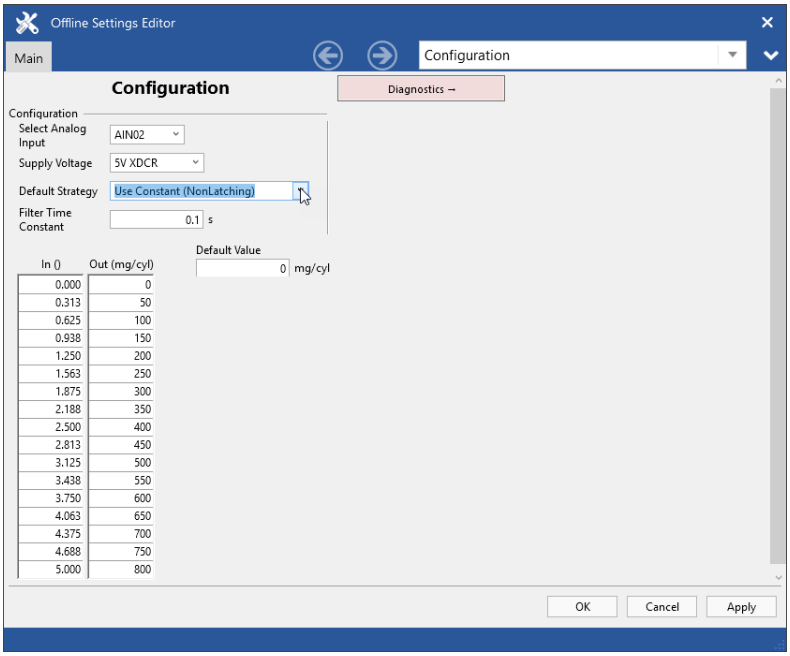


Figure 5-55. Example Setup 0-5 V to 0-850 mg/cyl Fuel Demand

If the signal is selected and connected properly, the LECM CRS system is setup to run the engine. When Run from the other speed control is initiated, a RUN command to the LECM CRS must also be provided, otherwise the fuel will not be enabled. A Stop command to the LECM will stop fueling the engine, so it will

initiate a stop. Shutdown functions from sensor failures, if setup that way, will also cause engine shutdown.

The LECM must be setup with the correct encoder setup, otherwise fuel injection will not be active. The LECM CRS speed control in-and outputs will not be active in this case.

Speed Control Setup

The following describes the setup of the LECM CRS Speed Control Software using the Toolkit setup guides, located on the first page of the Toolkit software. The setup guide will help you define the Speed Control functionality as is required for the application. Completing the setup guide will set calibrations and hardware definitions and enable/disable software logic parts as required.

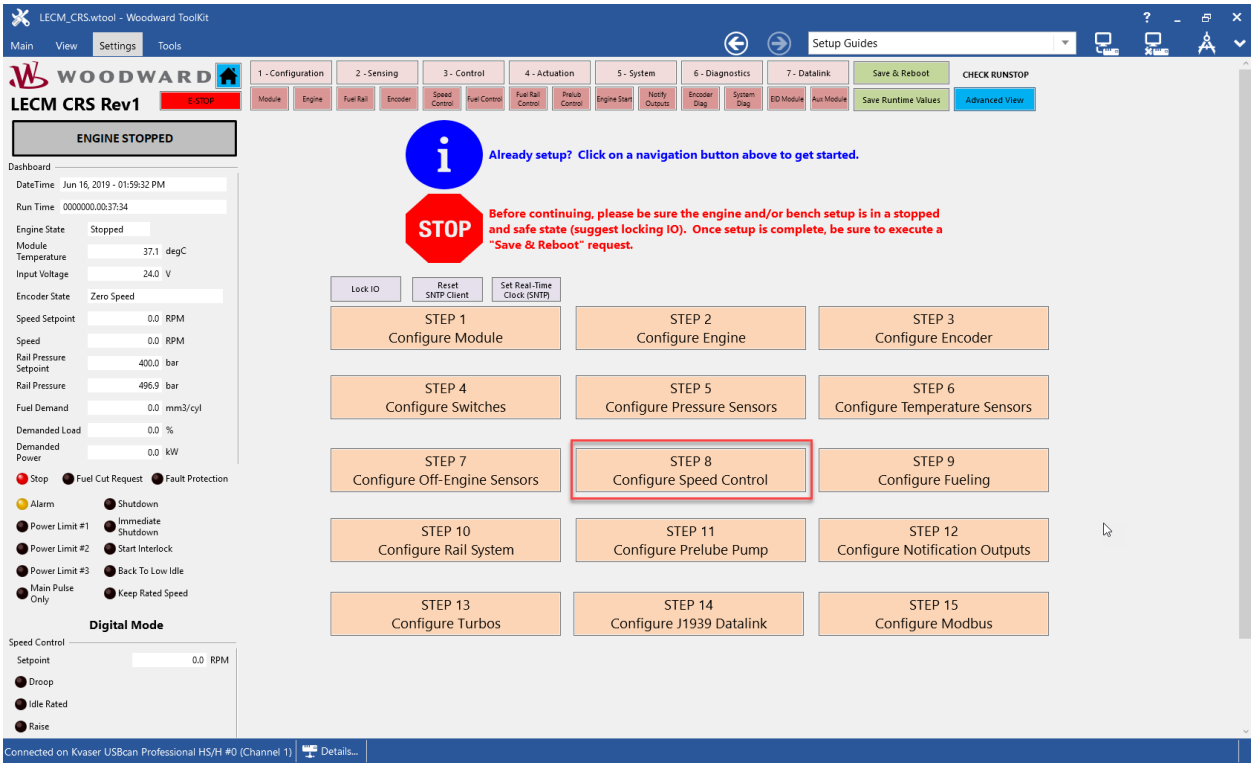


Figure 5-56. Toolkit Configuration Page

Clicking the Configure Speed Control button will open an offline editor window where some base explanation is provided on the icons used in this setup guide.

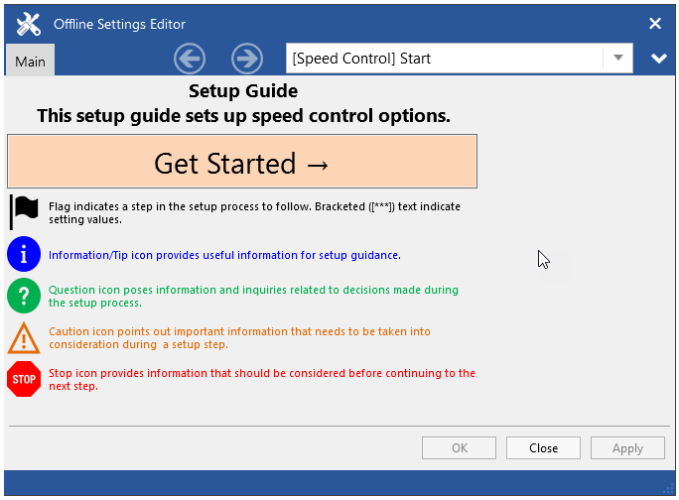


Figure 5-57. Setup Guide Start Page

Pushing the “Get Started→” button will bring you to the first page from the setup guide.

Using the “Next →” and “← Back” buttons will guide you through the setup procedure.

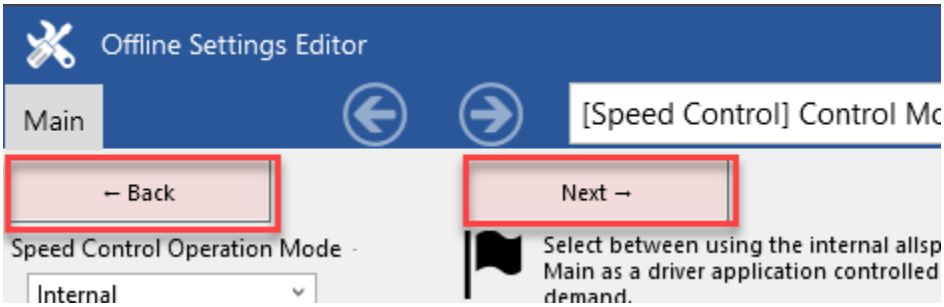


Figure 5-58. Setup Guide Navigation Buttons

[Speed Control] Control Mode

The [Speed Control] Control Mode page makes it possible to select between internal and external speed control. Selecting External Fuel Demand will disable all the speed control features inside the LECM CRS software and the fuel demand needs to come from an external speed control command to one of the LECM analog inputs. Most Woodward speed controls do have 0-200 mA output, which can be put across a 25 Ohm resistor to convert it to a 0-5 V signal.

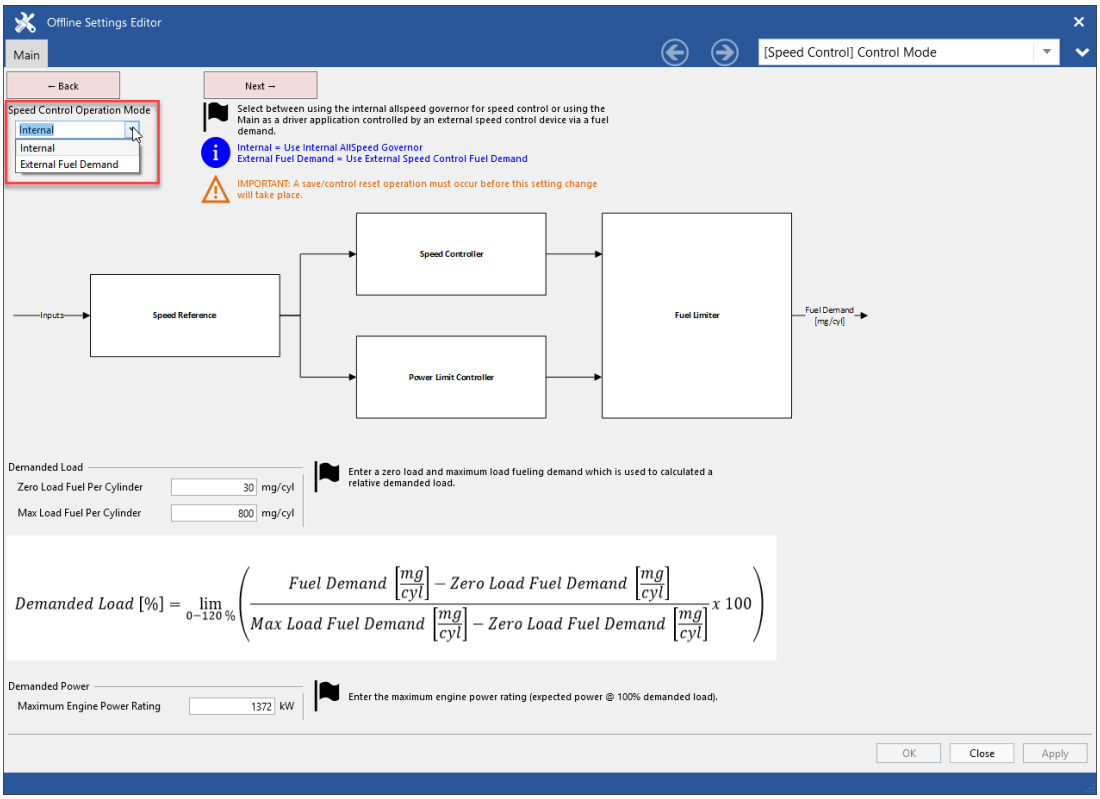


Figure 5-59. [Speed Control] Control Mode Offline Editor Page

The second part that needs to be setup on this page is the load calculation fuel amount.
The fuel amount at 0% load vs the fuel amount at full load (100%) must be set here.

The 0% load fuel amount is normally the fuel amount required when running the engine only. For an all speed engine, this will be the fuel amount when running the lowest speed setpoint without load. For a Genset this is normally the fuel amount when the engine is running at rated speed, no load condition. Setting these values as accurately as possible will result in a load indication, based on fuel amount.

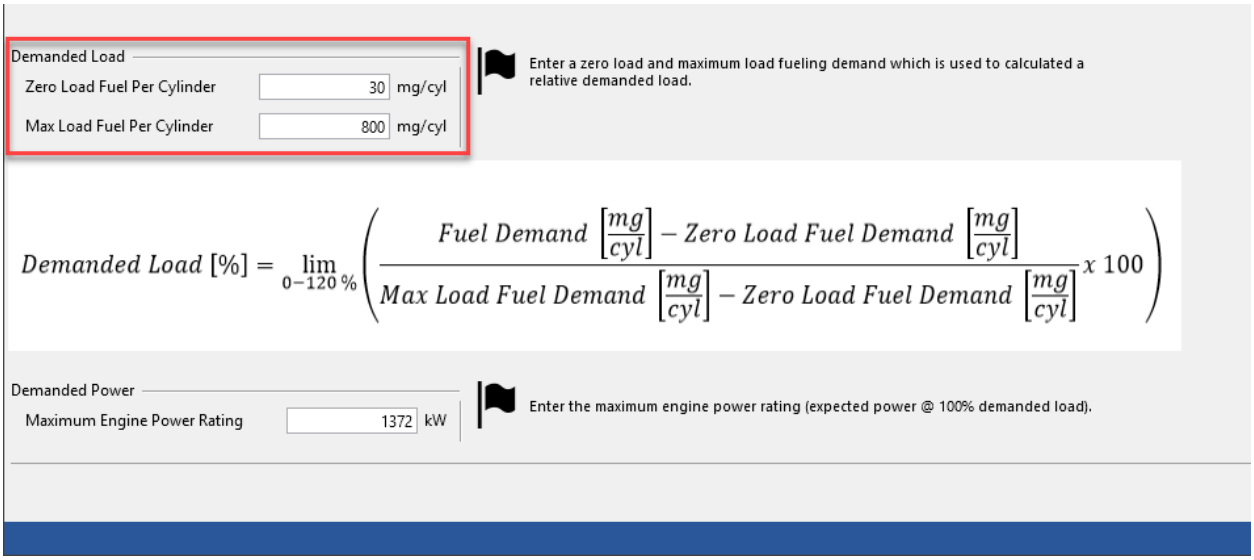


Figure 5-60. Demanded Load Setup

[Speed Control] External Fuel Demand

If the choice has been made to use an external speed control and the Next page has been pressed, the Setup Guide will navigate to the [Speed Control] Fuel Demand page. This page must be used to define the analog input signal used. A table can be used to setup the fuel demand relation to the analog input used. A Default fault strategy can be chosen in case the signal fails even as a filter constant time. It is advised to keep this filter constant time low to have good response on the engine when fuel demand is changing.

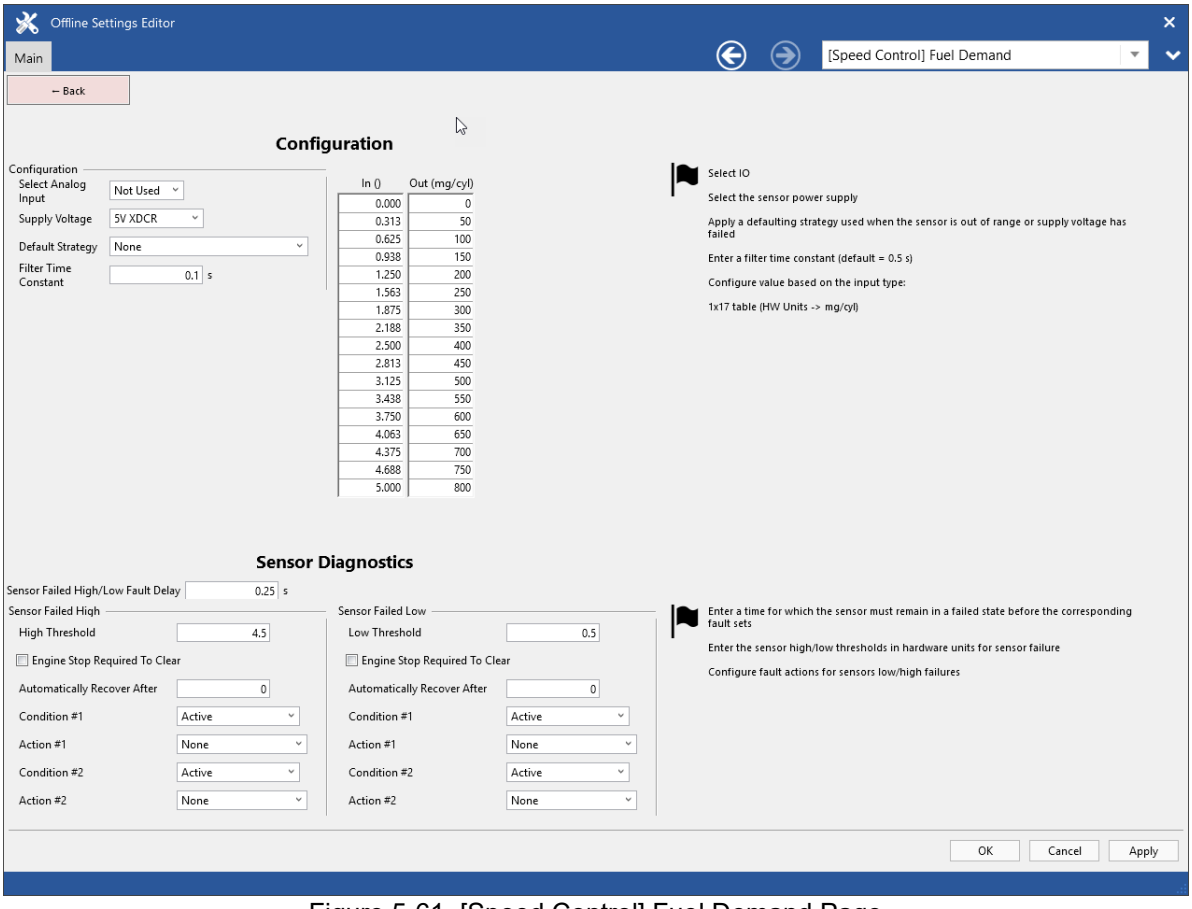


Figure 5-61. [Speed Control] Fuel Demand Page

Additional diagnostics can be setup in case the Fuel Demand Analog input signal is failing. A low and high threshold can be defined and up to two conditions and actions can be chosen in case the defined thresholds are exceeded. Also, the reset behavior can be defined for this particular analog input signal. As there is nothing else to define on the speed control part, when using external speed control selection this is the last page for the External Fuel Demand setup.

In case the internal speed control is selected, the Setup Guide will guide you through the remaining steps to define the speed control functionality. When changes to the default values have been made, hit the Apply button to upload the changes. This can take a few seconds and when it is done, you'll see the Close button appearing. Clicking this Close button will close the setup page. It is also possible, after changes are made, to hit the OK button to upload changes and close down the setup page directly.

[Speed Control] Speed Reference

The Speed Reference Setup page defines the speed reference setup for the internal speed control logic. Each button on this page will open another setup page to define in detail the required functionality.

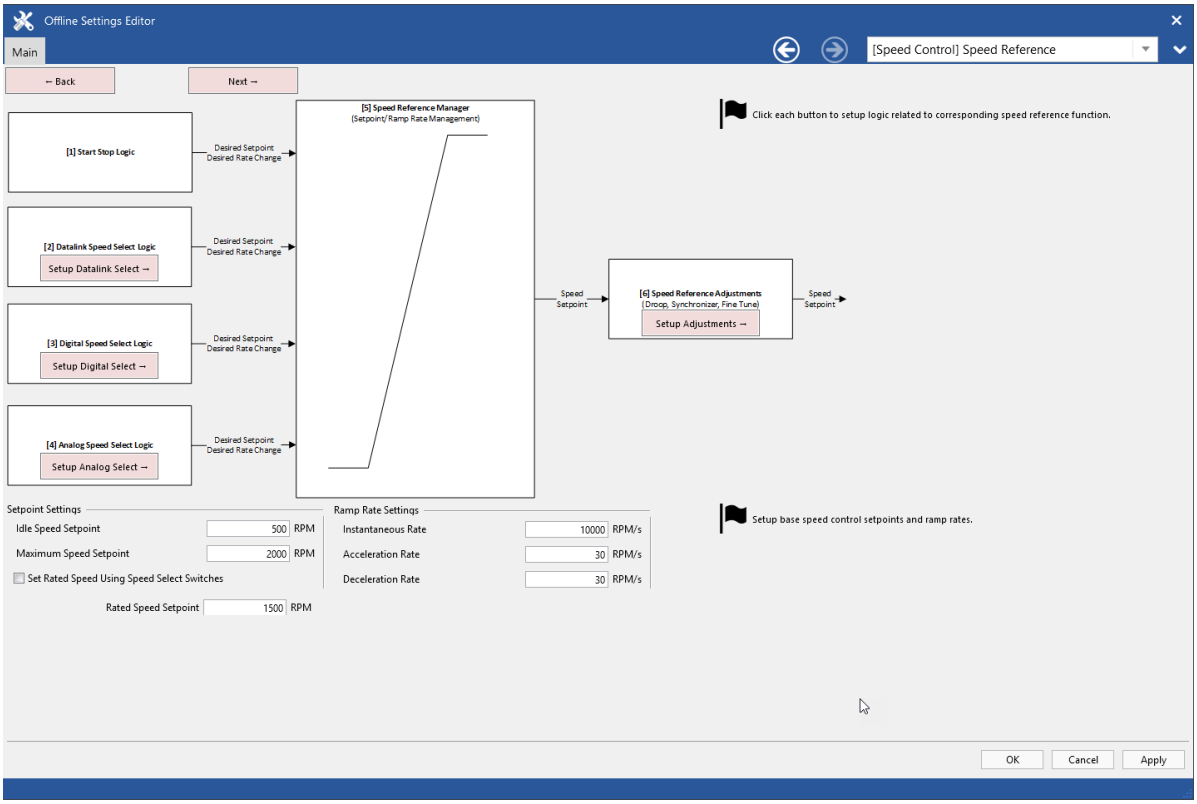


Figure 5-62. [Speed Control] Speed Reference Page

The following are additional details on each selection in the setup page.

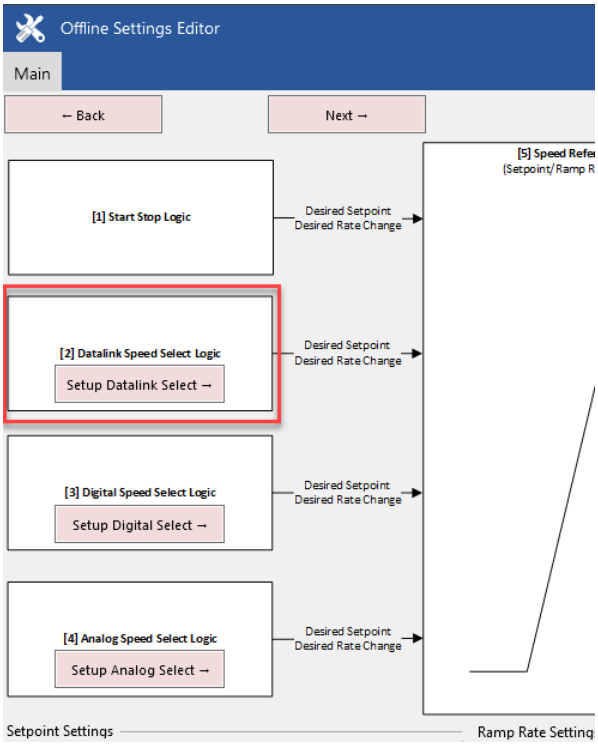


Figure 5-63. Setup Datalink Select

Selecting the Setup Datalink will open a setup page where it is possible to setup either J1939 Standard TSC1 signal or Modbus Related write signal.

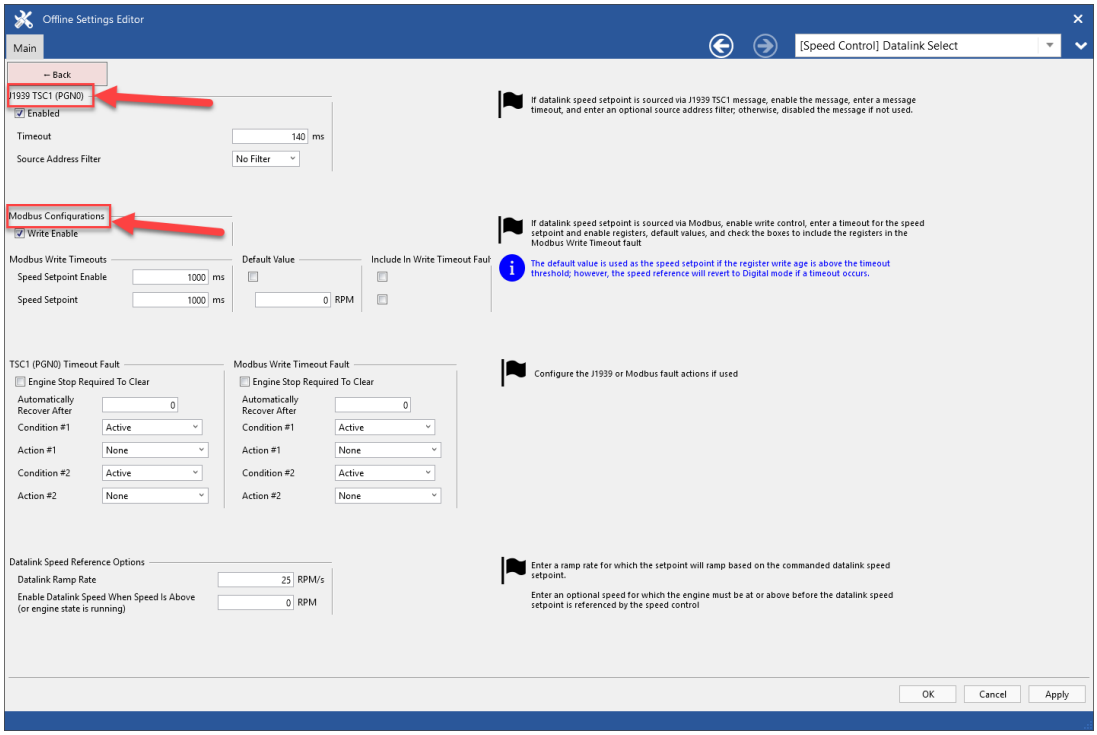


Figure 5-64. [Speed Control] Datalink Select

This setup page also makes it possible to setup the behavior in case of timeout faults for receiving the related messages. Up to two conditions and actions can be defined, even as the reset behavior.

Finally, a dedicated Ramp Rate can be setup when the Datalink setpoint is activated to select how fast the speed changes to the new setpoint. Once defined and setup, the "← Back" button can be pressed to return to the previous page.

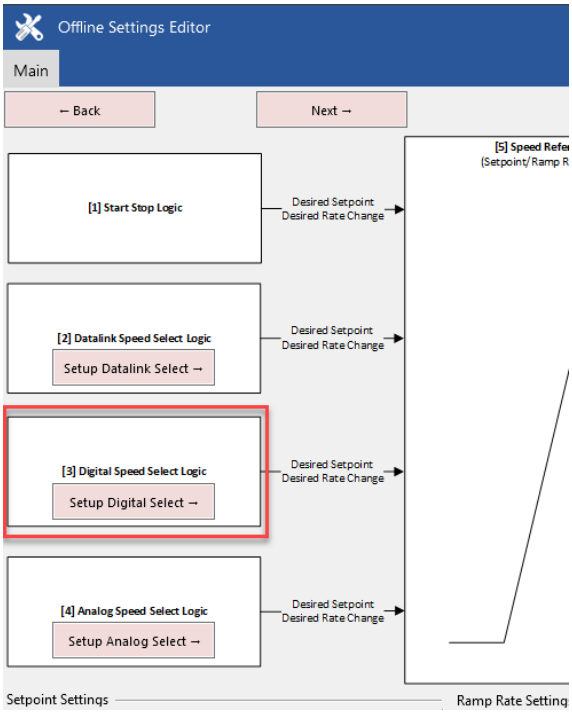


Figure 5-65. Setup Digital Select

Selecting the Setup Digital Select will open a setup page where it is possible to setup the parameters related to the digital speed setting logic.

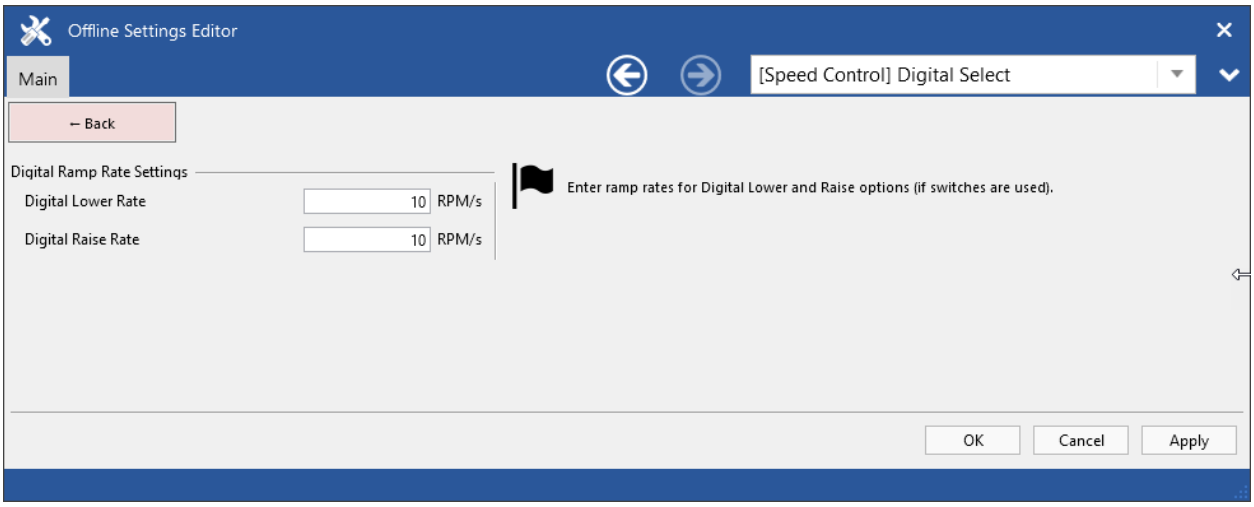


Figure 5-66. Digital Select Setup Page

This setup page can be used to define the ramp rates when Lower or Raise digital speed commands are provided when the engine will be running in digital speed mode, or in analog speed mode with the accelerator backup selected. Once defined and setup, the "← Back" button can be pressed to return to the previous page.

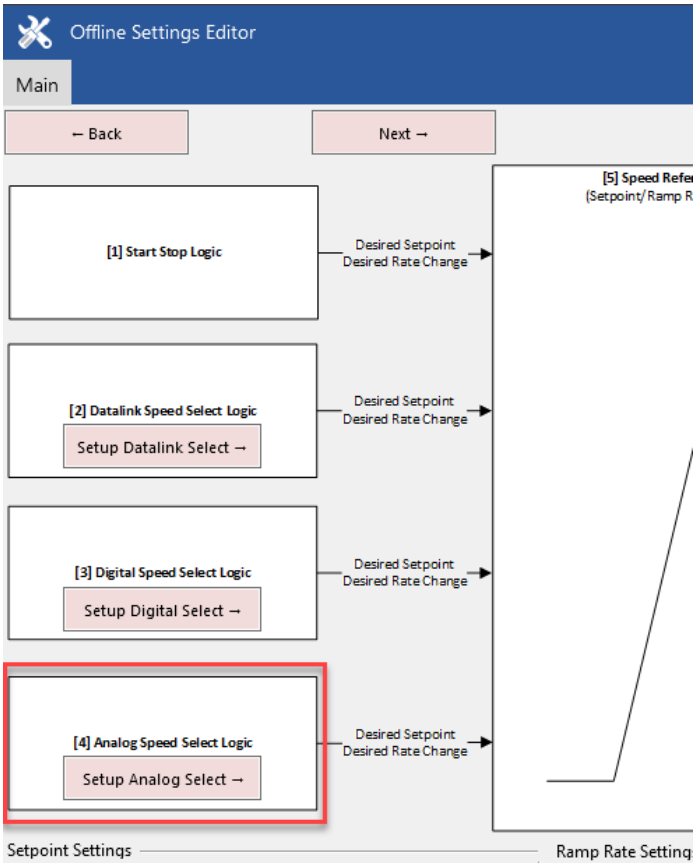


Figure 5-67. Setup Analog Select

Selecting the Setup Analog Select will open a setup page where it is possible to setup the parameters related to the analog speed setting logic. This setup page can be used to define the minimum and maximum analog speed setpoints and the ramp rates. Since the analog speed setting utilizes a deadband to increase or decrease the speed setpoint, the width of the deadbands and hysteresis are also adjustable. The graphical overview provides an illustration of the deadband logic.

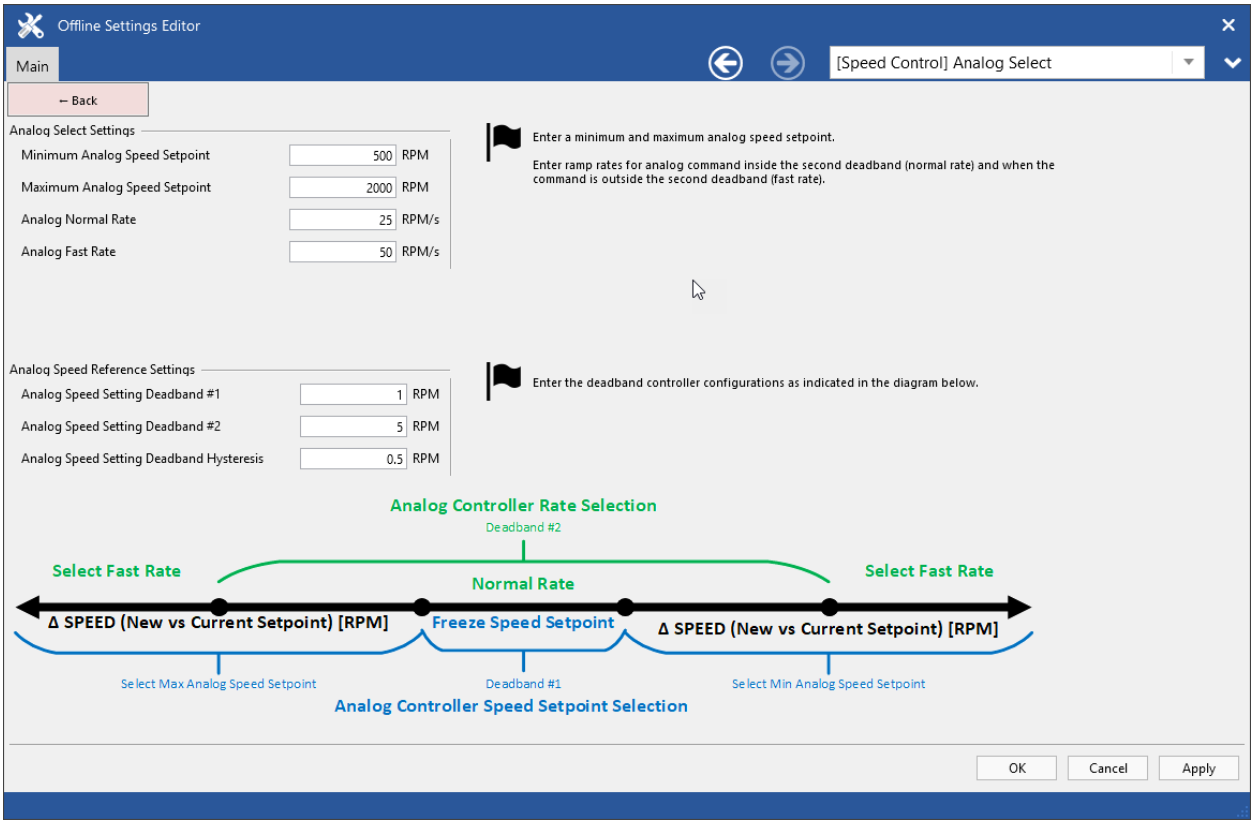


Figure 5-68. Analog Select Setup Page

Pressing the "← Back button" takes you to the overview page where another setup selection can be made.

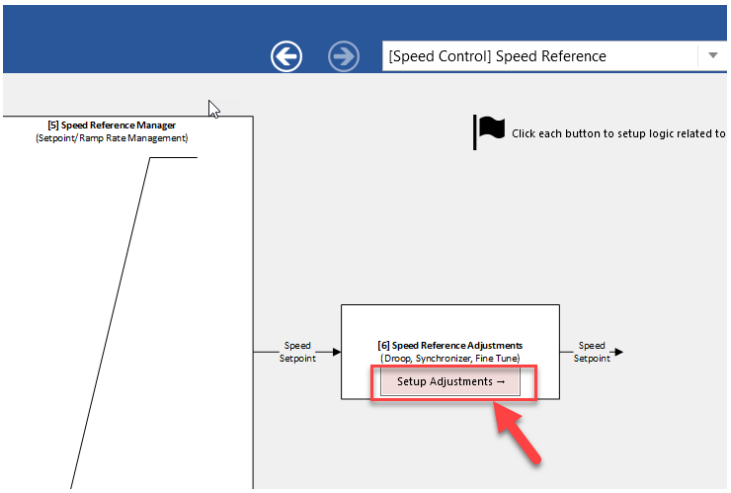


Figure 5-69. Speed Reference Adjustments

Selecting the Speed Reference Adjustments Setup will open a new page, displaying several options that can influence the actual speed reference setpoint. These options are:

A) Droop → Enable Droop using power input option. Enabling this function requires an analog input to be defined as Power Input. If not used, the internal Load Calculation based on fuel amount will be used for load %. This field also must be used to define a setpoint for the % amount.
B) Fine Tune Adjustment Bias → Bias of Speed reference meant for accurate speed setting. Enabling this function requires an analog input to be defined for this functionality.

C) Synchronizer Bias → Enabling this input will enable a +/- bias signal to the actual speed reference, which can be used to synchronize a generator to a live bus or grid. This input requires an analog input that needs to be defined for this functionality. An additional option can be used to disable the synchronizer when connected to the bus or grid, to prevent interference of the analog input when online.
D) GC2 Speed Governing Bias → This is a J1939 Standard CAN message that can be used to bias the speed reference. The 100% bias (max rpm amount) can be defined.
E) GC2 Speed Governing Bias → If enabled, the GC2 message will control the % of bias required.
F) GC1 Droop Amount → This is a J1939 Standard CAN message that can be used to define droop % amount over CAN bus. Based on load %, the droop will decrease speed setpoint accordingly.

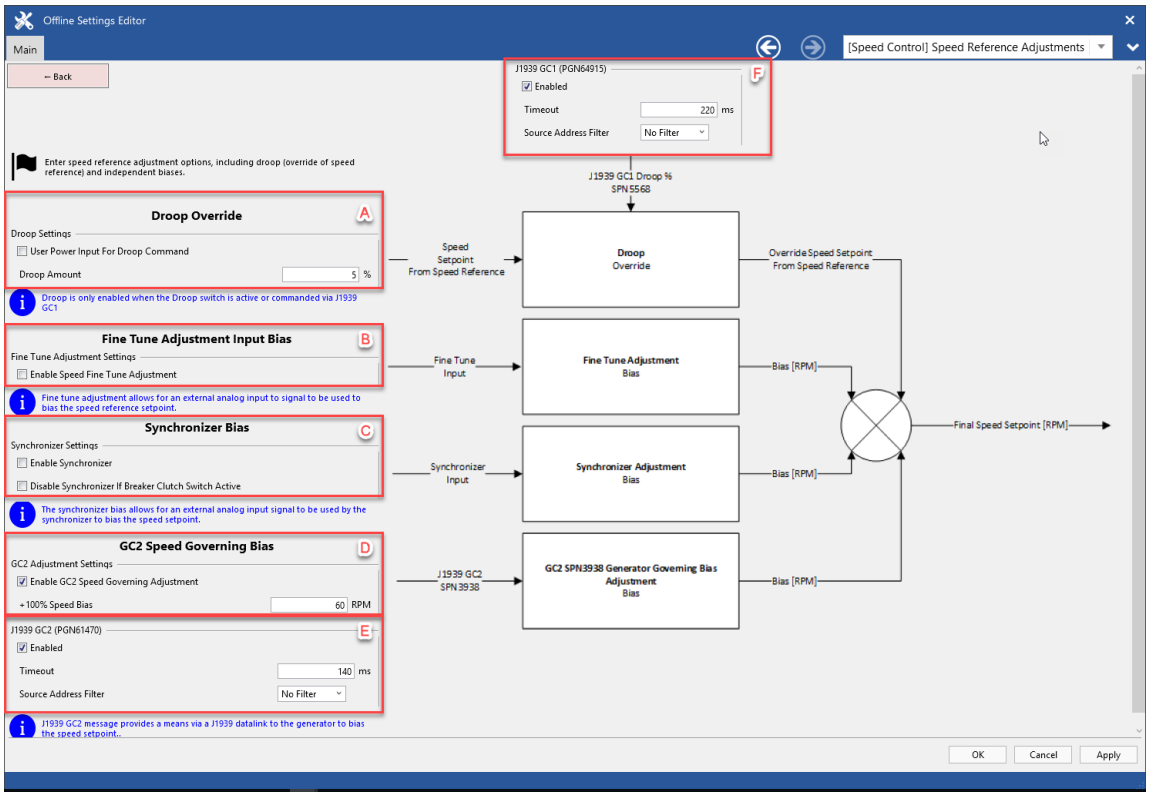


Figure 5-70. Speed Reference Adjustments

Pressing the "← Back button" takes you to the overview page where another setup selection can be made.

The remaining parameters on this page define the Speed Setpoints for Idle and Rated Speed, including the Ramp rates for ramping between the setpoints.

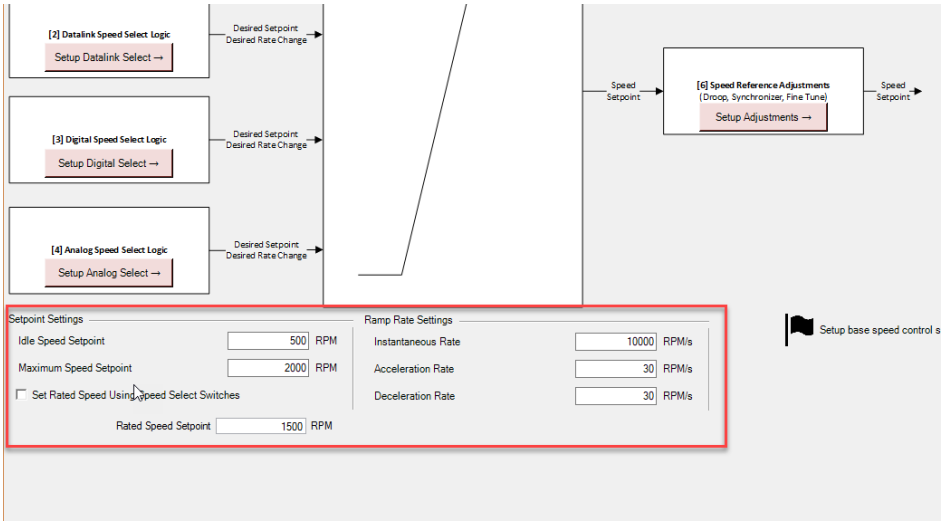


Figure 5-71. Speed Reference Setpoints and Ramp Rates

Pressing the “Next button →” on this page takes you to the [Speed Control] Speed Controller overview page where several selections concerning the speed control can be set. The correct setup of these selections is required to complete the speed control functionality.

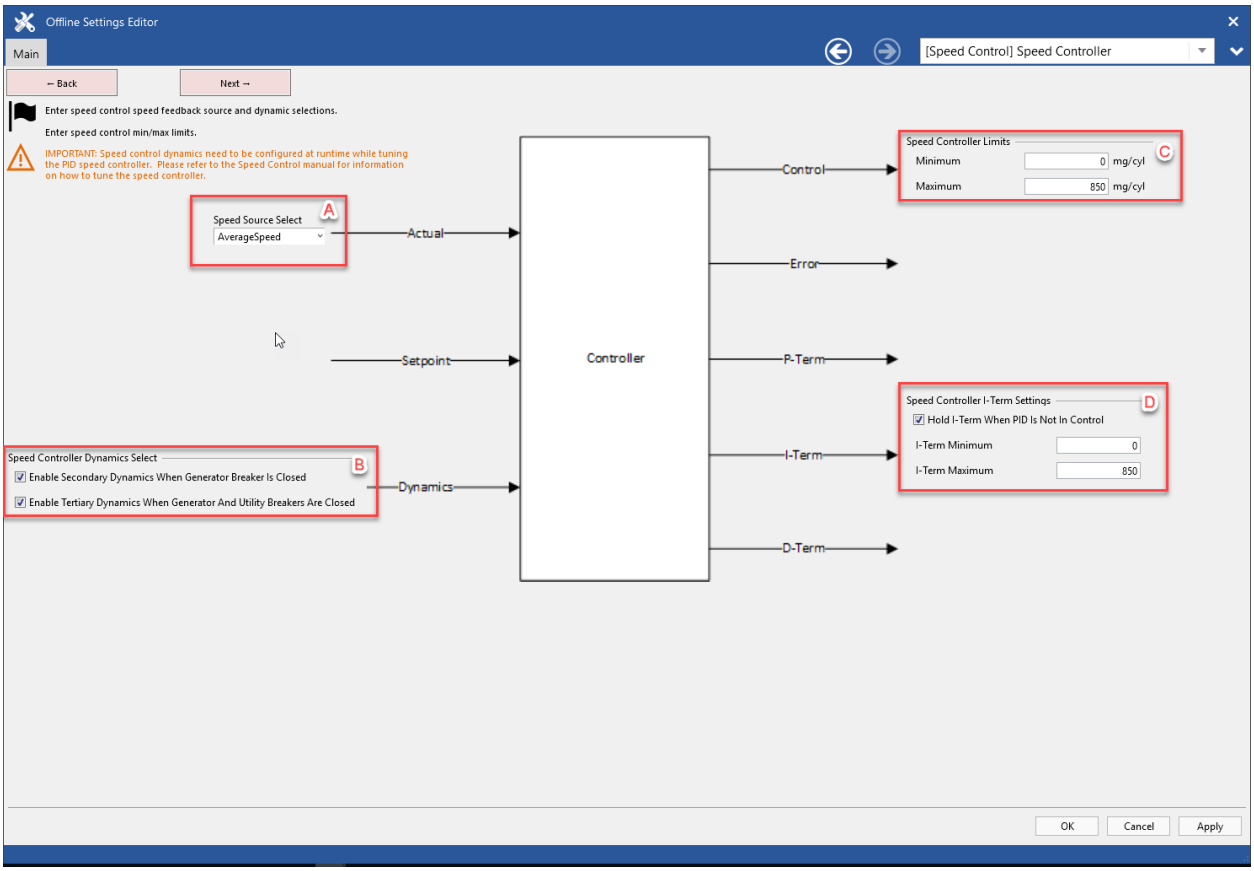


Figure 5-72. [Speed Control] Speed Controller Page

In the above picture, the four main items that can be setup are highlighted with a letter:

A) Speed Source Selection

This selection allows selection of the actual speed signal used as the actual speed source to the PID control. There are three possibilities as listed below:

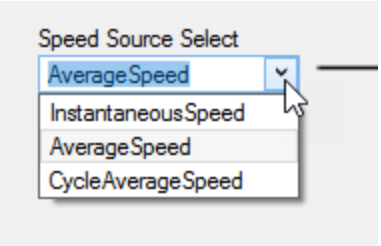


Figure 5-73. Speed Source Selection

Instantaneous Speed → Speed is the unfiltered measure of engine speed

AverageSpeed → Speed is calculated based upon the time taken to traverse adjacent RPM sample points (TDC angles).

CycleAverageSpeed → Speed is calculated using a measured cycle period (accumulation of prior speed sample point timespans) and updates on each sample point that has been defined for the active encoder source.

The instantaneous speed is the fastest updated speed but is also less stable. Instantaneous speed will provide for fast speed control but can also introduce irregularity for speed control. CycleAverageSpeed is the slowest updated speed signal and is therefore stable and provides steady control but might be too slow. The AverageSpeed is a compromise between Instantaneous and CycleAverageSpeed and is a good starting point.

B) Speed Controller Dynamics Select
It is possible to enable a secondary and tertiary dynamics set.

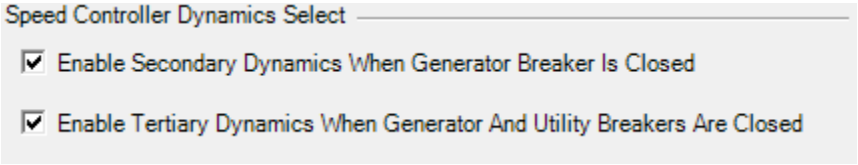


Figure 5-74. Speed Control Dynamics Select

Based on the feedback of the Generator Breaker / Clutch and/or Grid Breaker, the correct dynamics will be active to control the PID response.
Primary Dynamics → Generator Breaker / Clutch is open. Grid Breaker does not matter.
Secondary Dynamics → Generator Breaker / Clutch is closed AND Grid Breaker is open.
Tertiary Dynamics → Generator Breaker / Clutch is closed AND Grid Breaker is closed.

C) Speed Controller Limits
These are the limits of the PID output.

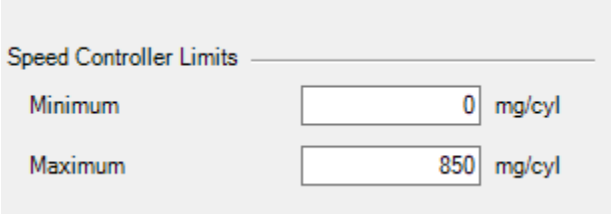


Figure 5-75. Speed Controller Limits

The minimum and maximum fuel amounts that the engine will need must be used here. The PID output range is determined by these values. If full load will not be reached, the maximum value must be increased. If the maximum value is too high, the accuracy of the PID is reduced.

D) Speed Controller I-Term Settings
These settings can be used to limit the integrator value of the PID.

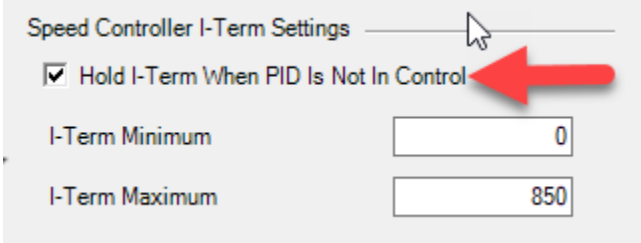


Figure 5-76. Speed Controller I-Term Settings

For the most party, these values are set equally to the PID range values. The selection of the 'Hold I-Term' indicated by the red arrow can be used to prevent the I-Term from integrating out when the PID is not in control. The I-Term will freeze its current value when PID is out of control, which can be due to a fuel limiter or any other logic that can take over the fuel control.

Pressing the "Next button →" on this page takes you to the "[Speed Control] Power Limit Controller" overview page where several selections concerning the power limit controller can be set.

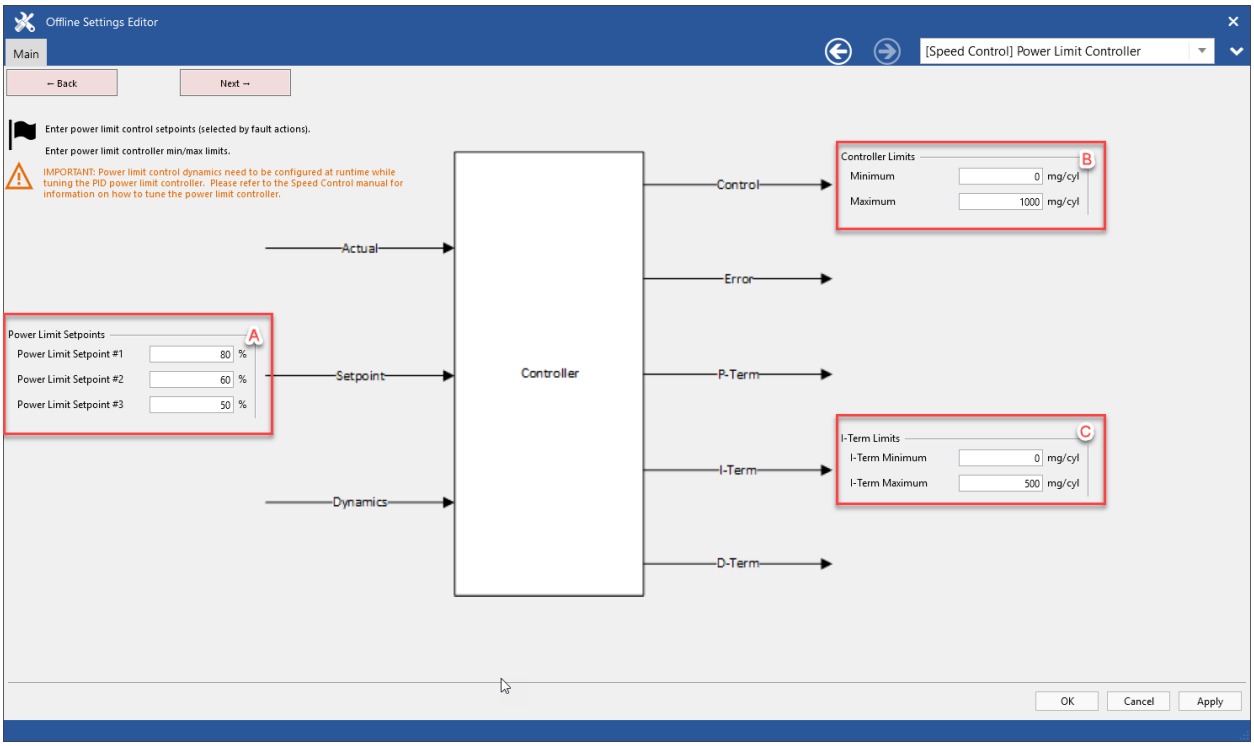


Figure 5-77. [Speed Control] Power Limit Controller Page

The Power Limit Controller reduces engine power to various calibrate able levels when critical faults are active. This setup page helps to define the parameters that are related to the power limiter. In the figure above, these sections are marked with a letter.

A) Power Limit Setpoints

Three different power limits can be setup. These limits need to be set in correspondence to the maximum load of the engine. 80% means 80% of the maximum rated engine load. Each power limit can be set as a Fault action for each fault that requires the power to be limited to protect the engine. NOTE: Care must be taken when in Genset Island mode, as limiting the power can result in under frequency.

B) Power Limit PID Controller Limits
The minimum and maximum fuel amounts that the engine will need, must be used here. Range of the Power Limit PID output is determined by these values.

C) I-Term Limits
These settings can be used to limit the integrator value of the PID.

Pressing the “Next button →” on this page takes you to the “[Speed Control] Fuel Limiting” overview page where several selections concerning the fuel limiters can be set.

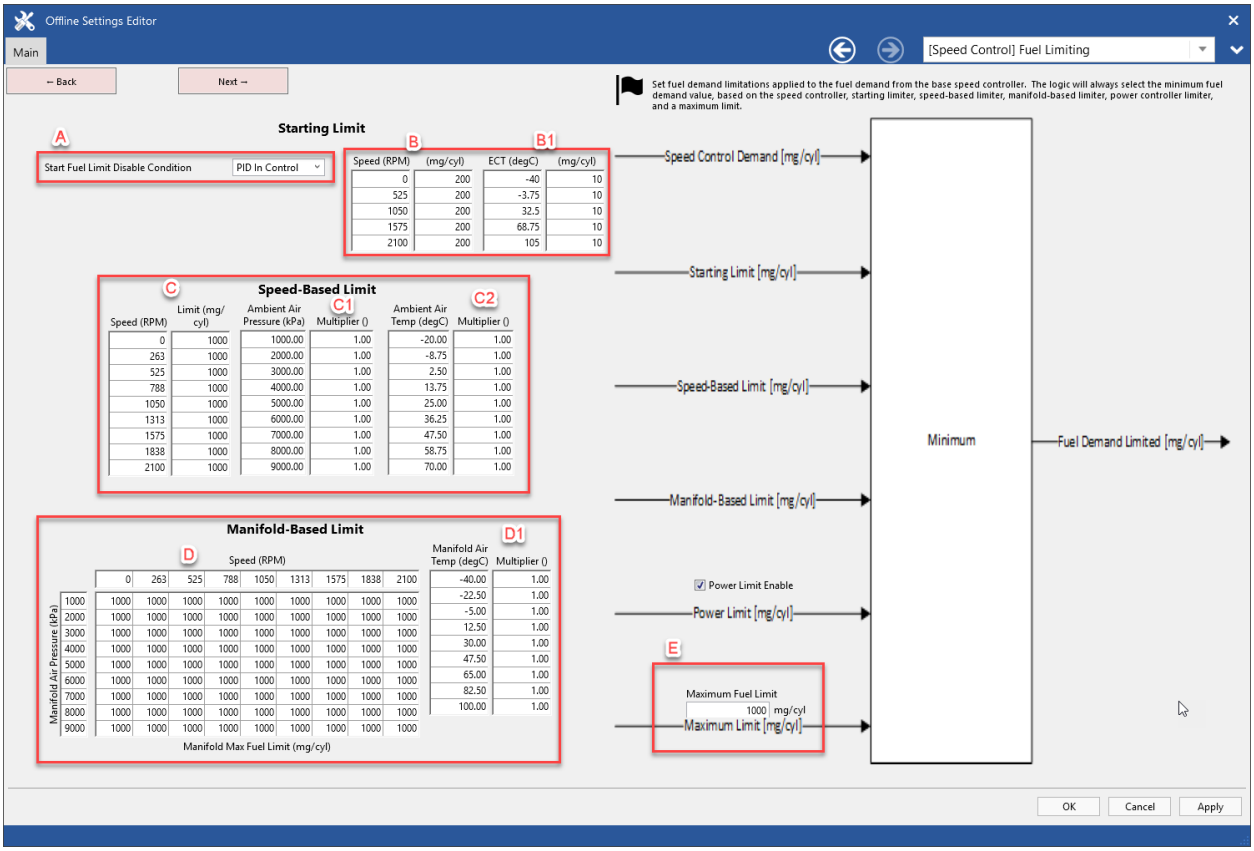


Figure 5-78. [Speed Control] Fuel Limiting Page

The Fuel Limiters are maximum fuel values that are allowed during engine operation. There are several variants available. The figure above has a selection of the possible fuel limiters, all indicated with a letter:

A) Start Fuel Limit Disable Condition
This option determines when the start fuel limiter will be switched off.

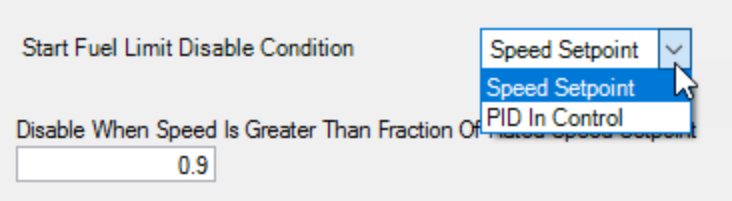


Figure 5-79. Start Fuel Limiter Disable Options

Either 'PID In Control' or 'Speed Setpoint' level reached can be selected here.
Speed Setpoint level reached provides a multiplier where you can define the fraction of rated speed level that the actual speed must reach before the limiter will be switched off.
PID in Control is feedback from the logic when PID has control of the engine, which is the most common way used to disable the start fuel limit.

B) Start Fuel Limiter
The Start Fuel Limiter is a speed dependent fuel limit value that can be compensated by the engine coolant temperature (ECT). The engine coolant temperature table (B1) is an addition on the Start Fuel Limit Table (B), so you can give more fuel when it is colder, for example.

C) Speed Based Fuel Limit
The Speed Based Fuel Limit is also called the Torque Fuel Limit. It is a speed-based fuel limiter to prevent overloading the engine at lower speeds. This feature is mostly used in all-speed applications, like marine propulsion and/or Pump Drive. This Speed Based Fuel Limit can be compensated by Ambient Air Pressure (C1) and Ambient Air Temperature (C2). These are multiplier factors that are applied to the base Speed Based Fuel Limit table.

D) Manifold Air Pressure Based Fuel Limit
This fuel limiter is a speed dependent, manifold air pressure-based fuel limiter.
This limiter can be compensated by the Manifold Air Temperature (D1), which is a multiplier factor to the base table.

E) Maximum Fuel Limit
This is the maximum fuel limit value that will be allowed to provide to the engine.

The Power Limiter is also one of the fuel limiters that can limit the fuel amount. It can be selected on this page to switch the power limiter to be active or not active.

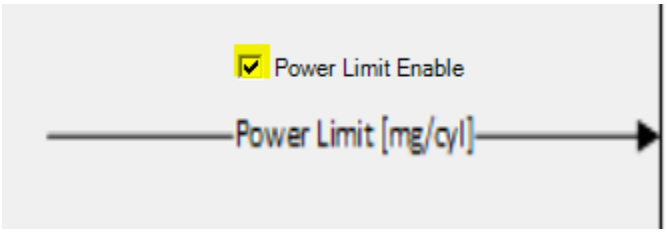


Figure 5-80. Enable/Disable of Power Limiter

All fuel limiters are coupled to an LSS-bus (Low Signal Select), which only allows the lowest value feeding that bus to pass.

Pressing the "Next button →" on this page takes you to the "[Speed Control] Complete" overview page where several selections concerning the load dump feature and Generator Breaker Idle Selected Monitor feature can be set.

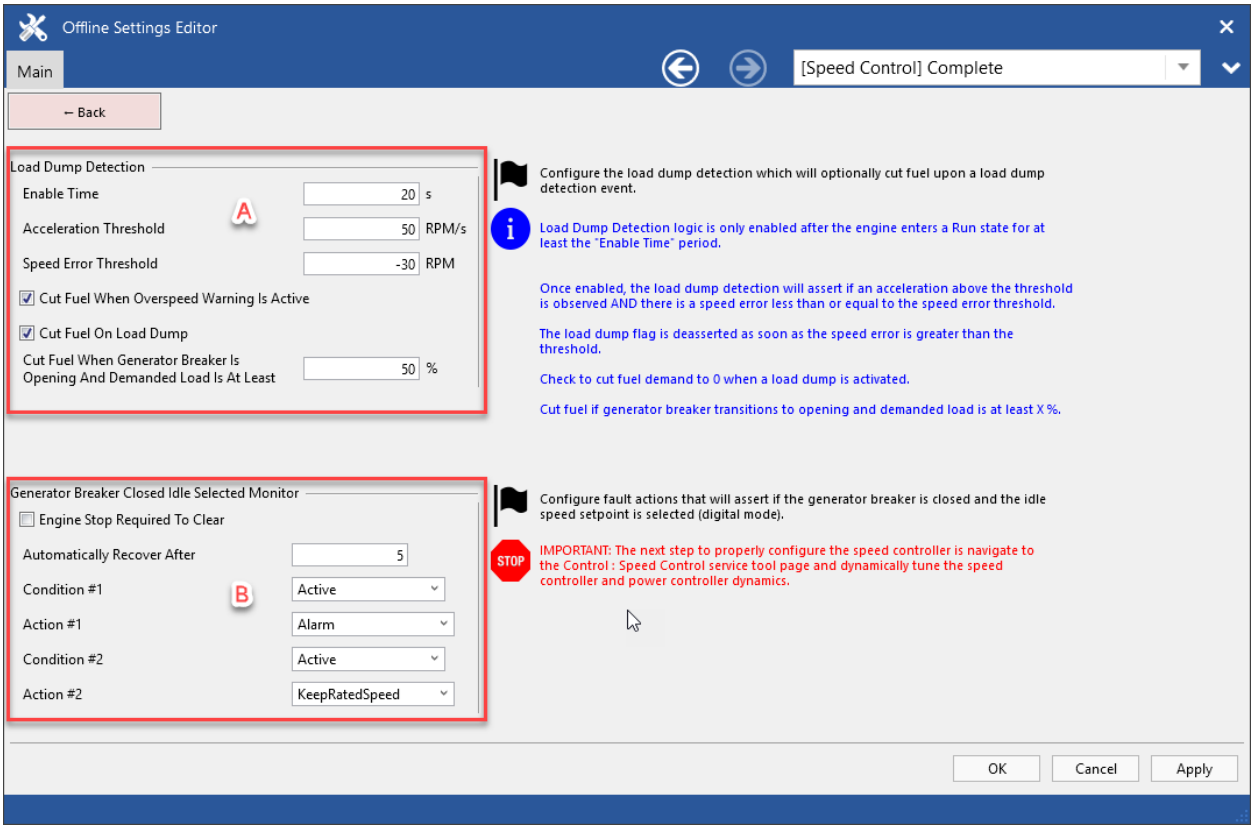


Figure 5-81. [Speed Control] Complete Page

A) Load Dump Detection
Load Dump Detection is a feature that will help reset the integrator value of the PID to zero, when big load changes are observed. The Load Dump feature can be very helpful when Load Rejection tests need to be performed when big load changes take place and fast reaction is required. Load Dump is based on actual speed acceleration in combination with a speed error (Speed Reference – actual speed). When both criteria match, the Load Dump feature will be activated. The Load Dump feature can be combined with a “Cut Fuel” logic, which immediately cuts fuel when active, independent of PID settings. This same “Cut Fuel” logic can also be activated when the generator breaker or clutch opens when the load is above a tunable percentage of load.

B) The Generator Breaker Closed Idle Selected Monitor is a feature that has the option to prevent the engine speed reference from ramping down to idle speed if the generator breaker is still closed. For several reasons, it may be possible that when an engine is online (Breaker Closed) that Idle Speed would be selected. In this case, this feature provides the possibility to block the speed reference to ramp to idle speed reference setpoint. Two actions can be taken when this is happening. Preventing the ramp down to idle speed setpoint can be blocked, when one of the actions is “KeepRatedSpeed”. This keeps engine at the rated speed as long as the breaker is closed.

When changes to the default values in this setup guide have been made, hit the Apply button to upload the changes. This can take a few seconds and when it is done, the Close button will appear. Clicking this button will close the setup page. After changes are made, it is also possible to hit the OK button to upload changes and close the setup page directly.

Chapter 6.
Control – Balancing

Introduction

The Control – Balancing blockset is used to define the Exhaust Gas Balancing control function, so it can be used inside the application. The Exhaust Gas Balancing control function provides a small offset in fuel injection to make the exhaust gas temperature of each cylinder close to equal, depending on certain settings.

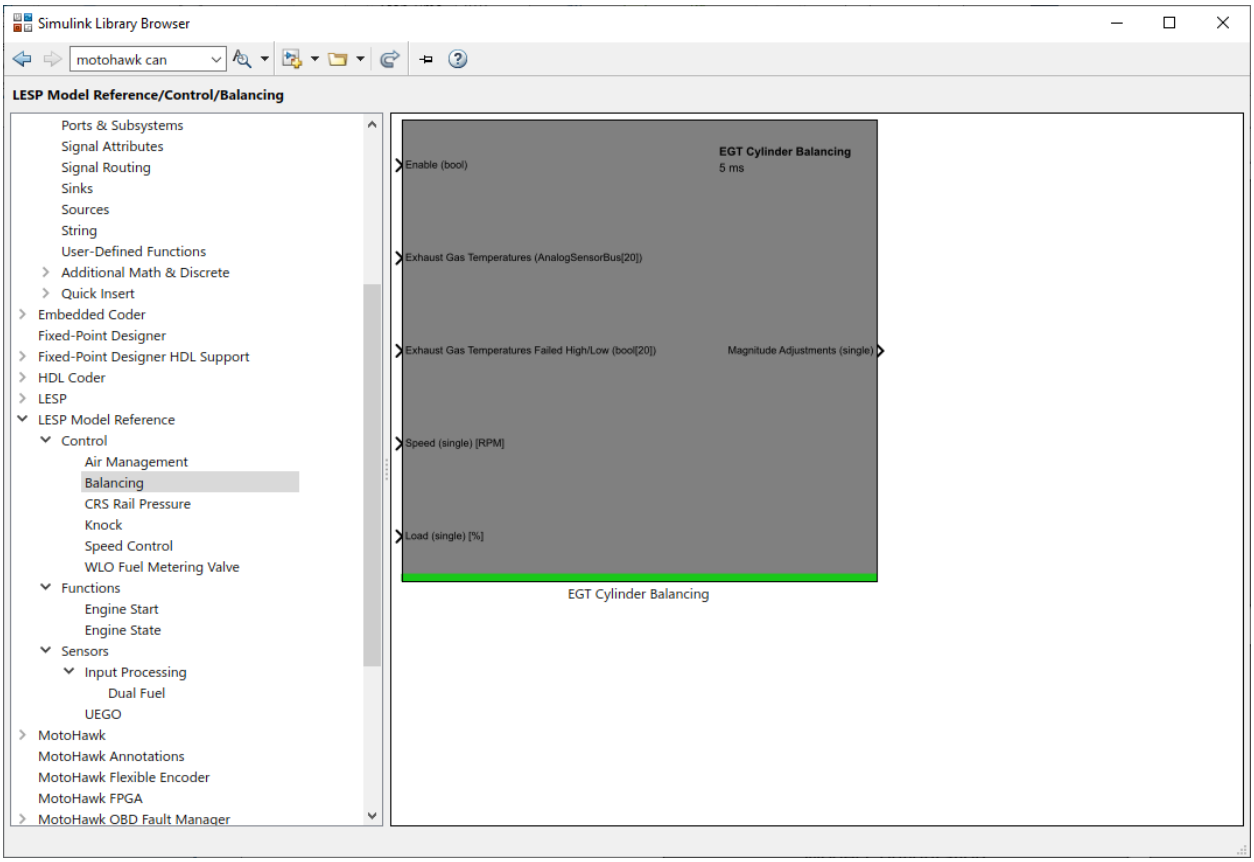


Figure 6-1. LESP Model Reference Control - Balancing Block

- EGT Cylinder Balancing
  This model reference block contains the logic to perform Exhaust Gas Balancing functionality inside the application. It requires the Exhaust Gas Temperature measurement of each individual cylinder to perform an offset for injection quantity to compensate for deviations between the different cylinders.

EGT Cylinder Balancing Block Interface

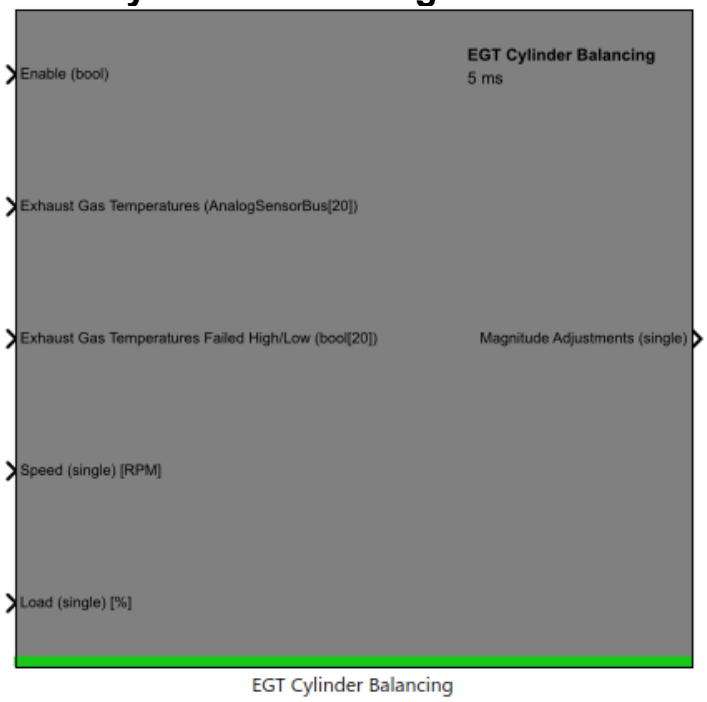


Figure 6-2. LESP Model Reference EGT Cylinder Balancing Block

Table 6-1. EGT Cylinder Balancing Block

Port	DataType	Description
[In] Enable	boolean	Input that confirms if the logic inside the block is enabled or not.
[In] Exhaust Gas Temperatures [20]	AnalogSensorBus	This input requires an array of the exhaust gas temperatures of each individual cylinder. Bus Details: AnalogSensorBus: - FilteredValue - MeasurementStatus_enum MeasurementStatus_enum: (lesp_measurement_status_enum) 0 – Valid 1 – Disabled 2 – SignalLow 3 – SignalHigh 4 – SensorSupplyFailed 5 – ReferenceSourceFailed 6 – OpenCircuit
[In] Exhaust Gas Temperatures Failed High/Low [20]	boolean	This input requires an array of the exhaust Gas Temperature Sensor faults of each individual cylinder. When temperature is low or high, it will be reported here, and the internal logic will use it to implement the correct strategy.
[In] Speed (RPM)	single	This input requires the actual Speed Signal in RPM as defined in the application. It will be used to define the speed threshold to enable the EGT correction logic.

[In] Load (%)	single	This input requires the actual Load Signal in % as defined in the application. It will be used to define the load threshold to enable the EGT correction logic.
[Out] Magnitude Adjustments	single	This output provides an array of the magnitude adjustments that needs to be added to the individual cylinder fuel amounts. This will have an effect on the exhaust temperatures, so the control logic inside the block can adjust/correct per cylinder when required.

Example Model Using the Blocks

Below is a very simple setup using MotoHawk Calibration and MotoHawk Probes blocks to make the model suitable for building.

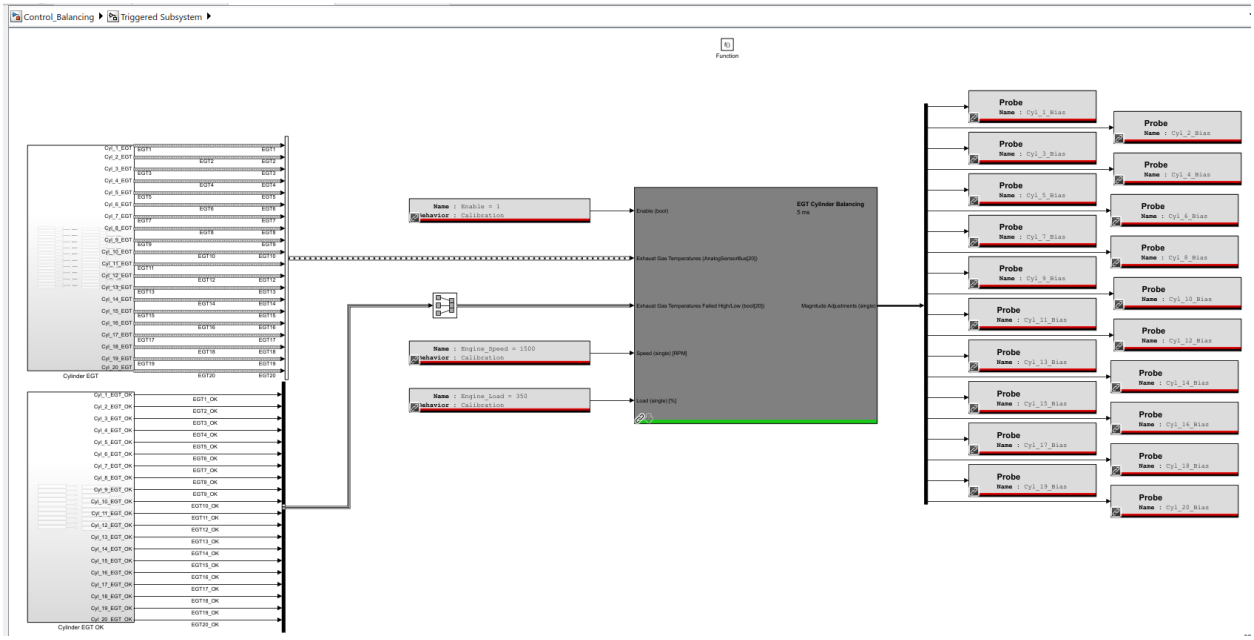


Figure 6-3. Simple Example Model Using the Control – Balancing Block

When compiling, the required files will generate and if the Toolkit required blocks are also added to the application model, it will generate the required SID files for the Toolkit HMI tool creation.

Default ToolKit Pages

Opening a new Toolkit application and selecting the correctly generated .sid file will look like the screenshot below.

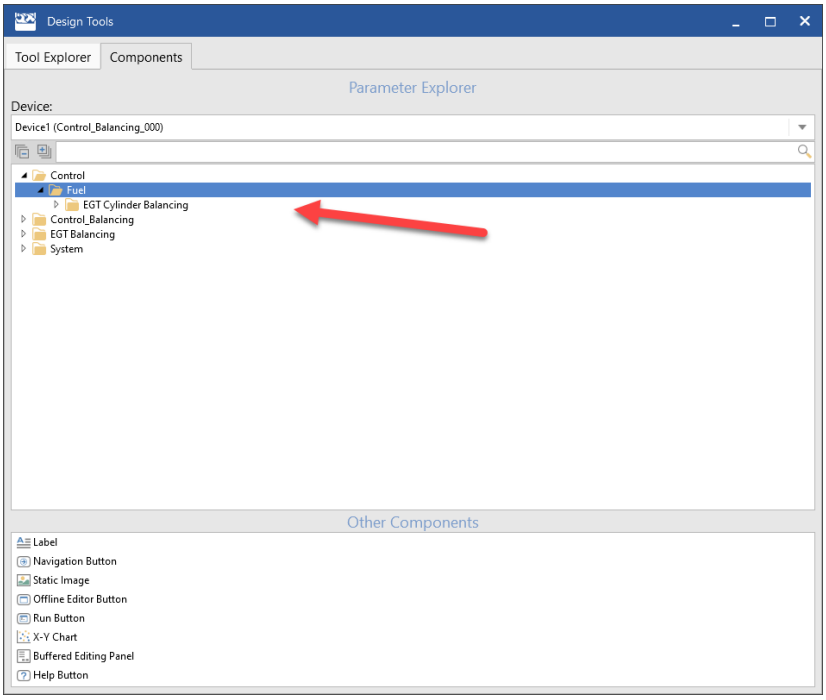
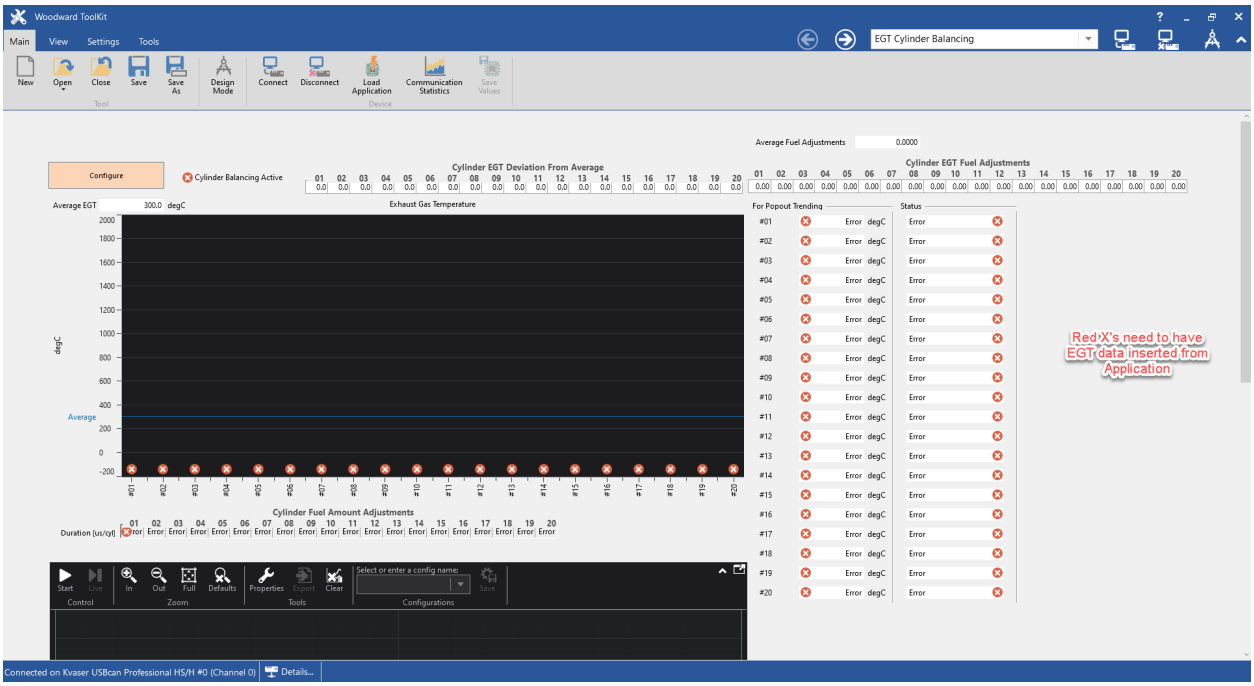


Figure 6-4. SID File Selection Control - Balancing Logic

Below are a few of the Toolkit pages that contain most of the defined parameters in the example model. Of course, the real sensors and data must be attached to complete it for the application that is being worked on. For that reason, red X's can be seen on some pages.



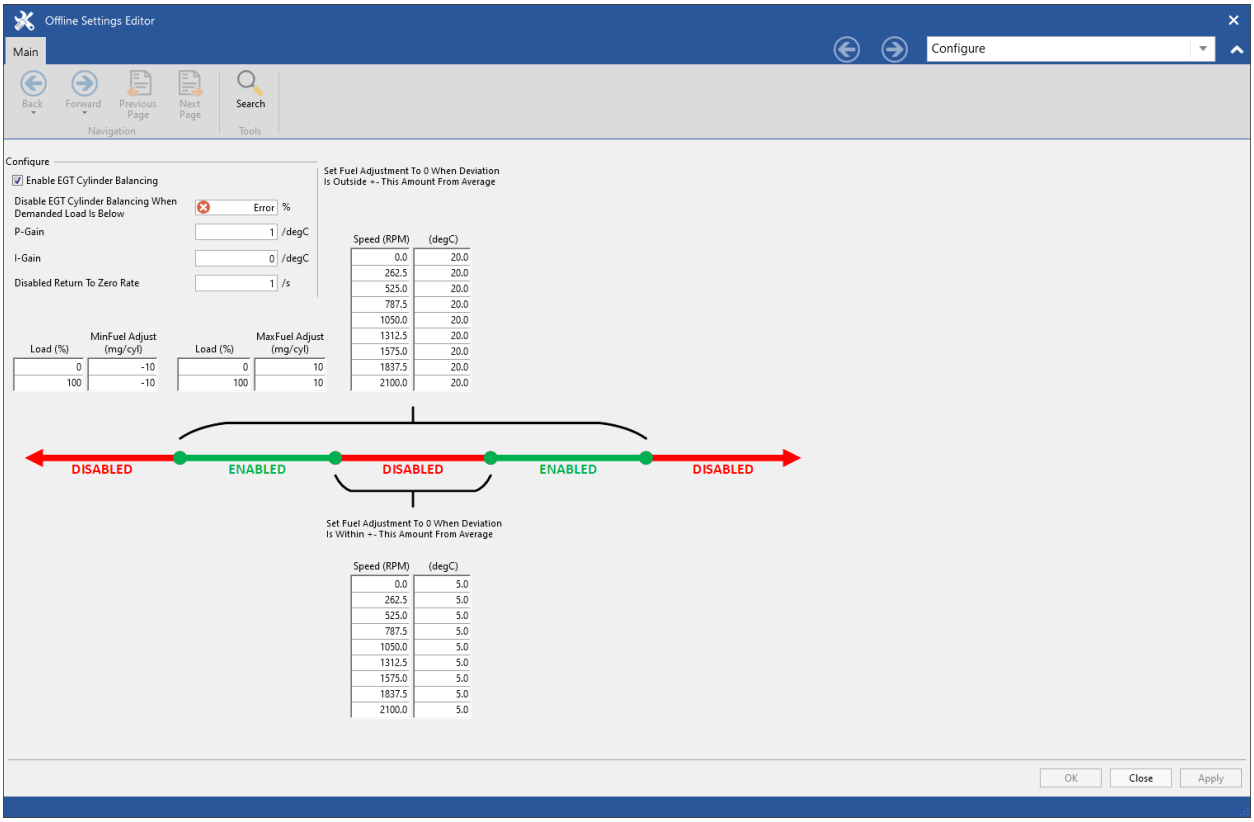


Figure 6-5. Examples of the Toolkit Pages

Functional Description – EGT Cylinder Balancing

The LESP Model Reference Block logic supports cylinder exhaust gas temperature-based cylinder balancing by biasing the fuel injection durations of each cylinder to normalize the exhaust gas temperature of the individual cylinders. This feature is called Exhaust Gas Temperature Balancing. The following describes more details on this feature and how to set it up.

Setup Exhaust Gas Temperature Sensors

To set up the exhaust gas temperature, the Toolkit can be used to define the correct LECM hardware selection for the correct cylinder exhaust as temperature sensor. In most cases, these will be the thermocouple sensors, which can be K-type or J-type (selectable).

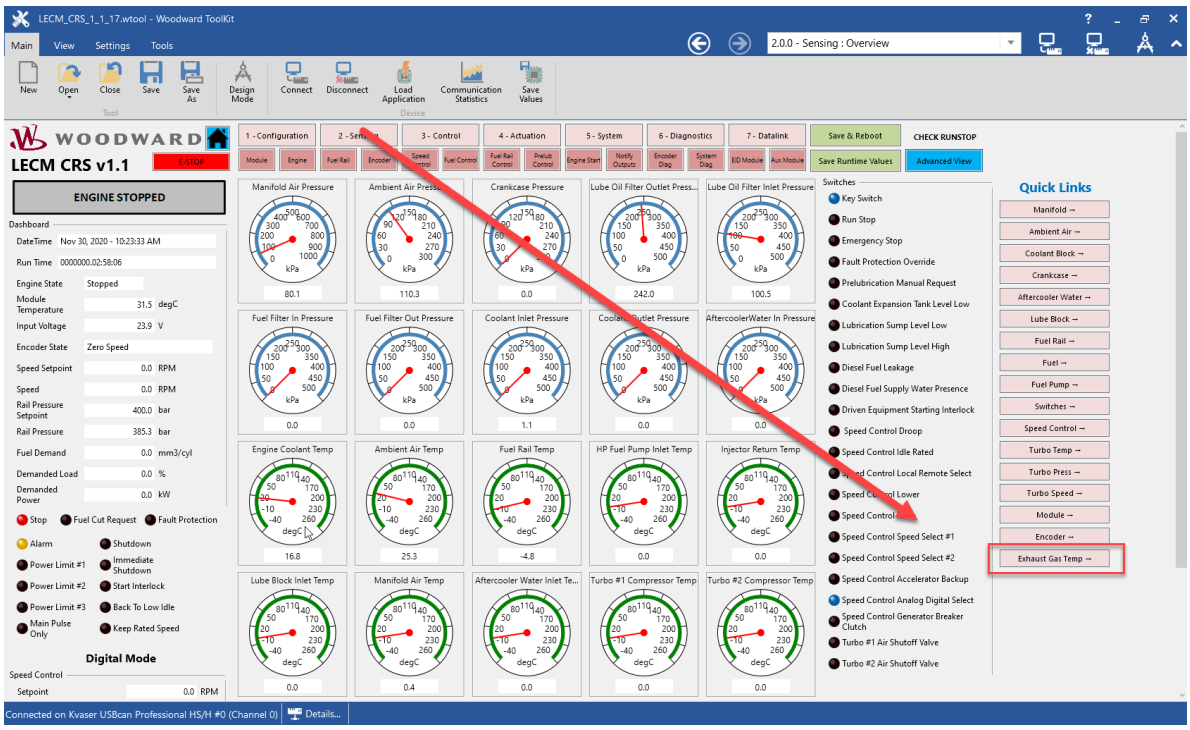


Figure 6-6. Setup of the Exhaust Gas Temperature Sensors

Once the selection is made, you can set up each sensor and select the correct sensor type. Filter time constant, default strategy should be followed in case of issues with the sensor and a default value.

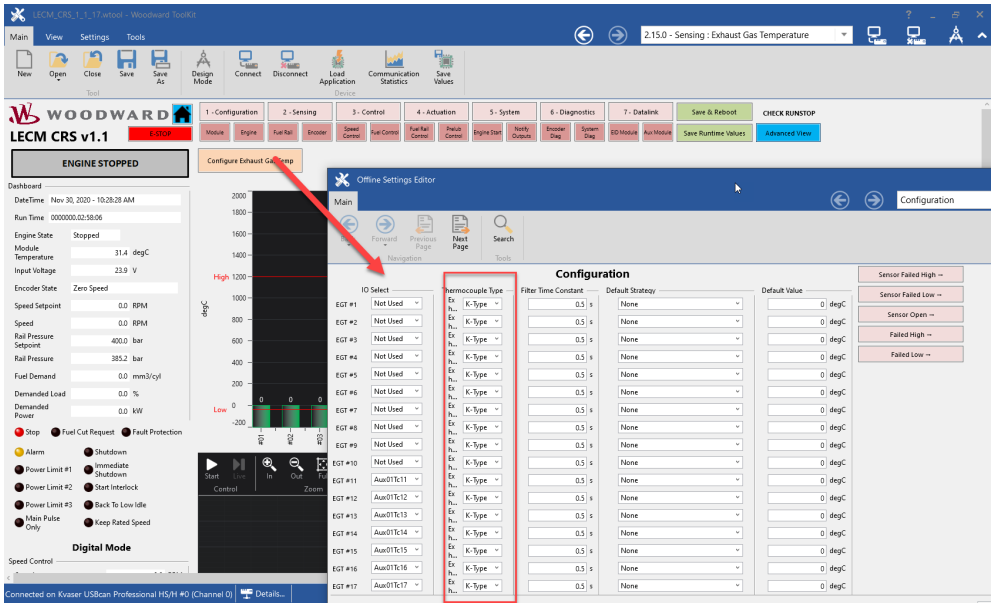


Figure 6-7. Exhaust Gas Temperature Configurations

The configuration screen also allows the possibility to set up the sensor failures and operating high and low temperature settings. Each will have the possibility to set up its own Fault Action.

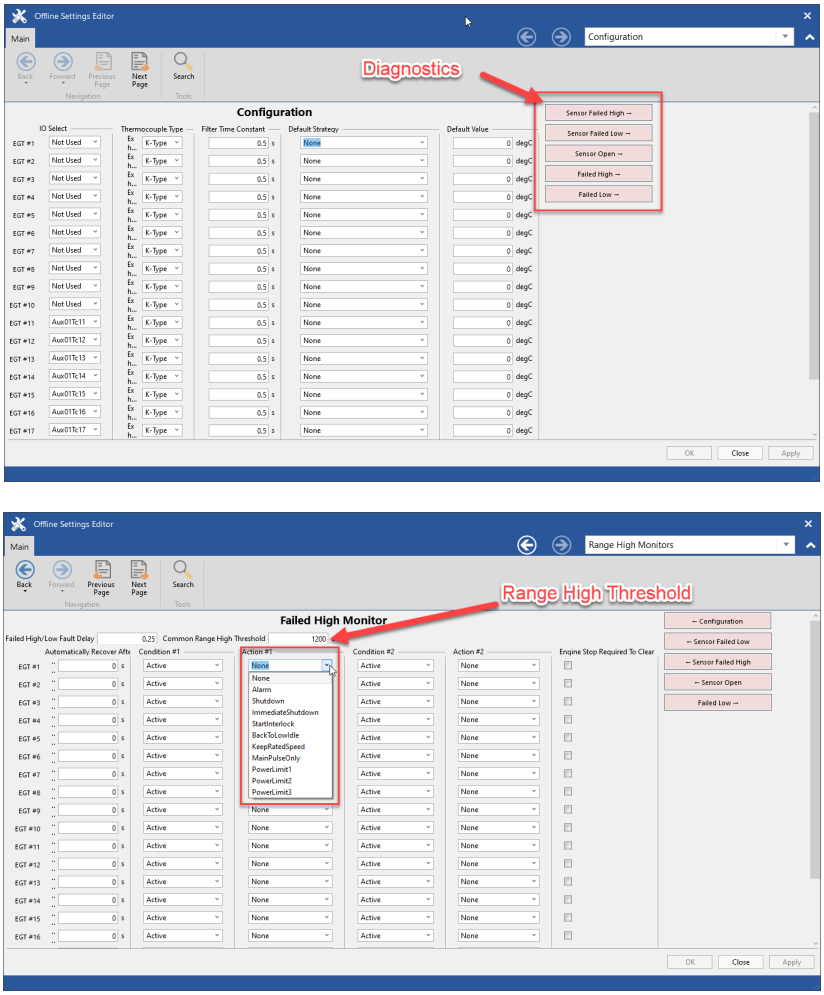


Figure 6-8. Exhaust Gas Temperature Diagnostics (Example Range High)

Exhaust Gas Balancing – Control Setup

Once the sensor setup is complete, the EGT Cylinder Balancing Control can be set up on the Service Tool page, as shown in the example below.

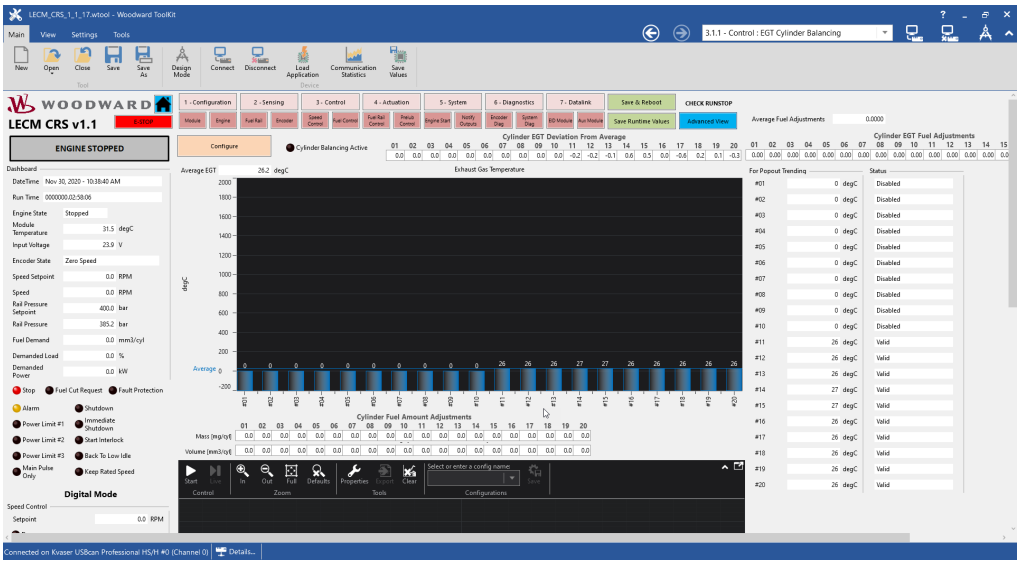


Figure 6-9. Exhaust Gas Temperature – Control

Clicking the Configure button will open the configuration window to set up the control behavior as required.

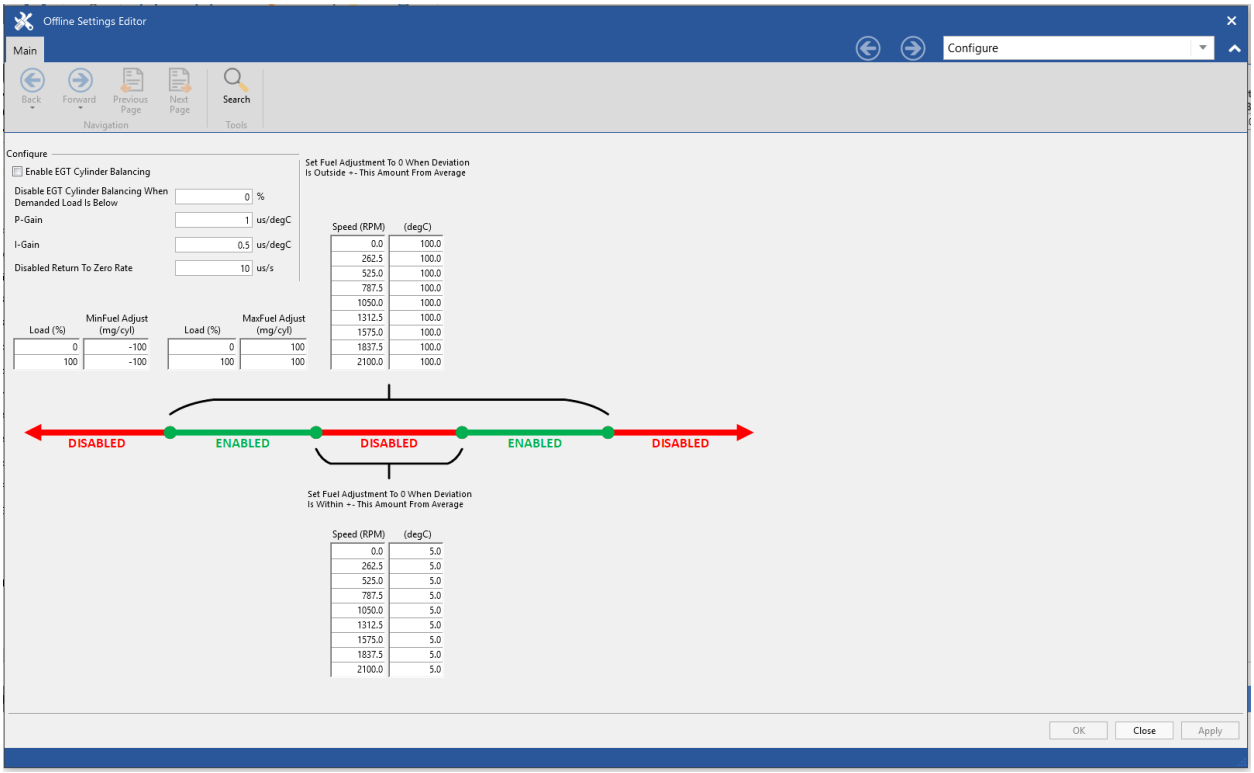
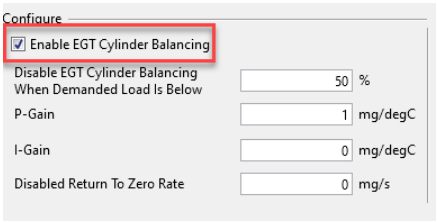


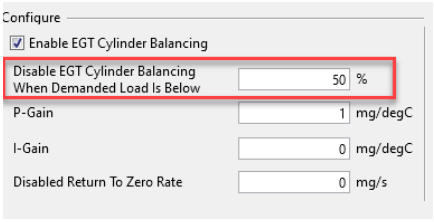
Figure 6-10. Exhaust Gas Temperature Balancing – Configuration Setup

The Exhaust Gas Temperature Balancing logic calculates an average of the EGT's and compares each cylinder EGT to the average of all. Depending on the difference, it will bias the fuel amount for each respective cylinder to normalize the temperature to the average value. This is done by a PI controller.

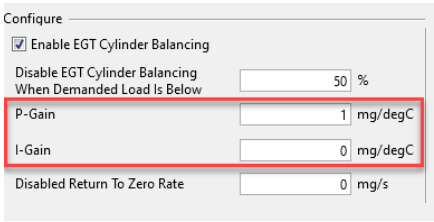
Once the EGT's are all in the defined range, the bias will stop and maintain until the EGT balancing is disabled by load setpoint or manually switched off.



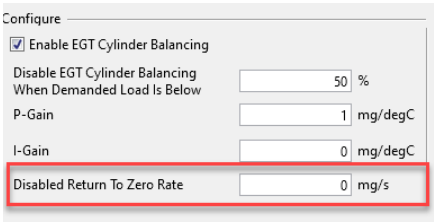
The Enable EGT Cylinder Balancing checkbox can be used to enable and disable the EGT Cylinder Balancing logic.
Checked → Activated,
Unchecked → De-Activated



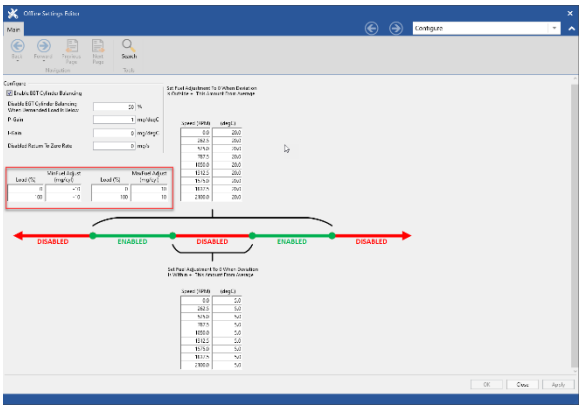
The 'Disable EGT Cylinder Balancing When Demanded Load is below' ….% parameter is used to automatically disable EGT balancing when the demanded load is below the defined threshold. Above this value, the EGT balancing will be active if the above mentioned 'Enable Checkbox' is selected.



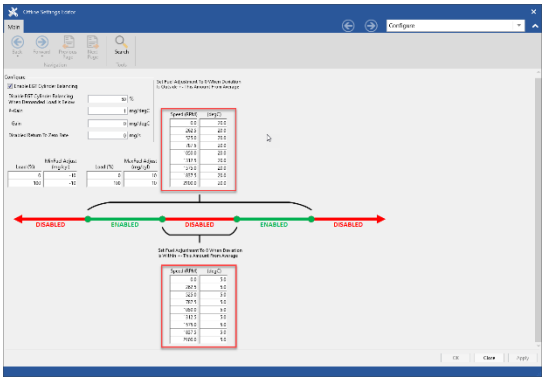
The EGT Cylinder balancing is a PI control loop. The Proportional-Gain and Integral Gain values can be set with these parameters. Values that are too high will make control unstable. Values that are too low will result in a slow response. During commissioning, the ideal settings must be determined.



When the EGT Cylinder balancing is switched off, this is the rate the duration bias will return to zero again. A value of zero will reset the cylinder trims to zero immediately.



These tables are demanded load dependent maximum bias values for the duration bias to correct the temperature. There is a lower limit value and a higher limit value. If correctly calibrated, the bias will stay within these limits.



The upper and lower limit of where the EGT Cylinder balance control will be active. These are speed dependent tables that can be set up. When there is too much deviation, the bias will stay on zero (0). The balancing will be active only in the green area of the picture.

Chapter 7.
Control – Air Management

Introduction

The ‘Control – Air Management’ blockset is used to define the throttle control function, so it can be used inside the application. Controlling the throttle will provide control over the air flow in the engine and can be used for starting performance improvements or for air/fuel ratio adjustments.

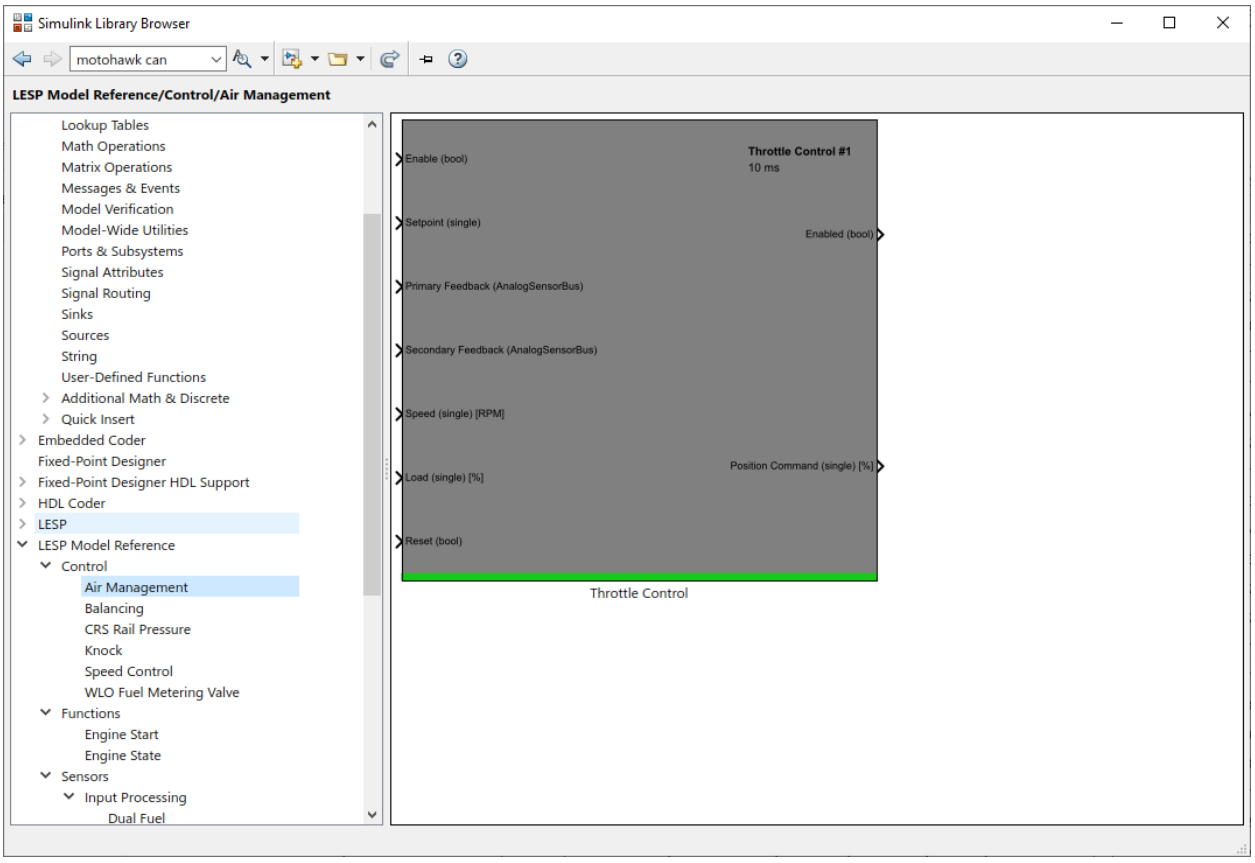


Figure 7-1. LESP Model Reference ‘Control – Throttle Control’ Block

- Control – Throttle Control
  This model reference block contains the logic to perform throttle control functionality inside the application. This can be used for air/fuel ratio control purposes.
  It can be used for real throttle valve control but can also be used to control bypass or wastegate valve.

Throttle Control Block Interface

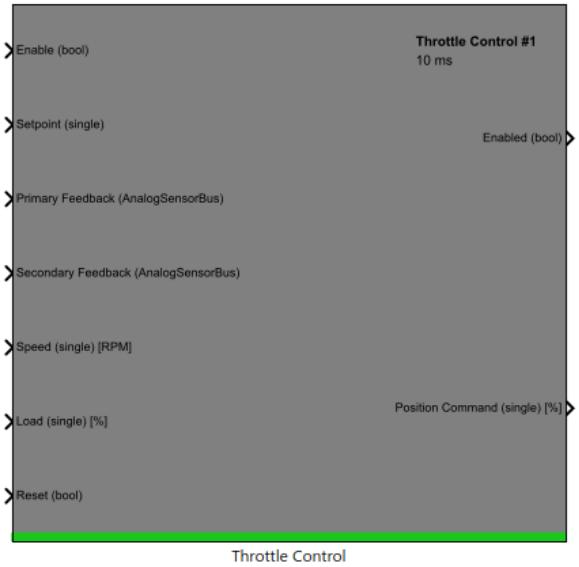


Figure 7-2. LESP Model Reference ‘Throttle Control’ Block

Table 7-1. ‘Throttle Control’ Block

Port	DataType	Description
[In] Enable	boolean	Input that defines if the logic inside the block is enabled or not.
[In] Setpoint	single	This input requires an array of the exhaust Gas Temperatures of each individual cylinder.
[In] Primary Feedback	AnalogSensorBus	This input requires the primary feedback of the throttle valve. This is normally an analog value that will be proportional to the valve position. Bus Details: AnalogSensorBus: - FilteredValue - MeasurementStatus_enum MeasurementStatus_enum: (lesp_measurement_status_enum) 0 – Valid 1 – Disabled 2 – SignalLow 3 – SignalHigh 4 – SensorSupplyFailed 5 – ReferenceSourceFailed 6 – OpenCircuit
[In] Secondary Feedback	AnalogSensorBus	This input requires the secondary feedback of the throttle valve. This normally comes through bus communication like CAN - J1939, CAN open, etc… Bus Details: AnalogSensorBus: - FilteredValue - MeasurementStatus_enum MeasurementStatus_enum: (lesp_measurement_status_enum)

		0 – Valid 1 – Disabled 2 – SignalLow 3 – SignalHigh 4 – SensorSupplyFailed 5 – ReferenceSourceFailed 6 – OpenCircuit
[In] Speed (RPM)	single	This input requires the actual Speed Signal in RPM as defined in the application. It will be used to define the speed threshold to enable the Throttle Control logic.
[In] Load (%)	single	This input requires the actual Load Signal in % as defined in the application. It will be used to define the load threshold to enable the Throttle Control logic.
[In] Reset	boolean	Input that requires the application to be reset and connected. It is required to reset a fault condition inside the Throttle Control reference block.
[Out] Enabled	boolean	This output provides feedback if the block is enabled or not.
[Out] Position Command (%)	Single	This output provides the position command for the throttle valve

Example Model Using the Blocks

Below is a very simple setup using MotoHawk Calibration and MotoHawk Probes blocks to make the model suitable for building.

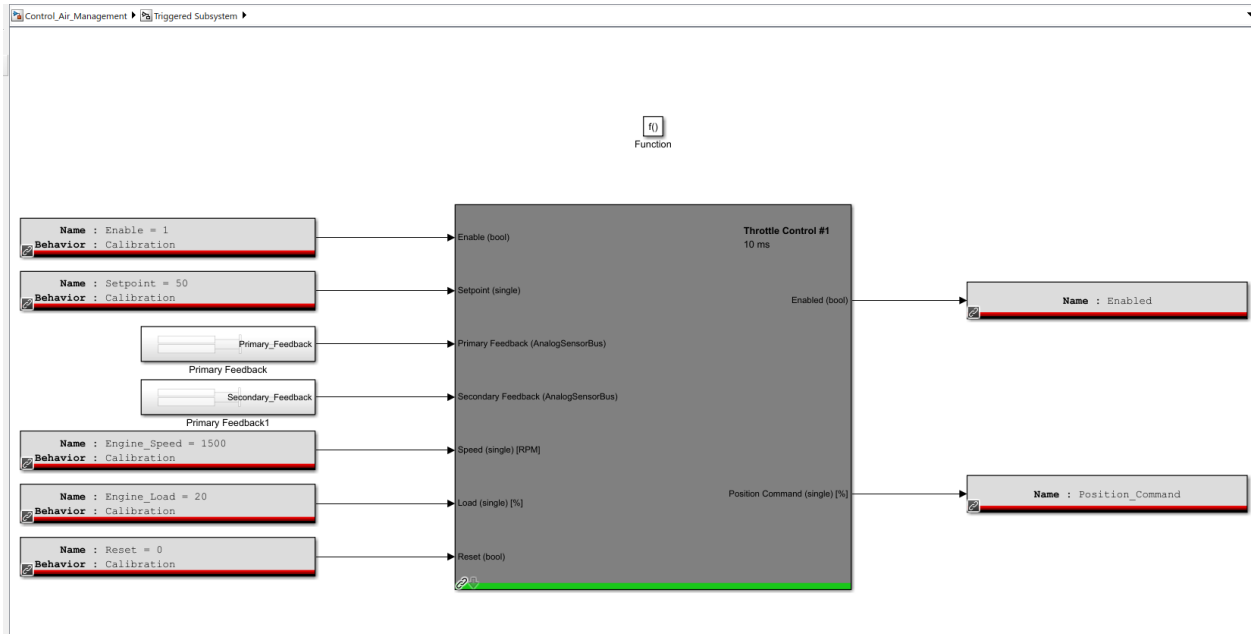


Figure 7-3. Simple Example Model, Using the Control - Air Management Block

The required files will generate when compiling. If the Toolkit required blocks are also added to the application model, it will generate the required SID files for the Toolkit HMI tool creation.

Default ToolKit Pages

Opening a new Toolkit application and selecting the correctly generated .sid file will look like the screenshot below.

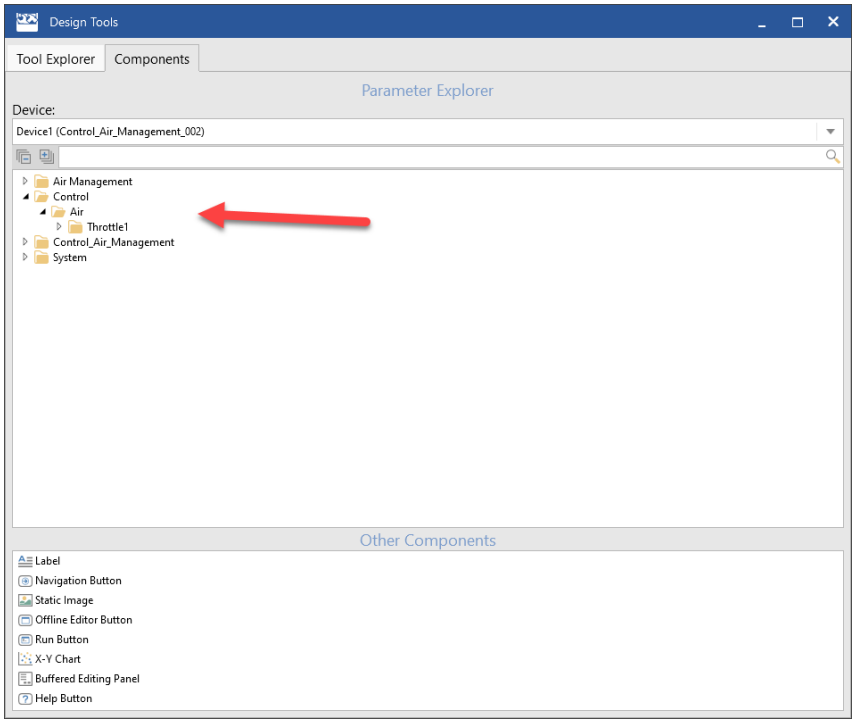
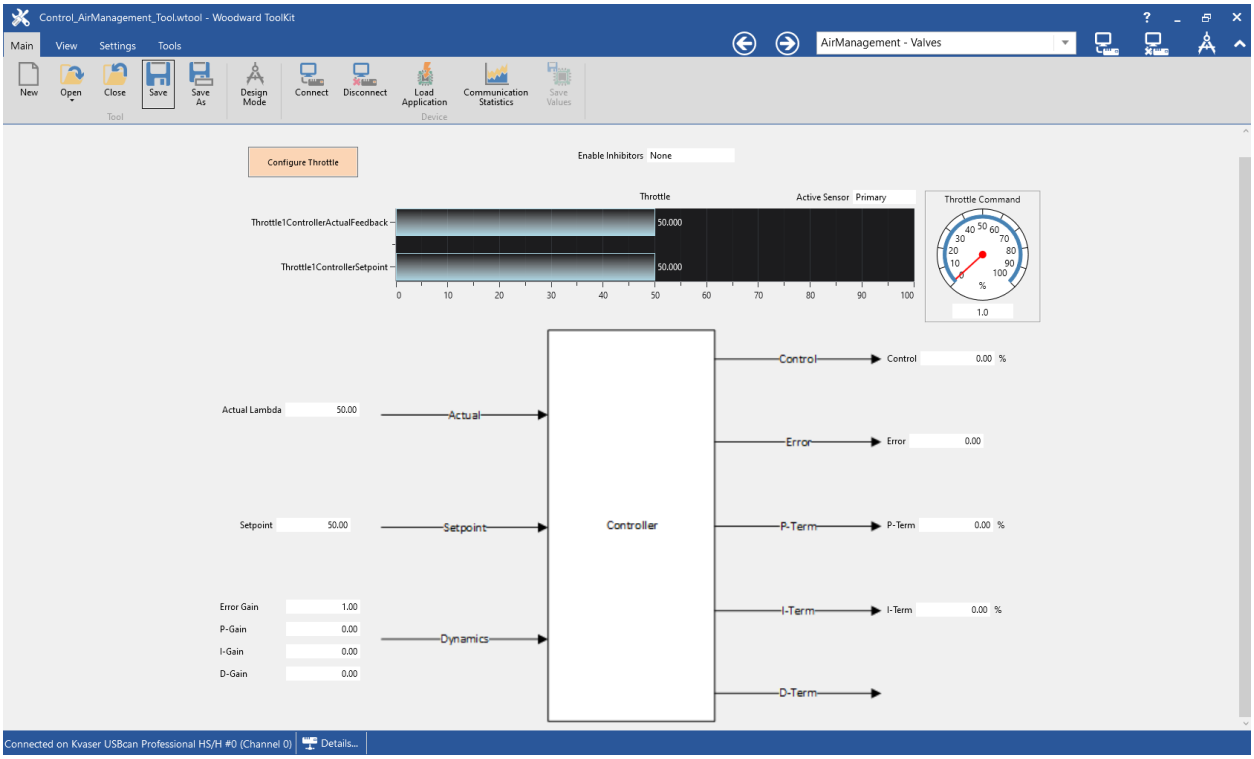
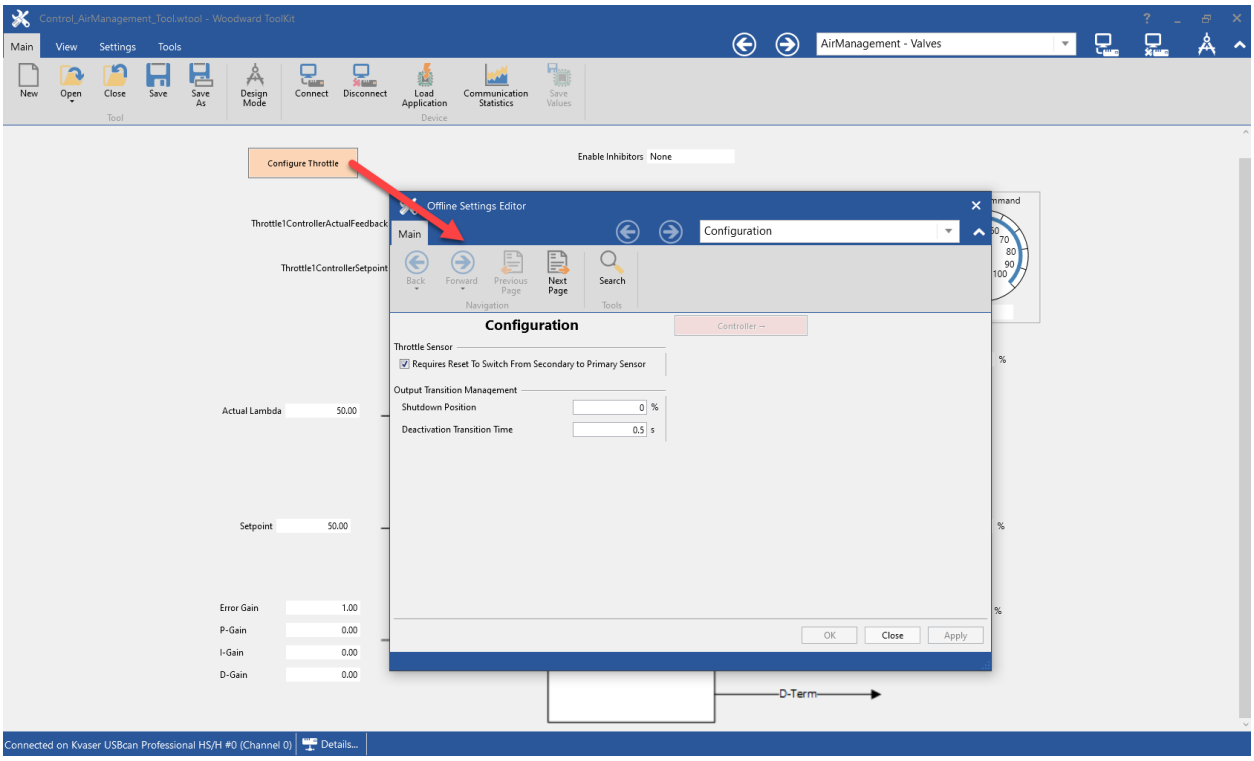


Figure 7-4. SID File Selection Control – Air Management Logic

Below are a few of the Toolkit pages that contain most of the defined parameters in the example model. Of course, the real sensors and data must be attached to complete it for the application that is being worked on. For that reason, red X's appear on some pages.


Control_AirManagement_Tool.wtool - Woodward ToolKit
Main
View
Settings
Tools
AirManagement - Valves
New
Open
Close
Save
Save As
Design Mode
Connect
Disconnect
Load Application
Communication Statistics
Save Values
Configure Throttle
Enable Inhibitors None
Throttle
Active Sensor Primary
Throttle1ControllerActualFeedback
Throttle1ControllerSetpoint
50.000
Throttle Command
Actual Lambda 50.00
Actual
Setpoint 50.00
Setpoint
Controller
Error Gain 1.00
P-Gain 0.00
I-Gain 0.00
D-Gain 0.00
Dynamics
Control
Control 0.00 %
Error
Error 0.00
P-Term
P-Term 0.00 %
I-Term
I-Term 0.00 %
D-Term
Connected on Kvaser USBcan Professional HS/H #0 (Channel 0)
Details...


Control_AirManagement_Tool.wtool - Woodward ToolKit
Main
View
Settings
Tools
AirManagement - Valves
Configure Throttle
Enable Inhibitors None
Offline Settings Editor
Configuration
Back
Forward
Previous Page
Next Page
Search
Navigation
Tools
Configuration
Throttle Sensor
Requires Reset To Switch From Secondary to Primary Sensor
Output Transition Management
Shutdown Position 0 %
Deactivation Transition Time 0.5 s
OK
Close
Apply
Throttle1ControllerActualFeedback
Throttle1ControllerSetpoint
Actual Lambda 50.00
Setpoint 50.00
Error Gain 1.00
P-Gain 0.00
I-Gain 0.00
D-Gain 0.00
D-Term
Connected on Kvaser USBcan Professional HS/H #0 (Channel 0)
Details...

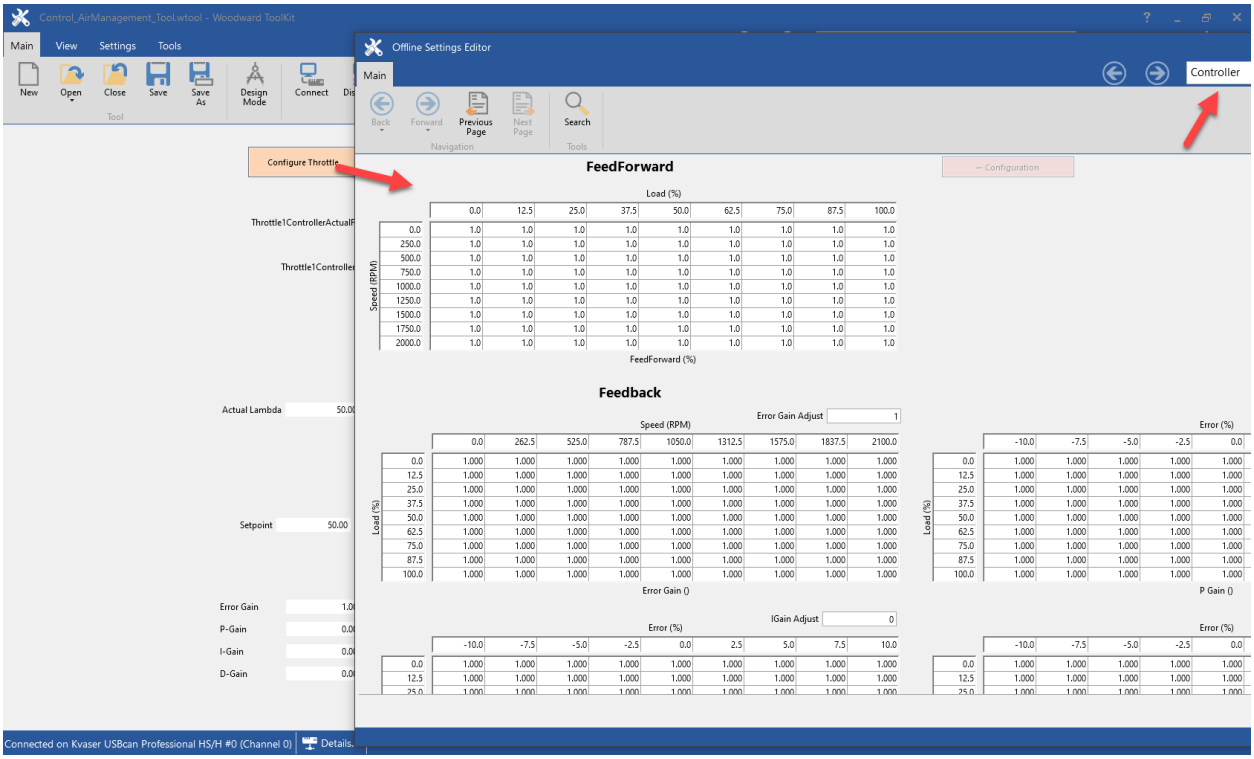


Figure 7-5. Examples of the Toolkit Pages

Functional Description – Throttle Control

The following describes the logic that has been setup for the Throttle and/or Bypass, and/or Wastegate Valves. The Throttle/Bypass/Wastegate are electronically controlled valves which can be used to setup the air fuel ratio (AFR) and influence starting behavior.

The AFR is normally set by means of feedforward tables, where the throttle can be additionally controlled in close loop control with UEGO sensor feedback. The following chapters will provide more details on each of the valves, using Woodward valves as an example.

F-Series Valve

Figure 7-6. Throttle Valve – F-Series ITB

Additional details on the F-series can be found in manual B26355.
The Bypass Valve Logic can be altered from the Toolkit page as shown below:

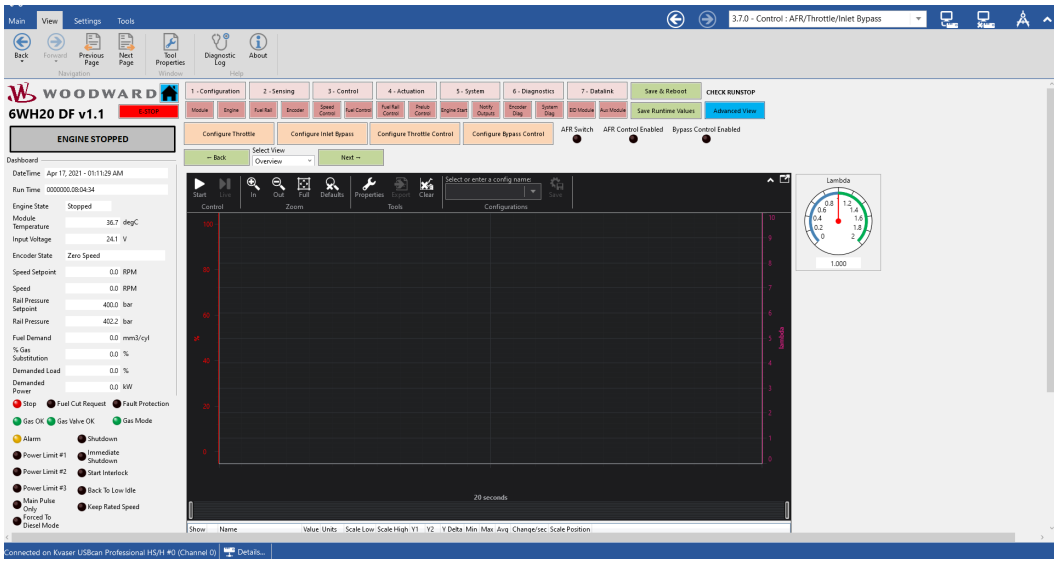


Figure 7-7. Toolkit Throttle/Bypass/Wastegate Valve Configuration Page

The offline configuration page for the Throttle/Bypass/Wastegate valve is used to setup the LECM control part for the command signal.

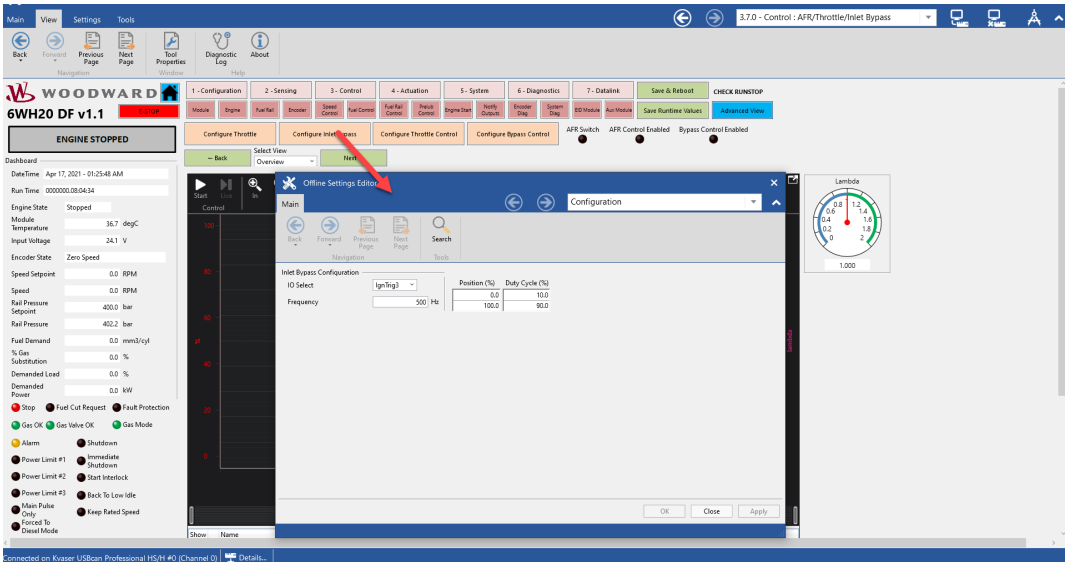


Figure 7-8. Throttle/Bypass/Wastegate Valve Configuration Setup

The position feedback of the F-Series Inlet Bypass valve can be set up from the Sensing Toolkit Page, selecting Throttle. The Toolkit screenshot below indicates the buttons to click.

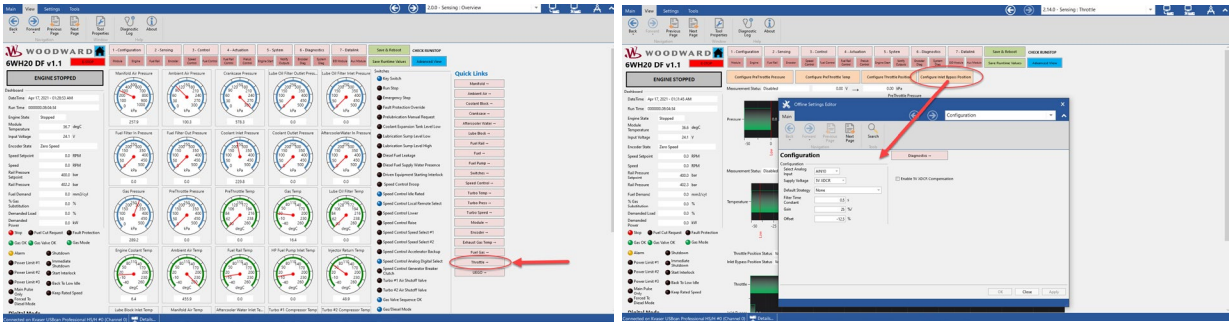


Figure 7-9. Bypass Inlet Valve Feedback Setup

When the Command and Feedback Signals are setup, additional diagnostics can be used to setup an alarm or special fault actions when a fault/mismatch is detected for this device.

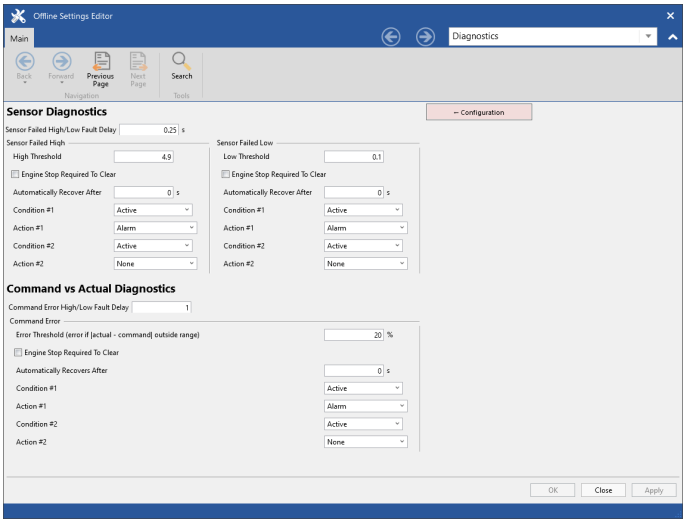


Figure 7-10. Throttle/Bypass/Wastegate Inlet Valve Diagnostics Setup

The position setup of the Throttle/Bypass/Wastegate valve during engine operation can be set from a table that is load and speed dependent. The opening position of the valve must be set during commissioning and fine-tuned over the whole engine operating range.

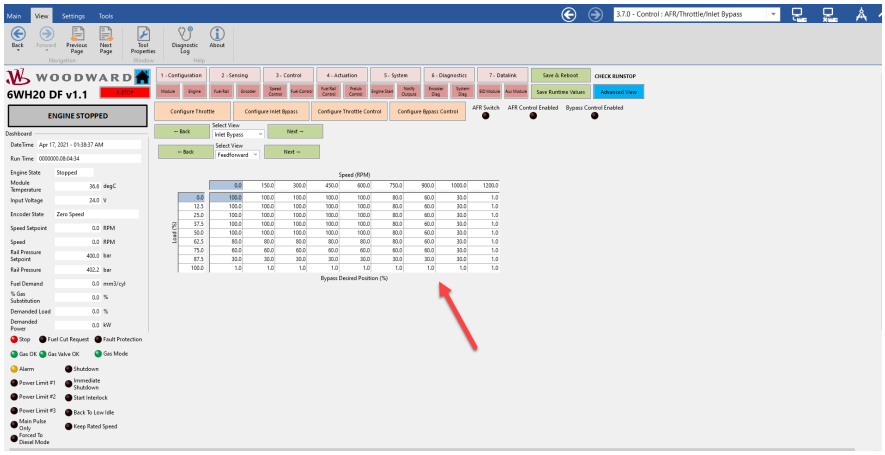


Figure 7-11. Bypass Inlet Valve Position Setup

Some diagnostics from the F-Series can also be monitored and viewed on the same page by selecting Diagnostics in the Select View drop down menu.

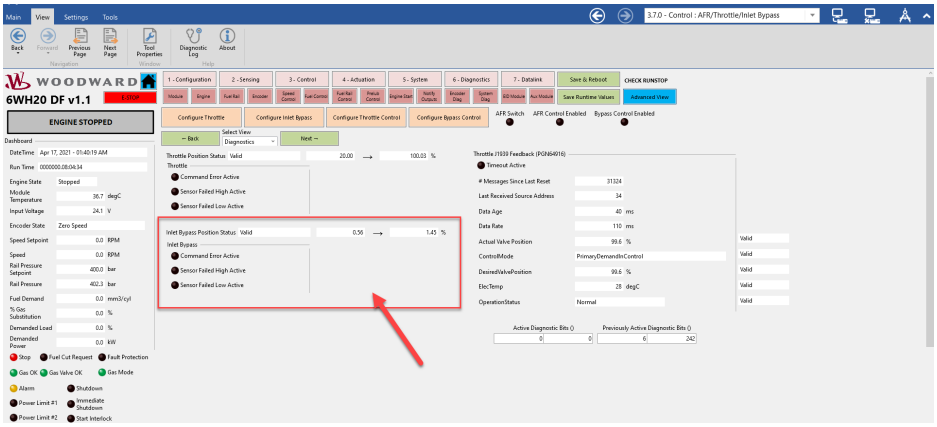


Figure 7-12. Throttle/Bypass/Wastegate Valve Diagnostics Monitor

The F-Series actuator has its own software tool (5418-2745) if more details are required during troubleshooting. Also, this tool is Toolkit-based and a special breakout cable is required to connect the unit to the PC via serial connection.

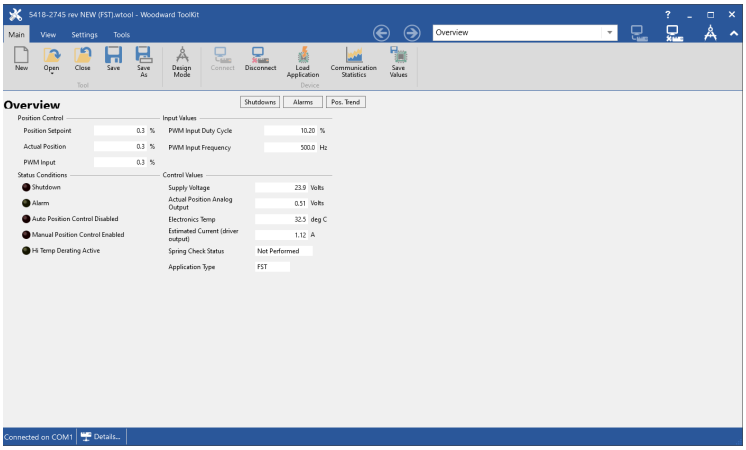


Figure 7-13. F-Series Actuator Tool

R-Series Valve

An R-Series actuator can also be used to control the Throttle/Bypass/Wastegate valve.

Figure 7-14. Throttle/Bypass/Wastegate Valve - R-Series Actuator

More details on the R-series can be found in manual B26845.

The Throttle/Bypass/Wastegate Valve Logic can be altered from the Toolkit page as shown in the example below.

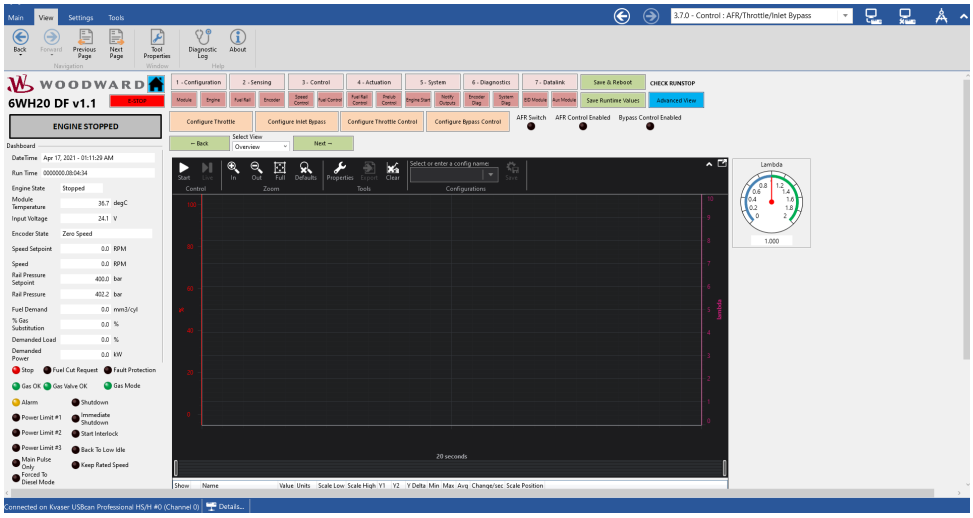


Figure 7-15. Toolkit Throttle/Bypass/Wastegate Valve Configuration Page

The offline configuration page for the Throttle/Bypass/Wastegate valve is used to setup the control part for the command signal.

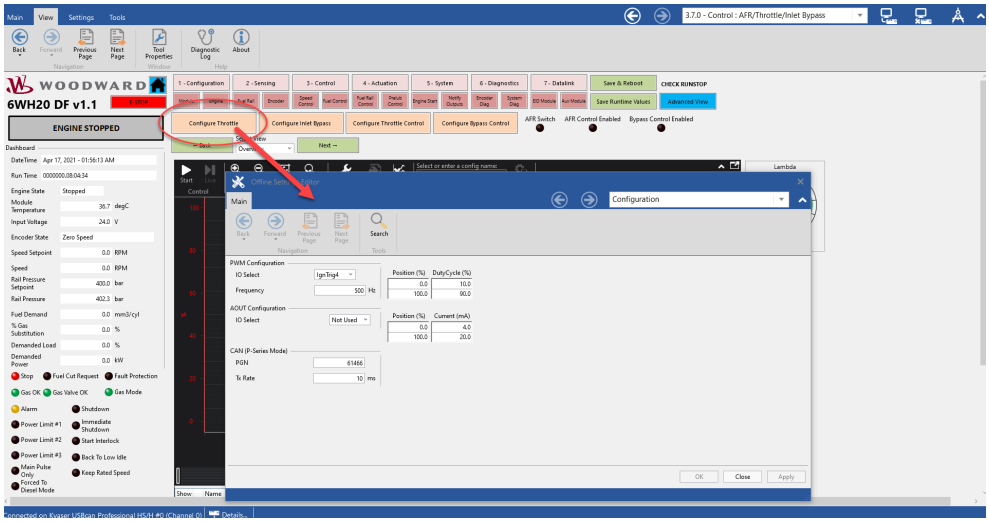


Figure 7-16. Throttle/Bypass/Wastegate Valve Configuration Setup

The position feedback for the R-Series Throttle valve can be set up from the Sensing Toolkit Page, selecting Throttle. The Toolkit screenshot below indicates which buttons to click.

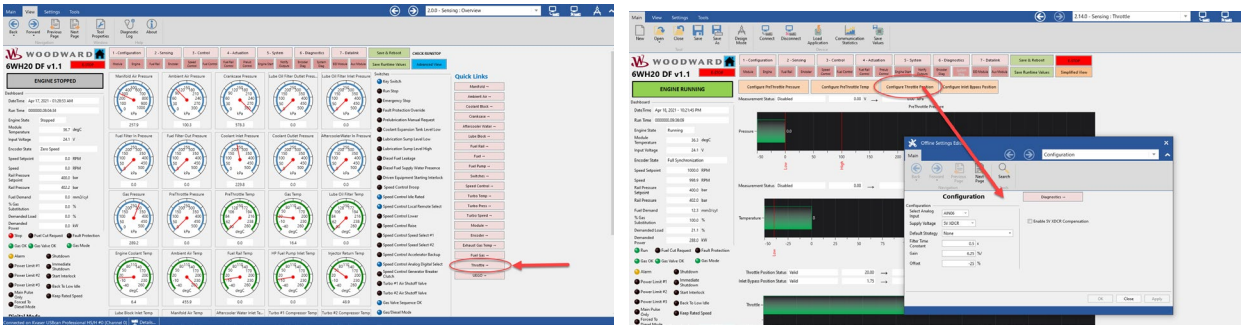


Figure 7-17. Throttle Valve Feedback Setup

Next to the Analog Command Signal and Feedback Signal from R-Series, it is also possible to connect the CAN link of the R-Series (CAN1) to the CAN1 of the LECM main board. In Toolkit, this setup can be configured by going to the page mentioned below.

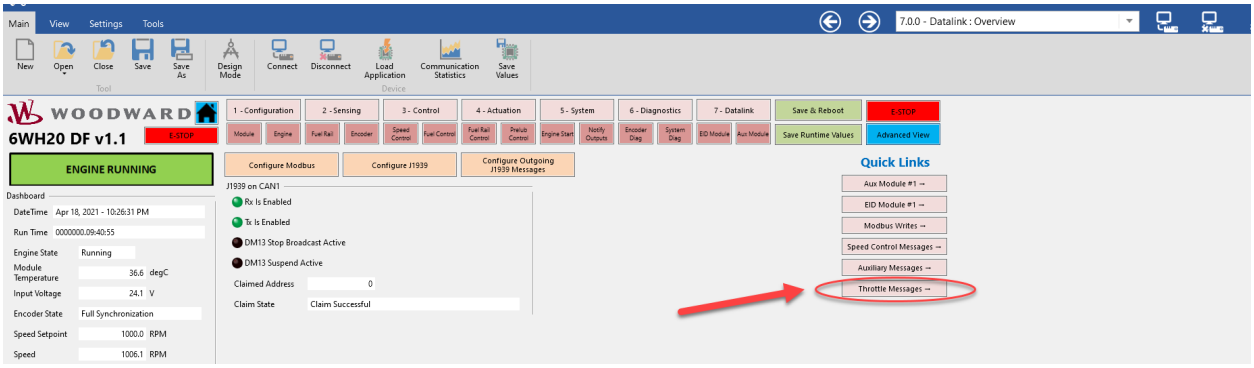


Figure 7-18. Throttle Valve Datalink

Clicking the “Throttle Messages → ” page will open another page that lets you define the required command and/or feedback messages for J1939, even as each of the signal’s timeout.
The Throttle Feedback Signal (PGN64916) provides more diagnostic info from the R-Series Throttle valve.

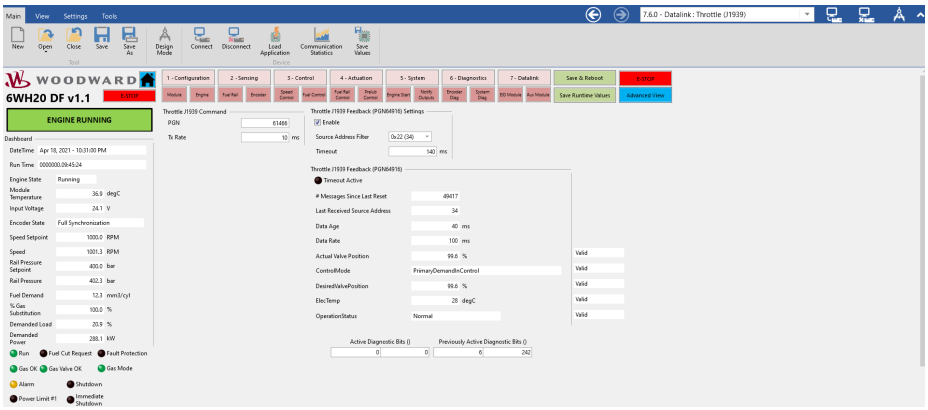


Figure 7-19. Throttle Valve J1939 Command/Feedback and Additional Info

When Command and Feedback Signals are set up, additional diagnostics can be used to setup Alarm or special Fault Actions when a fault/mismatch is detected for this device.

When CAN and analog signals are used, the switch between both (primary and secondary sources) is done automatically.

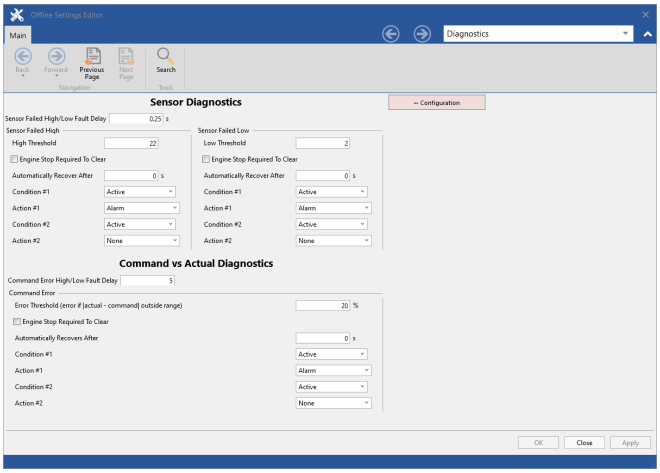


Figure 7-20. Throttle/Bypass/Wastegate Valve Diagnostics Setup

The position setup of the Throttle/Bypass/Wastegate valve during engine operation can be set from a table that is load and speed dependent. The opening position of the valve must be set during commissioning and fine-tuned over the whole engine operating range.

The control for throttle consists of two pieces: Feedforward Table and Feedback PID Control.

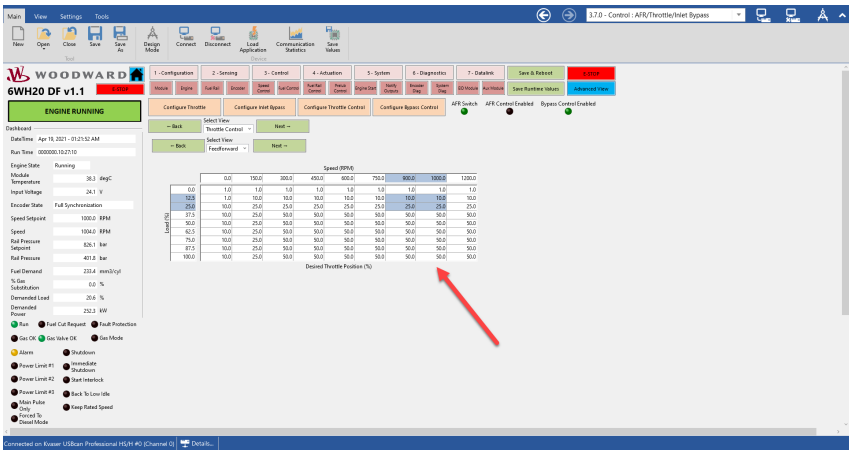


Figure 7-21. Throttle/Bypass/Wastegate Valve Feedforward Table Setup

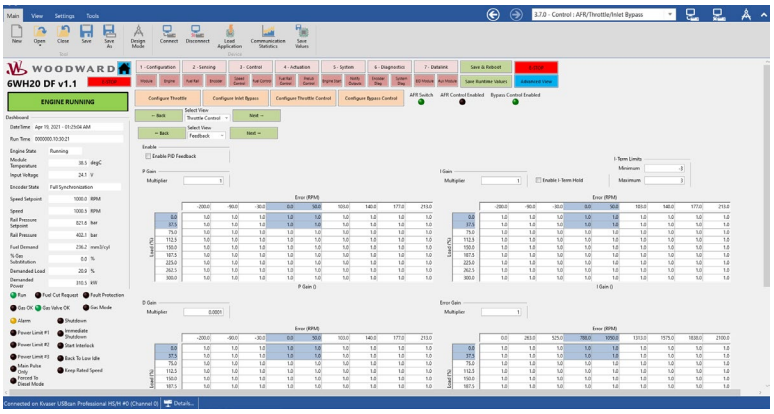


Figure 7-22. Throttle/Bypass/Wastegate Valve Feedback PID Setup

The Feedback PID setup controls the Throttle/Bypass/Wastegate valve based on the UEGO Sensor Feedback signal. The Overview page of the Throttle Control Valve shows a table where lambda desired can be set, based on speed and load.

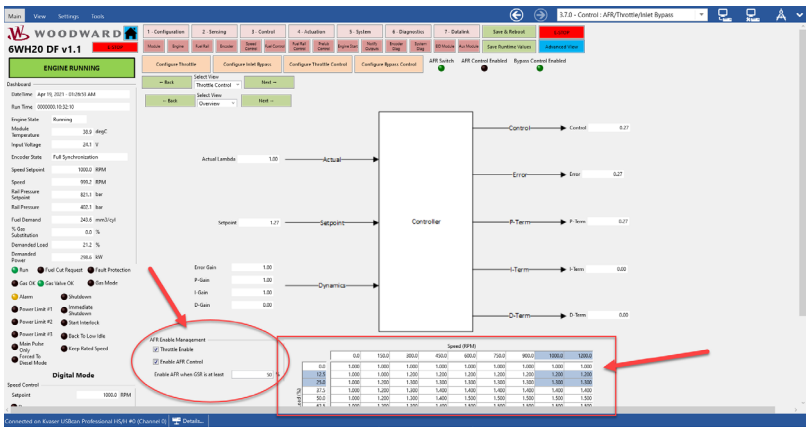


Figure 7-23. Throttle Valve Desired AFR Setup

The AFR close loop can be enabled/disabled with local parameters in Toolkit, but also by means of a discrete input (AFR Enable). This allows flexibility for when AFR control is required and when it is not.

The R-Series actuator has its own software tool (9927-2264) if more details are required during troubleshooting. This tool is Toolkit-based and a Kvaser CAN to USB cable is required to connect the unit to the PC.

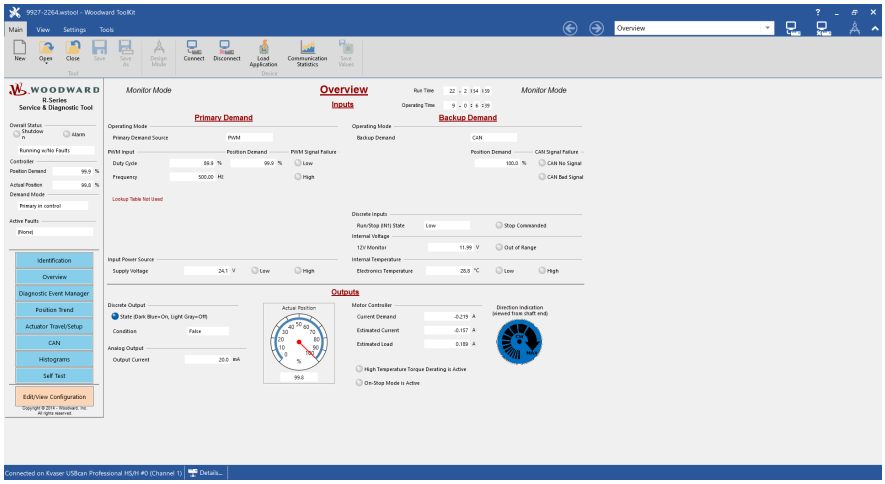


Figure 7-24. Throttle/Bypass/Wastegate Valve Service Tool

Chapter 8.
Functions – Engine State

Introduction

The ‘Functions – Engine State’ blockset is used to define the Engine State function so it can be used inside the application where used. The Engine State Default Strategy will provide State Flags, which will be useful in other parts of the application. Inside the blockset, two blocks are available as shown in the figure below.

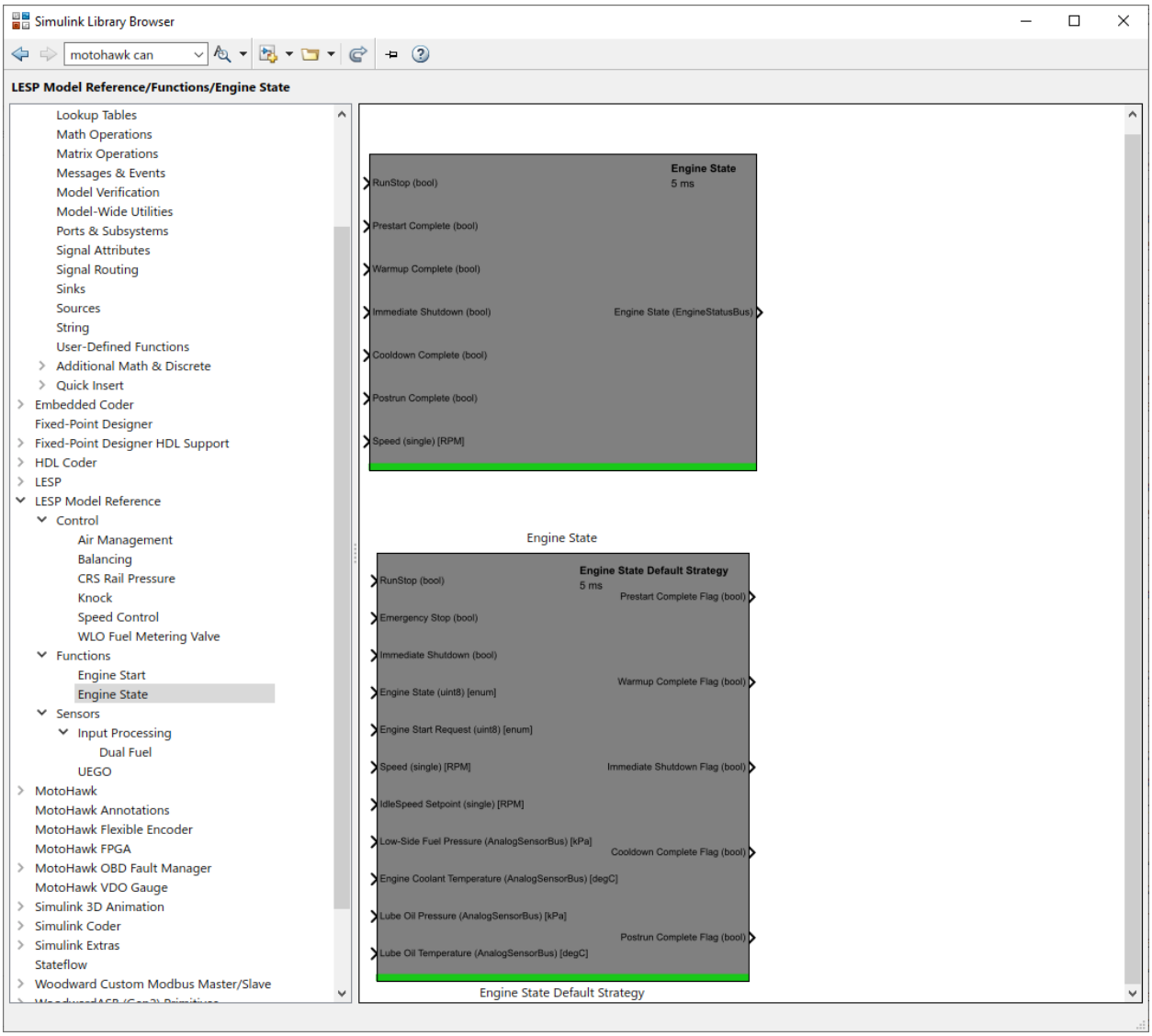


Figure 8-1. LESP Model Reference ‘Functions – Engine State’ Blocks

- Engine State Default Strategy
  This is the default strategy that is used in the standard WWD LESP software. It has several thresholds to transfer between states, which can be calibrated once this block is included in the application.

- Engine State
  If a ‘self-defined’ Engine State logic is preferred, it can be fed though this block to map it to the

EngineStatusBus that is used through more LESP-based blocksets. The ‘self-made’ logic for state definitions will then be merged into the EngineStatusBus output field.

Engine State Default Strategy Block Interface

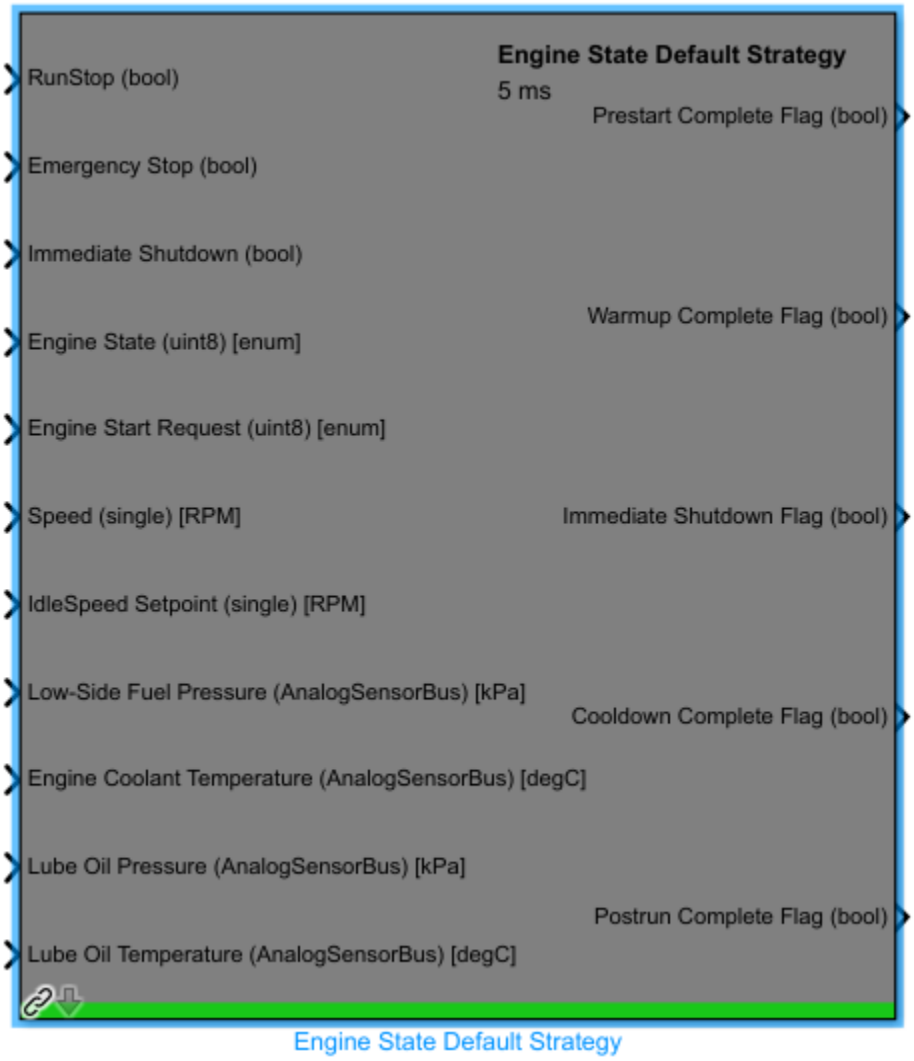


Figure 8-2. LESP Model Reference ‘Functions – Engine State’ Block

Table 8-1. ‘Speed Control Setpoints’ Block

Port	DataType	Description
[In] RunStop	boolean	Input that requires the RunStop status feedback in the application. Run – Engine is ready to run and SpeedControl will be active. Stop – Engine will shut down and go into Stop state.
[In] Emergency Stop	boolean	This input requires the Emergency Stop Input signal as defined in the application. Emergency Stop is a direct shutdown of the fuel to force engine stop.
[In] Immediate Shutdown	boolean	This input requires the Immediate Shutdown Input as defined in the application.

		Immediate Shutdown means the engine will be stopped instantly, not following any cooldown action or similar sequence.
[In] Engine State [enum]	uint8	The actual engine state the engine is operating in at that particular moment. These states (lesp_engine_state_enum) are defined as below: 0 – Stopped 1 – Prestart 2 – Starting 3 – Warmup 4 – Running 5 – Cooldown 6 – Stopping 7 – Postrun
[In] Engine Start Request	boolean	This input requires the actual Engine Start Request input as defined in the application.
[In] Speed [RPM]	single	This input requires the actual Speed Signal as defined in the application.
[In] IdleSpeed Setpoint [RPM]	single	This input requires the actual Idle Speed Setpoint as defined in the application.
[In] Low-Side Fuel Pressure [kPa]	AnalogSensorBus	This input requires the actual Low-Side Fuel Pressure as defined in the application. Bus Details: AnalogSensorBus: - FilteredValue - MeasurementStatus_enum MeasurementStatus_enum: (lesp_measurement_status_enum) 0 – Valid 1 – Disabled 2 – SignalLow 3 – SignalHigh 4 – SensorSupplyFailed 5 – ReferenceSourceFailed 6 – OpenCircuit
[In] Engine Coolant Temperature [degC]	AnalogSensorBus	This input requires the actual Engine Coolant Temperature as defined in the application. Bus Details: AnalogSensorBus: - FilteredValue - MeasurementStatus_enum MeasurementStatus_enum: (lesp_measurement_status_enum) 0 – Valid 1 – Disabled 2 – SignalLow 3 – SignalHigh 4 – SensorSupplyFailed 5 – ReferenceSourceFailed 6 – OpenCircuit
[In] Lube Oil Pressure [kPa]	AnalogSensorBus	This input requires the actual Lube Oil Pressure as defined in the application. Bus Details: AnalogSensorBus: - FilteredValue - MeasurementStatus_enum

		- MeasurementStatus_enum: (lesp_measurement_status_enum) 0 – Valid 1 – Disabled 2 – SignalLow 3 – SignalHigh 4 – SensorSupplyFailed 5 – ReferenceSourceFailed 6 – OpenCircuit
[In] Lube Oil Temperature [degC]	AnalogSensorBus	This input requires the actual Lube Oil Temperature as defined in the application. Bus Details: AnalogSensorBus: - FilteredValue - MeasurementStatus_enum MeasurementStatus_enum: (lesp_measurement_status_enum) 0 – Valid 1 – Disabled 2 – SignalLow 3 – SignalHigh 4 – SensorSupplyFailed 5 – ReferenceSourceFailed 6 – OpenCircuit
[Out] PreStart Complete Flag	boolean	This output provides Block feedback to indicate if the PreStart Engine State is complete or not.
[Out] Warmup Complete Flag	boolean	This output provides Block feedback to indicate if the WarmUp State is complete or not.
[Out] Immediate Shutdown Flag	boolean	This output provides Block feedback to indicate if an Immediate Shutdown is active or not.
[Out] Cooldown Complete Flag	boolean	This output provides Block feedback to indicate if the CoolDown Engine State is complete or not.
[Out] Postrun Complete Flag	boolean	This output provides Block feedback to indicate if the PostRun Engine State is complete or not.

Engine State Block Interface

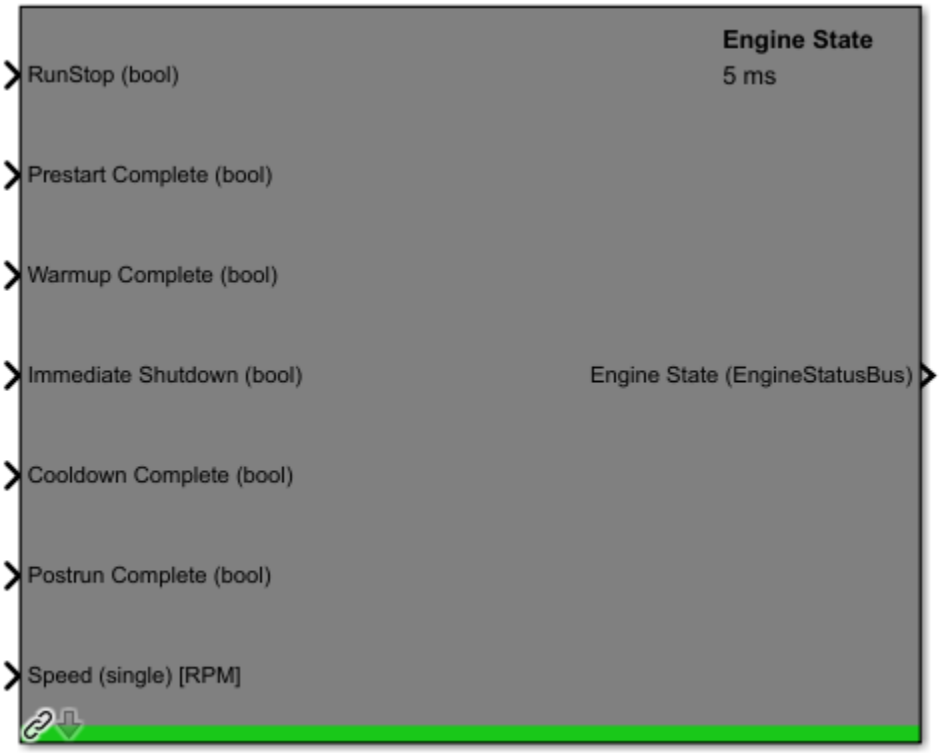


Figure 8-3. LESP Model Reference 'Functions – Engine State' Block

Table 8-2. 'Speed Control Setpoints' Block

Port	DataType	Description
[In] RunStop	boolean	Input that requires the RunStop status feedback in the application. Run – Engine is ready to run and SpeedControl will be active. Stop – Engine will shut down and go into Stop state.
[In] Prestart Complete	boolean	This input provides feedback to the Block to indicate if the PreStart Engine State is complete or not. It can be fed with a signal from the application's own created logic or from the 'Engine State Default Strategy' Block.
[In] Warmup Complete	Boolean	This input provides feedback to the Block to indicate if the WarmUp Engine State is complete or not. It can be fed with a signal from the application's own created logic or from the 'Engine State Default Strategy' Block.
[In] Immediate Shutdown	Boolean	This input requires the Immediate Shutdown Input as defined in the application. Immediate Shutdown means that the engine will be stopped instantly, not following any cooldown action or similar sequence.

[In] Cooldown Complete	Boolean	This input provides feedback to the Block to indicate if the CoolDown Engine State is complete or not. It can be fed with a signal from the application's own created logic or from the 'Engine State Default Strategy' Block.
[In] Postrun Complete	Boolean	This input provides feedback to the Block to indicate if the PostRun Engine State is complete or not. It can be fed with a signal from the application's own created logic or from the 'Engine State Default Strategy' Block.
[In] Speed [RPM]	Single	This input requires the actual Speed Signal as defined in the application.
[Out] Engine State	EngineStatusBus	The actual engine state that the engine is operating in at that particular moment. These states (lesp_engine_state_enum) are defined as below: 0 – Stopped 1 – Prestart 2 – Starting 3 – Warmup 4 – Running 5 – Cooldown 6 – Stopping 7 – Postrun

Example Model Using the Blocks

Below is a very simple setup using MotoHawk Calibration and MotoHawk Probes blocks to make the model suitable for building.

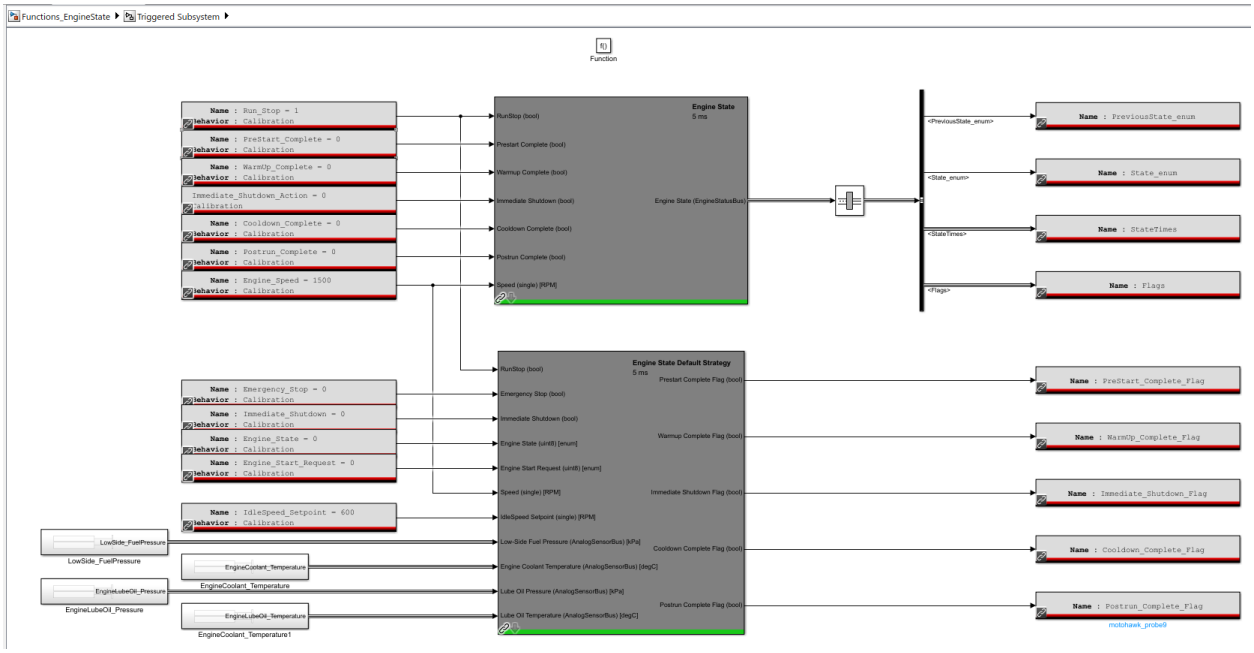


Figure 8-4. Simple Example Model, Using the Functions – Engine State Blocks

When compiling, it will generate the required files and if the Toolkit required blocks are also added to the application model, it will generate the required SID files for the Toolkit HMI tool creation.

Default ToolKit Pages

Opening a new Toolkit application and selecting the correctly generated .sid file will look like the below screenshot.

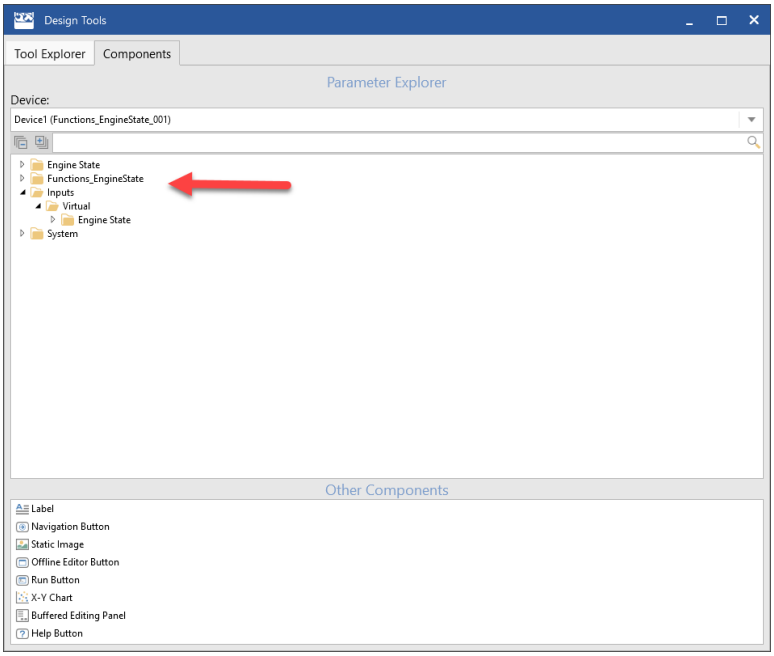
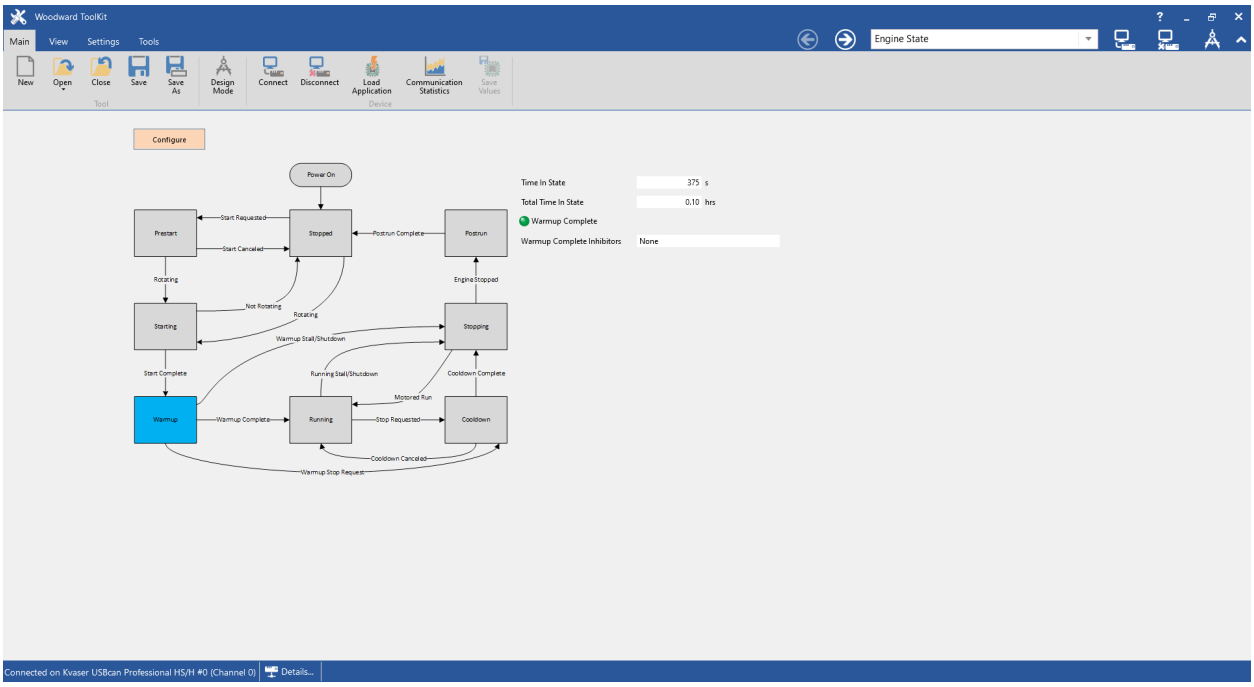


Figure 8-5. SID File Selection Functions – Engine States Logic

Below are a few of the Toolkit pages that will contain most of the defined parameters in the example model. Of course, the real sensors and data must be attached to complete it for the application that is being worked on. For that reason, red X's appear on some pages.



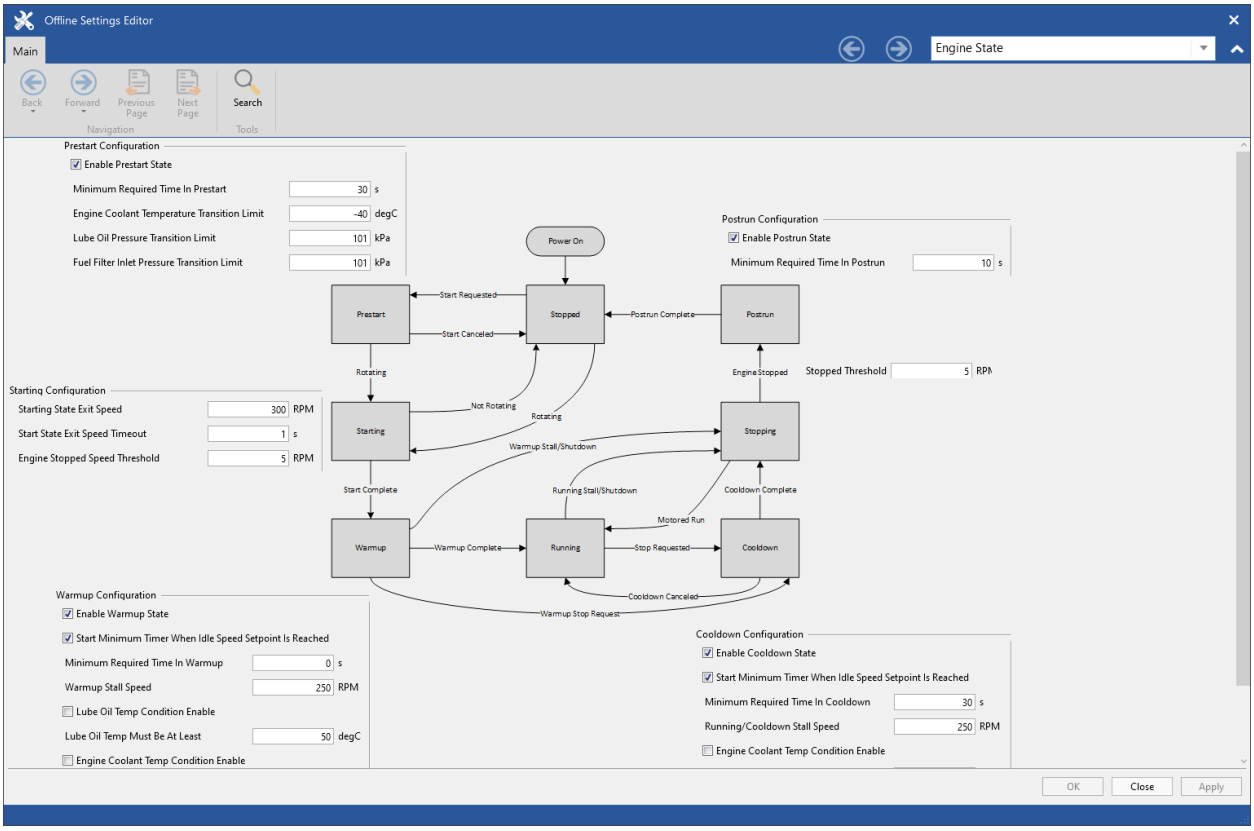


Figure 8-6. Examples of the Functions - Engine State Toolkit Pages

Functional Description - Engine State

The following describes the LECM CRS engine state and corresponding configurations. Although the engine states may remain constant across application variants, the sequencing/conditions/actions taken for each state may vary. The software supports 8 engine states (4 mandatory, 4 opt-out states) as illustrated in the figure below. The engine state is used to drive specific state-based control strategies and diagnostic behaviors.

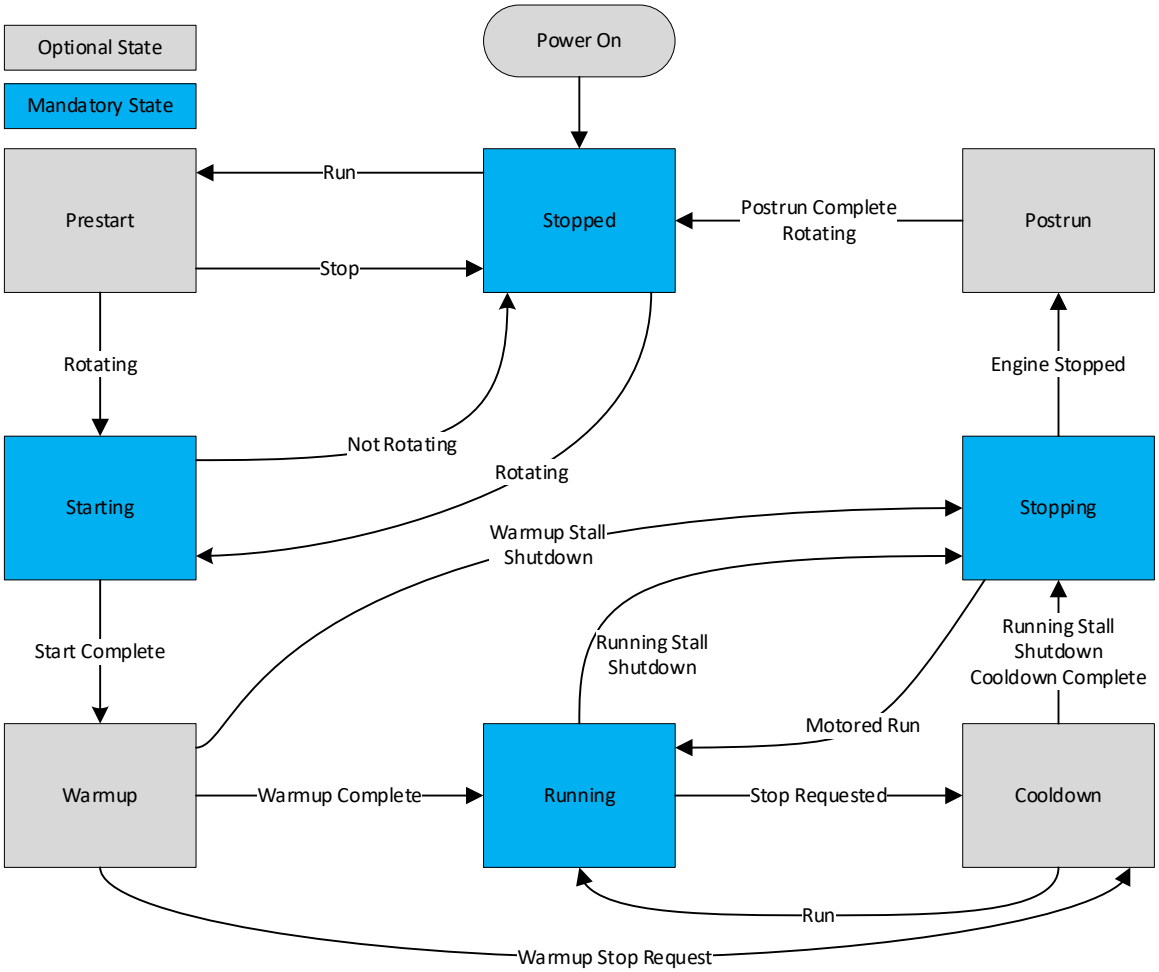


Figure 8-7. Engine State Overview

States

- **Stopped**
  - The stopped state signifies the engine in a stopped state and when a stop request is active before an engine start.
- **Prestart (optional)**
  - The prestart state is used to enable systems required to run before engine start and/or perform conditional checks before allowing the engine to start.
- **Starting**
  - The starting state reflects the engine rotating or actively engaged with the starter; speed is generally below idle speed. The starting state consists of a starting exit speed and timeout which will transition to the warmup state once speed is at or above the exit speed for the timeout period. A speed threshold for when to transition back to stopped is also provided.

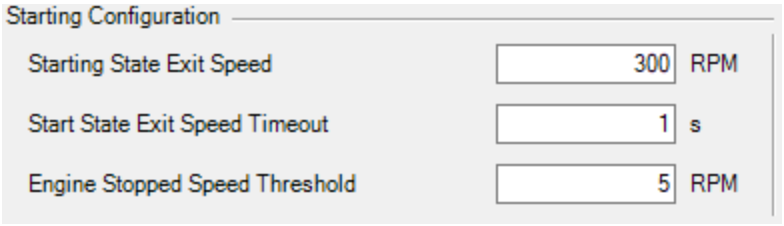


Figure 8-8. Engine Starting State Exit Conditions

- **Warmup (optional)**
  - The warmup state is equivalent to an engine run state; however, can it drive specific application behaviors to allow for different operational commands while the engine is warming up (e.g. stay at idle, single injection event, etc...).
- **Running**
  - The running state signifies the engine is operating in run mode.
- **Cooldown (optional)**
  - The cooldown state is equivalent to an engine run state; however, it can drive specific application behaviors to allow for different operational commands while the engine is cooling down (e.g. slower speed, single injection event, etc...) and is typically entered when a normal shutdown is requested.
- **Stopping**
  - The stopping state signifies all fueling commands are disabled and the engine is currently "rolling down," or stopping.
- **Postrun (optional)**
  - The postrun state is used to enable systems required to run after an engine stop (e.g. reduce rail pressure).

Default Strategies

Default Prestart Strategy

The standard software supports an optional prestart strategy with the following conditions that must be met before the prestart complete flag is asserted, allowing for an engine start request to commence:

1. Minimum time required to remain in prestart state.
2. Engine coolant temperature is at or above a threshold.
3. Lube oil pressure is at or above a threshold.
4. Fuel filter inlet pressure is at or above a threshold.

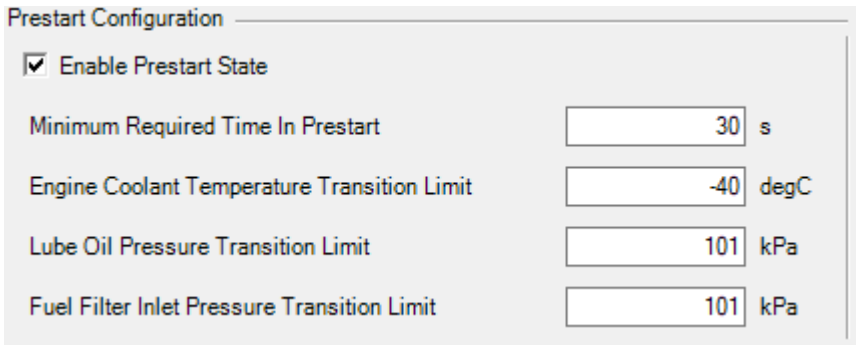


Figure 8-9. Prestart Configurations

Default Warmup Strategy

The standard software supports an optional warmup strategy that allows for customized fueling commands and holds the idle speed setpoint while the engine is in the warmup state. The following conditions that must be met before the warmup complete flag is asserted, allowing for normal engine run operational state to commence:

1. Minimum time required to remain in warmup state with an option to start the minimum time once the idle speed setpoint has been reached.
2. Lube oil temperature is at or above a threshold.
3. Engine coolant temperature is at or above a threshold.

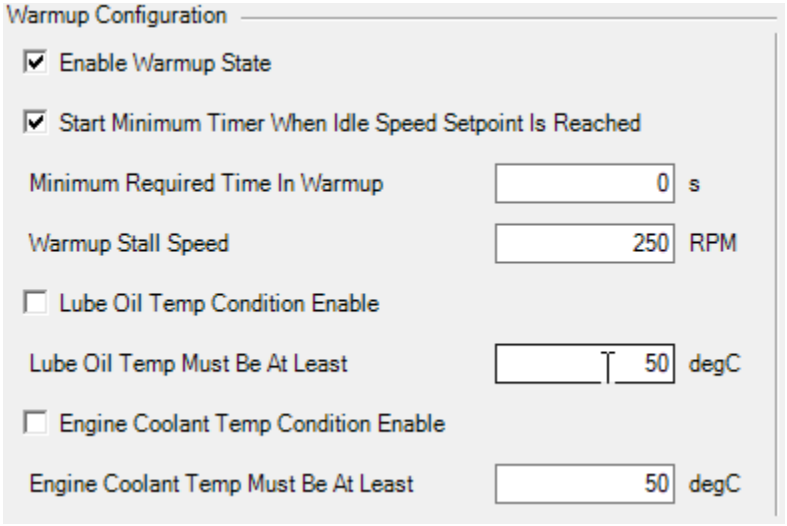


Figure 8-10. Warmup Configurations

The state also supports a stall speed that will transition to the Stopping state if speed drops below the stall threshold while in the Warmup state.

Default Cooldown Strategy

The standard software supports an optional cooldown strategy that allows for customized fueling commands and holds the idle speed setpoint while the engine is in the cooldown state. The following conditions that must be met before the cooldown complete flag is asserted, allowing for normal engine stop to commence:

1. Minimum time required to remain in cooldown state with an option to start the minimum time once the idle speed setpoint has been reached.
2. Lube Oil Temperature is at or below a threshold.
3. Engine Coolant Temperature is at or below a threshold.

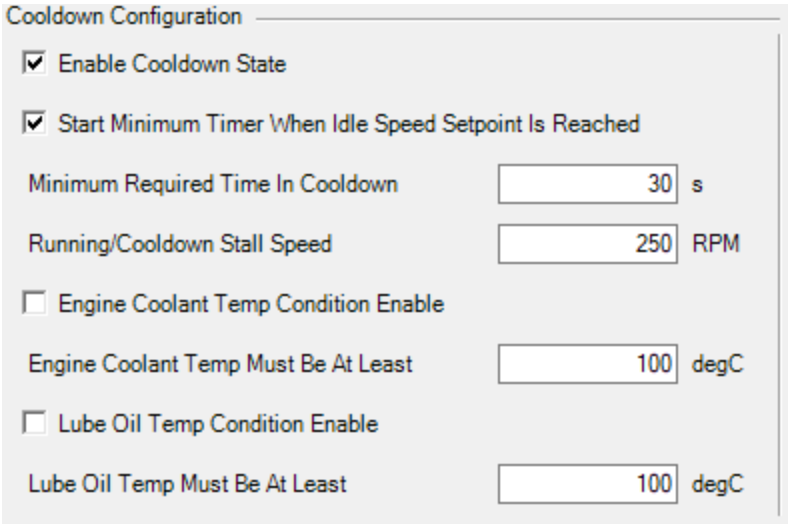


Figure 8-11. Cooldown Configurations

The state also supports a stall speed that will transition to the Stopping state if speed drops below the stall threshold while in the Cooldown state.

Default Postrun Strategy

The standard software supports an optional postrun strategy with the following conditions that must be met before the postrun complete flag is asserted, allowing the engine state to enter the Stopped state:

1. Minimum time required to remain in postrun state.

Figure 8-12. Postrun Configurations

Chapter 9.
Functions – Engine Start

Introduction

The ‘Functions – Engine Start’ blockset can be used for controlling engine Start and/or Engine Start Request. This includes starter motor control, automatic re-start, and cooldown logic for the starter motor. Inside the blockset, two blocks are available as shown in the screenshot below.

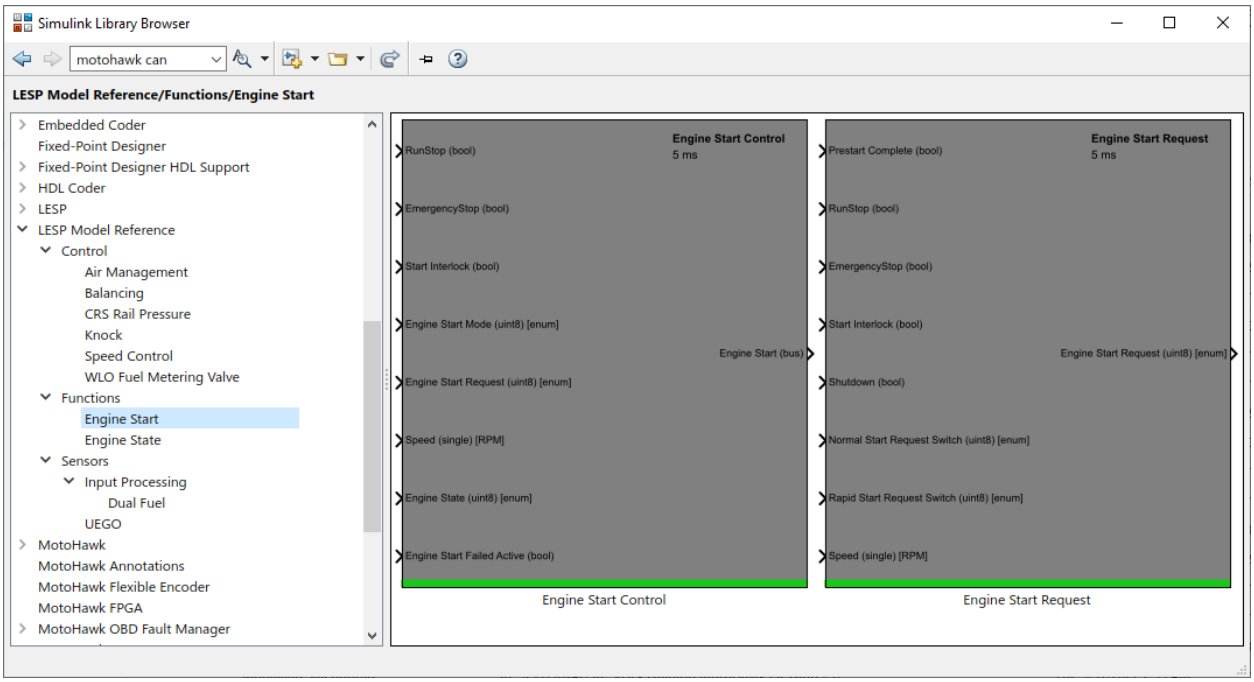


Figure 9-1. LESP Model Reference ‘Functions – Engine Start’ Blocks

- Engine Start Control
  This block can be used to define the Engine Start Control logic inside the application.

- Engine Start Request
  This block can be used to define the Engine Start Request.

Engine Start Control Block Interface

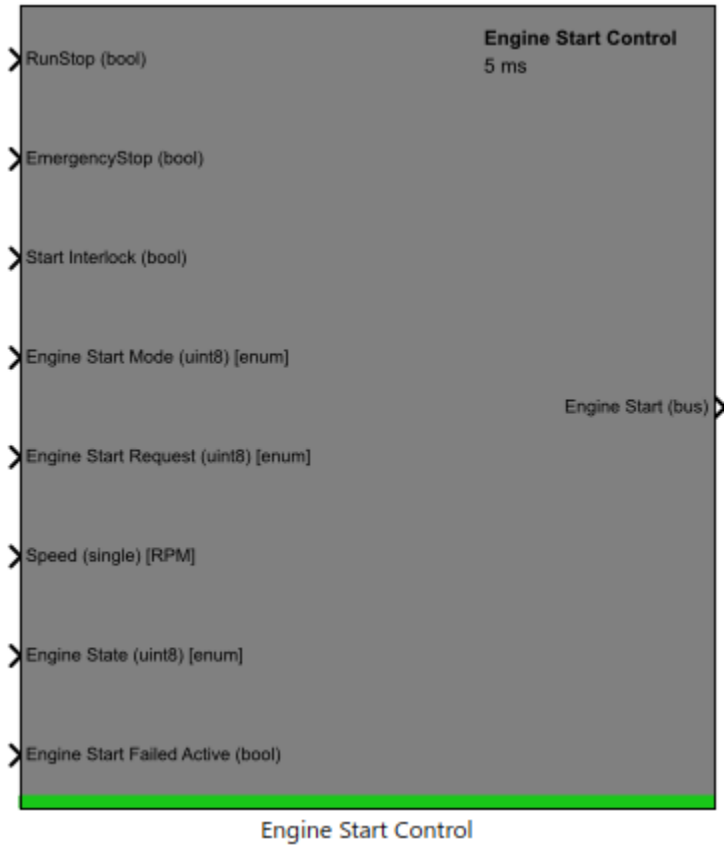


Figure 9-2. LESP Model Reference ‘Functions – Engine Start Control’ Block

Table 9-1. ‘Speed Control Setpoints’ Block

Port	DataType	Description
[In] RunStop	boolean	Input that requires the RunStop status feedback in the application. Run – Engine is ready to run and SpeedControl will be active. Stop – Engine will shut down and go into Stop state.
[In] Emergency Stop	boolean	This input requires the Emergency Stop input signal as defined in the application. Emergency Stop is a direct fuel shut down to force engine stop.
[In] Start Interlock	boolean	This input requires the Start Interlock input signal as defined in the application. Start Interlock indicates starting needs to be inhibited. An example of when this is required is if the turning wheel or engine is under maintenance.
[In] Engine Start Mode [enum]	uint8	Input that defines starting in Manual or in Auto Mode. Manual Mode requires manual activation of the starter motor each time after a start attempt. Automatic mode is 1 x starter motor activation, and when start is not successful, it will retry for X amount of times until the engine starts.

		Enum: lesp_engine_start_mode_enum 0 – Manual 1 - Auto
[In] Engine Start Request [enum]	uint8	This input requires the actual Engine Start Request input as defined in the application. It indicates ready or not, normal, or rapid start. enum: lesp_engine_start_request_enum 0 – NotReady 1 – Ready 2 – Normal 3 – Rapid
[In] Speed [RPM]	single	This input requires the actual Speed Signal as defined in the application.
[In] Engine State [enum]	uint8	The actual engine state the engine is operating in at that particular moment. These states (lesp_engine_state_enum) are defined as below: 0 – Stopped 1 – Prestart 2 – Starting 3 – Warmup 4 – Running 5 – Cooldown 6 – Stopping 7 – Postrun
[In] Engine Start Failed Active	boolean	This input requires the feedback if the engine start has failed. TRUE – Failed FALSE – Not Failed
[Out] Engine Start	Bus	Output that defines the Engine Start (bus). The Bus defines: -DiscreteOutput (bus) - IOSelect_enum - BehaviorSelect_enum - State_enum - DutyCycle_Pct - Frequency_Hz - Engine StartState_enum (lesp_engine_start_state_enum) 0 - Waiting For Start Request 1 - Starter Active 2 - Starter Cooldown 3 - Waiting For Reset 4 - Waiting For Stop Command 5 - Engine Running - EngineStartInterlockCause_enum (lesp_engine_start_interlock_cause_enum) 0 - None 1 - Stop Commanded 2 - Emergency Stop Active 3 - Start Interlock Active 4 - Request Not Available

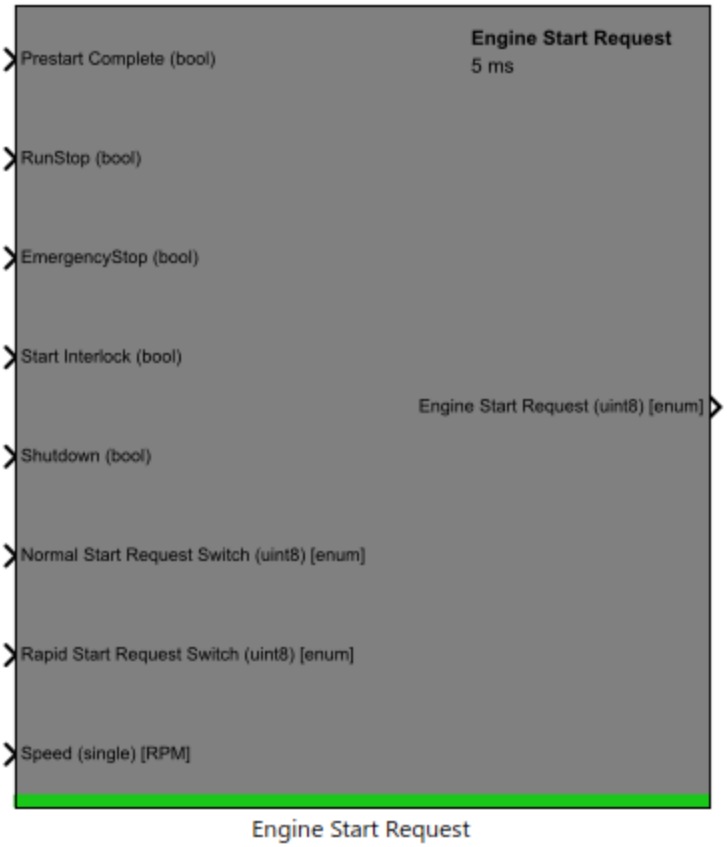


Figure 9-3. LESP Model Reference 'Functions – Engine Start Request' Block

Table 9-2. 'Speed Control Setpoints' Block

Port	DataType	Description
[In] Prestart Complete	boolean	Input that requires the RunStop status feedback in the application. Run – Engine is ready to run and SpeedControl is active. Stop – Engine will shut down and go into Stop state.
[In] RunStop	boolean	Input that requires the RunStop status feedback in the application. Run – Engine is ready to run and SpeedControl will be active. Stop – Engine will shut down and go into Stop state.
[In] Emergency Stop	boolean	This input requires the Emergency Stop input signal as defined in the application. Emergency Stop is a direct fuel shutdown to force engine stop.
[In] Start Interlock	boolean	This input requires the Start Interlock input signal as defined in the application. Start Interlock indicates that starting needs to be inhibited. An example for when this is required is if the turning wheel or engine is under maintenance.
[In] Shutdown	boolean	Input that requires the Shutdown status feedback in the application.

[In] Normal Start Request Switch [enum]	uint8	The Engine Start Request switch supports a switch input that requests a normal engine start when the engine start logic is configured for manual mode. enum: lesp_engine_start_request_enum 0 – Not Ready 1 – Ready 2 – Normal 3 – Rapid
[In] Rapid Start Request Switch [enum]	uint8	The Engine Rapid Start Request switch supports a switch input that requests an engine start (bounded by less constraints) when the engine start logic is configured for manual mode. enum: lesp_engine_start_request_enum 0 – Not Ready 1 – Ready 2 – Normal 3 – Rapid
[In] Speed [RPM]	single	This input requires the actual Speed Signal as defined in the application.
[Out] Engine Start Request [enum]	uint8	Output that defines the Engine Start request. It is used to activate the starter motor. enum: lesp_engine_start_request_enum 0 – Not Ready 1 – Ready 2 – Normal 3 – Rapid

Example Model Using the Blocks

Below is a very simple setup using MotoHawk Calibration and MotoHawk Probes blocks to make the model suitable for building.

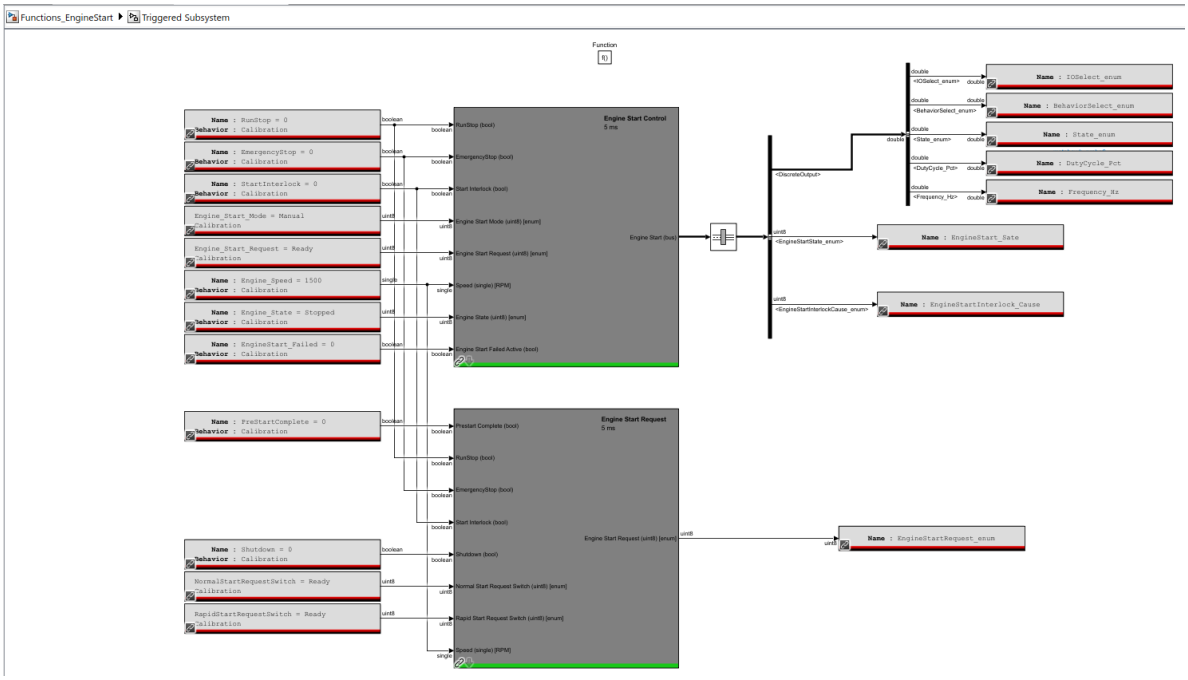


Figure 9-4. Simple Example Model, Using the Functions – Engine Start Blocks

When compiling, it will generate the required files and if the Toolkit required blocks are also added to the application model, it will generate the required SID files for the Toolkit HMI tool creation.

Default ToolKit Pages

Opening a new Toolkit application and selecting the correctly generated .sid file will look like the screenshot below.

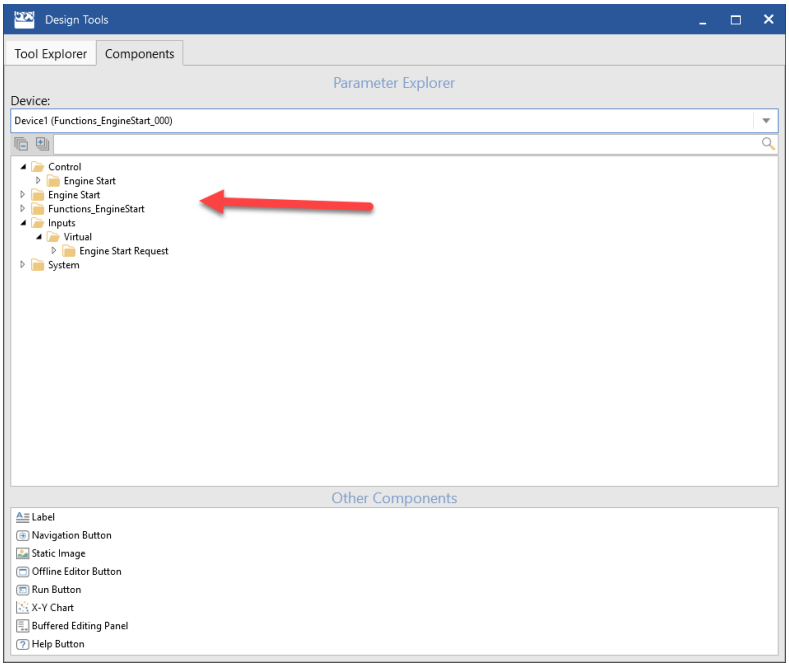
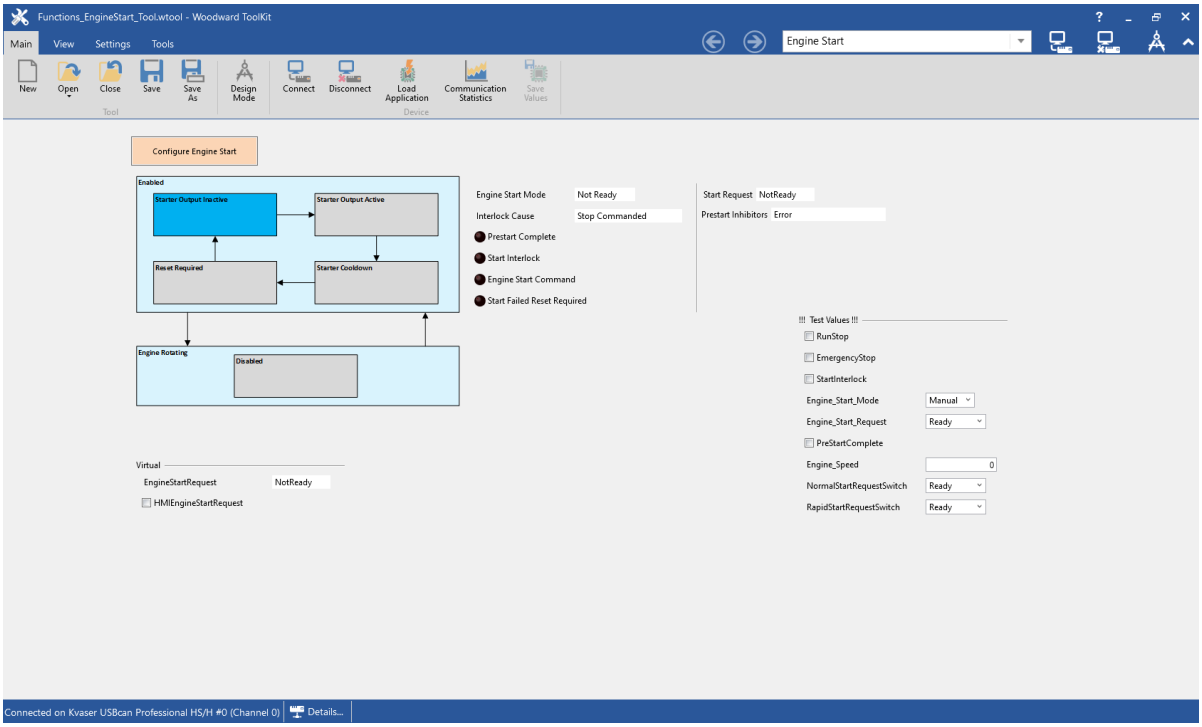
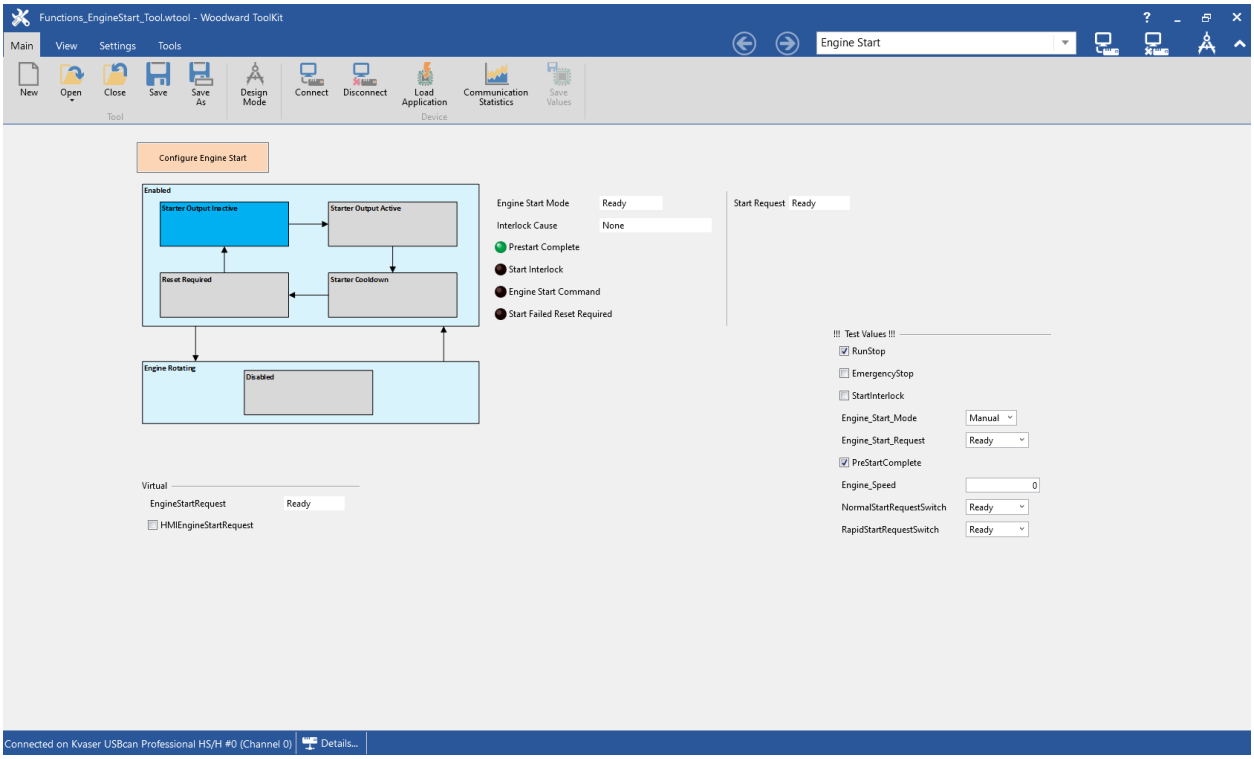
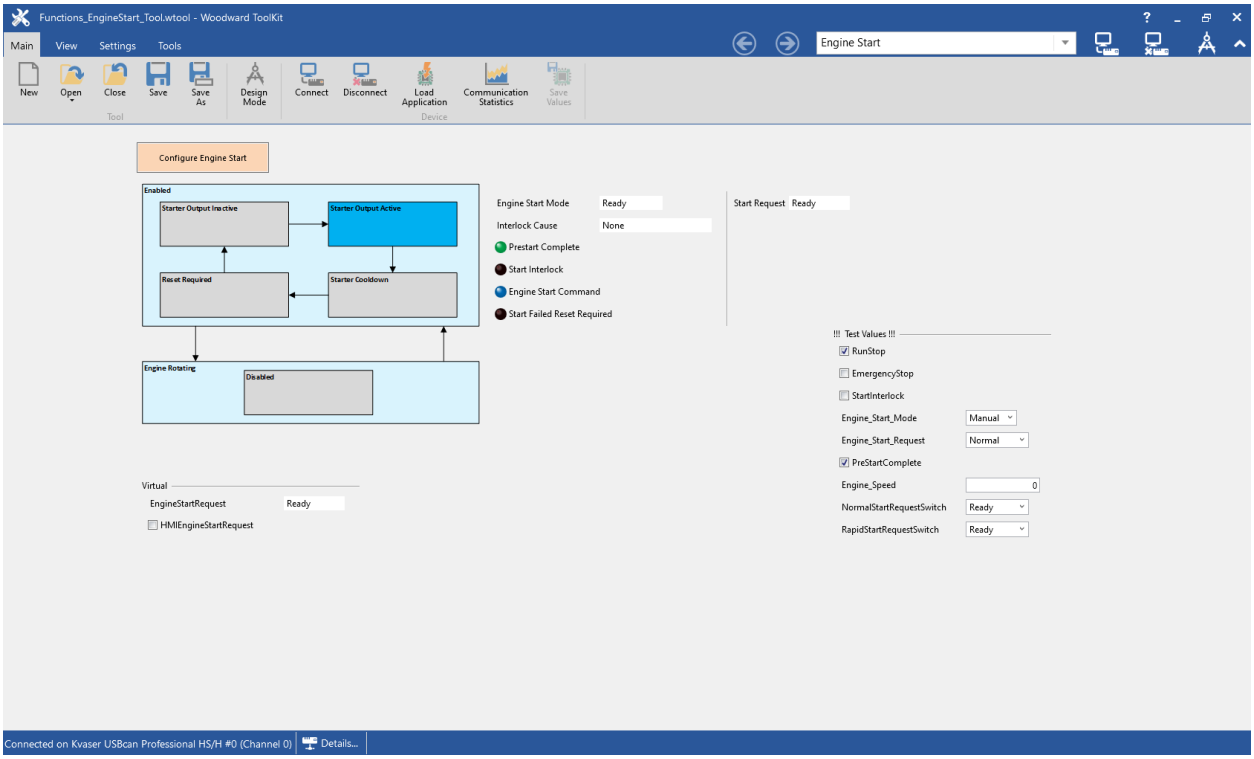


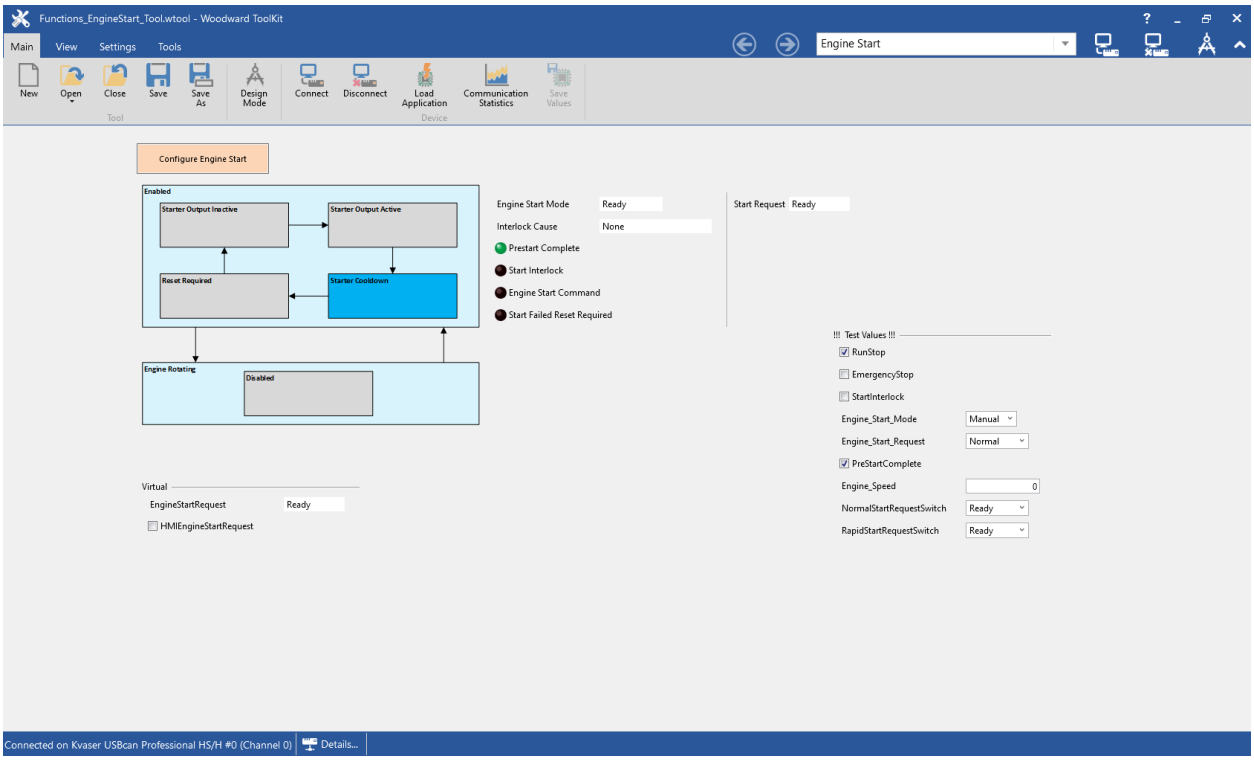
Figure 9-5. SID File Selection Functions – Engine Start Logic

The following are a few of the Toolkit pages that will contain most of the defined parameters in the example model. Of course, the real sensors and data must be attached to complete it for the application that is being worked on. For that reason, red X's appear on some pages.


Functions_EngineStart_Tool.wtool - Woodward ToolKit
Main
View
Settings
Tools
Engine Start
New
Open
Close
Save
Save As
Design Mode
Connect
Disconnect
Load Application
Communication Statistics
Save Values
Tool
Device
Configure Engine Start
Enabled
Starter Output Inactive
Starter Output Active
Reset Required
Starter Cooldown
Engine Rotating
Disabled
Engine Start Mode
Not Ready
Interlock Cause
Stop Commanded
Prestart Complete
Start Interlock
Engine Start Command
Start Failed Reset Required
Start Request
NotReady
Prestart Inhibitors
Error
!!! Test Values !!!
RunStop
EmergencyStop
StartInterlock
Engine_Start_Mode
Manual
Engine_Start_Request
Ready
PreStartComplete
Engine_Speed
0
NormalStartRequestSwitch
Ready
RapidStartRequestSwitch
Ready
Virtual
EngineStartRequest
NotReady
HMIEngineStartRequest
Connected on Kvaser USBcan Professional HS/H #0 (Channel 0)
Details...


Functions_EngineStart_Tool.wtool - Woodward ToolKit
Main
View
Settings
Tools
Engine Start
New
Open
Close
Save
Save As
Design Mode
Connect
Disconnect
Load Application
Communication Statistics
Save Values
Tool
Device
Configure Engine Start
Enabled
Starter Output Inactive
Starter Output Active
Reset Required
Starter Cooldown
Engine Rotating
Disabled
Engine Start Mode
Ready
Interlock Cause
None
Prestart Complete
Start Interlock
Engine Start Command
Start Failed Reset Required
Start Request
Ready
!!! Test Values !!!
RunStop
EmergencyStop
StartInterlock
Engine_Start_Mode
Manual
Engine_Start_Request
Ready
PreStartComplete
Engine_Speed
0
NormalStartRequestSwitch
Ready
RapidStartRequestSwitch
Ready
Virtual
EngineStartRequest
Ready
HMIEngineStartRequest
Connected on Kvaser USBcan Professional HS/H #0 (Channel 0)
Details...


Functions_EngineStart_Tool.wtool - Woodward ToolKit
Main
View
Settings
Tools
Engine Start
New
Open
Close
Save
Save As
Design Mode
Connect
Disconnect
Load Application
Communication Statistics
Save Values
Configure Engine Start
Enabled
Starter Output Inactive
Starter Output Active
Reset Required
Starter Cooldown
Engine Rotating
Disabled
Engine Start Mode
Ready
Interlock Cause
None
Prestart Complete
Start Interlock
Engine Start Command
Start Failed Reset Required
Start Request
Ready
!!! Test Values !!!
RunStop
EmergencyStop
StartInterlock
Engine_Start_Mode
Manual
Engine_Start_Request
Normal
PreStartComplete
Engine_Speed
0
NormalStartRequestSwitch
Ready
RapidStartRequestSwitch
Ready
Virtual
EngineStartRequest
Ready
HMIEngineStartRequest
Connected on Kvaser USBcan Professional HS/H #0 (Channel 0)
Details...


Functions_EngineStart_Tool.wtool - Woodward ToolKit
Main
View
Settings
Tools
Engine Start
New
Open
Close
Save
Save As
Design Mode
Connect
Disconnect
Load Application
Communication Statistics
Save Values
Configure Engine Start
Enabled
Starter Output Inactive
Starter Output Active
Reset Required
Starter Cooldown
Engine Rotating
Disabled
Engine Start Mode
Ready
Interlock Cause
None
Prestart Complete
Start Interlock
Engine Start Command
Start Failed Reset Required
Start Request
Ready
!!! Test Values !!!
RunStop
EmergencyStop
StartInterlock
Engine_Start_Mode
Manual
Engine_Start_Request
Normal
PreStartComplete
Engine_Speed
0
NormalStartRequestSwitch
Ready
RapidStartRequestSwitch
Ready
Virtual
EngineStartRequest
Ready
HMIEngineStartRequest
Connected on Kvaser USBcan Professional HS/H #0 (Channel 0)
Details...

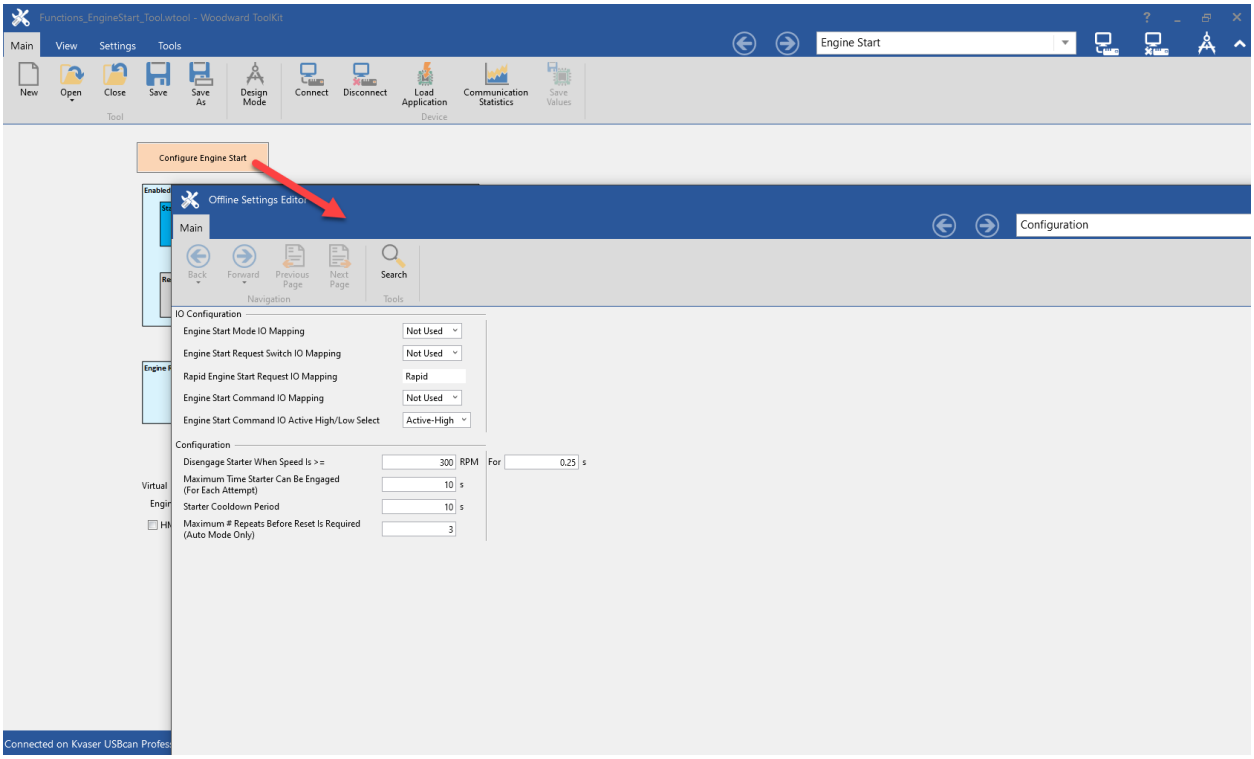


Figure 9-6. Example Functions - Engine Start Toolkit Pages

Functional Description - Engine Start

The engine start inputs are converted to a *start request* command to the start control logic as illustrated in the diagram below. The prestart complete flag (as described in Chapter 8) and start interlocks must be disabled before the start request will enter a *ready* state. The *start request* command includes the following:

- **Not Ready**
  - Indicates to the start controller the engine conditions are not valid for starter engagement.
- **Ready**
  - Indicates to the start controller the engine conditions are valid for starter engagement, awaiting a start request.
- **Normal**
  - Indicates an explicit normal start request has been initiated by the user.
- **Rapid**
  - Indicates an explicit rapid start request has been initiated by the user.

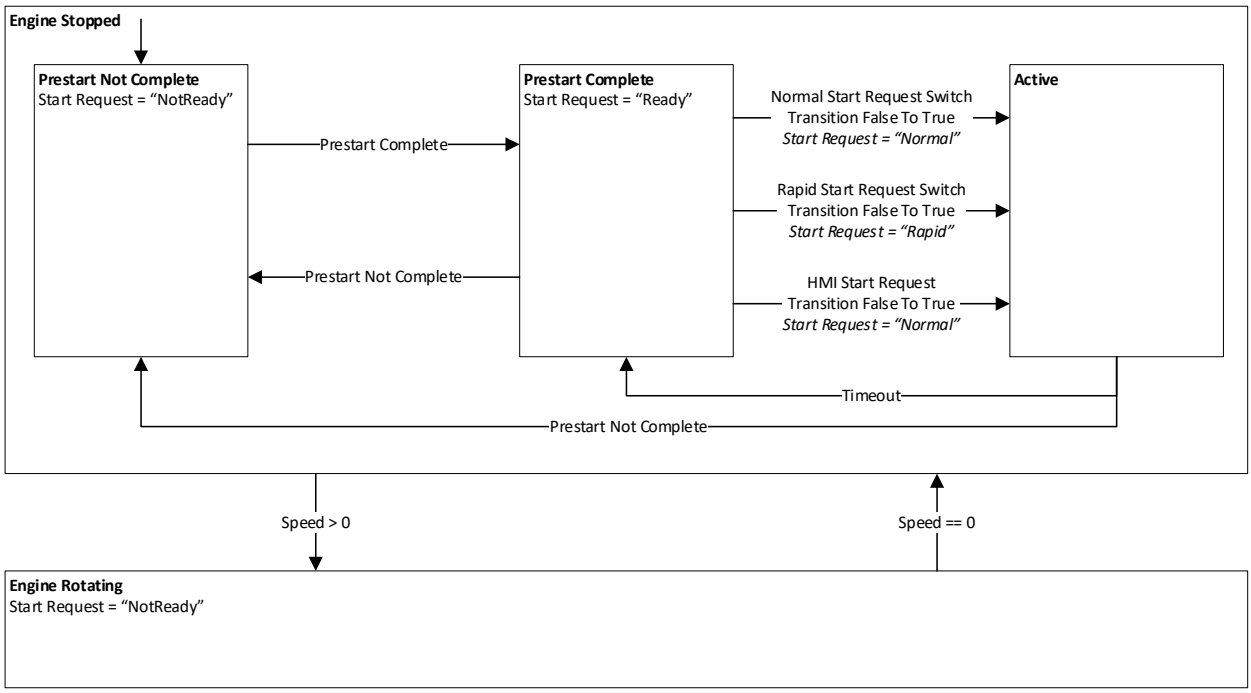


Figure 9-7. Engine Start Request Command Diagram

Engine Start Controller

The engine start controller consists of two modes of operation, Manual or Auto, and four controlling states: Inactive, Active, Cooldown, and Reset Required.

Inactive State

- The starter motor command is set to Inactive (off).
- All starting interlocks and enable conditions must be valid before a transition to the Active state is allowed.
- Manual Mode
  - The state will transition to Active if the start request command transitions from Ready to Normal or Rapid.

- Auto Mode
  - The state will transition to Active when the RunStop switch is equal to a Run state.

Active State

- The starter motor command is set to Active (on).
- The state will transition to Cooldown if the starter has been engaged for a user-specified timeout and the engine has not started, OR speed is observed above a starter engagement speed threshold for a user-specified time period.

Cooldown State

- The starter motor command is set to Inactive (off).
- Manual Mode
  - The state will transition to Inactive after a user-specified starter cooldown timeout period.
- Auto Mode
  - The state will transition to Inactive after a user-specified starter cooldown period AND the number of attempts limit has not been reached.
  - The state will transition to Reset Required after a user-specified starter cooldown period AND if the number of attempts limit has been reached.

Reset Required State

- The starter motor command is set to Inactive (off).
- The state will transition to Inactive if the mode is changed to Manual OR the RunStop is equal to Stop and an explicit reset command (or module reset) is initiated.
- # of attempts is reset to 0 after exiting the Reset Required state.

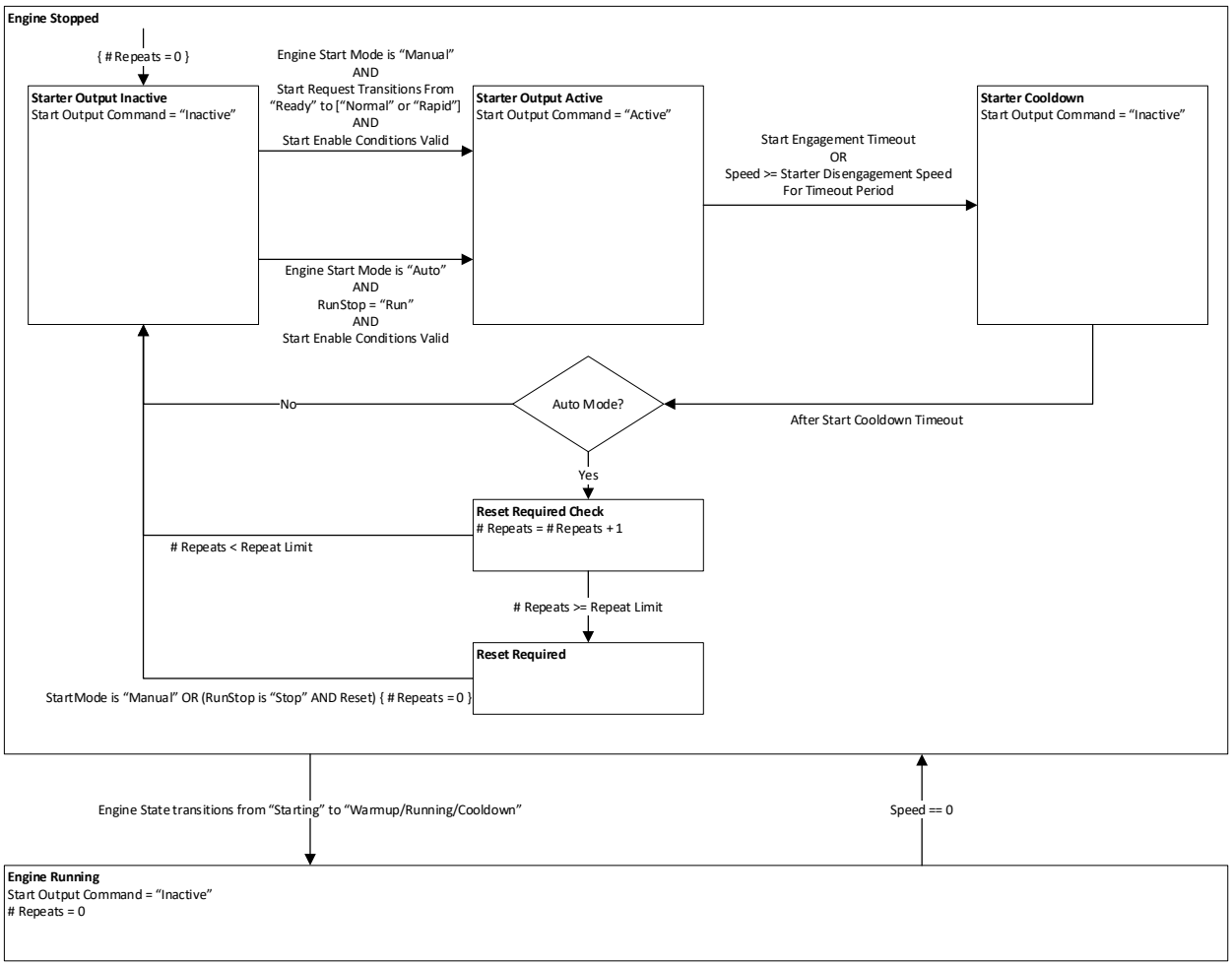


Figure 9-8. Engine Start Controller

The engine start logic can be viewed and configured on the "Control: Engine Start" service tool page.

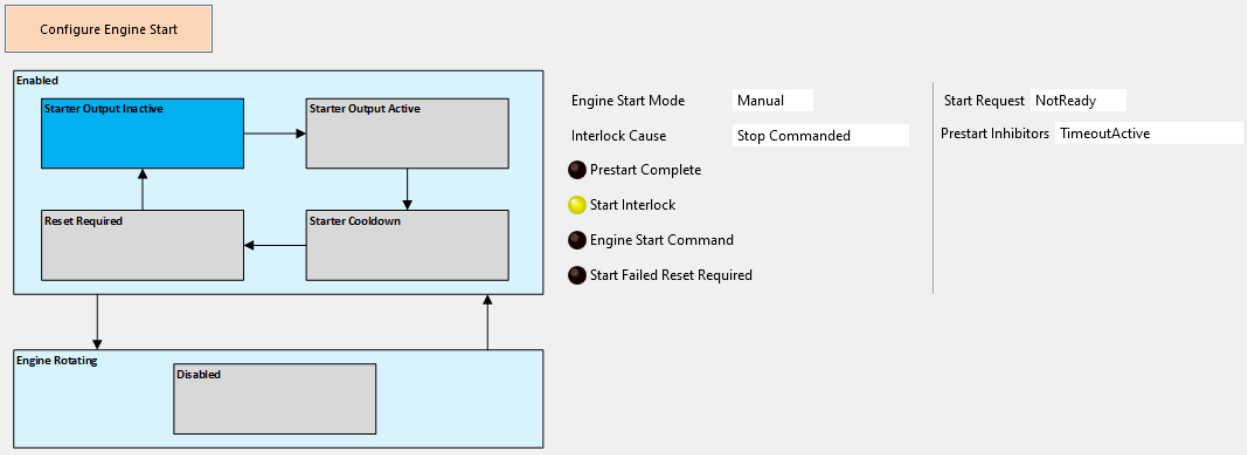


Figure 9-9. Engine Start Service Tool Page

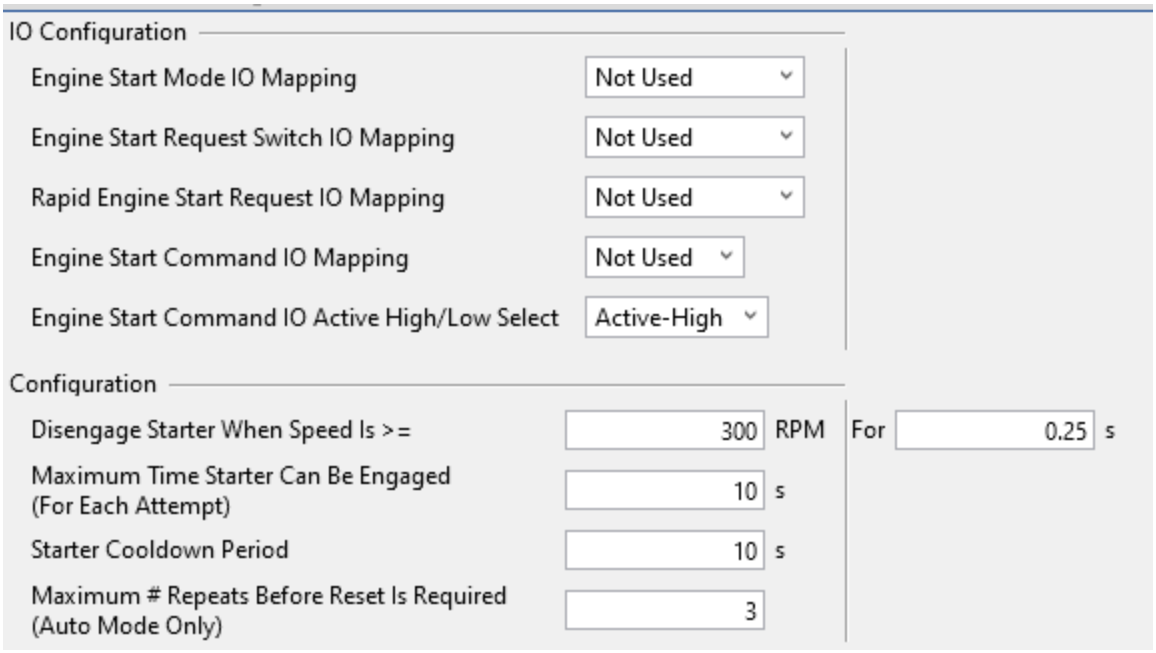


Figure 9-10. Engine Start Configurations

Chapter 10.
Sensors – Input Processing – CRS Dual Fuel

Introduction

The ‘Sensors – Input Processing CRS Dual Fuel’ blockset is used for Dual Fuel sensing purposes. Several blocks are available such as pressure sensing, speed sensing, discrete switch sensing, temperature sensing, and off-engine sensing.

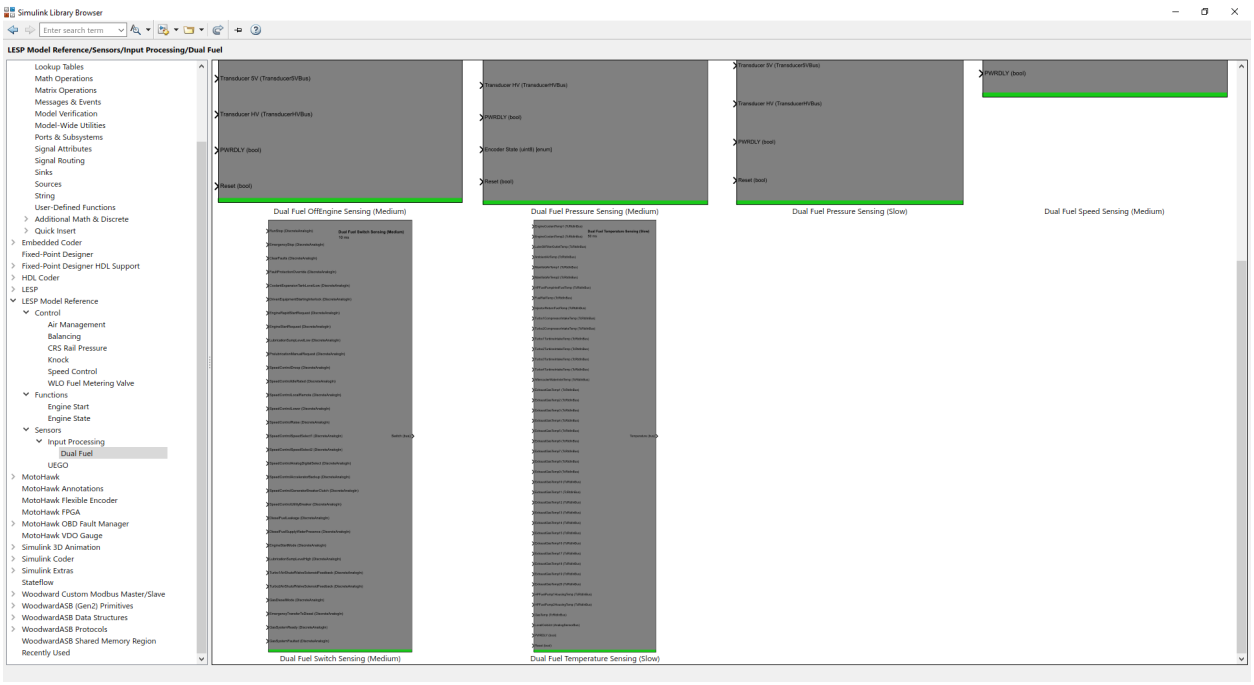


Figure 10-1. LESP Model Reference ‘Sensors – CRS Dual Fuel Input Processing’ Block

- CRS Dual Fuel OffEngine Sensing (Medium)
  This block can be used to process the Off-Engine Sensor Signals to be defined inside the application. It is called (Medium) because it is processed in 10 ms rate.

- CRS Dual Fuel Pressure Sensing (Medium)
  This block can be used to process the Pressure Sensor Signals to be defined inside the application. It is called (Medium) because it is processed in 10 ms rate.

- CRS Dual Fuel Pressure Sensing (Slow)
  This block can be used to process the Pressure Sensor Signals to be defined inside the application. It is called (Slow) because it is processed in 50 ms rate.

- CRS Dual Fuel Speed Sensing (Medium)
  This block can be used to process the Speed Sensor Signals to be defined inside the application. It is called (Medium) because it is processed in 10 ms rate.

- CRS Dual Fuel Switch Sensing (Medium)
  This block can be used to process the Discrete Switch Sensor Signals to be defined inside the application. It is called (Medium) because it is processed in 10 ms rate.

- CRS Dual Fuel Temperature Sensing (Slow)
  This block can be used to process the Temperature Sensor Signals to be defined inside the application. It is called (Medium) because it is processed in 50 ms rate.

CRS Dual Fuel OffEngine Sensing Block Interface

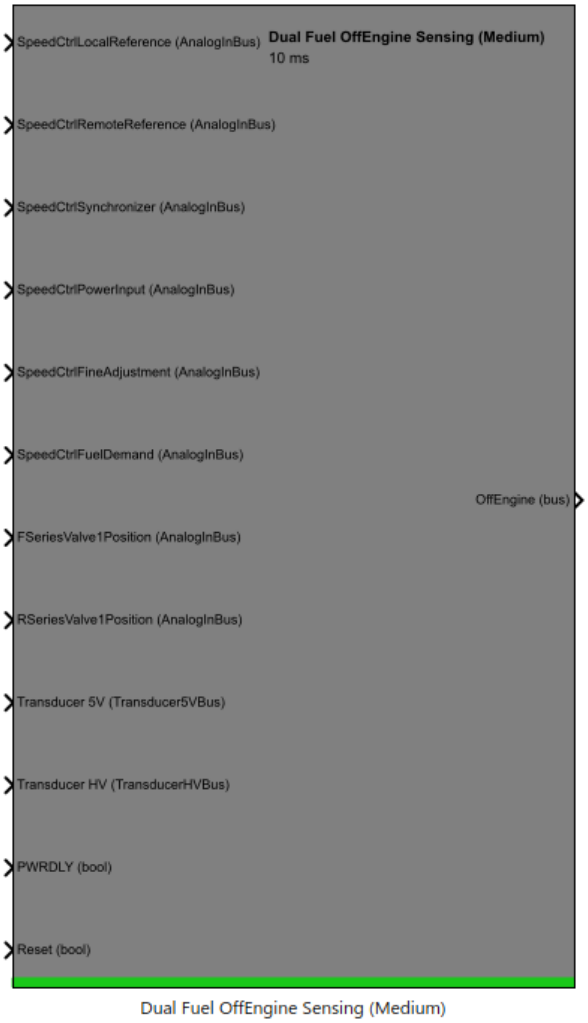


Figure 10-2. LESP Model Reference ‘Sensors – CRS Dual Fuel OffEngine Sensing’ Block

Table 10-1. ‘CRS Dual Fuel OffEngine Sensing (Medium)’ Block

Port	DataType	Description
[In] SpeedCtrlLocalReference	AnalogInBus	Input that requires the Speed Control Local Speed Reference setpoint for the application. This is normally an analog speed setting that comes from Local position (Engine Control Room). Bus Details: AnalogInBus: - Mode_enum - Value - ADC - ADCToHWUnitsGain - ADCToHWUnitsOffset - PullupResistorValue_ohms

		Mode_enum: (lesp_analog_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V (Mode 2) 0-1.25V (Mode 3) +-1V (Mode 4) +-2.5V (Mode 5) 4-20mA (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance (Mode 9) Thermocouple
[In] SpeedCtrlRemoteReference	AnalogInBus	Input that requires the Speed Control Remote Speed Reference setpoint for the application. This is normally an analog speed setting that comes from Remote position (Bridge Control). Bus Details: AnalogInBus: - Mode_enum - Value - ADC - ADCToHWUnitsGain - ADCToHWUnitsOffset - PullupResistorValue_ohms Mode_enum: (lesp_analog_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V (Mode 2) 0-1.25V (Mode 3) +-1V (Mode 4) +-2.5V (Mode 5) 4-20mA (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance (Mode 9) Thermocouple
[In] SpeedCtrlSynchronizer	AnalogInBus	Input that requires the Speed Control Synchronizer setpoint for the application. This is a bias signal that comes from a Synchronizer device, that will be used to increase/decrease engine speed to synchronize with other Gensets that are coupled to an electric network. Bus Details: AnalogInBus: - Mode_enum - Value - ADC - ADCToHWUnitsGain - ADCToHWUnitsOffset - PullupResistorValue_ohms Mode_enum: (lesp_analog_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V (Mode 2) 0-1.25V (Mode 3) +-1V

		(Mode 4) +-2.5V (Mode 5) 4-20mA (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance (Mode 9) Thermocouple
[In] SpeedCtrlPowerInput	AnalogInBus	Input that requires the Speed Control Power Input / kW input for the application. This is a signal that corresponds to the actual load measured (mostly a kW transducer for Gensets). Bus Details: AnalogInBus: - Mode_enum - Value - ADC - ADCToHWUnitsGain - ADCToHWUnitsOffset - PullupResistorValue_ohms Mode_enum: (lesp_analog_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V (Mode 2) 0-1.25V (Mode 3) +-1V (Mode 4) +-2.5V (Mode 5) 4-20mA (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance (Mode 9) Thermocouple
[In] SpeedCtrlFineAdjustment	AnalogInBus	Input that requires the Speed Control Fine Adjustment setpoint for the application. This is a bias signal that comes from an external device, that will be used to increase/decrease engine speed to fine tune the speed. Sometimes this is required during production validation testing. Bus Details: AnalogInBus: - Mode_enum - Value - ADC - ADCToHWUnitsGain - ADCToHWUnitsOffset - PullupResistorValue_ohms Mode_enum: (lesp_analog_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V (Mode 2) 0-1.25V (Mode 3) +-1V (Mode 4) +-2.5V (Mode 5) 4-20mA (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance (Mode 9) Thermocouple

[In] SpeedCtrlFuelDemand	AnalogInBus	Input that requires the Speed Control Fuel Demand for the application if an external Speed Control Device is used. It will correspond with the output signal of the external speed control. Bus Details: AnalogInBus: - Mode_enum - Value - ADC - ADCToHWUnitsGain - ADCToHWUnitsOffset - PullupResistorValue_ohms Mode_enum: (lesp_analog_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V (Mode 2) 0-1.25V (Mode 3) +-1V (Mode 4) +-2.5V (Mode 5) 4-20mA (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance (Mode 9) Thermocouple
[In] FSeriesValve1Position	AnalogInBus	Input that requires the F-Series Valve Position Setpoint. These valves can be used as Throttle, Bypass, and/or wastegate. Bus Details: AnalogInBus: - Mode_enum - Value - ADC - ADCToHWUnitsGain - ADCToHWUnitsOffset - PullupResistorValue_ohms Mode_enum: (lesp_analog_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V (Mode 2) 0-1.25V (Mode 3) +-1V (Mode 4) +-2.5V (Mode 5) 4-20mA (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance (Mode 9) Thermocouple
[In] RSeriesValve1Position	AnalogInBus	Input that requires the RSeries Valve Position Setpoint. These valves can be used as Throttle, Bypass, and/or Wastegate. Bus Details: AnalogInBus: - Mode_enum - Value - ADC - ADCToHWUnitsGain - ADCToHWUnitsOffset

		- PullupResistorValue_ohms Mode_enum: (lesp_analog_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V (Mode 2) 0-1.25V (Mode 3) +-1V (Mode 4) +-2.5V (Mode 5) 4-20mA (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance (Mode 9) Thermocouple
[In] Transducer 5V	Transducer5VBus	Input that requires the Transducer 5 V feedback value. This can be used for calculating ratiometric correction for the analog input signals. Bus Details: Transducer5VBus: - FilteredValue_V - MeasurementStatus_enum - RatiometricCorrectionFactor MeasurementStatus_enum: (lesp_measurement_status_enum) 0 – Valid 1 – Disabled 2 – SignalLow 3 – SignalHigh 4 – SensorSupplyFailed 5 – ReferenceSourceFailed 6 – OpenCircuit
[In] Transducer HV	TransducerHVBus	Input that requires the High Voltage (HV) Transducer feedback value. Bus Details: TransducerHighVoltBus: - FilteredValue_V - MeasurementStatus_enum - RatiometricCorrectionFactor - VoltageSelect_enum MeasurementStatus_enum: (lesp_measurement_status_enum) 0 – Valid 1 – Disabled 2 – SignalLow 3 – SignalHigh 4 – SensorSupplyFailed 5 – ReferenceSourceFailed 6 – OpenCircuit lesp_hv_transducer_select_enum: 0 – 12V 1 – 20V
[In] PWRDLY	boolean	This input requires the status of the PowerOn Delay that is required for some initialization logic.
[In] Reset	boolean	Input that requires the application to be reset and connected.

[Out] OffEngine	Bus	Output that contains the following signals, all of type (AnalogSensorBus): - SpeedCtrlLocalReference - SpeedCtrlRemoteReference - SpeedCtrlSynchronizer - SpeedCtrlPowerInput - SpeedCtrlFineAdjustment - SpeedCtrlFuelDemand - FSeriesValve1Position - RSeriesValve1Position Bus Details: AnalogSensorBus: - FilteredValue - MeasurementStatus_enum MeasurementStatus_enum: (lesp_measurement_status_enum) 0 – Valid 1 – Disabled 2 – SignalLow 3 – SignalHigh 4 – SensorSupplyFailed 5 – ReferenceSourceFailed 6 – OpenCircuit

CRS Dual Fuel Pressure Sensing (Medium) Block Interface

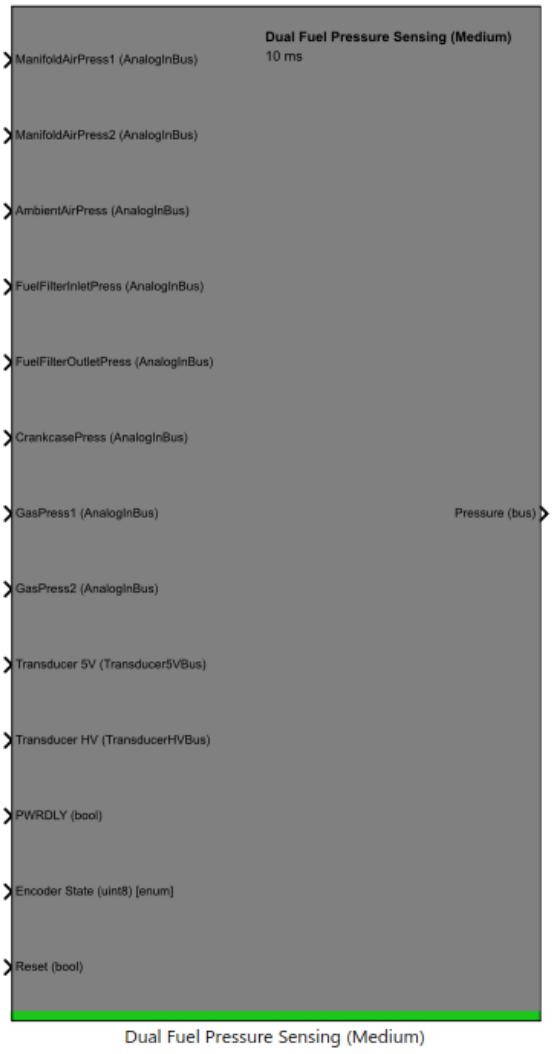


Figure 10-3. LESP Model Reference ‘Sensors – CRS Dual Fuel Pressure Sensing (Medium)’ Block

Table 10-2. ‘CRS Dual Fuel Pressure Sensing (Medium)’ Block

Port	DataType	Description
[In] ManifoldAirPress1	AnalogInBus	Input that requires the Engine Manifold Air Pressure Sensor #1 input signal. It is the one of the redundant sensors, measuring actual engine manifold air pressure. Bus Details: AnalogInBus: - Mode_enum - Value - ADC - ADCToHWUnitsGain - ADCToHWUnitsOffset - PullupResistorValue_ohms Mode_enum: (lesp_analog_in_mode_enum) (Mode 0) Undefined

		(Mode 1) 0-5V (Mode 2) 0-1.25V (Mode 3) +-1V (Mode 4) +-2.5V (Mode 5) 4-20mA (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance (Mode 9) Thermocouple
[In] ManifoldAirPress2	AnalogInBus	Input that requires the Engine Manifold Air Pressure Sensor #2 input signal. It is the one of the redundant sensors, measuring actual engine manifold air pressure. Bus Details: AnalogInBus: - Mode_enum - Value - ADC - ADCToHWUnitsGain - ADCToHWUnitsOffset - PullupResistorValue_ohms Mode_enum: (lesp_analog_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V (Mode 2) 0-1.25V (Mode 3) +-1V (Mode 4) +-2.5V (Mode 5) 4-20mA (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance (Mode 9) Thermocouple
[In] AmbientAirPress	AnalogInBus	Input that requires the Ambient Air Pressure input signal. This signal measures the actual ambient air pressure, which is required for the logic and absolute/relative pressure calculations. Bus Details: AnalogInBus: - Mode_enum - Value - ADC - ADCToHWUnitsGain - ADCToHWUnitsOffset - PullupResistorValue_ohms Mode_enum: (lesp_analog_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V (Mode 2) 0-1.25V (Mode 3) +-1V (Mode 4) +-2.5V (Mode 5) 4-20mA (Mode 6) RTD Low Impedance

		(Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance (Mode 9) Thermocouple
[In] FuelFilterInletPress	AnalogInBus	Input that requires the Fuel Filter Inlet Pressure input signal. This signal is the fuel pressure from tank to filter inlet. Bus Details: AnalogInBus: - Mode_enum - Value - ADC - ADCToHWUnitsGain - ADCToHWUnitsOffset - PullupResistorValue_ohms Mode_enum: (lesp_analog_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V (Mode 2) 0-1.25V (Mode 3) +-1V (Mode 4) +-2.5V (Mode 5) 4-20mA (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance (Mode 9) Thermocouple
[In] FuelFilterOutletPress	AnalogInBus	Input that requires the Fuel Filter Outlet Pressure input signal. This signal is the fuel pressure going to the engine. Bus Details: AnalogInBus: - Mode_enum - Value - ADC - ADCToHWUnitsGain - ADCToHWUnitsOffset - PullupResistorValue_ohms Mode_enum: (lesp_analog_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V (Mode 2) 0-1.25V (Mode 3) +-1V (Mode 4) +-2.5V (Mode 5) 4-20mA (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance (Mode 9) Thermocouple
[In] CrankCasePress	AnalogInBus	Input that requires the Crank Case Pressure input signal. This sensor is required to monitor for excessive crankcase pressure, which can lead to dangerous situations.

		Bus Details: AnalogInBus: - Mode_enum - Value - ADC - ADCToHWUnitsGain - ADCToHWUnitsOffset - PullupResistorValue_ohms Mode_enum: (lesp_analog_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V (Mode 2) 0-1.25V (Mode 3) +-1V (Mode 4) +-2.5V (Mode 5) 4-20mA (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance (Mode 9) Thermocouple
[In] GasPress1	AnalogInBus	Input that requires the Gas Pressure Sensor #1 input signal. It is the one of the redundant sensors, measuring actual gas pressure. Bus Details: AnalogInBus: - Mode_enum - Value - ADC - ADCToHWUnitsGain - ADCToHWUnitsOffset - PullupResistorValue_ohms Mode_enum: (lesp_analog_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V (Mode 2) 0-1.25V (Mode 3) +-1V (Mode 4) +-2.5V (Mode 5) 4-20mA (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance (Mode 9) Thermocouple
[In] GasPress2	AnalogInBus	Input that requires the Gas Pressure Sensor #2 input signal. It is one of the redundant sensors, measuring actual gas pressure. Bus Details: AnalogInBus: - Mode_enum - Value - ADC - ADCToHWUnitsGain - ADCToHWUnitsOffset - PullupResistorValue_ohms Mode_enum:

		(lesp_analog_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V (Mode 2) 0-1.25V (Mode 3) +-1V (Mode 4) +-2.5V (Mode 5) 4-20mA (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance (Mode 9) Thermocouple
[In] Transducer 5V	Transducer5VBus	Input that requires the Transducer 5 V feedback value. This can be used for calculating ratiometric correction for the analog input signals. Bus Details: Transducer5VBus: - FilteredValue_V - MeasurementStatus_enum - RatiometricCorrectionFactor MeasurementStatus_enum: (lesp_measurement_status_enum) 0 – Valid 1 – Disabled 2 – SignalLow 3 – SignalHigh 4 – SensorSupplyFailed 5 – ReferenceSourceFailed 6 – OpenCircuit
[In] Transducer HV	TransducerHVBus	Input that requires the High Voltage (HV) Transducer feedback value. Bus Details: TransducerHighVoltBus: - FilteredValue_V - MeasurementStatus_enum - RatiometricCorrectionFactor - VoltageSelect_enum MeasurementStatus_enum: (lesp_measurement_status_enum) 0 – Valid 1 – Disabled 2 – SignalLow 3 – SignalHigh 4 – SensorSupplyFailed 5 – ReferenceSourceFailed 6 – OpenCircuit lesp_hv_transducer_select_enum: 0 – 12V 1 – 20V
[In] PWRDLY	boolean	This input requires the status of the PowerOn Delay that is required for some initialization logic.
[In] EncoderState (enum)	uint8	This input requires the EncoderState enum input from the encoder logic as defined in the application. lesp_encoder_state_enum:

		0 – Not Created 1 – ZeroSpeed 2 – Rotating 3 – Partial Synchronization 4 – Full Synchronization Pending 5 – Full Synchronization
[In] Reset	boolean	Input that requires the application to be reset and connected.
[Out] Pressure	Bus	Output that contains the following signals, all of type (AnalogSensorbus): - ManifoldAirPress1 - ManifoldAirPress2 - ManifoldAirPress - AmbientAirPress - FuelFilterInletPress - FuelFilterOutletPress - FuelFilterDifferentialPress - CrankcasePress - GasPress1 - GasPress2 - GasPress Bus Details: AnalogSensorBus: - FilteredValue - MeasurementStatus_enum MeasurementStatus_enum: (lesp_measurement_status_enum) 0 – Valid 1 – Disabled 2 – SignalLow 3 – SignalHigh 4 – SensorSupplyFailed 5 – ReferenceSourceFailed 6 – OpenCircuit

CRS Dual Fuel Pressure Sensing (Slow) Block Interface

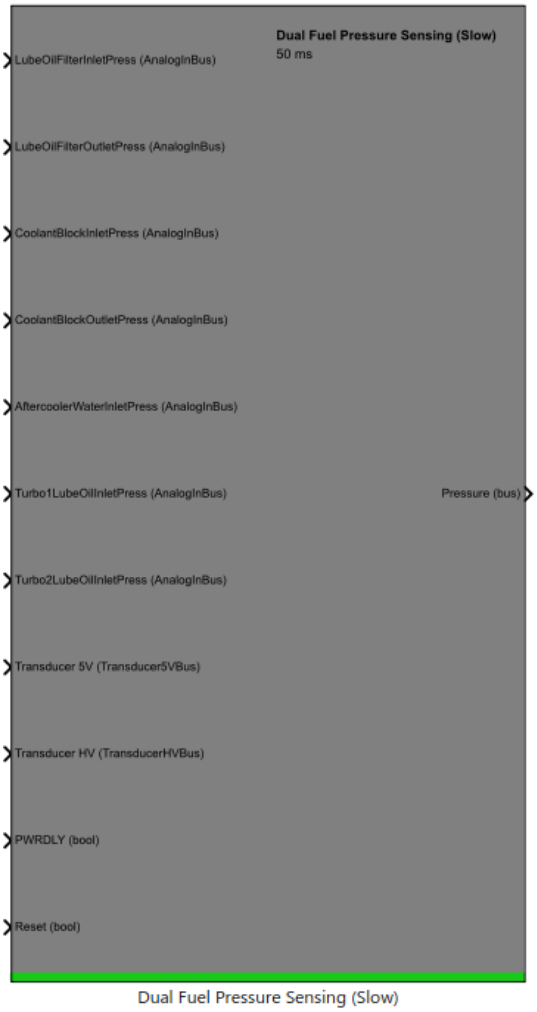


Figure 10-4. LESP Model Reference ‘Sensors – CRS Dual Fuel Pressure Sensing (Slow)’ Block

Table 10-3. ‘CRS Dual Fuel Pressure Sensing (Slow)’ Block

Port	DataType	Description
[In] LubeOilFilterInletPress	AnalogInBus	Input that requires the LubeOil Filter Inlet Pressure input signal. This is the pressure before/at LubeOil filter inlet. Bus Details: AnalogInBus: - Mode_enum - Value - ADC - ADCToHWUnitsGain - ADCToHWUnitsOffset - PullupResistorValue_ohms Mode_enum: (lesp_analog_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V (Mode 2) 0-1.25V (Mode 3) +-1V

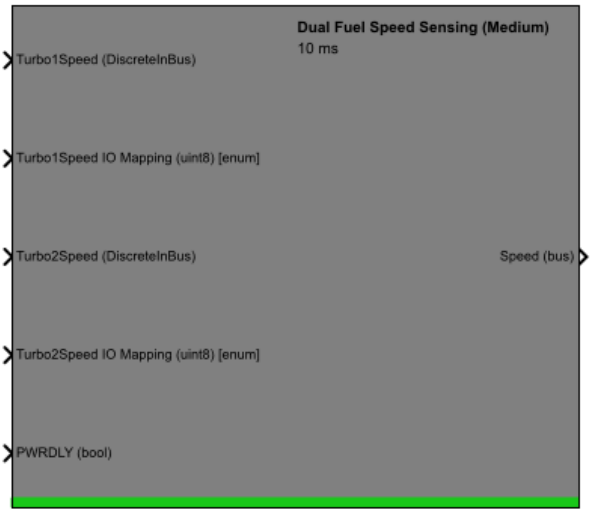
		(Mode 4) +-2.5V (Mode 5) 4-20mA (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance (Mode 9) Thermocouple
[In] LubeOilFilterOutletPress	AnalogInBus	Input that requires the LubeOil Filter Outlet Pressure input signal. This is the pressure after LubeOil filter at engine LubeOil inlet. Bus Details: AnalogInBus: - Mode_enum - Value - ADC - ADCToHWUnitsGain - ADCToHWUnitsOffset - PullupResistorValue_ohms Mode_enum: (lesp_analog_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V (Mode 2) 0-1.25V (Mode 3) +-1V (Mode 4) +-2.5V (Mode 5) 4-20mA (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance (Mode 9) Thermocouple
[In] CoolantBlockInletPress	AnalogInBus	Input that requires the Coolant Block Inlet Pressure input signal. This is the coolant pressure at Engine Block Inlet. Bus Details: AnalogInBus: - Mode_enum - Value - ADC - ADCToHWUnitsGain - ADCToHWUnitsOffset - PullupResistorValue_ohms Mode_enum: (lesp_analog_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V (Mode 2) 0-1.25V (Mode 3) +-1V (Mode 4) +-2.5V (Mode 5) 4-20mA (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance (Mode 9) Thermocouple
[In] CoolantBlockOutletPress	AnalogInBus	Input that requires the Coolant Block Outlet Pressure input signal. This is the coolant pressure at Engine Block Outlet.

		Bus Details: AnalogInBus: - Mode_enum - Value - ADC - ADCToHWUnitsGain - ADCToHWUnitsOffset - PullupResistorValue_ohms Mode_enum: (lesp_analog_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V (Mode 2) 0-1.25V (Mode 3) +-1V (Mode 4) +-2.5V (Mode 5) 4-20mA (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance (Mode 9) Thermocouple
[In] AfterCoolerWaterInletPress	AnalogInBus	Input that requires the AfterCooler Water Inlet Pressure input signal. This is the water inlet pressure at AfterCooler inlet. Bus Details: AnalogInBus: - Mode_enum - Value - ADC - ADCToHWUnitsGain - ADCToHWUnitsOffset - PullupResistorValue_ohms Mode_enum: (lesp_analog_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V (Mode 2) 0-1.25V (Mode 3) +-1V (Mode 4) +-2.5V (Mode 5) 4-20mA (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance (Mode 9) Thermocouple
[In] Turbo1LubeOilInletPress	AnalogInBus	Input that requires the TurboCharger #1 LubeOil Inlet Pressure input signal. This is the LubeOil inlet pressure for the TC#1. Bus Details: AnalogInBus: - Mode_enum - Value - ADC - ADCToHWUnitsGain - ADCToHWUnitsOffset - PullupResistorValue_ohms Mode_enum: (lesp_analog_in_mode_enum)

		(Mode 0) Undefined (Mode 1) 0-5V (Mode 2) 0-1.25V (Mode 3) +-1V (Mode 4) +-2.5V (Mode 5) 4-20mA (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance (Mode 9) Thermocouple
[In] Turbo2LubeOilInletPress	AnalogInBus	Input that requires the TurboCharger #2 LubeOil Inlet Pressure input signal. This is the LubeOil inlet pressure for the TC#2. Bus Details: AnalogInBus: - Mode_enum - Value - ADC - ADCToHWUnitsGain - ADCToHWUnitsOffset - PullupResistorValue_ohms Mode_enum: (lesp_analog_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V (Mode 2) 0-1.25V (Mode 3) +-1V (Mode 4) +-2.5V (Mode 5) 4-20mA (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance (Mode 9) Thermocouple
[In] Transducer 5V	Transducer5VBus	Input that requires the Transducer 5 V feedback value. This can be used for calculating ratiometric correction for the analog input signals. Bus Details: Transducer5VBus: - FilteredValue_V - MeasurementStatus_enum - RatiometricCorrectionFactor MeasurementStatus_enum: (lesp_measurement_status_enum) 0 – Valid 1 – Disabled 2 – SignalLow 3 – SignalHigh 4 – SensorSupplyFailed 5 – ReferenceSourceFailed 6 – OpenCircuit
[In] Transducer HV	TransducerHVBus	Input that requires the High Voltage (HV) Transducer feedback value. Bus Details: TransducerHighVoltBus: - FilteredValue_V - MeasurementStatus_enum

		- RatiometricCorrectionFactor - VoltageSelect_enum MeasurementStatus_enum: (lesp_measurement_status_enum) 0 – Valid 1 – Disabled 2 – SignalLow 3 – SignalHigh 4 – SensorSupplyFailed 5 – ReferenceSourceFailed 6 – OpenCircuit lesp_hv_transducer_select_enum: 0 – 12V 1 – 20V
[In] PWRDLY	boolean	This input requires the status of the PowerOn Delay that is required for some initialization logic.
[In] Reset	boolean	Input that requires to have the reset of the application connected.
[Out] Pressure	Bus	Output that contains the following signals, all of type (AnalogSensorbus): - LubeOilFilterInletPress - LubeOilFilterOutletPress - LubeOilFilterDifferentialPress - CoolantBlockInletPress - CoolantBlockOutletPress - AfterCoolerWaterInletPress - Turbo1LubeOilInletPress - Turbo2LubeOilInletPress Bus Details: AnalogSensorBus: - FilteredValue - MeasurementStatus_enum MeasurementStatus_enum: (lesp_measurement_status_enum) 0 – Valid 1 – Disabled 2 – SignalLow 3 – SignalHigh 4 – SensorSupplyFailed 5 – ReferenceSourceFailed 6 – OpenCircuit

CRS Dual Fuel Speed Sensing (Medium) Block Interface



Dual Fuel Speed Sensing (Medium)

Figure 10-5. LESP Model Reference ‘Sensors – CRS Dual Fuel Speed Sensing (Medium)’ Block

Table 10-4. ‘Dual Fuel Speed Sensing (Medium)’ Block

Port	DataType	Description
[In] Turbo1Speed	DiscreteInBus	Input that requires the TurboCharger #1 speed input signal. This is the actual speed measured at the Turbocharger #1 wheel. Bus Details: DiscreteInBus: - DiscreteState - Frequency_Hz - DutyCycle_Pct
[In] Turbo1SpeedIOMapping (enum)	uint8	Input that requires the TurboCharger #1 Speed IOMapping signal. This is the IOMapping for the speed measurement at the Turbocharger #1 wheel. enum Details: (lecm_frequency_in_mapping_enum) 0 – Not Used 1 – MainSpeed1 2 – MainSpeed2 3 – MainSpeed3 4 – MainSpeed4 5 – EIDSpeed1 6 – EIDSpeed2 7 – AuxSpeed1 8 – AuxSpeed2

[In] Turbo2Speed	DiscreteInBus	Input that requires the TurboCharger #2 speed input signal. This is the actual speed measured at the Turbocharger #2 wheel. Bus Details: DiscreteInBus: - DiscreteState - Frequency_Hz - DutyCycle_Pct
[In] Turbo2SpeedIOMapping (enum)	uint8	Input that requires the TurboCharger #2 speed IOMapping signal. This is the IOMapping for the speed measurement at the Turbocharger #2 wheel. enum Details: (lecm_frequency_in_mapping_enum) 0 – Not Used 1 – MainSpeed1 2 – MainSpeed2 3 – MainSpeed3 4 – MainSpeed4 5 – EIDSpeed1 6 – EIDSpeed2 7 – AuxSpeed1 8 – AuxSpeed2
[In] PWRDLY	boolean	This input requires the status of the PowerOn Delay that is required for some initialization logic.
[Out] Speed	Bus	Output that contains the following signals, all type (AnalogSensorbus): - Turbo1Speed - Turbo2Speed Bus Details: AnalogSensorBus: - FilteredValue - MeasurementStatus_enum MeasurementStatus_enum: (lesp_measurement_status_enum) 0 – Valid 1 – Disabled 2 – SignalLow 3 – SignalHigh 4 – SensorSupplyFailed 5 – ReferenceSourceFailed 6 – OpenCircuit

CRS Dual Fuel Switch Sensing (Medium) Block Interface

Figure 10-6. LESP Model Reference ‘Sensors – CRS Dual Fuel Switch Sensing (Medium)’ Block

Table 10-5. ‘Dual Fuel Switch Sensing (Medium)’ Block

Port	DataType	Description
[In] RunStop	DiscreteAnalogIn	Input that requires the RunStop status feedback in the application. Run – Engine is ready to run and SpeedControl will be active. Stop – Engine will shut down and go into Stop state.

		DiscreteAnalogIn: - Type_enum - AnalogADC - DiscreteState Type_enum: (lesp_discrete_analog_in_mapping_type_enum) 0 – None 1 – Discrete 2 – Analog
[In] EmergencyStop	DiscreteAnalogIn	This input requires the Emergency Stop Input signal as defined in the application. Emergency Stop indicates direct fuel shutdown to force engine stop. DiscreteAnalogIn: - Type_enum - AnalogADC - DiscreteState Type_enum: (lesp_discrete_analog_in_mapping_type_enum) 0 – None 1 – Discrete 2 – Analog
[In] ClearFaults	DiscreteAnalogIn	This input requires the Clear Faults Input signal as defined in the application. It is meant to clear any fault that might be active on the used channels. DiscreteAnalogIn: - Type_enum - AnalogADC - DiscreteState Type_enum: (lesp_discrete_analog_in_mapping_type_enum) 0 – None 1 – Discrete 2 – Analog
[In] FaultProtectionOverride	DiscreteAnalogIn	This input requires the Fault Protection Override Input signal as defined in the application. It is meant to override any shutdown in case of emergencies. This is typical onboard vessels as an ultimate rescue to save the vessel at all costs. DiscreteAnalogIn: - Type_enum - AnalogADC - DiscreteState Type_enum: (lesp_discrete_analog_in_mapping_type_enum) 0 – None 1 – Discrete 2 – Analog
[In] CoolantExpansionTankLevelLow	DiscreteAnalogIn	This input requires the Tank Level Low feedback from the Coolant Expansion Tank.

		When level drops below a certain minimum, this input will be activated. DiscreteAnalogIn: - Type_enum - AnalogADC - DiscreteState Type_enum: (lesp_discrete_analog_in_mapping_type_enum) 0 – None 1 – Discrete 2 – Analog
[In] DrivenEquipmentStartInterlock	DiscreteAnalogIn	This input requires the Start Interlock from any driven equipment. Examples are Turning Wheel engaged or any other engine maintenance blocking. When this is active, the engine must be blocked from any start attempt. DiscreteAnalogIn: - Type_enum - AnalogADC - DiscreteState Type_enum: (lesp_discrete_analog_in_mapping_type_enum) 0 – None 1 – Discrete 2 – Analog
[In] EngineRapidStartRequest	DiscreteAnalogIn	This input requires the Rapid Start Input signal as defined in the application. Rapid Start will override pre-start sequencing and enable starting possibility directly. DiscreteAnalogIn: - Type_enum - AnalogADC - DiscreteState Type_enum: (lesp_discrete_analog_in_mapping_type_enum) 0 – None 1 – Discrete 2 – Analog
[In] EngineStartRequest	DiscreteAnalogIn	This input requires the Request for Engine Start in the application. When this input becomes active, the engine Start will be executed (when allowed in the application logic). DiscreteAnalogIn: - Type_enum - AnalogADC - DiscreteState Type_enum: (lesp_discrete_analog_in_mapping_type_enum) 0 – None 1 – Discrete 2 – Analog

[In] LubricationSumpLevelLow	DiscreteAnalogIn	This input requires the Low Level feedback from the Lubrication Oil Sump. When oil level drops below a certain minimum level, this input should be activated. DiscreteAnalogIn: - Type_enum - AnalogADC - DiscreteState Type_enum: (lesp_discrete_analog_in_mapping_type_enum) 0 – None 1 – Discrete 2 – Analog
[In] PreLubricationManualRequest	DiscreteAnalogIn	This input requires the manual request for starting the Pre-lubrication Oil Pump inside the application. DiscreteAnalogIn: - Type_enum - AnalogADC - DiscreteState Type_enum: (lesp_discrete_analog_in_mapping_type_enum) 0 – None 1 – Discrete 2 – Analog
[In] SpeedControlDroop	DiscreteAnalogIn	This input requires the selection if Droop is required in the application or not. When not selected, isochronous operation will be active. Droop is a reduction in speed reference when load is increased. This is used for stability and loadsharing purposes. DiscreteAnalogIn: - Type_enum - AnalogADC - DiscreteState Type_enum: (lesp_discrete_analog_in_mapping_type_enum) 0 – None 1 – Discrete 2 – Analog
[In] SpeedControlIdleRated	DiscreteAnalogIn	This input requires the Idle/Rated Speed Selection for the application. This will be active when Speedcontrol is in Digital Speed Settings mode. DiscreteAnalogIn: - Type_enum - AnalogADC - DiscreteState Type_enum: (lesp_discrete_analog_in_mapping_type_enum) 0 – None

		1 – Discrete 2 – Analog
[In] SpeedControlLocalRemote	DiscreteAnalogIn	This input requires the selection input between Local and Remote Speed reference setpoint inside the application. This selection is active when speed control is in Analog Speed Setting mode. DiscreteAnalogIn: - Type_enum - AnalogADC - DiscreteState Type_enum: (lesp_discrete_analog_in_mapping_type_enum) 0 – None 1 – Discrete 2 – Analog
[In] SpeedControlLower	DiscreteAnalogIn	This input requires the Lower Speed Selection for the application. This will be active when Speedcontrol is in Digital Speed Settings mode. DiscreteAnalogIn: - Type_enum - AnalogADC - DiscreteState Type_enum: (lesp_discrete_analog_in_mapping_type_enum) 0 – None 1 – Discrete 2 – Analog
[In] SpeedControlRaise	DiscreteAnalogIn	This input requires the Raise Speed Selection for the application. This will be active when Speedcontrol is in Digital Speed Settings mode. DiscreteAnalogIn: - Type_enum - AnalogADC - DiscreteState Type_enum: (lesp_discrete_analog_in_mapping_type_enum) 0 – None 1 – Discrete 2 – Analog
[In] SpeedControlSpeedSelect1	DiscreteAnalogIn	This input requires the Speed Select1 status for the application. This will be active when Speedcontrol is in Digital Speed Settings mode, and together with the SpeedControlSpeedSelect2, determines the actual rated speed setpoint. A combination of 4 setpoints can be made with both input definitions and normally comes from wiring harness jumpers. DiscreteAnalogIn:

		- Type_enum - AnalogADC - DiscreteState Type_enum: (lesp_discrete_analog_in_mapping_type_enum) 0 – None 1 – Discrete 2 – Analog
[In] SpeedControlSpeedSelect2	DiscreteAnalogIn	This input requires the Speed Select2 status for the application. This will be active when Speedcontrol is in Digital Speed Settings mode, and together with the SpeedControlSpeedSelect1, determines the actual rated speed setpoint. A combination of 4 setpoints can be made with both input definitions and normally comes from wiring harness jumpers. DiscreteAnalogIn: - Type_enum - AnalogADC - DiscreteState Type_enum: (lesp_discrete_analog_in_mapping_type_enum) 0 – None 1 – Discrete 2 – Analog
[In] SpeedControlAnalogDigitalSelect	DiscreteAnalogIn	This input requires the Analog/Discrete Speed Setting Selection input for the application. This selection determines if the analog or digital speed related inputs will be active or not. DiscreteAnalogIn: - Type_enum - AnalogADC - DiscreteState Type_enum: (lesp_discrete_analog_in_mapping_type_enum) 0 – None 1 – Discrete 2 – Analog
[In] SpeedControlAcceleratorBackup	DiscreteAnalogIn	This input requires the Accelerator Backup Speed Selection for the application. This will be active when Speedcontrol is in Analog Speed Settings mode and enables the digital Raise/Lower commands to be used, even when in Analog Speed Setting mode. DiscreteAnalogIn: - Type_enum - AnalogADC - DiscreteState Type_enum: (lesp_discrete_analog_in_mapping_type_enum) 0 – None

		1 – Discrete 2 – Analog
[In] SpeedControlGeneratorBreakerClutch	DiscreteAnalogIn	This input requires the feedback of either the Generator Breaker (Genset) or the Gearbox Clutch (Marine/Mechanical) for the application. When this input changes, it also changes the dynamics selection of the PID for the Speed Control. DiscreteAnalogIn: - Type_enum - AnalogADC - DiscreteState Type_enum: (lesp_discrete_analog_in_mapping_type_enum) 0 – None 1 – Discrete 2 – Analog
[In] SpeedControlUtilityBreaker	DiscreteAnalogIn	This input requires feedback from the Utility Breaker (Genset) for the application. When this input changes, it also changes the dynamics selection of the PID for the Speed Control. DiscreteAnalogIn: - Type_enum - AnalogADC - DiscreteState Type_enum: (lesp_discrete_analog_in_mapping_type_enum) 0 – None 1 – Discrete 2 – Analog
[In] DieselFuelLeakage	DiscreteAnalogIn	This input requires feedback from the Fuel Leakage Detection inside the application. DiscreteAnalogIn: - Type_enum - AnalogADC - DiscreteState Type_enum: (lesp_discrete_analog_in_mapping_type_enum) 0 – None 1 – Discrete 2 – Analog
[In] DieselFuelSupplyWaterPresence	DiscreteAnalogIn	This input requires feedback from the Water Presence Detection inside the Diesel Fuel Supply for the application. DiscreteAnalogIn: - Type_enum - AnalogADC - DiscreteState Type_enum: (lesp_discrete_analog_in_mapping_type_enum)

		0 – None 1 – Discrete 2 – Analog
[In] EngineStartMode	DiscreteAnalogIn	This input requires feedback from the Engine Start Mode inside the application. This can be either Manual Mode or Auto Mode. DiscreteAnalogIn: - Type_enum - AnalogADC - DiscreteState Type_enum: (lesp_discrete_analog_in_mapping_type_enum) 0 – None 1 – Discrete 2 – Analog
[In] LubricationSumpLevelHigh	DiscreteAnalogIn	This input requires the High Level feedback from the Lubrication Oil Sump. When oil level raises above a certain maximum level, this input should be activated. DiscreteAnalogIn: - Type_enum - AnalogADC - DiscreteState Type_enum: (lesp_discrete_analog_in_mapping_type_enum) 0 – None 1 – Discrete 2 – Analog
[In] Turbo1AirShutoffValveSolenoidFeedback	DiscreteAnalogIn	This input requires the feedback status from the Turbo1 Air Shutoff valve solenoid. It indicates whether the valve is in the open or closed position. DiscreteAnalogIn: - Type_enum - AnalogADC - DiscreteState Type_enum: (lesp_discrete_analog_in_mapping_type_enum) 0 – None 1 – Discrete 2 – Analog
[In] Turbo2AirShutoffValveSolenoidFeedback	DiscreteAnalogIn	This input requires the feedback status from the Turbo2 Air Shutoff valve solenoid. It indicates whether the valve is in the open or closed position. DiscreteAnalogIn: - Type_enum - AnalogADC - DiscreteState Type_enum: (lesp_discrete_analog_in_mapping_type_enum)

		0 – None 1 – Discrete 2 – Analog
[In] GasDieselmode	DiscreteAnalogIn	This input requires the selection status for operating the engine in Diesel or in Gas-Diesel mode. When changing this input, the transfer from full diesel to gas-diesel or from gas-diesel to full diesel will need to be activated when the application conditions allow. DiscreteAnalogIn: - Type_enum - AnalogADC - DiscreteState Type_enum: (lesp_discrete_analog_in_mapping_type_enum) 0 – None 1 – Discrete 2 – Analog
[In] EmergencyTransferToDiesel	DiscreteAnalogIn	This input requires the selection status for Emergency transfer back to Diesel mode when in Gas-Diesel mode. When active, instant switch over to full Diesel mode will be required. This is normally done as result of engine safety (e.g. due to gas leak or engine sensor exceeding certain levels). DiscreteAnalogIn: - Type_enum - AnalogADC - DiscreteState Type_enum: (lesp_discrete_analog_in_mapping_type_enum) 0 – None 1 – Discrete 2 – Analog
[In] GasSystemReady	DiscreteAnalogIn	This input requires the feedback from the Gas Valve Unit (GVU) to indicate that the valves are in good condition, or the Gas System is ready and allows the engine to transfer from Diesel into Gas-Diesel mode. DiscreteAnalogIn: - Type_enum - AnalogADC - DiscreteState Type_enum: (lesp_discrete_analog_in_mapping_type_enum) 0 – None 1 – Discrete 2 – Analog
[In] GasSystemFaulted	DiscreteAnalogIn	This input requires the feedback from the GVU to indicate that the Gas System has Faulted. This input is normally used to transfer back from Gas-Diesel into Full Diesel mode.

		DiscreteAnalogIn: - Type_enum - AnalogADC - DiscreteState Type_enum: (lesp_discrete_analog_in_mapping_type_enum) 0 – None 1 – Discrete 2 – Analog
[Out] Switch	Bus	Output that contains the following signals, all of type (uint8): - RunStop - EmergencyStop - ClearFaults - FaultProtectionOverride - CoolantExpansionTankLevelLow - DrivenEquipmentStartingInterlock - EngineRapidStartRequest - EngineStartRequest - LubricationSumpLevelLow - PrelubricationManualRequest - SpeedControlDroop - SpeedControlIdleRated - SpeedControlLocalRemote - SpeedControlLower - SpeedControlRaise - SpeedControlSpeedSelect1 - SpeedControlSpeedSelect2 - SpeedControlAnalogDigitalSelect - SpeedControlAcceleratorBackup - SpeedControlGeneratorBreakerClutch - SpeedControlUtilityBreaker - DieselFuelLeakage - DieselFuelSupplyWaterPresence - EngineStartMode - LubricationSumpLevelHigh - Turbo1AirShutoffValveSolenoid- Feedback - Turbo2AirShutoffValveSolenoid- Feedback - GasDieselMode - EmergencyTransferToDiesel - GasSystemReady - GasSystemFaulted

CRS Dual Fuel Temperature Sensing (Slow) Block Interface

Figure 10-7. LESP Model Reference ‘Sensors – CRS Dual Fuel Temperature Sensing (Slow)’ Block

Table 10-6. ‘CRS Dual Fuel Temperature Sensing (Slow)’ Block

Port	DataType	Description
[In] EngineCoolantTemp1	TcRtdInBus	Input that requires the Engine Coolant Temperature #1 input signal. TcRtdInBus:

		- Mode_enum - Status_enum - PullupResistorValue_ohms - Value Mode_enum: (lesp_tcrtd_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance (Mode 9) Thermocouple (Mode 10) Datalink degC Status_enum: (lesp_tcrtd_in_status_enum) 0 – NotAvailable 1 – Error 2 – Open 3 – Valid
[In] EngineCoolantTemp2	TcRtdInBus	Input that requires the Engine Coolant Temperature #2 input signal. TcRtdInBus: - Mode_enum - Status_enum - PullupResistorValue_ohms - Value Mode_enum: (lesp_tcrtd_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance (Mode 9) Thermocouple (Mode 10) Datalink degC Status_enum: (lesp_tcrtd_in_status_enum) 0 – NotAvailable 1 – Error 2 – Open 3 – Valid
[In] LubeOilFilterOutletTemp	TcRtdInBus	Input that requires the Lubrication Oil Filter Outlet Temperature input signal. TcRtdInBus: - Mode_enum - Status_enum - PullupResistorValue_ohms - Value Mode_enum: (lesp_tcrtd_in_mode_enum)

		(Mode 0) Undefined (Mode 1) 0-5V (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance (Mode 9) Thermocouple (Mode 10) Datalink degC Status_enum: (lesp_tcrtd_in_status_enum) 0 – NotAvailable 1 – Error 2 – Open 3 – Valid
[In] AmbientAirTemp	TcRtdInBus	Input that requires the Ambient Air Temperature input signal. TcRtdInBus: - Mode_enum - Status_enum - PullupResistorValue_ohms - Value Mode_enum: (lesp_tcrtd_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance (Mode 9) Thermocouple (Mode 10) Datalink degC Status_enum: (lesp_tcrtd_in_status_enum) 0 – NotAvailable 1 – Error 2 – Open 3 – Valid
[In] ManifoldAirTemp1	TcRtdInBus	Input that requires the Engine Manifold Air #1 Temperature input signal. TcRtdInBus: - Mode_enum - Status_enum - PullupResistorValue_ohms - Value Mode_enum: (lesp_tcrtd_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance (Mode 9) Thermocouple (Mode 10) Datalink degC

		Status_enum: (lesp_tcrtd_in_status_enum) 0 – NotAvailable 1 – Error 2 – Open 3 – Valid
[In] ManifoldAirTemp2	TcRtdInBus	Input that requires the Engine Manifold Air #2 Temperature input signal. TcRtdInBus: - Mode_enum - Status_enum - PullupResistorValue_ohms - Value Mode_enum: (lesp_tcrtd_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance (Mode 9) Thermocouple (Mode 10) Datalink degC Status_enum: (lesp_tcrtd_in_status_enum) 0 – NotAvailable 1 – Error 2 – Open 3 – Valid
[In] HPFuelPumpInletFuelTemp	TcRtdInBus	Input that requires the High Pressure Pump Inlet Fuel Temperature input signal. TcRtdInBus: - Mode_enum - Status_enum - PullupResistorValue_ohms - Value Mode_enum: (lesp_tcrtd_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance (Mode 9) Thermocouple (Mode 10) Datalink degC Status_enum: (lesp_tcrtd_in_status_enum) 0 – NotAvailable 1 – Error 2 – Open 3 – Valid

[In] FuelRailTemp	TcRtdInBus	Input that requires the Fuel Rail Temperature input signal. TcRtdInBus: - Mode_enum - Status_enum - PullupResistorValue_ohms - Value Mode_enum: (lesp_tcrtd_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance (Mode 9) Thermocouple (Mode 10) Datalink degC Status_enum: (lesp_tcrtd_in_status_enum) 0 – NotAvailable 1 – Error 2 – Open 3 – Valid
[In] InjectorReturnFuelTemp	TcRtdInBus	Input that requires the Injector Return Line Fuel Temperature input signal. TcRtdInBus: - Mode_enum - Status_enum - PullupResistorValue_ohms - Value Mode_enum: (lesp_tcrtd_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance (Mode 9) Thermocouple (Mode 10) Datalink degC Status_enum: (lesp_tcrtd_in_status_enum) 0 – NotAvailable 1 – Error 2 – Open 3 – Valid
[In] Turbo1CompressorIntakeTemp	TcRtdInBus	Input that requires the Turbocharger #1 Compressor Side Intake Temperature input signal. TcRtdInBus: - Mode_enum - Status_enum - PullupResistorValue_ohms

		- Value Mode_enum: (lesp_tcrtd_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance (Mode 9) Thermocouple (Mode 10) Datalink degC Status_enum: (lesp_tcrtd_in_status_enum) 0 – NotAvailable 1 – Error 2 – Open 3 – Valid
[In] Turbo2CompressorIntakeTemp	TcRtdInBus	Input that requires the Turbocharger #2 Compressor Side Intake Temperature input signal. TcRtdInBus: - Mode_enum - Status_enum - PullupResistorValue_ohms - Value Mode_enum: (lesp_tcrtd_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance (Mode 9) Thermocouple (Mode 10) Datalink degC Status_enum: (lesp_tcrtd_in_status_enum) 0 – NotAvailable 1 – Error 2 – Open 3 – Valid
[In] Turbo1TurbineIntakeTemp	TcRtdInBus	Input that requires the Turbocharger #1 Turbine Side Intake Temperature input signal. TcRtdInBus: - Mode_enum - Status_enum - PullupResistorValue_ohms - Value Mode_enum: (lesp_tcrtd_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V (Mode 6) RTD Low Impedance

		(Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance (Mode 9) Thermocouple (Mode 10) Datalink degC Status_enum: (lesp_tcrtd_in_status_enum) 0 – NotAvailable 1 – Error 2 – Open 3 – Valid
[In] Turbo2TurbineIntakeTemp	TcRtdInBus	Input that requires the Turbocharger #2 Turbine Side Intake Temperature input signal. TcRtdInBus: - Mode_enum - Status_enum - PullupResistorValue_ohms - Value Mode_enum: (lesp_tcrtd_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance (Mode 9) Thermocouple (Mode 10) Datalink degC Status_enum: (lesp_tcrtd_in_status_enum) 0 – NotAvailable 1 – Error 2 – Open 3 – Valid
[In] Turbo3TurbineIntakeTemp	TcRtdInBus	Input that requires the Turbocharger #3 Turbine Side Intake Temperature input signal. TcRtdInBus: - Mode_enum - Status_enum - PullupResistorValue_ohms - Value Mode_enum: (lesp_tcrtd_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance (Mode 9) Thermocouple (Mode 10) Datalink degC Status_enum: (lesp_tcrtd_in_status_enum)

		0 – NotAvailable 1 – Error 2 – Open 3 – Valid
[In] Turbo4TurbineIntakeTemp	TcRtdInBus	Input that requires the Turbocharger #4 Turbine Side Intake Temperature input signal. TcRtdInBus: - Mode_enum - Status_enum - PullupResistorValue_ohms - Value Mode_enum: (lesp_tcrtd_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance (Mode 9) Thermocouple (Mode 10) Datalink degC Status_enum: (lesp_tcrtd_in_status_enum) 0 – NotAvailable 1 – Error 2 – Open 3 – Valid
[In] AfterCoolerWaterInletTemp	TcRtdInBus	Input that requires the Water Inlet Temperature of the AfterCooler input signal. TcRtdInBus: - Mode_enum - Status_enum - PullupResistorValue_ohms - Value Mode_enum: (lesp_tcrtd_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance (Mode 9) Thermocouple (Mode 10) Datalink degC Status_enum: (lesp_tcrtd_in_status_enum) 0 – NotAvailable 1 – Error 2 – Open 3 – Valid
[In] ExhaustGasTemp1	TcRtdInBus	Input that requires the Exhaust Gas Temperature of Cylinder #1 input signal.

		TcRtdInBus: - Mode_enum - Status_enum - PullupResistorValue_ohms - Value Mode_enum: (lesp_tcrtd_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance (Mode 9) Thermocouple (Mode 10) Datalink degC Status_enum: (lesp_tcrtd_in_status_enum) 0 – NotAvailable 1 – Error 2 – Open 3 – Valid
[In] ExhaustGasTemp2	TcRtdInBus	Input that requires the Exhaust Gas Temperature of Cylinder #2 input signal. TcRtdInBus: - Mode_enum - Status_enum - PullupResistorValue_ohms - Value Mode_enum: (lesp_tcrtd_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance (Mode 9) Thermocouple (Mode 10) Datalink degC Status_enum: (lesp_tcrtd_in_status_enum) 0 – NotAvailable 1 – Error 2 – Open 3 – Valid
[In] ExhaustGasTemp3	TcRtdInBus	Input that requires the Exhaust Gas Temperature of Cylinder #3 input signal. TcRtdInBus: - Mode_enum - Status_enum - PullupResistorValue_ohms - Value Mode_enum:

		(lesp_tcrtd_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance (Mode 9) Thermocouple (Mode 10) Datalink degC Status_enum: (lesp_tcrtd_in_status_enum) 0 – NotAvailable 1 – Error 2 – Open 3 – Valid
[In] ExhaustGasTemp4	TcRtdInBus	Input that requires the Exhaust Gas Temperature of Cylinder #4 input signal. TcRtdInBus: - Mode_enum - Status_enum - PullupResistorValue_ohms - Value Mode_enum: (lesp_tcrtd_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance (Mode 9) Thermocouple (Mode 10) Datalink degC Status_enum: (lesp_tcrtd_in_status_enum) 0 – NotAvailable 1 – Error 2 – Open 3 – Valid
[In] ExhaustGasTemp5	TcRtdInBus	Input that requires the Exhaust Gas Temperature of Cylinder #5 input signal. TcRtdInBus: - Mode_enum - Status_enum - PullupResistorValue_ohms - Value Mode_enum: (lesp_tcrtd_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance (Mode 9) Thermocouple

		(Mode 10) Datalink degC Status_enum: (lesp_tcrtd_in_status_enum) 0 – NotAvailable 1 – Error 2 – Open 3 – Valid
[In] ExhaustGasTemp6	TcRtdInBus	Input that requires the Exhaust Gas Temperature of Cylinder #6 input signal. TcRtdInBus: - Mode_enum - Status_enum - PullupResistorValue_ohms - Value Mode_enum: (lesp_tcrtd_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance (Mode 9) Thermocouple (Mode 10) Datalink degC Status_enum: (lesp_tcrtd_in_status_enum) 0 – NotAvailable 1 – Error 2 – Open 3 – Valid
[In] ExhaustGasTemp7	TcRtdInBus	Input that requires the Exhaust Gas Temperature of Cylinder #7 input signal. TcRtdInBus: - Mode_enum - Status_enum - PullupResistorValue_ohms - Value Mode_enum: (lesp_tcrtd_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance (Mode 9) Thermocouple (Mode 10) Datalink degC Status_enum: (lesp_tcrtd_in_status_enum) 0 – NotAvailable 1 – Error 2 – Open

		3 – Valid
[In] ExhaustGasTemp8	TcRtdInBus	Input that requires the Exhaust Gas Temperature of Cylinder #8 input signal. TcRtdInBus: - Mode_enum - Status_enum - PullupResistorValue_ohms - Value Mode_enum: (lesp_tcrtd_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance (Mode 9) Thermocouple (Mode 10) Datalink degC Status_enum: (lesp_tcrtd_in_status_enum) 0 – NotAvailable 1 – Error 2 – Open 3 – Valid
[In] ExhaustGasTemp9	TcRtdInBus	Input that requires the Exhaust Gas Temperature of Cylinder #9 input signal. TcRtdInBus: - Mode_enum - Status_enum - PullupResistorValue_ohms - Value Mode_enum: (lesp_tcrtd_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance (Mode 9) Thermocouple (Mode 10) Datalink degC Status_enum: (lesp_tcrtd_in_status_enum) 0 – NotAvailable 1 – Error 2 – Open 3 – Valid
[In] ExhaustGasTemp10	TcRtdInBus	Input that requires the Exhaust Gas Temperature of Cylinder #10 input signal. TcRtdInBus:

		- Mode_enum - Status_enum - PullupResistorValue_ohms - Value Mode_enum: (lesp_tcrtd_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance (Mode 9) Thermocouple (Mode 10) Datalink degC Status_enum: (lesp_tcrtd_in_status_enum) 0 – NotAvailable 1 – Error 2 – Open 3 – Valid
[In] ExhaustGasTemp11	TcRtdInBus	Input that requires the Exhaust Gas Temperature of Cylinder #11 input signal. TcRtdInBus: - Mode_enum - Status_enum - PullupResistorValue_ohms - Value Mode_enum: (lesp_tcrtd_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance (Mode 9) Thermocouple (Mode 10) Datalink degC Status_enum: (lesp_tcrtd_in_status_enum) 0 – NotAvailable 1 – Error 2 – Open 3 – Valid
[In] ExhaustGasTemp12	TcRtdInBus	Input that requires the Exhaust Gas Temperature of Cylinder #12 input signal. TcRtdInBus: - Mode_enum - Status_enum - PullupResistorValue_ohms - Value Mode_enum: (lesp_tcrtd_in_mode_enum)

		(Mode 0) Undefined (Mode 1) 0-5V (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance (Mode 9) Thermocouple (Mode 10) Datalink degC Status_enum: (lesp_tcrtd_in_status_enum) 0 – NotAvailable 1 – Error 2 – Open 3 – Valid
[In] ExhaustGasTemp13	TcRtdInBus	Input that requires the Exhaust Gas Temperature of Cylinder #13 input signal. TcRtdInBus: - Mode_enum - Status_enum - PullupResistorValue_ohms - Value Mode_enum: (lesp_tcrtd_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance (Mode 9) Thermocouple (Mode 10) Datalink degC Status_enum: (lesp_tcrtd_in_status_enum) 0 – NotAvailable 1 – Error 2 – Open 3 – Valid
[In] ExhaustGasTemp14	TcRtdInBus	Input that requires the Exhaust Gas Temperature of Cylinder #14 input signal. TcRtdInBus: - Mode_enum - Status_enum - PullupResistorValue_ohms - Value Mode_enum: (lesp_tcrtd_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance (Mode 9) Thermocouple (Mode 10) Datalink degC

		Status_enum: (lesp_tcrtd_in_status_enum) 0 – NotAvailable 1 – Error 2 – Open 3 – Valid
[In] ExhaustGasTemp15	TcRtdInBus	Input that requires the Exhaust Gas Temperature of Cylinder #15 input signal. TcRtdInBus: - Mode_enum - Status_enum - PullupResistorValue_ohms - Value Mode_enum: (lesp_tcrtd_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance (Mode 9) Thermocouple (Mode 10) Datalink degC Status_enum: (lesp_tcrtd_in_status_enum) 0 – NotAvailable 1 – Error 2 – Open 3 – Valid
[In] ExhaustGasTemp16	TcRtdInBus	Input that requires the Exhaust Gas Temperature of Cylinder #16 input signal. TcRtdInBus: - Mode_enum - Status_enum - PullupResistorValue_ohms - Value Mode_enum: (lesp_tcrtd_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance (Mode 9) Thermocouple (Mode 10) Datalink degC Status_enum: (lesp_tcrtd_in_status_enum) 0 – NotAvailable 1 – Error 2 – Open 3 – Valid

[In] ExhaustGasTemp17	TcRtdInBus	Input that requires the Exhaust Gas Temperature of Cylinder #17 input signal. TcRtdInBus: - Mode_enum - Status_enum - PullupResistorValue_ohms - Value Mode_enum: (lesp_tcrtd_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance (Mode 9) Thermocouple (Mode 10) Datalink degC Status_enum: (lesp_tcrtd_in_status_enum) 0 – NotAvailable 1 – Error 2 – Open 3 – Valid
[In] ExhaustGasTemp18	TcRtdInBus	Input that requires the Exhaust Gas Temperature of Cylinder #18 input signal. TcRtdInBus: - Mode_enum - Status_enum - PullupResistorValue_ohms - Value Mode_enum: (lesp_tcrtd_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance (Mode 9) Thermocouple (Mode 10) Datalink degC Status_enum: (lesp_tcrtd_in_status_enum) 0 – NotAvailable 1 – Error 2 – Open 3 – Valid
[In] ExhaustGasTemp19	TcRtdInBus	Input that requires the Exhaust Gas Temperature of Cylinder #19 input signal. TcRtdInBus: - Mode_enum - Status_enum

		- PullupResistorValue_ohms - Value Mode_enum: (lesp_tcrtd_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance (Mode 9) Thermocouple (Mode 10) Datalink degC Status_enum: (lesp_tcrtd_in_status_enum) 0 – NotAvailable 1 – Error 2 – Open 3 – Valid
[In] ExhaustGasTemp20	TcRtdInBus	Input that requires the Exhaust Gas Temperature of Cylinder #20 input signal. TcRtdInBus: - Mode_enum - Status_enum - PullupResistorValue_ohms - Value Mode_enum: (lesp_tcrtd_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance (Mode 9) Thermocouple (Mode 10) Datalink degC Status_enum: (lesp_tcrtd_in_status_enum) 0 – NotAvailable 1 – Error 2 – Open 3 – Valid
[In] HPFuelPump1HousingTemp	TcRtdInBus	Input that requires the High Pressure Pump #1 Housing Temperature input signal. TcRtdInBus: - Mode_enum - Status_enum - PullupResistorValue_ohms - Value Mode_enum: (lesp_tcrtd_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V

		(Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance (Mode 9) Thermocouple (Mode 10) Datalink degC Status_enum: (lesp_tcrtd_in_status_enum) 0 – NotAvailable 1 – Error 2 – Open 3 – Valid
[In] HPFuelPump2HousingTemp	TcRtdInBus	Input that requires the High Pressure Pump #2 Housing Temperature input signal. TcRtdInBus: - Mode_enum - Status_enum - PullupResistorValue_ohms - Value Mode_enum: (lesp_tcrtd_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance (Mode 9) Thermocouple (Mode 10) Datalink degC Status_enum: (lesp_tcrtd_in_status_enum) 0 – NotAvailable 1 – Error 2 – Open 3 – Valid
[In] GasTemp	TcRtdInBus	Input that requires the Gas Temperature input signal. TcRtdInBus: - Mode_enum - Status_enum - PullupResistorValue_ohms - Value Mode_enum: (lesp_tcrtd_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance (Mode 9) Thermocouple (Mode 10) Datalink degC Status_enum: (lesp_tcrtd_in_status_enum)

		0 – NotAvailable 1 – Error 2 – Open 3 – Valid
[In] LocalColdJct	AnalogSensorBus	Input that requires the Local Cold Junction Temperature input signal. AnalogSensorBus: - FilteredValue - MeasurementStatus_enum MeasurementStatus_enum: (lesp_measurement_status_enum) 0 – Valid 1 – Disabled 2 – SignalLow 3 – SignalHigh 4 – SensorSupplyFailed 5 – ReferenceSourceFailed 6 – OpenCircuit
[In] PWRDLY	boolean	This input requires the status of the PowerOn Delay that is required for some initialization logic.
[In] Reset	boolean	Input that requires the application reset and connected.
[Out] Temperature	Bus	Output that contains the following signals, all type (AnalogSensorBus): - EngineCoolantTemp1 - EngineCoolantTemp2 - EngineCoolantTemp - LubeOilFilterOutletTemp - AmbientAirTemp - ManifoldAirTemp1 - ManifoldAirTemp2 - ManifoldAirTemp - HPFuelPumpInletFuelTemp - FuelRailTemp - InjectorReturnFuelTemp - Turbo1CompressorIntakeTemp - Turbo2CompressorIntakeTemp - Turbo1TurbineIntakeTemp - Turbo2TurbineIntakeTemp - Turbo3TurbineIntakeTemp - Turbo4TurbineIntakeTemp - AfterCoolerWaterInletTemp - ExhaustGasTemp1 - ExhaustGasTemp2 - ExhaustGasTemp3 - ExhaustGasTemp4 - ExhaustGasTemp5 - ExhaustGasTemp6 - ExhaustGasTemp7 - ExhaustGasTemp8 - ExhaustGasTemp9 - ExhaustGasTemp10 - ExhaustGasTemp11 - ExhaustGasTemp12 - ExhaustGasTemp13

		- ExhaustGasTemp14 - ExhaustGasTemp15 - ExhaustGasTemp16 - ExhaustGasTemp17 - ExhaustGasTemp18 - ExhaustGasTemp19 - ExhaustGasTemp20 - HPFuelPump1HousingTemp - HPFuelPump2HousingTemp - GasTemp Bus Details: AnalogSensorBus: - FilteredValue - MeasurementStatus_enum MeasurementStatus_enum: (lesp_measurement_status_enum) 0 – Valid 1 – Disabled 2 – SignalLow 3 – SignalHigh 4 – SensorSupplyFailed 5 – ReferenceSourceFailed 6 – OpenCircuit

Example Model Using the Blocks

Below is a very simple setup using MotoHawk Calibration and MotoHawk Probes blocks to make the model suitable for building.

Figure 10-8. Simple Example Model, Using the Sensors - CRS Interface Blocks

The required files will generate while compiling, and if the Toolkit required blocks are also added to the application model, it will generate the required SID files for the Toolkit HMI tool creation.

Default ToolKit Pages

Opening a new Toolkit application and selecting the correctly generated .sid file will look like the screenshot below.

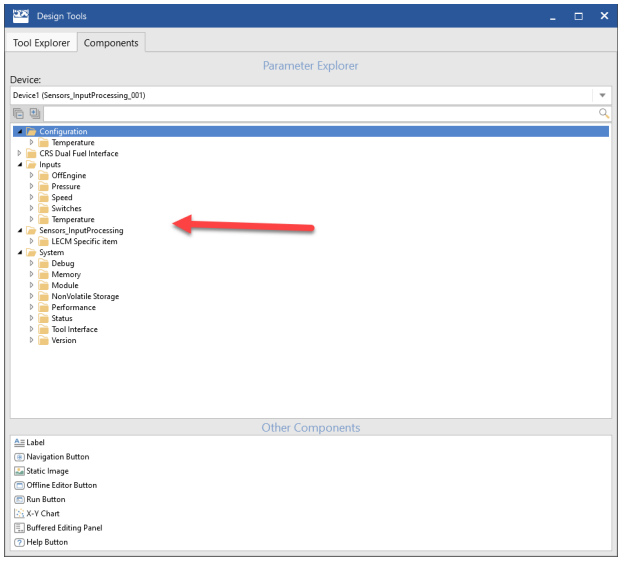
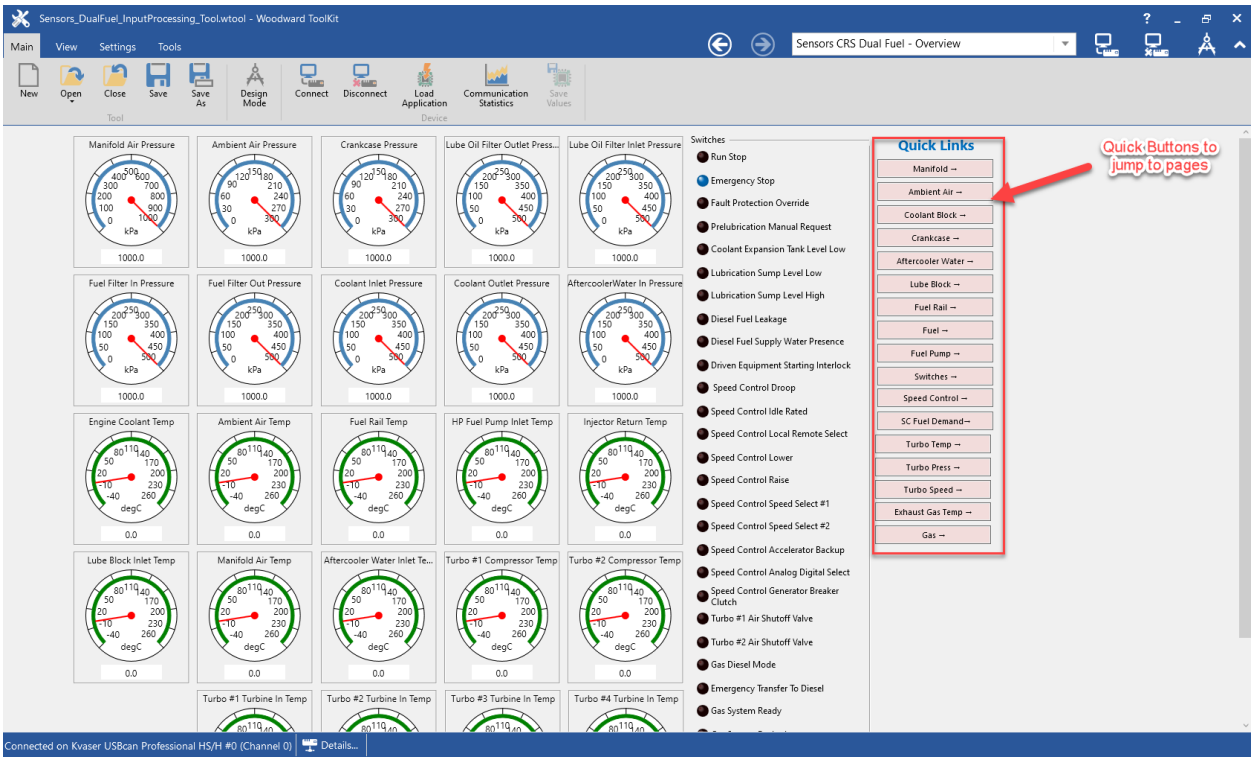
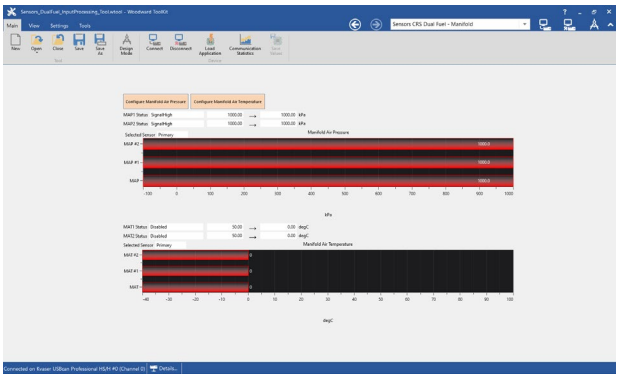
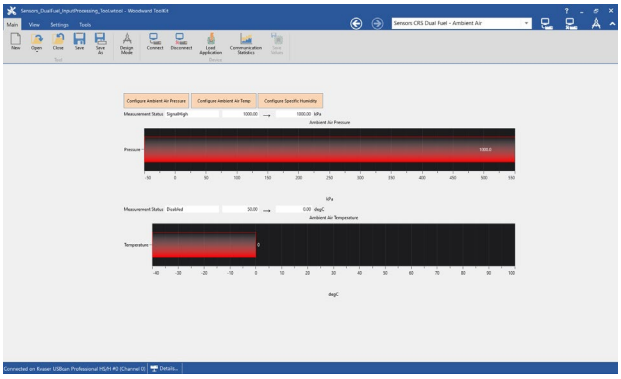


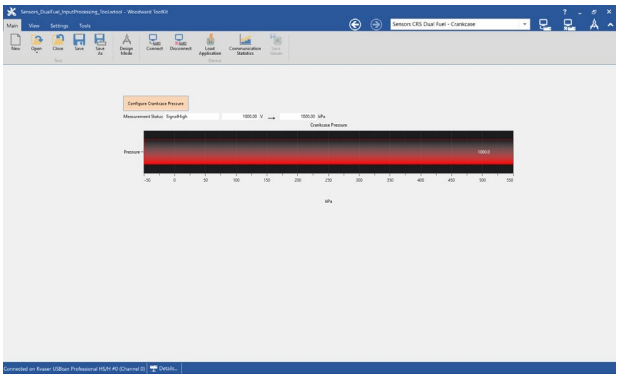
Figure 10-9. SID File Selection Sensors – CRS Dual Fuel Interface Logic

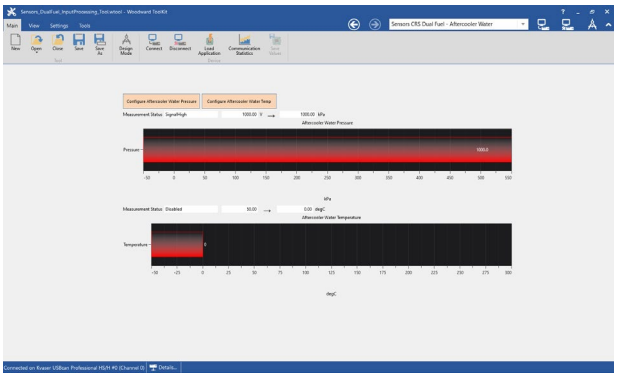
Below are a few of the Toolkit pages that will contain most of the defined parameters in the example model. Of course, the real sensors and data must be attached to complete it for the application that is being worked on. For that reason, red X's appear on some pages.

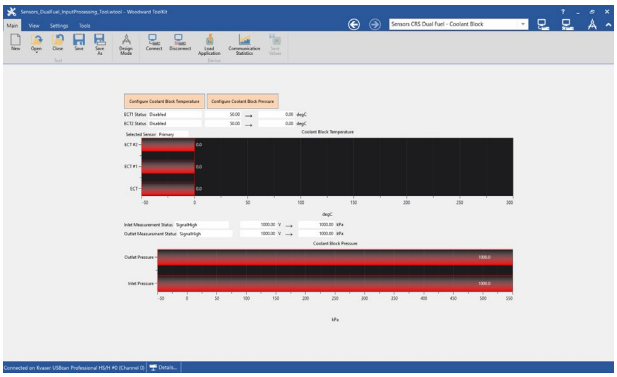


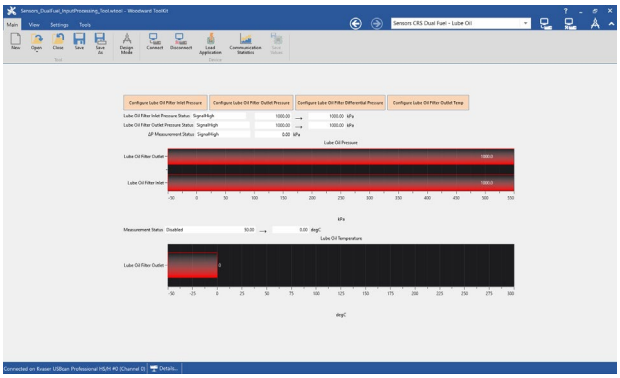


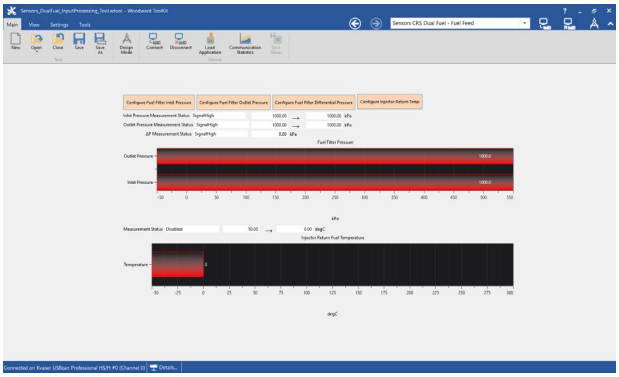












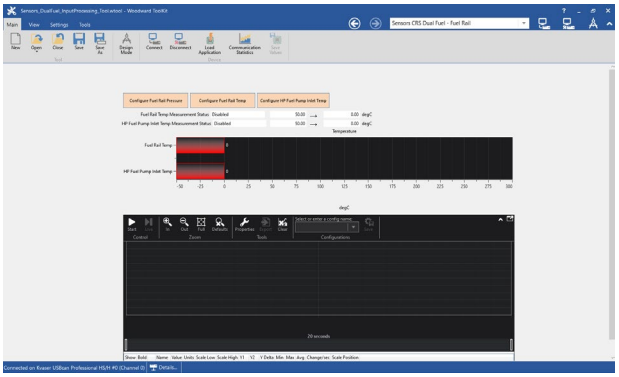


Figure 10-10. Examples of the Sensors-Input CRS Dual Fuel Toolkit Pages

Chapter 11.
Sensors – Input Processing - CRS

Introduction

The ‘Sensors – Input Processing CRS’ blockset is used for Common Rail System (CRS) sensing purposes. Several blocks are available like pressure sensing, speed sensing, discrete switch sensing, temperature sensing and off-engine sensing.

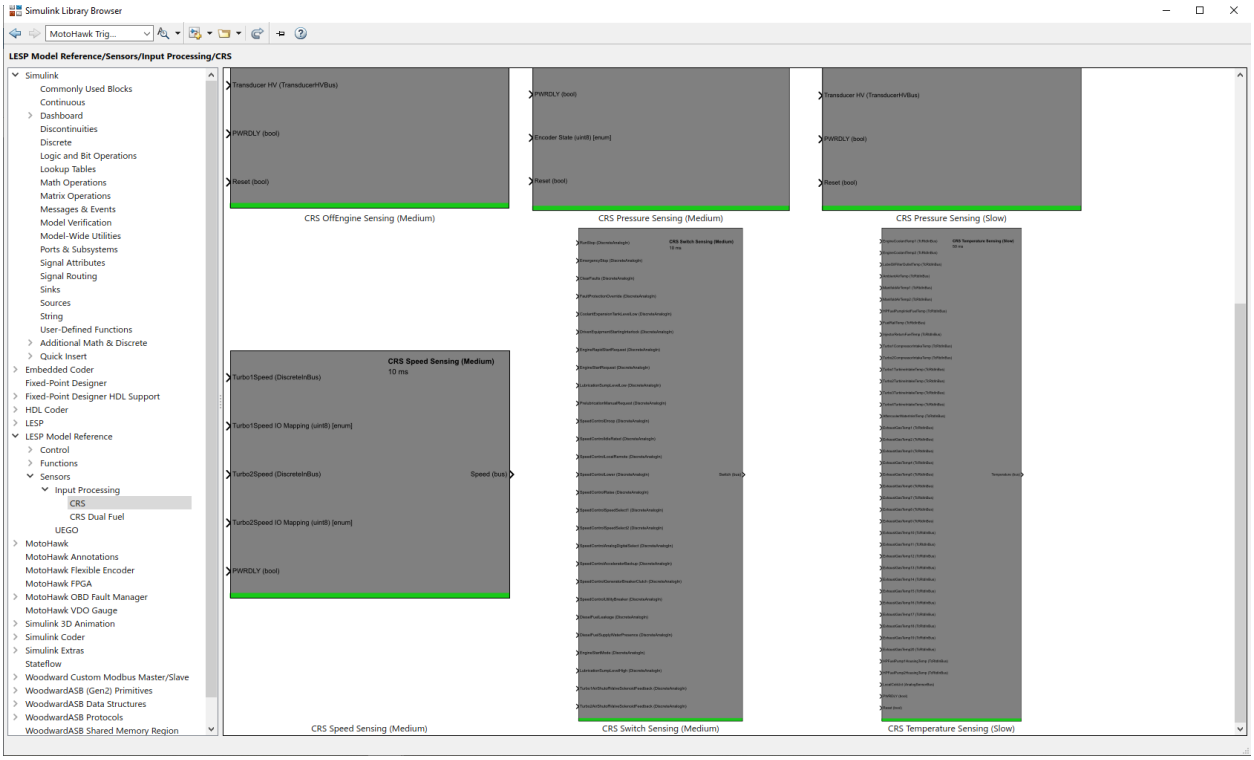


Figure 11-1. LESP Model Reference ‘Sensors – CRS Input Processing’ Block

- CRS OffEngine Sensing (Medium)
  This block can be used to process the Off-Engine Sensor Signals to be defined inside the application. It is called (Medium) because it is processed in 10 ms rate.

- CRS Pressure Sensing (Medium)
  This block can be used to process the Pressure Sensor Signals to be defined inside the application. It is called (Medium) because it is processed in 10 ms rate.

- CRS Pressure Sensing (Slow)
  This block can be used to process the Pressure Sensor Signals to be defined inside the application. It is called (Slow) because it is processed in 50 ms rate.

- CRS Speed Sensing (Medium)
  This block can be used to process the Speed Sensor Signals to be defined inside the application. It is called (Medium) because it is processed in 10 ms rate.

- CRS Switch Sensing (Medium)
  This block can be used to process the Discrete Switch Sensor Signals to be defined inside the application. It is called (Medium) because it is processed in 10 ms rate.

- CRS Temperature Sensing (Slow)
  This block can be used to process the Temperature Sensor Signals to be defined inside the application. It is called (Medium) because it is processed in 50 ms rate.

CRS OffEngine Sensing Block Interface

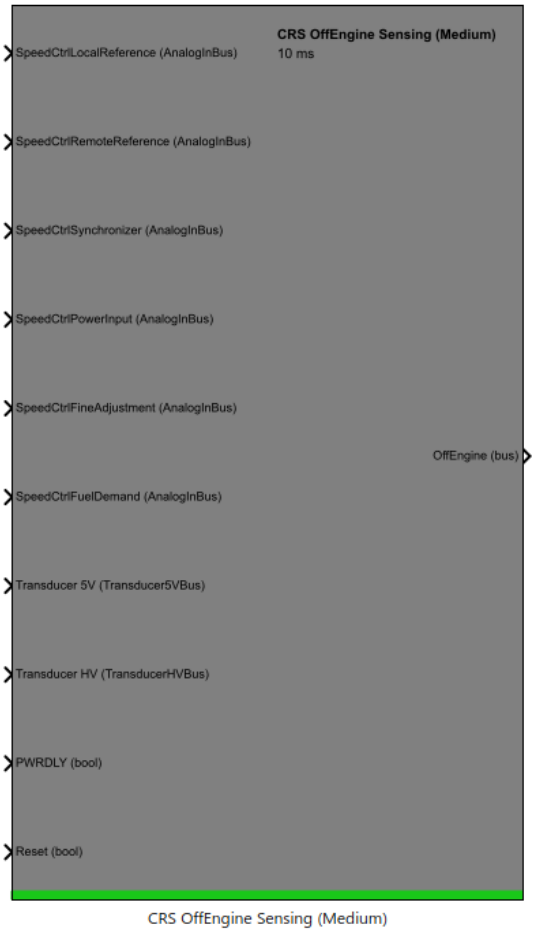


Figure 11-2. LESP Model Reference ‘Sensors – CRS OffEngine Sensing’ Block

Table 11-1. ‘CRS OffEngine Sensing (Medium)’ Block

Port	DataType	Description
[In] SpeedCtrlLocalReference	AnalogInBus	Input that requires the Speed Control Local Speed Reference setpoint for the application. This is normally an analog speed setting that comes from local position (Engine Control Room). Bus Details: AnalogInBus: - Mode_enum - Value - ADC - ADCToHWUnitsGain - ADCToHWUnitsOffset - PullupResistorValue_ohms Mode_enum: (lesp_analog_in_mode_enum)

		(Mode 0) Undefined (Mode 1) 0-5V (Mode 2) 0-1.25V (Mode 3) +-1V (Mode 4) +-2.5V (Mode 5) 4-20mA (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance (Mode 9) Thermocouple
[In] SpeedCtrlRemoteReference	AnalogInBus	Input that requires the Speed Control Remote Speed Reference setpoint for the application. This is normally an analog speed setting that comes from remote position (Bridge Control). Bus Details: AnalogInBus: - Mode_enum - Value - ADC - ADCToHWUnitsGain - ADCToHWUnitsOffset - PullupResistorValue_ohms Mode_enum: (lesp_analog_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V (Mode 2) 0-1.25V (Mode 3) +-1V (Mode 4) +-2.5V (Mode 5) 4-20mA (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance (Mode 9) Thermocouple
[In] SpeedCtrlSynchronizer	AnalogInBus	Input that requires the Speed Control Synchronizer setpoint for the application. This is a bias signal that comes from a synchronizer device, used to increase/decrease engine speed to synchronize with other Gensets that are coupled to an electric network. Bus Details: AnalogInBus: - Mode_enum - Value - ADC - ADCToHWUnitsGain - ADCToHWUnitsOffset - PullupResistorValue_ohms Mode_enum: (lesp_analog_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V (Mode 2) 0-1.25V (Mode 3) +-1V (Mode 4) +-2.5V (Mode 5) 4-20mA

		(Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance (Mode 9) Thermocouple
[In] SpeedCtrlPowerInput	AnalogInBus	Input that requires the Speed Control Power Input / kW input for the application. This is signal that corresponds with the actual load measured (mostly a kW transducer for Gensets). Bus Details: AnalogInBus: - Mode_enum - Value - ADC - ADCToHWUnitsGain - ADCToHWUnitsOffset - PullupResistorValue_ohms Mode_enum: (lesp_analog_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V (Mode 2) 0-1.25V (Mode 3) +-1V (Mode 4) +-2.5V (Mode 5) 4-20mA (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance (Mode 9) Thermocouple
[In] SpeedCtrlFineAdjustment	AnalogInBus	Input that requires the Speed Control Fine Adjustment setpoint for the application. This is a bias signal that comes from an external device, used to increase/decrease engine speed to fine tune the speed. Sometimes this is required during production validation testing. Bus Details: AnalogInBus: - Mode_enum - Value - ADC - ADCToHWUnitsGain - ADCToHWUnitsOffset - PullupResistorValue_ohms Mode_enum: (lesp_analog_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V (Mode 2) 0-1.25V (Mode 3) +-1V (Mode 4) +-2.5V (Mode 5) 4-20mA (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance (Mode 9) Thermocouple

[In] SpeedCtrlFuelDemand	AnalogInBus	Input that requires the Speed Control Fuel Demand for the application, if an external Speed Control Device is used. It will correspond with the output signal of the external speed control. Bus Details: AnalogInBus: - Mode_enum - Value - ADC - ADCToHWUnitsGain - ADCToHWUnitsOffset - PullupResistorValue_ohms Mode_enum: (lesp_analog_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V (Mode 2) 0-1.25V (Mode 3) +-1V (Mode 4) +-2.5V (Mode 5) 4-20mA (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance (Mode 9) Thermocouple
[In] Transducer 5V	Transducer5VBus	Input that requires the Transducer 5 V feedback value. This can be used for calculating ratiometric correction for the analog input signals. Bus Details: Transducer5VBus: - FilteredValue_V - MeasurementStatus_enum - RatiometricCorrectionFactor MeasurementStatus_enum: (lesp_measurement_status_enum) 0 – Valid 1 – Disabled 2 – SignalLow 3 – SignalHigh 4 – SensorSupplyFailed 5 – ReferenceSourceFailed 6 – OpenCircuit
[In] Transducer HV	TransducerHVBus	Input that requires the High Voltage (HV) Transducer feedback value. Bus Details: TransducerHighVoltBus: - FilteredValue_V - MeasurementStatus_enum - RatiometricCorrectionFactor - VoltageSelect_enum MeasurementStatus_enum: (lesp_measurement_status_enum) 0 – Valid 1 – Disabled 2 – SignalLow 3 – SignalHigh 4 – SensorSupplyFailed

		5 – ReferenceSourceFailed 6 – OpenCircuit lesp_hv_transducer_select_enum: 0 – 12V 1 – 20V
[In] PWRDLY	boolean	This input requires the status of the PowerOn Delay that is required for some initialization logic.
[In] Reset	boolean	Input that requires the application reset and connected.
[Out] OffEngine	Bus	Output that contains the following signals, all of type (AnalogSensorbus): - SpeedCtrlLocalReference - SpeedCtrlRemoteReference - SpeedCtrlSynchronizer - SpeedCtrlPowerInput - SpeedCtrlFineAdjustment - SpeedCtrlFuelDemand Bus Details: AnalogSensorBus: - FilteredValue - MeasurementStatus_enum MeasurementStatus_enum: (lesp_measurement_status_enum) 0 – Valid 1 – Disabled 2 – SignalLow 3 – SignalHigh 4 – SensorSupplyFailed 5 – ReferenceSourceFailed 6 – OpenCircuit

CRS Pressure Sensing (Medium) Block Interface

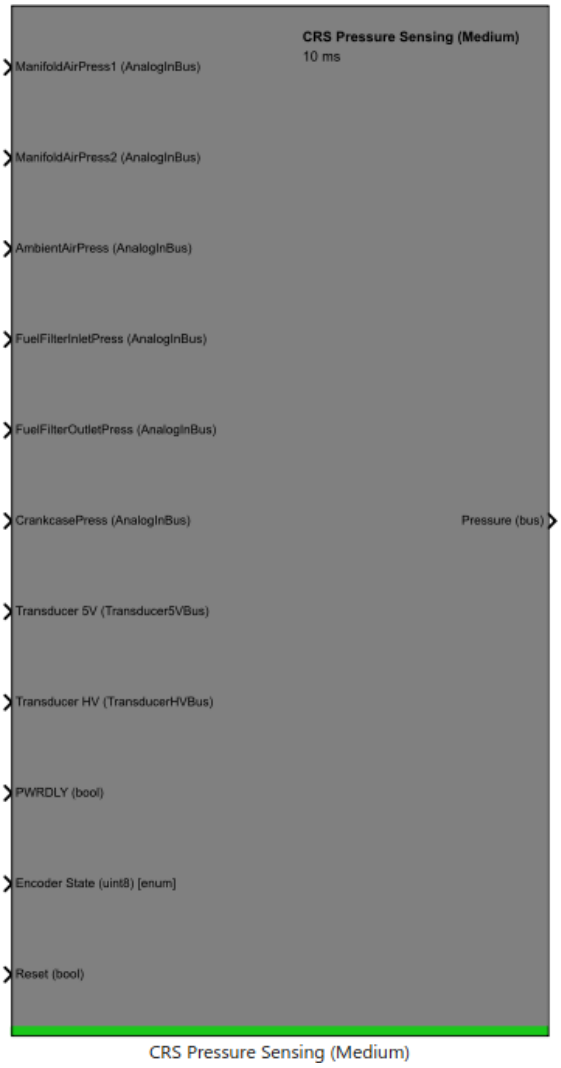


Figure 11-3. LESP Model Reference ‘Sensors – CRS Pressure Sensing (Medium)’ Block

Table 11-2. ‘CRS Pressure Sensing (Medium)’ Block

Port	DataType	Description
[In] ManifoldAirPress1	AnalogInBus	Input that requires the Engine Manifold Air Pressure Sensor #1 input signal. It is one of the redundant sensors, measuring actual engine manifold air pressure. Bus Details: AnalogInBus: - Mode_enum - Value - ADC - ADCToHWUnitsGain - ADCToHWUnitsOffset - PullupResistorValue_ohms Mode_enum: (lesp_analog_in_mode_enum) (Mode 0) Undefined

		(Mode 1) 0-5V (Mode 2) 0-1.25V (Mode 3) +-1V (Mode 4) +-2.5V (Mode 5) 4-20mA (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance
[In] ManifoldAirPress2	AnalogInBus	Input that requires the Engine Manifold Air Pressure Sensor #2 input signal. It is one of the redundant sensors, measuring actual engine manifold air pressure. Bus Details: AnalogInBus: - Mode_enum - Value - ADC - ADCToHWUnitsGain - ADCToHWUnitsOffset - PullupResistorValue_ohms Mode_enum: (lesp_analog_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V (Mode 2) 0-1.25V (Mode 3) +-1V (Mode 4) +-2.5V (Mode 5) 4-20mA (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance
[In] AmbientAirPress	AnalogInBus	Input that requires the Ambient Air Pressure input signal. This signal measures the actual ambient air pressure, which is required for the logic and absolute/relative pressure calculations. Bus Details: AnalogInBus: - Mode_enum - Value - ADC - ADCToHWUnitsGain - ADCToHWUnitsOffset - PullupResistorValue_ohms Mode_enum: (lesp_analog_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V (Mode 2) 0-1.25V (Mode 3) +-1V (Mode 4) +-2.5V (Mode 5) 4-20mA (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance

[In] FuelFilterInletPress	AnalogInBus	Input that requires the Fuel Filter Inlet Pressure input signal. This signal is the fuel pressure from tank to filter inlet. Bus Details: AnalogInBus: - Mode_enum - Value - ADC - ADCToHWUnitsGain - ADCToHWUnitsOffset - PullupResistorValue_ohms Mode_enum: (lesp_analog_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V (Mode 2) 0-1.25V (Mode 3) +-1V (Mode 4) +-2.5V (Mode 5) 4-20mA (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance
[In] FuelFilterOutletPress	AnalogInBus	Input that requires the Fuel Filter Outlet Pressure input signal. This signal is the fuel pressure going to the engine. Bus Details: AnalogInBus: - Mode_enum - Value - ADC - ADCToHWUnitsGain - ADCToHWUnitsOffset - PullupResistorValue_ohms Mode_enum: (lesp_analog_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V (Mode 2) 0-1.25V (Mode 3) +-1V (Mode 4) +-2.5V (Mode 5) 4-20mA (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance
[In] CrankCasePress	AnalogInBus	Input that requires the Crank Case Pressure input signal. This sensor is required to monitor excessive crankcase pressure, which can lead to dangerous situations. Bus Details: AnalogInBus: - Mode_enum - Value - ADC - ADCToHWUnitsGain - ADCToHWUnitsOffset - PullupResistorValue_ohms

		Mode_enum: (lesp_analog_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V (Mode 2) 0-1.25V (Mode 3) +-1V (Mode 4) +-2.5V (Mode 5) 4-20mA (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance
[In] Transducer 5V	Transducer5VBus	Input that requires Transducer 5 V feedback value. This can be used for calculating ratiometric correction for the analog input signals. Bus Details: Transducer5VBus: - FilteredValue_V - MeasurementStatus_enum - RatiometricCorrectionFactor MeasurementStatus_enum: (lesp_measurement_status_enum) 0 – Valid 1 – Disabled 2 – SignalLow 3 – SignalHigh 4 – SensorSupplyFailed 5 – ReferenceSourceFailed 6 – OpenCircuit
[In] Transducer HV	TransducerHVBus	Input that requires the High Voltage (HV) Transducer feedback value. Bus Details: TransducerHighVoltBus: - FilteredValue_V - MeasurementStatus_enum - RatiometricCorrectionFactor - VoltageSelect_enum MeasurementStatus_enum: (lesp_measurement_status_enum) 0 – Valid 1 – Disabled 2 – SignalLow 3 – SignalHigh 4 – SensorSupplyFailed 5 – ReferenceSourceFailed 6 – OpenCircuit lesp_hv_transducer_select_enum: 0 – 12V 1 – 20V
[In] PWRDLY	boolean	This input requires the status of the PowerOn Delay that is required for some initialization logic.
[In] EncoderState (enum)	uint8	This input requires the EncoderState enum input from the encoder logic as defined in the application. lesp_encoder_state_enum: 0 – Not Created

		1 – ZeroSpeed 2 – Rotating 3 – Partial Synchronization 4 – Full Synchronization Pending 5 – Full Synchronization
[In] Reset	boolean	Input that requires the application reset and connected.
[Out] Pressure	Bus	Output that contains the following signals, all of type (AnalogSensorbus): - ManifoldAirPress1 - ManifoldAirPress2 - ManifoldAirPress - AmbientAirPress - FuelFilterInletPress - FuelFilterOutletPress - FuelFilterDifferentialPress - CrankcasePress Bus Details: AnalogSensorBus: - FilteredValue - MeasurementStatus_enum MeasurementStatus_enum: (lesp_measurement_status_enum) 0 – Valid 1 – Disabled 2 – SignalLow 3 – SignalHigh 4 – SensorSupplyFailed 5 – ReferenceSourceFailed 6 – OpenCircuit

CRS Pressure Sensing (Slow) Block Interface

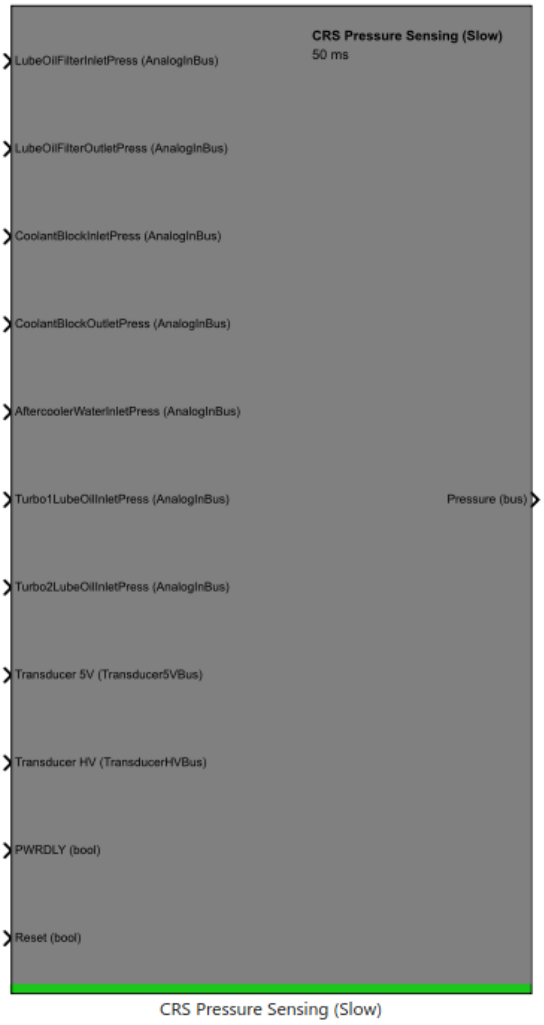


Figure 11-4. LESP Model Reference ‘Sensors – CRS Pressure Sensing (Slow)’ Block

Table 11-3. ‘CRS Pressure Sensing (Slow)’ Block

Port	DataType	Description
[In] LubeOilFilterInletPress	AnalogInBus	Input that requires the LubeOil Filter Inlet Pressure input signal. This is the pressure before/at LubeOil filter inlet. Bus Details: AnalogInBus: - Mode_enum - Value - ADC - ADCToHWUnitsGain - ADCToHWUnitsOffset - PullupResistorValue_ohms Mode_enum: (lesp_analog_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V (Mode 2) 0-1.25V

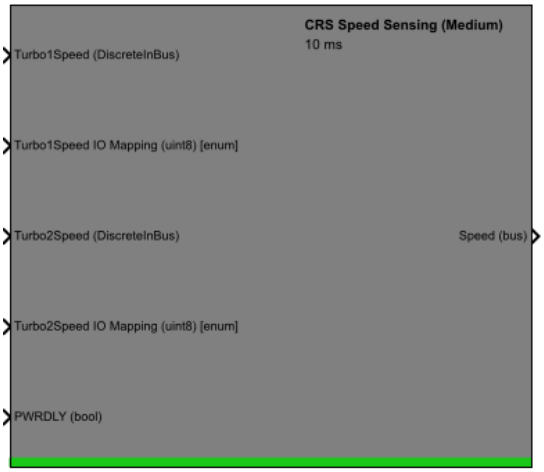
		(Mode 3) +-1V (Mode 4) +-2.5V (Mode 5) 4-20mA (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance
[In] LubeOilFilterOutletPress	AnalogInBus	Input that requires the LubeOil Filter Outlet Pressure input signal. This is the pressure after LubeOil filter at engine LubeOil inlet. Bus Details: AnalogInBus: - Mode_enum - Value - ADC - ADCToHWUnitsGain - ADCToHWUnitsOffset - PullupResistorValue_ohms Mode_enum: (lesp_analog_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V (Mode 2) 0-1.25V (Mode 3) +-1V (Mode 4) +-2.5V (Mode 5) 4-20mA (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance
[In] CoolantBlockInletPress	AnalogInBus	Input that requires the Coolant Block Inlet Pressure input signal. This is the coolant pressure at Engine Block Inlet. Bus Details: AnalogInBus: - Mode_enum - Value - ADC - ADCToHWUnitsGain - ADCToHWUnitsOffset - PullupResistorValue_ohms Mode_enum: (lesp_analog_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V (Mode 2) 0-1.25V (Mode 3) +-1V (Mode 4) +-2.5V (Mode 5) 4-20mA (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance
[In] CoolantBlockOutletPress	AnalogInBus	Input that requires the Coolant Block Outlet Pressure input signal. This is the coolant pressure at Engine Block Outlet.

		Bus Details: AnalogInBus: - Mode_enum - Value - ADC - ADCToHWUnitsGain - ADCToHWUnitsOffset - PullupResistorValue_ohms Mode_enum: (lesp_analog_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V (Mode 2) 0-1.25V (Mode 3) +-1V (Mode 4) +-2.5V (Mode 5) 4-20mA (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance
[In] AfterCoolerWaterInletPress	AnalogInBus	Input that requires the AfterCooler Water Inlet Pressure input signal. This is the water inlet pressure at AfterCooler inlet. Bus Details: AnalogInBus: - Mode_enum - Value - ADC - ADCToHWUnitsGain - ADCToHWUnitsOffset - PullupResistorValue_ohms Mode_enum: (lesp_analog_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V (Mode 2) 0-1.25V (Mode 3) +-1V (Mode 4) +-2.5V (Mode 5) 4-20mA (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance
[In] Turbo1LubeOilInletPress	AnalogInBus	Input that requires the TurboCharger #1 LubeOil Inlet Pressure input signal. This is the LubeOil inlet pressure for the TC#1. Bus Details: AnalogInBus: - Mode_enum - Value - ADC - ADCToHWUnitsGain - ADCToHWUnitsOffset - PullupResistorValue_ohms Mode_enum: (lesp_analog_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V

		(Mode 2) 0-1.25V (Mode 3) +-1V (Mode 4) +-2.5V (Mode 5) 4-20mA (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance
[In] Turbo2LubeOilInletPress	AnalogInBus	Input that requires the TurboCharger #2 LubeOil Inlet Pressure input signal. This is the LubeOil inlet pressure for the TC#2. Bus Details: AnalogInBus: - Mode_enum - Value - ADC - ADCToHWUnitsGain - ADCToHWUnitsOffset - PullupResistorValue_ohms Mode_enum: (lesp_analog_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V (Mode 2) 0-1.25V (Mode 3) +-1V (Mode 4) +-2.5V (Mode 5) 4-20mA (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance
[In] Transducer 5V	Transducer5VBus	Input that requires the Transducer 5 V feedback value. This can be used for calculating ratiometric correction for the analog input signals. Bus Details: Transducer5VBus: - FilteredValue_V - MeasurementStatus_enum - RatiometricCorrectionFactor MeasurementStatus_enum: (lesp_measurement_status_enum) 0 – Valid 1 – Disabled 2 – SignalLow 3 – SignalHigh 4 – SensorSupplyFailed 5 – ReferenceSourceFailed 6 – OpenCircuit
[In] Transducer HV	TransducerHVBus	Input that requires the High Voltage (HV) Transducer feedback value. Bus Details: TransducerHighVoltBus: - FilteredValue_V - MeasurementStatus_enum - RatiometricCorrectionFactor - VoltageSelect_enum MeasurementStatus_enum:

		(lesp_measurement_status_enum) 0 – Valid 1 – Disabled 2 – SignalLow 3 – SignalHigh 4 – SensorSupplyFailed 5 – ReferenceSourceFailed 6 – OpenCircuit lesp_hv_transducer_select_enum: 0 – 12V 1 – 20V
[In] PWRDLY	boolean	This input requires the status of the PowerOn Delay that is required for some initialization logic.
[In] Reset	boolean	Input that requires the application reset and connected.
[Out] Pressure	Bus	Output that contains the following signals, all of type (AnalogSensorbus): - LubeOilFilterInletPress - LubeOilFilterOutletPress - LubeOilFilterDifferentialPress - CoolantBlockInletPress - CoolantBlockOutletPress - AfterCoolerWaterInletPress - Turbo1LubeOilInletPress - Turbo2LubeOilInletPress Bus Details: AnalogSensorBus: - FilteredValue - MeasurementStatus_enum MeasurementStatus_enum: (lesp_measurement_status_enum) 0 – Valid 1 – Disabled 2 – SignalLow 3 – SignalHigh 4 – SensorSupplyFailed 5 – ReferenceSourceFailed 6 – OpenCircuit

CRS Speed Sensing (Medium) Block Interface



CRS Speed Sensing (Medium)

Figure 11-5. LESP Model Reference ‘Sensors – CRS Speed Sensing (Medium)’ Block

Table 11-4. ‘CRS Speed Sensing (Medium)’ Block

Port	DataType	Description
[In] Turbo1Speed	DiscreteInBus	Input that requires the TurboCharger #1 Speed input signal. This is the actual speed measured at the Turbocharger #1 wheel. Bus Details: DiscreteInBus: - DiscreteState - Frequency_Hz - DutyCycle_Pct
[In] Turbo1SpeedIOMapping (enum)	uint8	Input that requires the TurboCharger #1 Speed IOMapping signal. This is the IOMapping for the speed measurement at the Turbocharger #1 wheel. enum Details: (lecm_frequency_in_mapping_enum) 0 – Not Used 1 – MainSpeed1 2 – MainSpeed2 3 – MainSpeed3 4 – MainSpeed4 5 – EIDSpeed1 6 – EIDSpeed2 7 – AuxSpeed1 8 – AuxSpeed2
[In] Turbo2Speed	DiscreteInBus	Input that requires the TurboCharger #2 Speed input signal. This is the actual

		speed measured at the Turbocharger #2 wheel. Bus Details: DiscreteInBus: - DiscreteState - Frequency_Hz - DutyCycle_Pct
[In] Turbo2SpeedIOMapping (enum)	uint8	Input that requires the TurboCharger #2 Speed IOMapping signal. This is the IOMapping for the speed measurement at the Turbocharger #2 wheel. enum Details: (lecm_frequency_in_mapping_enum) 0 – Not Used 1 – MainSpeed1 2 – MainSpeed2 3 – MainSpeed3 4 – MainSpeed4 5 – EIDSpeed1 6 – EIDSpeed2 7 – AuxSpeed1 8 – AuxSpeed2
[In] PWRDLY	boolean	This input requires the status of the PowerOn Delay that is required for some initialization logic.
[Out] Speed	Bus	Output that contains the following signals, all type (AnalogSensorbus): - Turbo1Speed - Turbo2Speed Bus Details: AnalogSensorBus: - FilteredValue - MeasurementStatus_enum MeasurementStatus_enum: (lesp_measurement_status_enum) 0 – Valid 1 – Disabled 2 – SignalLow 3 – SignalHigh 4 – SensorSupplyFailed 5 – ReferenceSourceFailed 6 – OpenCircuit

CRS Switch Sensing (Medium) Block Interface

Figure 11-6. LESP Model Reference ‘Sensors – CRS Switch Sensing (Medium)’ Block

Table 11-5. ‘CRS Switch Sensing (Medium)’ Block

Port	DataType	Description
[In] RunStop	DiscreteAnalogIn	Input that requires the RunStop status feedback in the application. Run – Engine is ready to run and SpeedControl will be active. Stop – Engine will shut down and go into Stop state. DiscreteAnalogIn: - Type_enum

		- AnalogADC - DiscreteState Type_enum: (lesp_discrete_analog_in_mapping_type_enum) 0 – None 1 – Discrete 2 – Analog
[In] EmergencyStop	DiscreteAnalogIn	This input requires the Emergency Stop Input signal as defined in the application. Emergency Stop is a direct fuel shutdown to force the engine to stop. DiscreteAnalogIn: - Type_enum - AnalogADC - DiscreteState Type_enum: (lesp_discrete_analog_in_mapping_type_enum) 0 – None 1 – Discrete 2 – Analog
[In] ClearFaults	DiscreteAnalogIn	This input requires the Clear Faults Input signal as defined in the application. It is meant to clear any fault that might be active on the used channels. DiscreteAnalogIn: - Type_enum - AnalogADC - DiscreteState Type_enum: (lesp_discrete_analog_in_mapping_type_enum) 0 – None 1 – Discrete 2 – Analog
[In] FaultProtectionOverride	DiscreteAnalogIn	This input requires the Fault Protection Override Input signal as defined in the application. It is meant to override any shutdown in case of emergencies. This is typical for vessels as the ultimate rescue to save the vessel at any cost. DiscreteAnalogIn: - Type_enum - AnalogADC - DiscreteState Type_enum: (lesp_discrete_analog_in_mapping_type_enum) 0 – None 1 – Discrete 2 – Analog
[In] CoolantExpansionTankLevelLow	DiscreteAnalogIn	This input requires the Tank Level Low feedback from the coolant expansion tank. When the level drops below a certain minimum, this input will be activated.

		DiscreteAnalogIn: - Type_enum - AnalogADC - DiscreteState Type_enum: (lesp_discrete_analog_in_mapping_type_enum) 0 – None 1 – Discrete 2 – Analog
[In] DrivenEquipmentStartInterlock	DiscreteAnalogIn	This input requires the Start Interlock from any driven equipment. Examples are turning wheel engaged or any other engine maintenance blocking. When this is active, the engine must be blocked from any start attempt. DiscreteAnalogIn: - Type_enum - AnalogADC - DiscreteState Type_enum: (lesp_discrete_analog_in_mapping_type_enum) 0 – None 1 – Discrete 2 – Analog
[In] EngineRapidStartRequest	DiscreteAnalogIn	This input requires the Rapid Start Input signal as defined in the application. Rapid Start will override Pre-start sequencing and enables starting possibility directly. DiscreteAnalogIn: - Type_enum - AnalogADC - DiscreteState Type_enum: (lesp_discrete_analog_in_mapping_type_enum) 0 – None 1 – Discrete 2 – Analog
[In] EngineStartRequest	DiscreteAnalogIn	This input requires the Request for Engine Start in the application. When this input becomes active, the engine Start will be executed (when allowed in the application logic). DiscreteAnalogIn: - Type_enum - AnalogADC - DiscreteState Type_enum: (lesp_discrete_analog_in_mapping_type_enum) 0 – None 1 – Discrete 2 – Analog
[In] LubricationSumpLevelLow	DiscreteAnalogIn	This input requires the Low Level feedback from the lubrication oil sump.

		When the oil level drops below a certain minimum level, this input should be activated. DiscreteAnalogIn: - Type_enum - AnalogADC - DiscreteState Type_enum: (lesp_discrete_analog_in_mapping_type_enum) 0 – None 1 – Discrete 2 – Analog
[In] PreLubricationManualRequest	DiscreteAnalogIn	This input requires the manual request for starting the Pre-lubrication Oil Pump inside the application. DiscreteAnalogIn: - Type_enum - AnalogADC - DiscreteState Type_enum: (lesp_discrete_analog_in_mapping_type_enum) 0 – None 1 – Discrete 2 – Analog
[In] SpeedControlDroop	DiscreteAnalogIn	This input requires a selection if Droop is required in the application or not. When not selected, isochronous operation will be active. Droop is a reduction in speed reference when load is increased. This is used for stability and loadsharing purposes. DiscreteAnalogIn: - Type_enum - AnalogADC - DiscreteState Type_enum: (lesp_discrete_analog_in_mapping_type_enum) 0 – None 1 – Discrete 2 – Analog
[In] SpeedControlIdleRated	DiscreteAnalogIn	This input requires the Idle/Rated Speed Selection for the application. This will be active when Speedcontrol is in Digital Speed Settings mode. DiscreteAnalogIn: - Type_enum - AnalogADC - DiscreteState Type_enum: (lesp_discrete_analog_in_mapping_type_enum) 0 – None 1 – Discrete 2 – Analog

[In] SpeedControlLocalRemote	DiscreteAnalogIn	This input requires the selection input between Local and Remote Speed reference setpoint inside the application. This selection is active when speed control is in Analog Speed Setting mode. DiscreteAnalogIn: - Type_enum - AnalogADC - DiscreteState Type_enum: (lesp_discrete_analog_in_mapping_type_enum) 0 – None 1 – Discrete 2 – Analog
[In] SpeedControlLower	DiscreteAnalogIn	This input requires the Lower Speed Selection for the application. This will be active when Speedcontrol is in Digital Speed Settings mode. DiscreteAnalogIn: - Type_enum - AnalogADC - DiscreteState Type_enum: (lesp_discrete_analog_in_mapping_type_enum) 0 – None 1 – Discrete 2 – Analog
[In] SpeedControlRaise	DiscreteAnalogIn	This input requires the Raise Speed Selection for the application. This will be active when Speedcontrol is in Digital Speed Settings mode. DiscreteAnalogIn: - Type_enum - AnalogADC - DiscreteState Type_enum: (lesp_discrete_analog_in_mapping_type_enum) 0 – None 1 – Discrete 2 – Analog
[In] SpeedControlSpeedSelect1	DiscreteAnalogIn	This input requires the Speed Select1 status for the application. This will be active when Speedcontrol is in Digital Speed Settings mode and determines, together with the SpeedControlSpeedSelect2, the actual rated speed setpoint. A combination of 4 setpoints can be made with both input definitions and normally comes from wiring harness jumpers. DiscreteAnalogIn: - Type_enum - AnalogADC

		- DiscreteState Type_enum: (lesp_discrete_analog_in_mapping_type_enum) 0 – None 1 – Discrete 2 – Analog
[In] SpeedControlSpeedSelect2	DiscreteAnalogIn	This input requires the Speed Select2 status for the application. This will be active when Speedcontrol is in Digital Speed Settings mode and determines, together with the SpeedControlSpeedSelect1, the actual rated speed setpoint. A combination of 4 setpoints can be made with both input definitions and normally comes from wiring harness jumpers. DiscreteAnalogIn: - Type_enum - AnalogADC - DiscreteState Type_enum: (lesp_discrete_analog_in_mapping_type_enum) 0 – None 1 – Discrete 2 – Analog
[In] SpeedControlAnalogDigitalSelect	DiscreteAnalogIn	This input requires the selection input for Analog/Discrete Speed Setting Selection for the application. This selection determines if the analog or digital speed related inputs will be active or not. DiscreteAnalogIn: - Type_enum - AnalogADC - DiscreteState Type_enum: (lesp_discrete_analog_in_mapping_type_enum) 0 – None 1 – Discrete 2 – Analog
[In] SpeedControlAcceleratorBackup	DiscreteAnalogIn	This input requires the Accelerator Backup Speed Selection for the application. This will be active when Speedcontrol is in Analog Speed Settings mode and enables the digital Raise/Lower commands to be used, even when in Analog Speed Setting mode. DiscreteAnalogIn: - Type_enum - AnalogADC - DiscreteState Type_enum: (lesp_discrete_analog_in_mapping_type_enum) 0 – None 1 – Discrete

		2 – Analog
[In] SpeedControlGeneratorBreakerClutch	DiscreteAnalogIn	This input requires the feedback of either the Generator Breaker (Genset) or the Gearbox Clutch (Marine/Mechanical) for the application. When this input changes, it also changes the dynamics selection of the PID for the Speed Control. DiscreteAnalogIn: - Type_enum - AnalogADC - DiscreteState Type_enum: (lesp_discrete_analog_in_mapping_type_enum) 0 – None 1 – Discrete 2 – Analog
[In] SpeedControlUtilityBreaker	DiscreteAnalogIn	This input requires the feedback of the Utility Breaker (Genset) for the application. When this input changes, it also changes the dynamics selection of the PID for the Speed Control. DiscreteAnalogIn: - Type_enum - AnalogADC - DiscreteState Type_enum: (lesp_discrete_analog_in_mapping_type_enum) 0 – None 1 – Discrete 2 – Analog
[In] DieselFuelLeakage	DiscreteAnalogIn	This input requires the feedback of the Fuel Leakage Detection inside the application. DiscreteAnalogIn: - Type_enum - AnalogADC - DiscreteState Type_enum: (lesp_discrete_analog_in_mapping_type_enum) 0 – None 1 – Discrete 2 – Analog
[In] DieselFuelSupplyWaterPresence	DiscreteAnalogIn	This input requires the feedback of the Water Presence Detection inside the Diesel Fuel Supply for the application. DiscreteAnalogIn: - Type_enum - AnalogADC - DiscreteState Type_enum: (lesp_discrete_analog_in_mapping_type_enum) 0 – None

		1 – Discrete 2 – Analog
[In] EngineStartMode	DiscreteAnalogIn	This input requires the feedback of the Engine Start Mode inside the application. This can be either Manual Mode or Auto Mode. DiscreteAnalogIn: - Type_enum - AnalogADC - DiscreteState Type_enum: (lesp_discrete_analog_in_mapping_type_enum) 0 – None 1 – Discrete 2 – Analog
[In] LubricationSumpLevelHigh	DiscreteAnalogIn	This input requires the High Level feedback from the lubrication oil sump. When the oil level rises above a certain maximum level, this input should be activated. DiscreteAnalogIn: - Type_enum - AnalogADC - DiscreteState Type_enum: (lesp_discrete_analog_in_mapping_type_enum) 0 – None 1 – Discrete 2 – Analog
[In] Turbo1AirShutoffValveSolenoidFeedback	DiscreteAnalogIn	This input requires the feedback status from the Turbo1 Air Shutoff valve solenoid. It indicates whether the valve is in the open or closed position. DiscreteAnalogIn: - Type_enum - AnalogADC - DiscreteState Type_enum: (lesp_discrete_analog_in_mapping_type_enum) 0 – None 1 – Discrete 2 – Analog
[In] Turbo2AirShutoffValveSolenoidFeedback	DiscreteAnalogIn	This input requires the feedback status from the Turbo2 Air Shutoff valve solenoid. It indicates whether the valve is in the open or closed position. DiscreteAnalogIn: - Type_enum - AnalogADC - DiscreteState Type_enum: (lesp_discrete_analog_in_mapping_type_enum) 0 – None

		1 – Discrete 2 – Analog
[Out] Switch	Bus	Output that contains the following signals, all type (uint8): - RunStop - EmergencyStop - ClearFaults - FaultProtectionOverride - CoolantExpansionTankLevelLow - DrivenEquipmentStartingInterlock - EngineRapidStartRequest - EngineStartRequest - LubricationSumpLevelLow - PrelubricationManualRequest - SpeedControlDroop - SpeedControlIdleRated - SpeedControlLocalRemote - SpeedControlLower - SpeedControlRaise - SpeedControlSpeedSelect1 - SpeedControlSpeedSelect2 - SpeedControlAnalogDigitalSelect - SpeedControlAcceleratorBackup - SpeedControlGeneratorBreakerClutch - SpeedControlUtilityBreaker - DieselFuelLeakage - DieselFuelSupplyWaterPresence - EngineStartMode - LubricationSumpLevelHigh - Turbo1AirShutoffValveSolenoid- Feedback - Turbo2AirShutoffValveSolenoid- Feedback

CRS Temperature Sensing (Slow) Block Interface

Figure 11-7. LESP Model Reference ‘Sensors – CRS Temperature Sensing (Slow)’ Block

Table 11-6. ‘CRS Temperature Sensing (Slow)’ Block

Port	DataType	Description
[In] EngineCoolantTemp1	TcRtdInBus	Input that requires the Engine Coolant Temperature #1 input signal. TcRtdInBus:

		- Mode_enum - Status_enum - PullupResistorValue_ohms - Value Mode_enum: (lesp_tcrtd_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance (Mode 9) Thermocouple (Mode 10) Datalink degC Status_enum: (lesp_tcrtd_in_status_enum) 0 – NotAvailable 1 – Error 2 – Open 3 – Valid
[In] EngineCoolantTemp2	TcRtdInBus	Input that requires the Engine Coolant Temperature #2 input signal. TcRtdInBus: - Mode_enum - Status_enum - PullupResistorValue_ohms - Value Mode_enum: (lesp_tcrtd_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance (Mode 9) Thermocouple (Mode 10) Datalink degC Status_enum: (lesp_tcrtd_in_status_enum) 0 – NotAvailable 1 – Error 2 – Open 3 – Valid
[In] LubeOilFilterOutletTemp	TcRtdInBus	Input that requires the Lubrication Oil Filter Outlet Temperature input signal. TcRtdInBus: - Mode_enum - Status_enum - PullupResistorValue_ohms - Value Mode_enum: (lesp_tcrtd_in_mode_enum)

		(Mode 0) Undefined (Mode 1) 0-5V (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance (Mode 9) Thermocouple (Mode 10) Datalink degC Status_enum: (lesp_tcrtd_in_status_enum) 0 – NotAvailable 1 – Error 2 – Open 3 – Valid
[In] AmbientAirTemp	TcRtdInBus	Input that requires the Ambient Air Temperature input signal. TcRtdInBus: - Mode_enum - Status_enum - PullupResistorValue_ohms - Value Mode_enum: (lesp_tcrtd_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance (Mode 9) Thermocouple (Mode 10) Datalink degC Status_enum: (lesp_tcrtd_in_status_enum) 0 – NotAvailable 1 – Error 2 – Open 3 – Valid
[In] ManifoldAirTemp1	TcRtdInBus	Input that requires the Engine Manifold Air #1 Temperature input signal. TcRtdInBus: - Mode_enum - Status_enum - PullupResistorValue_ohms - Value Mode_enum: (lesp_tcrtd_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance (Mode 9) Thermocouple (Mode 10) Datalink degC

		Status_enum: (lesp_tcrtd_in_status_enum) 0 – NotAvailable 1 – Error 2 – Open 3 – Valid
[In] ManifoldAirTemp2	TcRtdInBus	Input that requires the Engine Manifold Air #2 Temperature input signal. TcRtdInBus: - Mode_enum - Status_enum - PullupResistorValue_ohms - Value Mode_enum: (lesp_tcrtd_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance (Mode 9) Thermocouple (Mode 10) Datalink degC Status_enum: (lesp_tcrtd_in_status_enum) 0 – NotAvailable 1 – Error 2 – Open 3 – Valid
[In] HPFuelPumpInletFuelTemp	TcRtdInBus	Input that requires the High Pressure Pump Inlet Fuel Temperature input signal. TcRtdInBus: - Mode_enum - Status_enum - PullupResistorValue_ohms - Value Mode_enum: (lesp_tcrtd_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance (Mode 9) Thermocouple (Mode 10) Datalink degC Status_enum: (lesp_tcrtd_in_status_enum) 0 – NotAvailable 1 – Error 2 – Open 3 – Valid

[In] FuelRailTemp	TcRtdInBus	Input that requires the Fuel Rail Temperature input signal. TcRtdInBus: - Mode_enum - Status_enum - PullupResistorValue_ohms - Value Mode_enum: (lesp_tcrtd_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance (Mode 9) Thermocouple (Mode 10) Datalink degC Status_enum: (lesp_tcrtd_in_status_enum) 0 – NotAvailable 1 – Error 2 – Open 3 – Valid
[In] InjectorReturnFuelTemp	TcRtdInBus	Input that requires the Injector Return line Fuel Temperature input signal. TcRtdInBus: - Mode_enum - Status_enum - PullupResistorValue_ohms - Value Mode_enum: (lesp_tcrtd_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance (Mode 9) Thermocouple (Mode 10) Datalink degC Status_enum: (lesp_tcrtd_in_status_enum) 0 – NotAvailable 1 – Error 2 – Open 3 – Valid
[In] Turbo1CompressorIntakeTemp	TcRtdInBus	Input that requires the Turbocharger #1 Compressor Side Intake Temperature input signal. TcRtdInBus: - Mode_enum - Status_enum - PullupResistorValue_ohms

		- Value Mode_enum: (lesp_tcrtd_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance (Mode 9) Thermocouple (Mode 10) Datalink degC Status_enum: (lesp_tcrtd_in_status_enum) 0 – NotAvailable 1 – Error 2 – Open 3 – Valid
[In] Turbo2CompressorIntakeTemp	TcRtdInBus	Input that requires the Turbocharger #2 Compressor Side Intake Temperature input signal. TcRtdInBus: - Mode_enum - Status_enum - PullupResistorValue_ohms - Value Mode_enum: (lesp_tcrtd_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance (Mode 9) Thermocouple (Mode 10) Datalink degC Status_enum: (lesp_tcrtd_in_status_enum) 0 – NotAvailable 1 – Error 2 – Open 3 – Valid
[In] Turbo1TurbineIntakeTemp	TcRtdInBus	Input that requires the Turbocharger #1 Turbine Side Intake Temperature input signal. TcRtdInBus: - Mode_enum - Status_enum - PullupResistorValue_ohms - Value Mode_enum: (lesp_tcrtd_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V (Mode 6) RTD Low Impedance

		(Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance (Mode 9) Thermocouple (Mode 10) Datalink degC Status_enum: (lesp_tcrtd_in_status_enum) 0 – NotAvailable 1 – Error 2 – Open 3 – Valid
[In] Turbo2TurbineIntakeTemp	TcRtdInBus	Input that requires the Turbocharger #2 Turbine Side Intake Temperature input signal. TcRtdInBus: - Mode_enum - Status_enum - PullupResistorValue_ohms - Value Mode_enum: (lesp_tcrtd_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance (Mode 9) Thermocouple (Mode 10) Datalink degC Status_enum: (lesp_tcrtd_in_status_enum) 0 – NotAvailable 1 – Error 2 – Open 3 – Valid
[In] Turbo3TurbineIntakeTemp	TcRtdInBus	Input that requires the Turbocharger #3 Turbine Side Intake Temperature input signal. TcRtdInBus: - Mode_enum - Status_enum - PullupResistorValue_ohms - Value Mode_enum: (lesp_tcrtd_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance (Mode 9) Thermocouple (Mode 10) Datalink degC Status_enum: (lesp_tcrtd_in_status_enum)

		0 – NotAvailable 1 – Error 2 – Open 3 – Valid
[In] Turbo4TurbineIntakeTemp	TcRtdInBus	Input that requires the Turbocharger #4 Turbine Side Intake Temperature input signal. TcRtdInBus: - Mode_enum - Status_enum - PullupResistorValue_ohms - Value Mode_enum: (lesp_tcrtd_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance (Mode 9) Thermocouple (Mode 10) Datalink degC Status_enum: (lesp_tcrtd_in_status_enum) 0 – NotAvailable 1 – Error 2 – Open 3 – Valid
[In] AfterCoolerWaterInletTemp	TcRtdInBus	Input that requires the Water Inlet Temperature of the AfterCooler input signal. TcRtdInBus: - Mode_enum - Status_enum - PullupResistorValue_ohms - Value Mode_enum: (lesp_tcrtd_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance (Mode 9) Thermocouple (Mode 10) Datalink degC Status_enum: (lesp_tcrtd_in_status_enum) 0 – NotAvailable 1 – Error 2 – Open 3 – Valid
[In] ExhaustGasTemp1	TcRtdInBus	Input that requires the Exhaust Gas Temperature of Cylinder #1 input signal.

		TcRtdInBus: - Mode_enum - Status_enum - PullupResistorValue_ohms - Value Mode_enum: (lesp_tcrtd_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance (Mode 9) Thermocouple (Mode 10) Datalink degC Status_enum: (lesp_tcrtd_in_status_enum) 0 – NotAvailable 1 – Error 2 – Open 3 – Valid
[In] ExhaustGasTemp2	TcRtdInBus	Input that requires the Exhaust Gas Temperature of Cylinder #2 input signal. TcRtdInBus: - Mode_enum - Status_enum - PullupResistorValue_ohms - Value Mode_enum: (lesp_tcrtd_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance (Mode 9) Thermocouple (Mode 10) Datalink degC Status_enum: (lesp_tcrtd_in_status_enum) 0 – NotAvailable 1 – Error 2 – Open 3 – Valid
[In] ExhaustGasTemp3	TcRtdInBus	Input that requires the Exhaust Gas Temperature of Cylinder #3 input signal. TcRtdInBus: - Mode_enum - Status_enum - PullupResistorValue_ohms - Value Mode_enum:

		(lesp_tcrtd_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance (Mode 9) Thermocouple (Mode 10) Datalink degC Status_enum: (lesp_tcrtd_in_status_enum) 0 – NotAvailable 1 – Error 2 – Open 3 – Valid
[In] ExhaustGasTemp4	TcRtdInBus	Input that requires the Exhaust Gas Temperature of Cylinder #4 input signal. TcRtdInBus: - Mode_enum - Status_enum - PullupResistorValue_ohms - Value Mode_enum: (lesp_tcrtd_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance (Mode 9) Thermocouple (Mode 10) Datalink degC Status_enum: (lesp_tcrtd_in_status_enum) 0 – NotAvailable 1 – Error 2 – Open 3 – Valid
[In] ExhaustGasTemp5	TcRtdInBus	Input that requires the Exhaust Gas Temperature of Cylinder #5 input signal. TcRtdInBus: - Mode_enum - Status_enum - PullupResistorValue_ohms - Value Mode_enum: (lesp_tcrtd_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance (Mode 9) Thermocouple

		(Mode 10) Datalink degC Status_enum: (lesp_tcrtd_in_status_enum) 0 – NotAvailable 1 – Error 2 – Open 3 – Valid
[In] ExhaustGasTemp6	TcRtdInBus	Input that requires the Exhaust Gas Temperature of Cylinder #6 input signal. TcRtdInBus: - Mode_enum - Status_enum - PullupResistorValue_ohms - Value Mode_enum: (lesp_tcrtd_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance (Mode 9) Thermocouple (Mode 10) Datalink degC Status_enum: (lesp_tcrtd_in_status_enum) 0 – NotAvailable 1 – Error 2 – Open 3 – Valid
[In] ExhaustGasTemp7	TcRtdInBus	Input that requires the Exhaust Gas Temperature of Cylinder #7 input signal. TcRtdInBus: - Mode_enum - Status_enum - PullupResistorValue_ohms - Value Mode_enum: (lesp_tcrtd_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance (Mode 9) Thermocouple (Mode 10) Datalink degC Status_enum: (lesp_tcrtd_in_status_enum) 0 – NotAvailable 1 – Error 2 – Open

		3 – Valid
[In] ExhaustGasTemp8	TcRtdInBus	Input that requires the Exhaust Gas Temperature of Cylinder #8 input signal. TcRtdInBus: - Mode_enum - Status_enum - PullupResistorValue_ohms - Value Mode_enum: (lesp_tcrtd_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance (Mode 9) Thermocouple (Mode 10) Datalink degC Status_enum: (lesp_tcrtd_in_status_enum) 0 – NotAvailable 1 – Error 2 – Open 3 – Valid
[In] ExhaustGasTemp9	TcRtdInBus	Input that requires the Exhaust Gas Temperature of Cylinder #9 input signal. TcRtdInBus: - Mode_enum - Status_enum - PullupResistorValue_ohms - Value Mode_enum: (lesp_tcrtd_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance (Mode 9) Thermocouple (Mode 10) Datalink degC Status_enum: (lesp_tcrtd_in_status_enum) 0 – NotAvailable 1 – Error 2 – Open 3 – Valid
[In] ExhaustGasTemp10	TcRtdInBus	Input that requires the Exhaust Gas Temperature of Cylinder #10 input signal. TcRtdInBus:

		- Mode_enum - Status_enum - PullupResistorValue_ohms - Value Mode_enum: (lesp_tcrtd_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance (Mode 9) Thermocouple (Mode 10) Datalink degC Status_enum: (lesp_tcrtd_in_status_enum) 0 – NotAvailable 1 – Error 2 – Open 3 – Valid
[In] ExhaustGasTemp11	TcRtdInBus	Input that requires the Exhaust Gas Temperature of Cylinder #11 input signal. TcRtdInBus: - Mode_enum - Status_enum - PullupResistorValue_ohms - Value Mode_enum: (lesp_tcrtd_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance (Mode 9) Thermocouple (Mode 10) Datalink degC Status_enum: (lesp_tcrtd_in_status_enum) 0 – NotAvailable 1 – Error 2 – Open 3 – Valid
[In] ExhaustGasTemp12	TcRtdInBus	Input that requires the Exhaust Gas Temperature of Cylinder #12 input signal. TcRtdInBus: - Mode_enum - Status_enum - PullupResistorValue_ohms - Value Mode_enum: (lesp_tcrtd_in_mode_enum)

		(Mode 0) Undefined (Mode 1) 0-5V (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance (Mode 9) Thermocouple (Mode 10) Datalink degC Status_enum: (lesp_tcrtd_in_status_enum) 0 – NotAvailable 1 – Error 2 – Open 3 – Valid
[In] ExhaustGasTemp13	TcRtdInBus	Input that requires the Exhaust Gas Temperature of Cylinder #13 input signal. TcRtdInBus: - Mode_enum - Status_enum - PullupResistorValue_ohms - Value Mode_enum: (lesp_tcrtd_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance (Mode 9) Thermocouple (Mode 10) Datalink degC Status_enum: (lesp_tcrtd_in_status_enum) 0 – NotAvailable 1 – Error 2 – Open 3 – Valid
[In] ExhaustGasTemp14	TcRtdInBus	Input that requires the Exhaust Gas Temperature of Cylinder #14 input signal. TcRtdInBus: - Mode_enum - Status_enum - PullupResistorValue_ohms - Value Mode_enum: (lesp_tcrtd_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance (Mode 9) Thermocouple (Mode 10) Datalink degC

		Status_enum: (lesp_tcrtd_in_status_enum) 0 – NotAvailable 1 – Error 2 – Open 3 – Valid
[In] ExhaustGasTemp15	TcRtdInBus	Input that requires the Exhaust Gas Temperature of Cylinder #15 input signal. TcRtdInBus: - Mode_enum - Status_enum - PullupResistorValue_ohms - Value Mode_enum: (lesp_tcrtd_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance (Mode 9) Thermocouple (Mode 10) Datalink degC Status_enum: (lesp_tcrtd_in_status_enum) 0 – NotAvailable 1 – Error 2 – Open 3 – Valid
[In] ExhaustGasTemp16	TcRtdInBus	Input that requires the Exhaust Gas Temperature of Cylinder #16 input signal. TcRtdInBus: - Mode_enum - Status_enum - PullupResistorValue_ohms - Value Mode_enum: (lesp_tcrtd_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance (Mode 9) Thermocouple (Mode 10) Datalink degC Status_enum: (lesp_tcrtd_in_status_enum) 0 – NotAvailable 1 – Error 2 – Open 3 – Valid

[In] ExhaustGasTemp17	TcRtdInBus	Input that requires the Exhaust Gas Temperature of Cylinder #17 input signal. TcRtdInBus: - Mode_enum - Status_enum - PullupResistorValue_ohms - Value Mode_enum: (lesp_tcrtd_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance (Mode 9) Thermocouple (Mode 10) Datalink degC Status_enum: (lesp_tcrtd_in_status_enum) 0 – NotAvailable 1 – Error 2 – Open 3 – Valid
[In] ExhaustGasTemp18	TcRtdInBus	Input that requires the Exhaust Gas Temperature of Cylinder #18 input signal. TcRtdInBus: - Mode_enum - Status_enum - PullupResistorValue_ohms - Value Mode_enum: (lesp_tcrtd_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance (Mode 9) Thermocouple (Mode 10) Datalink degC Status_enum: (lesp_tcrtd_in_status_enum) 0 – NotAvailable 1 – Error 2 – Open 3 – Valid
[In] ExhaustGasTemp19	TcRtdInBus	Input that requires the Exhaust Gas Temperature of Cylinder #19 input signal. TcRtdInBus: - Mode_enum - Status_enum

		- PullupResistorValue_ohms - Value Mode_enum: (lesp_tcrtd_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance (Mode 9) Thermocouple (Mode 10) Datalink degC Status_enum: (lesp_tcrtd_in_status_enum) 0 – NotAvailable 1 – Error 2 – Open 3 – Valid
[In] ExhaustGasTemp20	TcRtdInBus	Input that requires the Exhaust Gas Temperature of Cylinder #20 input signal. TcRtdInBus: - Mode_enum - Status_enum - PullupResistorValue_ohms - Value Mode_enum: (lesp_tcrtd_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance (Mode 9) Thermocouple (Mode 10) Datalink degC Status_enum: (lesp_tcrtd_in_status_enum) 0 – NotAvailable 1 – Error 2 – Open 3 – Valid
[In] HPFuelPump1HousingTemp	TcRtdInBus	Input that requires the High Pressure Pump #1 Housing Temperature input signal. TcRtdInBus: - Mode_enum - Status_enum - PullupResistorValue_ohms - Value Mode_enum: (lesp_tcrtd_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V

		(Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance (Mode 9) Thermocouple (Mode 10) Datalink degC Status_enum: (lesp_tcrtd_in_status_enum) 0 – NotAvailable 1 – Error 2 – Open 3 – Valid
[In] HPFuelPump2HousingTemp	TcRtdInBus	Input that requires the High Pressure Pump #2 Housing Temperature input signal. TcRtdInBus: - Mode_enum - Status_enum - PullupResistorValue_ohms - Value Mode_enum: (lesp_tcrtd_in_mode_enum) (Mode 0) Undefined (Mode 1) 0-5V (Mode 6) RTD Low Impedance (Mode 7) RTD Medium Impedance (Mode 8) RTD High Impedance (Mode 9) Thermocouple (Mode 10) Datalink degC Status_enum: (lesp_tcrtd_in_status_enum) 0 – NotAvailable 1 – Error 2 – Open 3 – Valid
[In] LocalColdJct	AnalogSensorBus	Input that requires the Local Cold Junction Temperature input signal. AnalogSensorBus: - FilteredValue - MeasurementStatus_enum MeasurementStatus_enum: (lesp_measurement_status_enum) 0 – Valid 1 – Disabled 2 – SignalLow 3 – SignalHigh 4 – SensorSupplyFailed 5 – ReferenceSourceFailed 6 – OpenCircuit
[In] PWRDLY	boolean	This input requires the status of the PowerOn Delay that is required for some initialization logic.
[In] Reset	boolean	Input that requires the application to be reset and connected.

[Out] Temperature	Bus	Output that contains the following signals, all type (AnalogSensorBus): - EngineCoolantTemp1 - EngineCoolantTemp2 - EngineCoolantTemp - LubeOilFilterOutletTemp - AmbientAirTemp - ManifoldAirTemp1 - ManifoldAirTemp2 - ManifoldAirTemp - HPFuelPumpInletFuelTemp - FuelRailTemp - InjectorReturnFuelTemp - Turbo1CompressorIntakeTemp - Turbo2CompressorIntakeTemp - Turbo1TurbineIntakeTemp - Turbo2TurbineIntakeTemp - Turbo3TurbineIntakeTemp - Turbo4TurbineIntakeTemp - AfterCoolerWaterInletTemp - ExhaustGasTemp1 - ExhaustGasTemp2 - ExhaustGasTemp3 - ExhaustGasTemp4 - ExhaustGasTemp5 - ExhaustGasTemp6 - ExhaustGasTemp7 - ExhaustGasTemp8 - ExhaustGasTemp9 - ExhaustGasTemp10 - ExhaustGasTemp11 - ExhaustGasTemp12 - ExhaustGasTemp13 - ExhaustGasTemp14 - ExhaustGasTemp15 - ExhaustGasTemp16 - ExhaustGasTemp17 - ExhaustGasTemp18 - ExhaustGasTemp19 - ExhaustGasTemp20 - HPFuelPump1HousingTemp - HPFuelPump2HousingTemp Bus Details: AnalogSensorBus: - FilteredValue - MeasurementStatus_enum MeasurementStatus_enum: (lesp_measurement_status_enum) 0 – Valid 1 – Disabled 2 – SignalLow 3 – SignalHigh 4 – SensorSupplyFailed 5 – ReferenceSourceFailed 6 – OpenCircuit

Example Model Using the Blocks

Below is a very simple setup using MotoHawk Calibration and MotoHawk Probes blocks to make the model suitable for building.

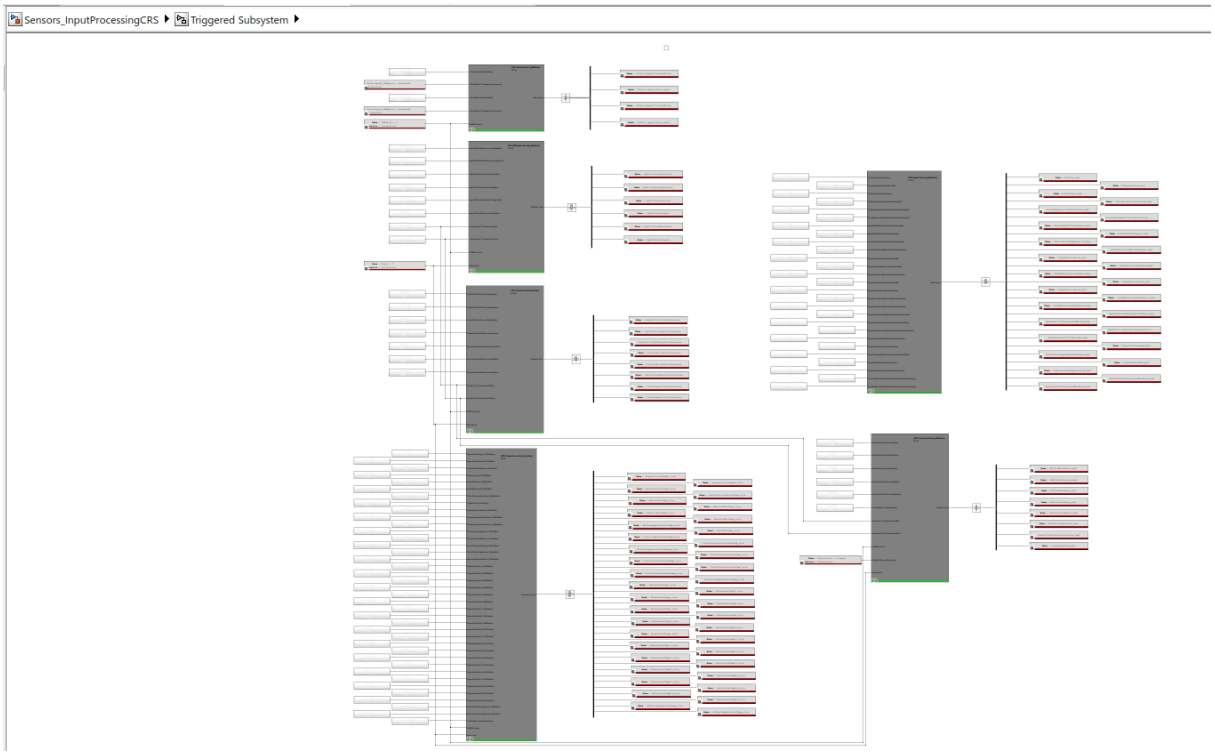


Figure 11-8. Simple Example Model, Using the Sensors - CRS Interface Block

The required files will generate when compiling, and if the Toolkit required blocks are also added to the application model, it will generate the required SID files for the Toolkit HMI tool creation.

Default ToolKit Pages

Opening a new Toolkit application and selecting the correctly generated .sid file will look like the screenshot below.

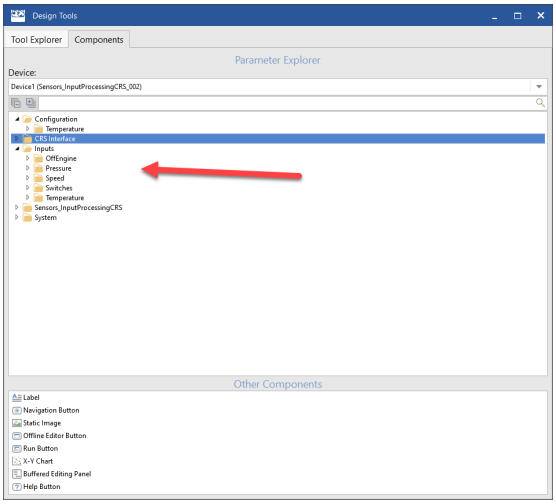
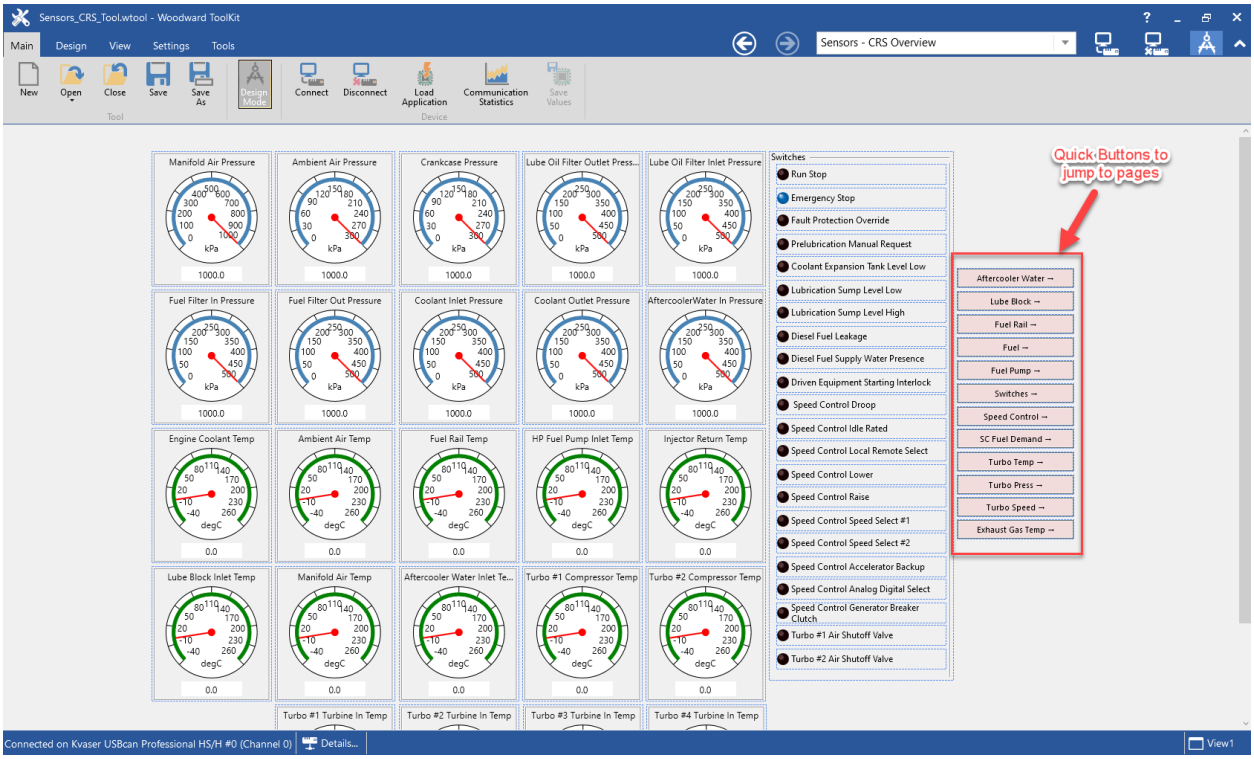
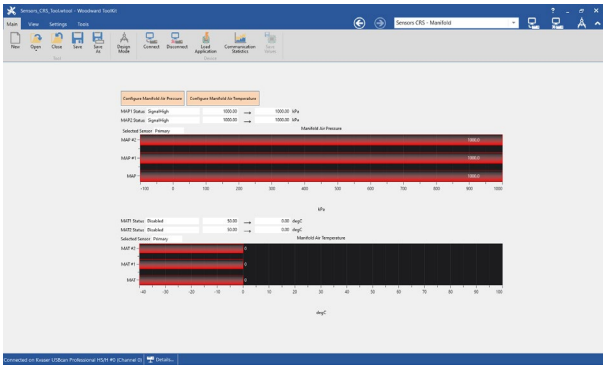
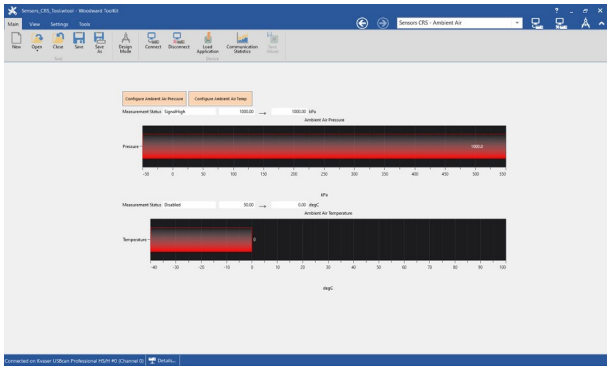


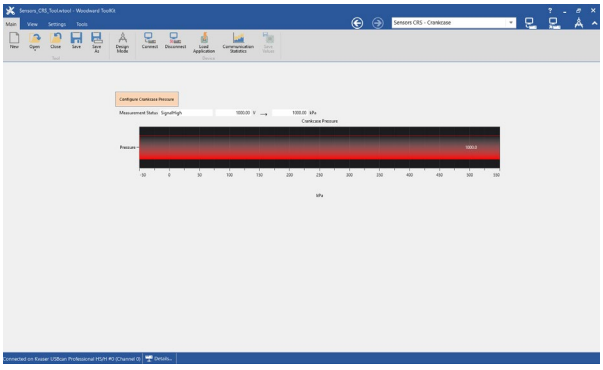
Figure 11-9. SID File Selection Sensors – CRS Interface Logic

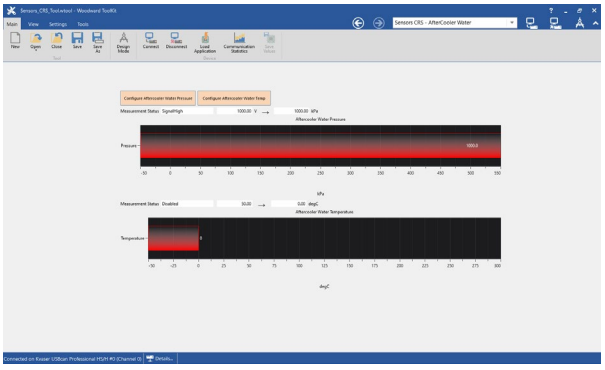
Below are a few of the Toolkit pages that will contain most of the defined parameters in the example model. Of course, the real sensors and data must be attached to complete it for the application that is being worked on. For that reason, red X's appear on some pages.

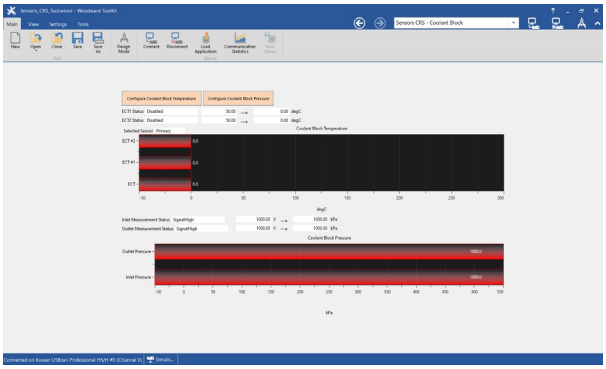


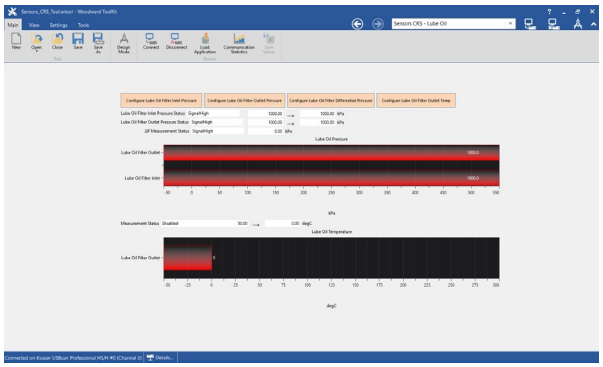


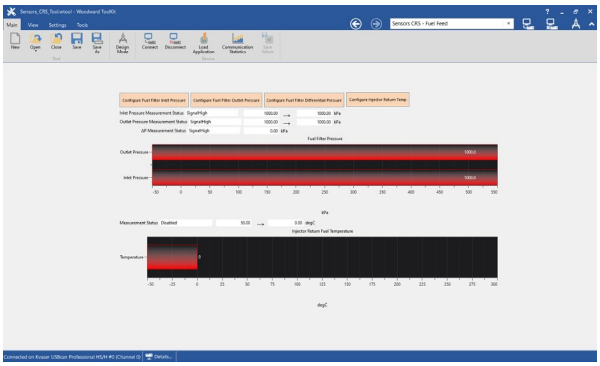


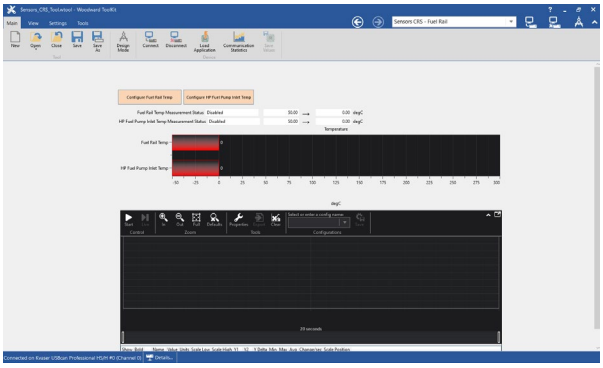


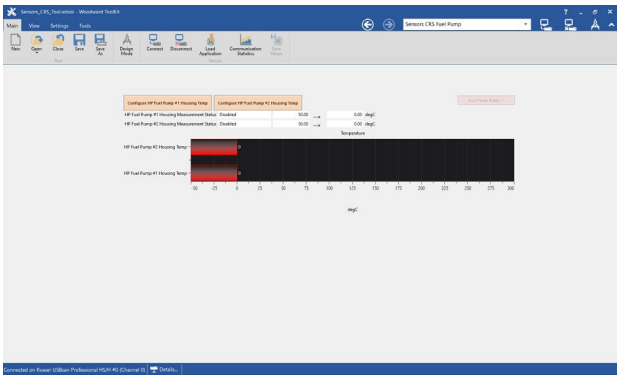


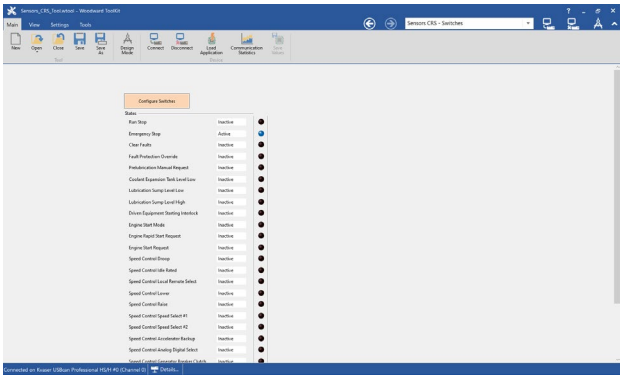


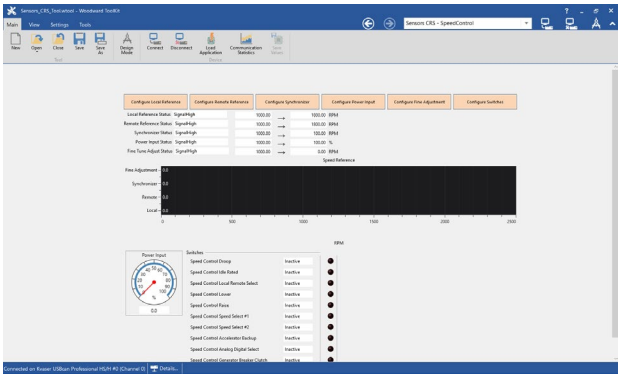


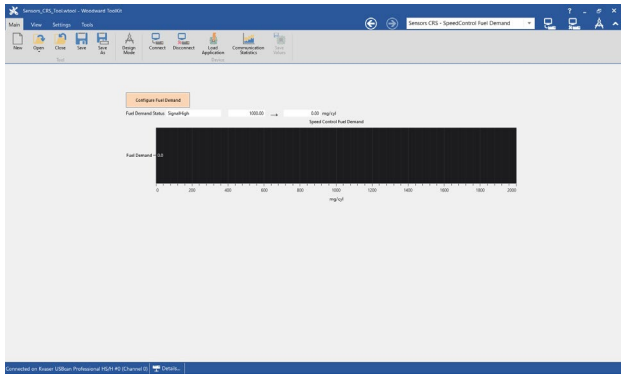












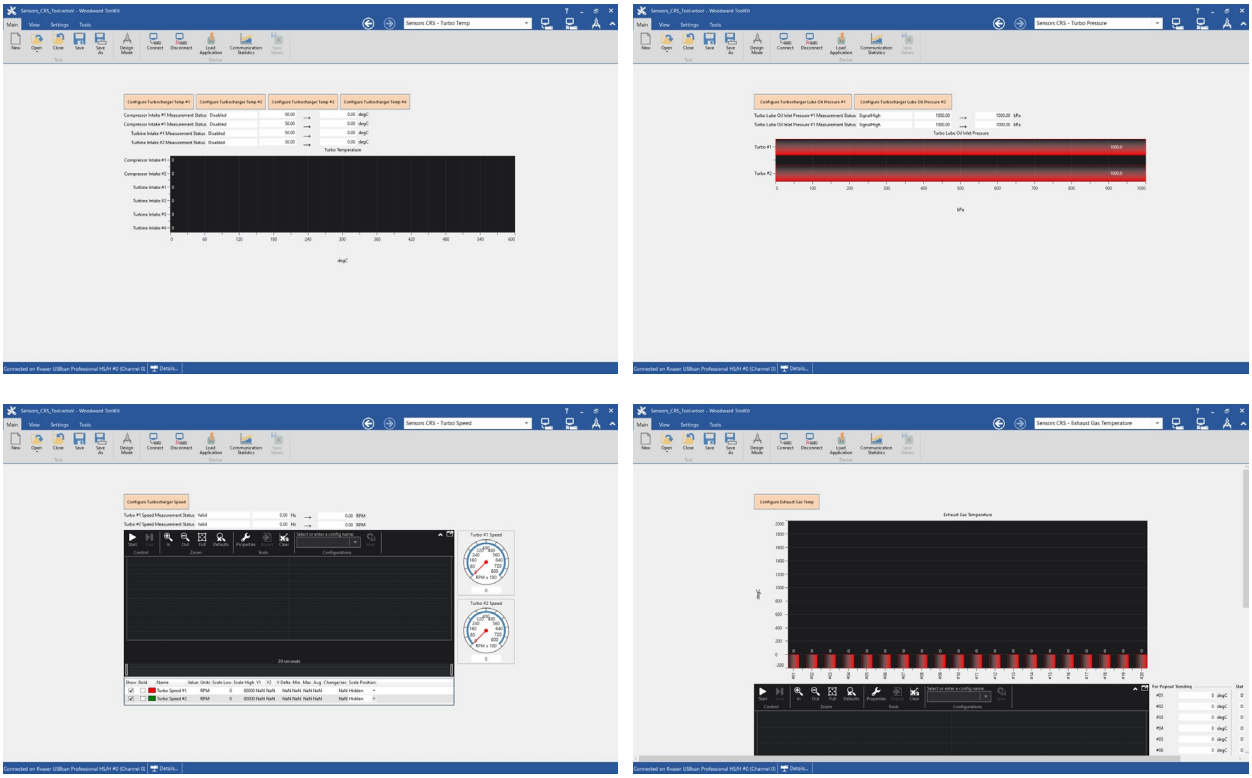


Figure 11-10. Examples of the Sensors-Input CRS Dual Fuel Toolkit Pages

Chapter 12.
Sensors – UEGO

Introduction

The ‘Sensors – UEGO’ blockset is used to activate, read, and control the Universal Exhaust Gas Oxygen (UEGO) Sensor inside the application. UEGO is also sometimes referred to as Wide Band Lambda sensors. UEGO sensors are designed to measure the amount of oxygen in an exhaust gas flow. This measurement is used to infer the air/fuel ratio at the time of combustion, ignition timing corrections, and the proportions of pollutants entering the catalytic converter.

Inside the blockset, one block is available as shown in the screenshot below.

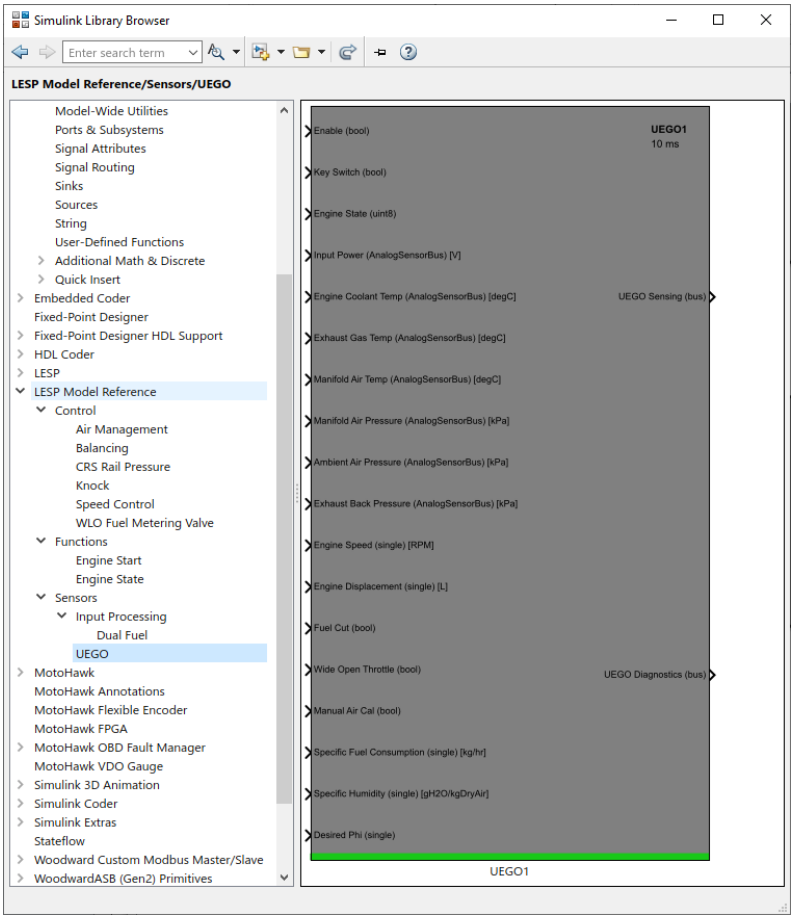


Figure 12-1. LESP Model Reference ‘Sensors – UEGO’ Block

- Sensors – UEGO Model Reference Block
  UEGO sensors are more complex inputs than the typical pressure or temperature sensor. There is a control loop required to create the measurement signal, a heater control circuit to maintain the sensor at the proper operating temperature, two voltages to be corrected, sensor protection schemes, and a calibration to be maintained.

UEGO1 Block Interface

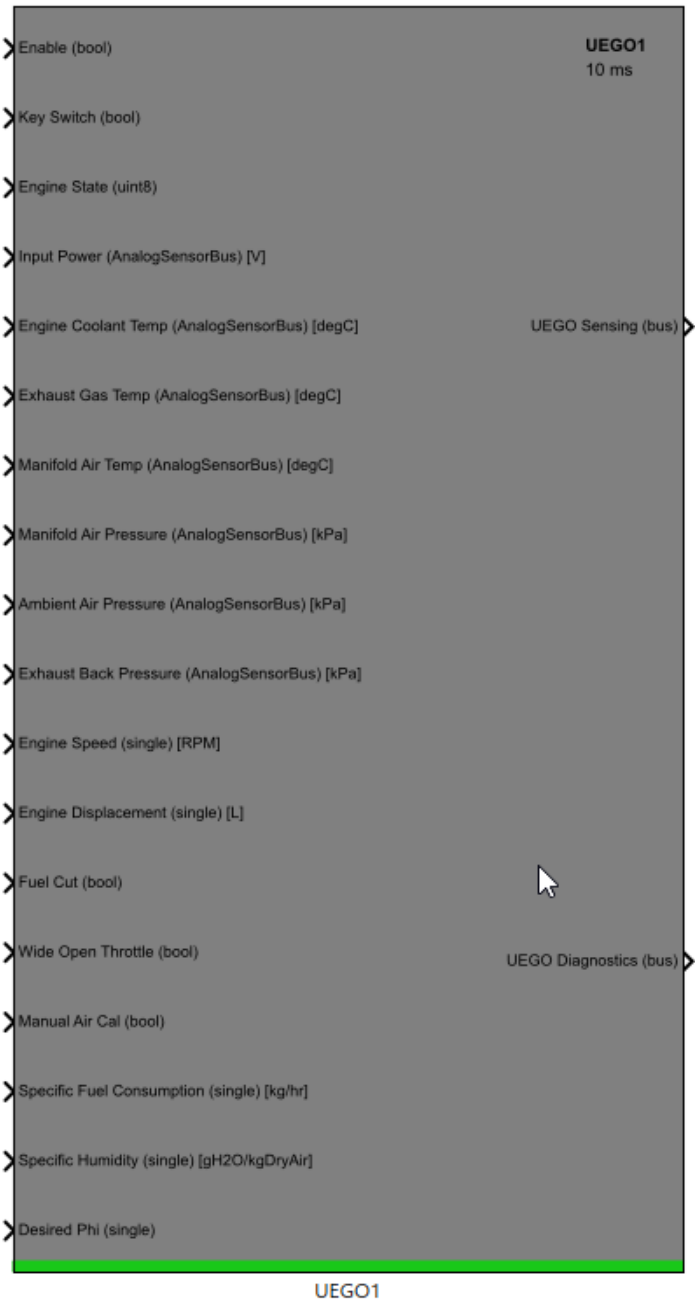


Figure 12-2. LESP Model Reference ‘Sensors – UEGO1’ Block

Table 12-1. ‘UEGO1’ Block

Port	DataType	Description
[In] Enable	Boolean	Input to enable / disable the UEGO logic. True → Active False → Inactive
[In] Key Switch	Boolean	The actual status of the Key Switch input at that moment in the application. This field is used for UEGO logic.

		Emergency Stop is a direct fuel shutdown to force engine stop.
[In] Engine State	uint8	The actual engine state the engine is operating in at that particular moment. These states (lesp_engine_state_enum) are defined as below: 0 – Stopped 1 – Prestart 2 – Starting 3 – Warmup 4 – Running 5 – Cooldown 6 – Stopping 7 – Postrun
[In] Input Power (V)	AnalogSensorBus	The actual input power supply in volts at that moment in the application. This field is used for UEGO logic. Bus Details: AnalogSensorBus: - FilteredValue - MeasurementStatus_enum MeasurementStatus_enum: (lesp_measurement_status_enum) 0 – Valid 1 – Disabled 2 – SignalLow 3 – SignalHigh 4 – SensorSupplyFailed 5 – ReferenceSourceFailed 6 – OpenCircuit
[In] Engine Coolant Temp (degC)	AnalogSensorBus	The actual engine coolant temperature in degC at that moment in the application. This field is used for UEGO logic. Bus Details: AnalogSensorBus: - FilteredValue - MeasurementStatus_enum MeasurementStatus_enum: (lesp_measurement_status_enum) 0 – Valid 1 – Disabled 2 – SignalLow 3 – SignalHigh 4 – SensorSupplyFailed 5 – ReferenceSourceFailed 6 – OpenCircuit
[In] Exhaust Gas Temp (degC)	AnalogSensorBus	The actual exhaust gas temperature in degC at that moment in the application. This field is used for UEGO logic. Bus Details: AnalogSensorBus: - FilteredValue - MeasurementStatus_enum MeasurementStatus_enum: (lesp_measurement_status_enum)

		0 – Valid 1 – Disabled 2 – SignalLow 3 – SignalHigh 4 – SensorSupplyFailed 5 – ReferenceSourceFailed 6 – OpenCircuit
[In] Manifold Air Temp (degC)	AnalogSensorBus	The actual manifold air temperature in degC at that moment in the application. This field is used for UEGO logic. Bus Details: AnalogSensorBus: - FilteredValue - MeasurementStatus_enum MeasurementStatus_enum: (lesp_measurement_status_enum) 0 – Valid 1 – Disabled 2 – SignalLow 3 – SignalHigh 4 – SensorSupplyFailed 5 – ReferenceSourceFailed 6 – OpenCircuit
[In] Manifold Air Pressure (kPa)	AnalogSensorBus	The actual manifold air pressure in kPa at that moment in the application. This field is used for UEGO logic. Bus Details: AnalogSensorBus: - FilteredValue - MeasurementStatus_enum MeasurementStatus_enum: (lesp_measurement_status_enum) 0 – Valid 1 – Disabled 2 – SignalLow 3 – SignalHigh 4 – SensorSupplyFailed 5 – ReferenceSourceFailed 6 – OpenCircuit
[In] Ambient Air Pressure (kPa)	AnalogSensorBus	The actual ambient air pressure in kPa at that moment in the application. This field is used for UEGO logic. Bus Details: AnalogSensorBus: - FilteredValue - MeasurementStatus_enum MeasurementStatus_enum: (lesp_measurement_status_enum) 0 – Valid 1 – Disabled 2 – SignalLow 3 – SignalHigh 4 – SensorSupplyFailed 5 – ReferenceSourceFailed 6 – OpenCircuit

[In] Exhaust Back Pressure (kPa)	AnalogSensorBus	The actual exhaust back pressure in kPa at that moment in the application. This field is used for UEGO logic. Bus Details: AnalogSensorBus: - FilteredValue - MeasurementStatus_enum MeasurementStatus_enum: (lesp_measurement_status_enum) 0 – Valid 1 – Disabled 2 – SignalLow 3 – SignalHigh 4 – SensorSupplyFailed 5 – ReferenceSourceFailed 6 – OpenCircuit
[In] Engine Speed (RPM)	single	The actual engine speed at that moment in the application. This field is used for UEGO logic.
[In] Engine Displacement (L)	single	Engine displacement in [L]. This info is in the spec sheet of the Engine (OEM).
[In] Fuel Cut	boolean	This input parameter defines if fuel cut is active or not. It is important for the logic to have this status feedback. Fuel cut means that fuel is shut-off instantly.
[In] Wide Open Throttle	boolean	This input parameter defines if the throttle is wide open. It is important for the logic to have this status feedback.
[In] Manual Air Cal	boolean	Input that defines if manual air calibration is wanted. The manual air calibration process involves removing the sensor from its mounting location so that it is exposed to pure air, turning on the heater to bring it up to proper temperature, and calculating a correction factor by comparing the measured pumping current to the ideal pumping current.
[In] Specific Fuel Consumption (kg/hr)	single	Specific fuel consumption in [kg/hr]
[In] Specific Humidity (gH2O/kgDryAir)	single	Specific humidity in [gH2O/kgDryAir]
[In] Desired Phi	single	Desired Phi value of the air and exhaust intake flow.
[Out] UEGO Sensing	bus	Sensing signals from the UEGO sensor logic. Signals in the bus are: - State_enum - Phi - Lambda - O2Pct - Desired phi - Desired Lambda

[Out] UEGO Diagnostics	bus	Diagnostics coming from the UEGO sensor logic. Signals in the bus are: - HeaterControlDiag (FaultBus) - HeaterRangeHighDiag (FaultBus) - HeaterRangeLowDiag (FaultBus) - HeaterOpenShortDiag (FaultBus) - VMDiag (FaultBus) - UNDiag (FaultBus) - IADiag (FaultBus) - URRangeHighDiag (FaultBus) - URRangeLowDiag (FaultBus) - UARangeHighDiag (FaultBus) - UARangeLowDiag (FaultBus) - SensorInactiveDiag (FaultBus) - AirCalLowerLimitDiag (Faultbus) - AirCalUpperLimitDiag (Faultbus) - AirCalMeasInvalid - AirCalCorrFactEnabled FaultBus: - AssertionStatus - ConditionalStatus - InhibitStatus

Example Model Using the Blocks

Below is a very simple setup using MotoHawk Calibration and MotoHawk Probes blocks to make the model suitable for building.

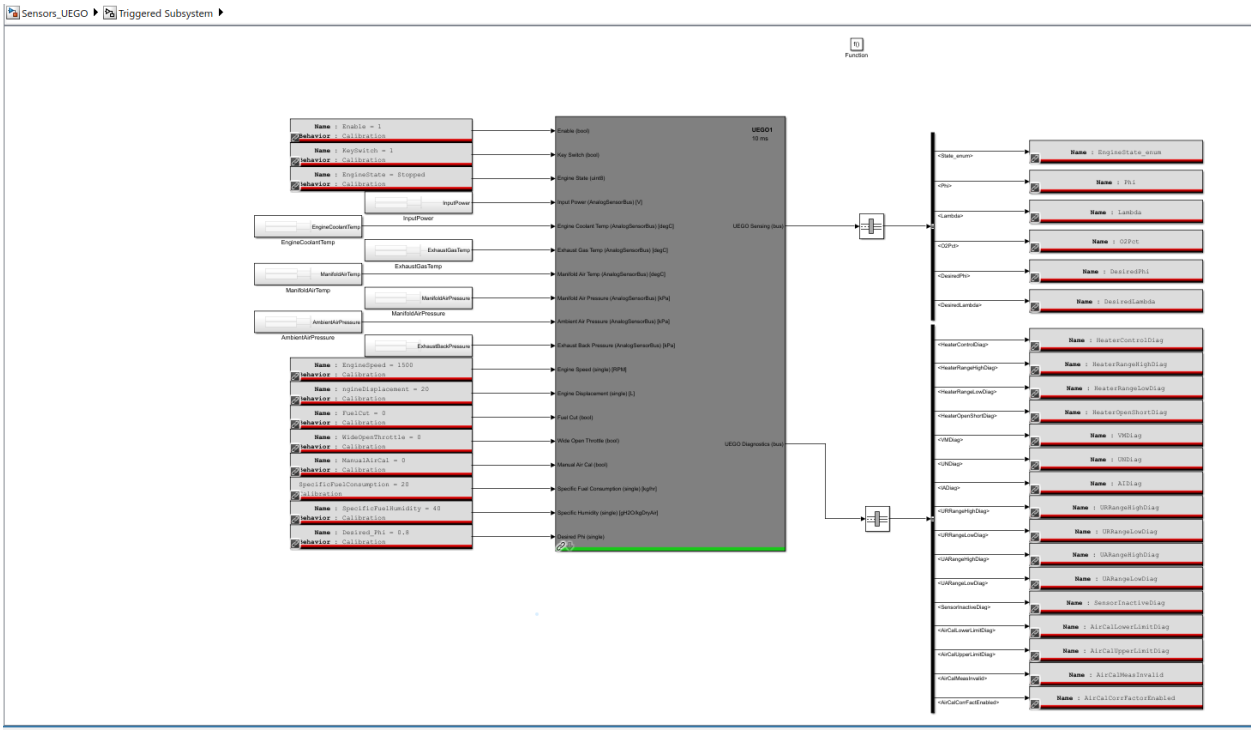


Figure 12-3. Simple Example Model, Using the Sensors – UEGO1 Block

The required files will generate when compiling, and if the Toolkit required blocks are also added to the application model, it will generate the required SID files for the Toolkit HMI tool creation.

Default ToolKit Pages

Opening a new Toolkit application and selecting the correctly generated .sid file will look like the screenshot below.

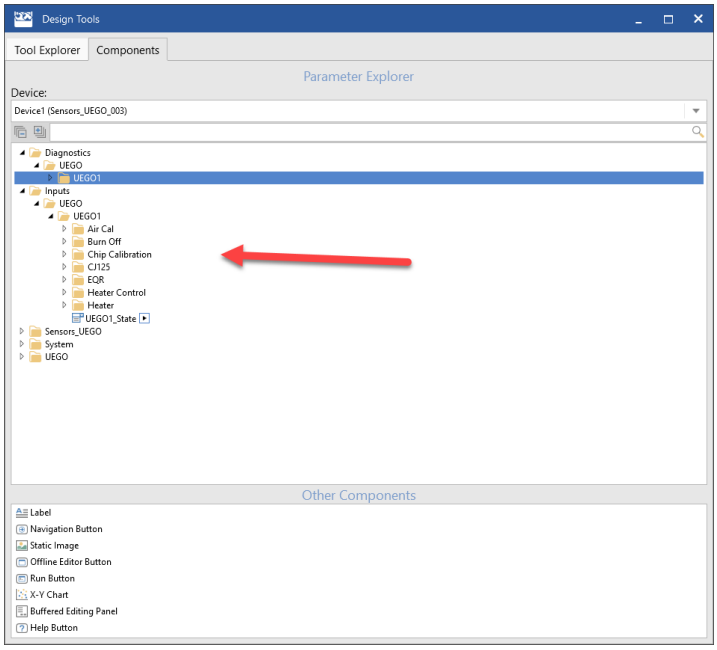
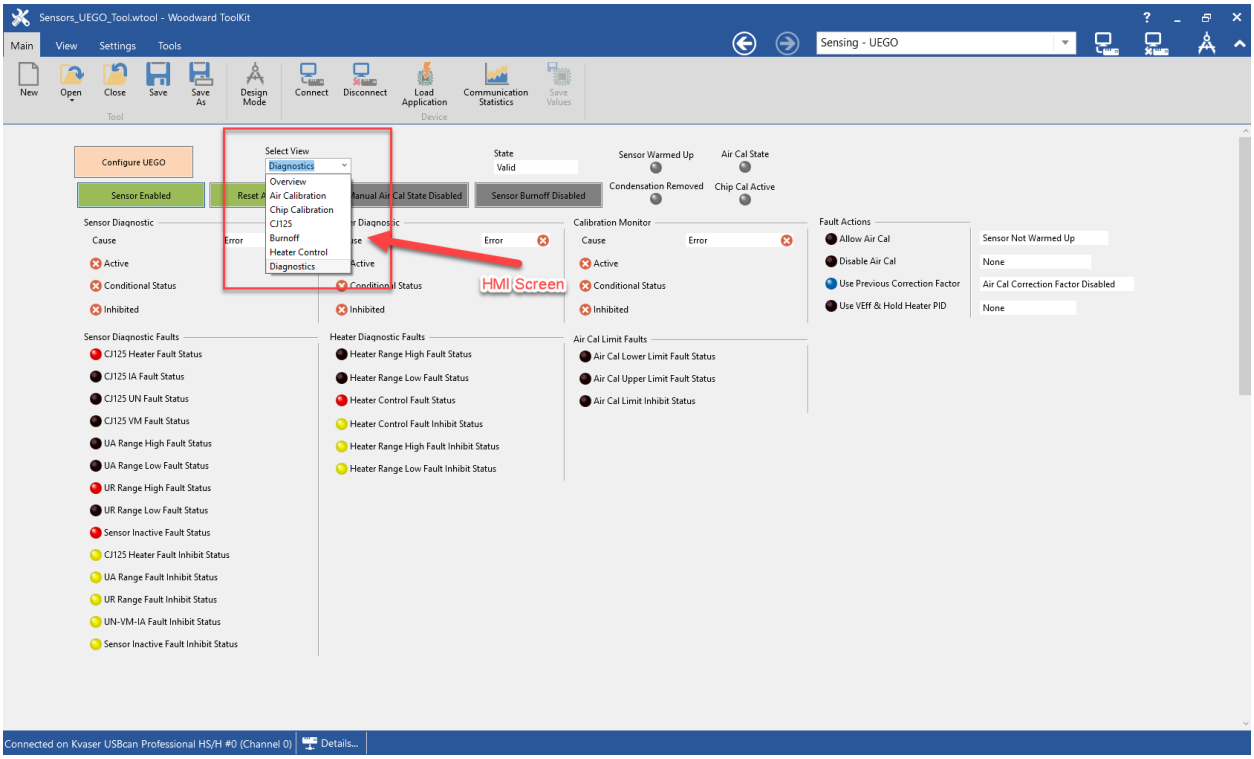
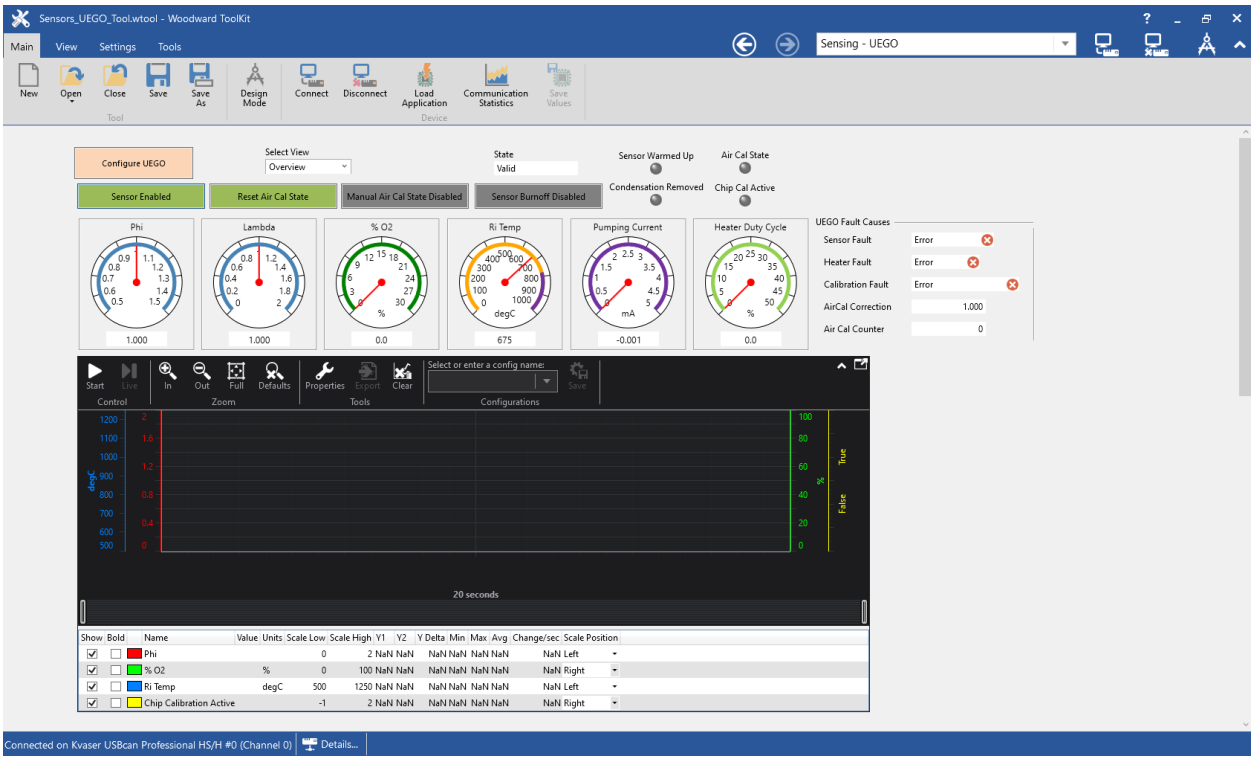
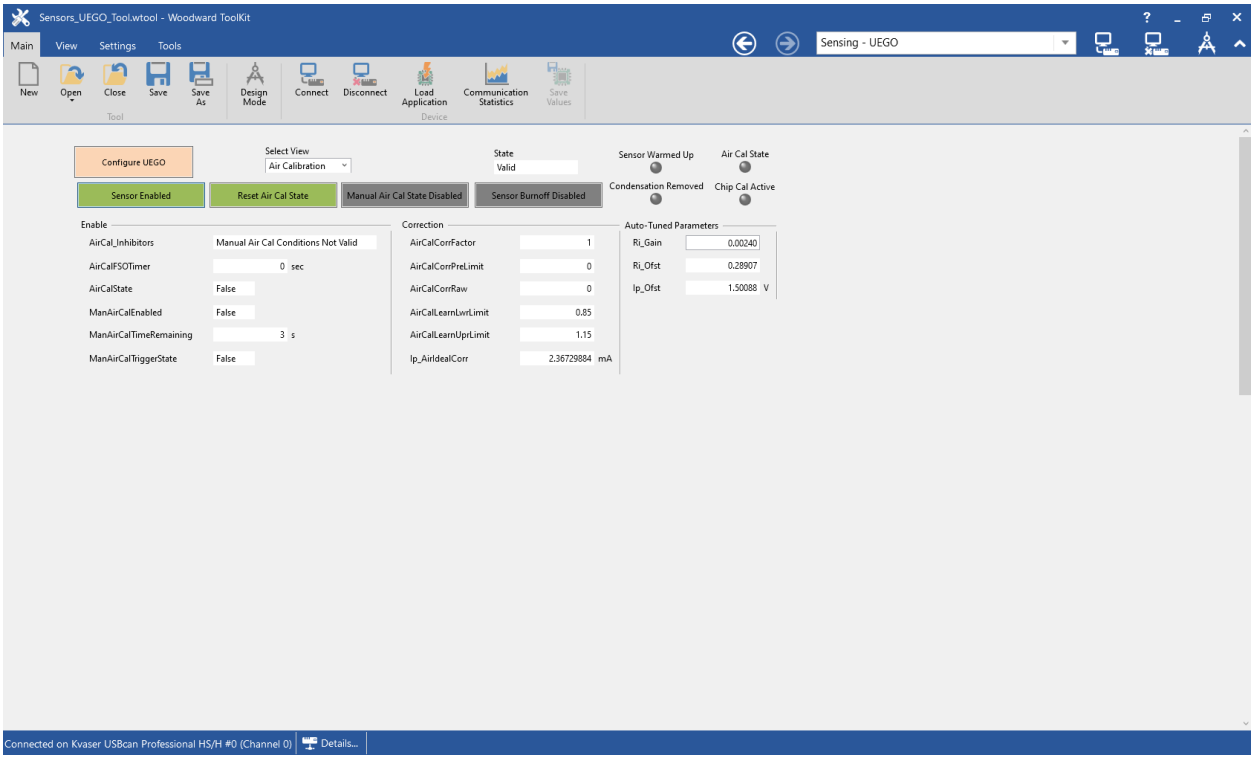


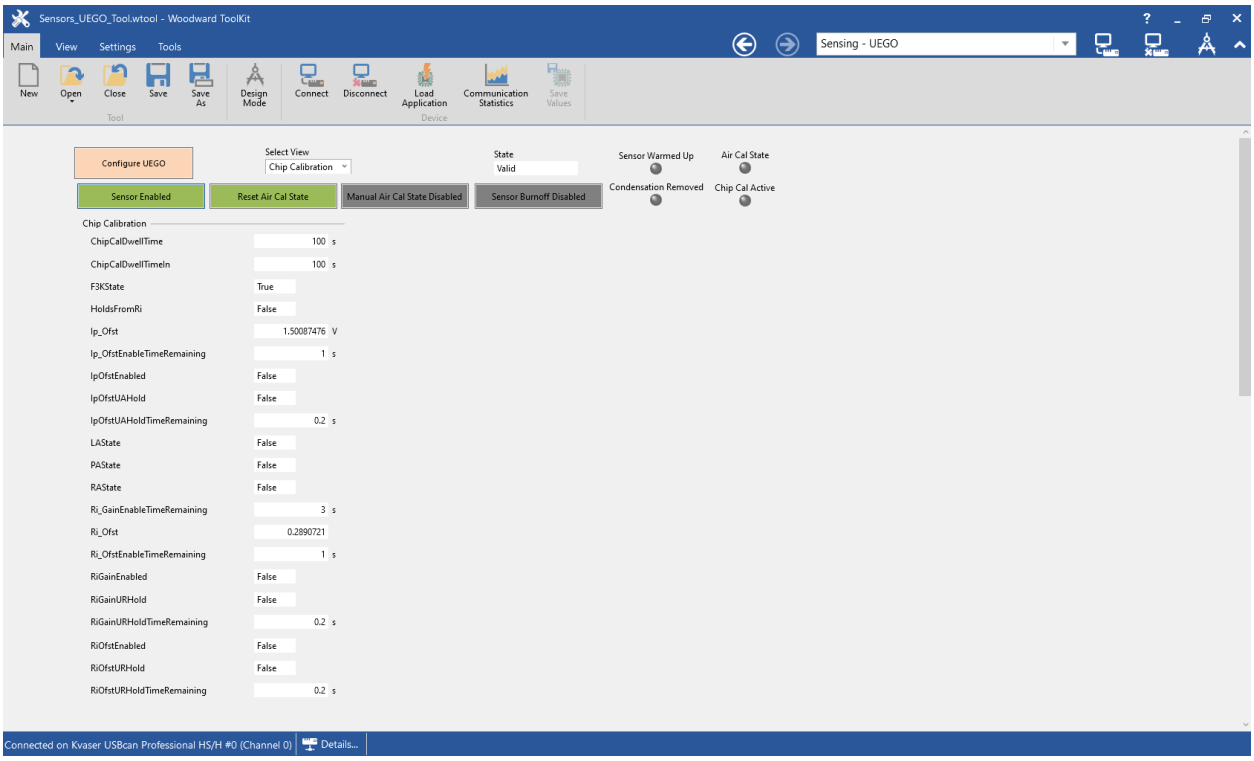
Figure 12-4. SID File Selection Sensors – UEGO

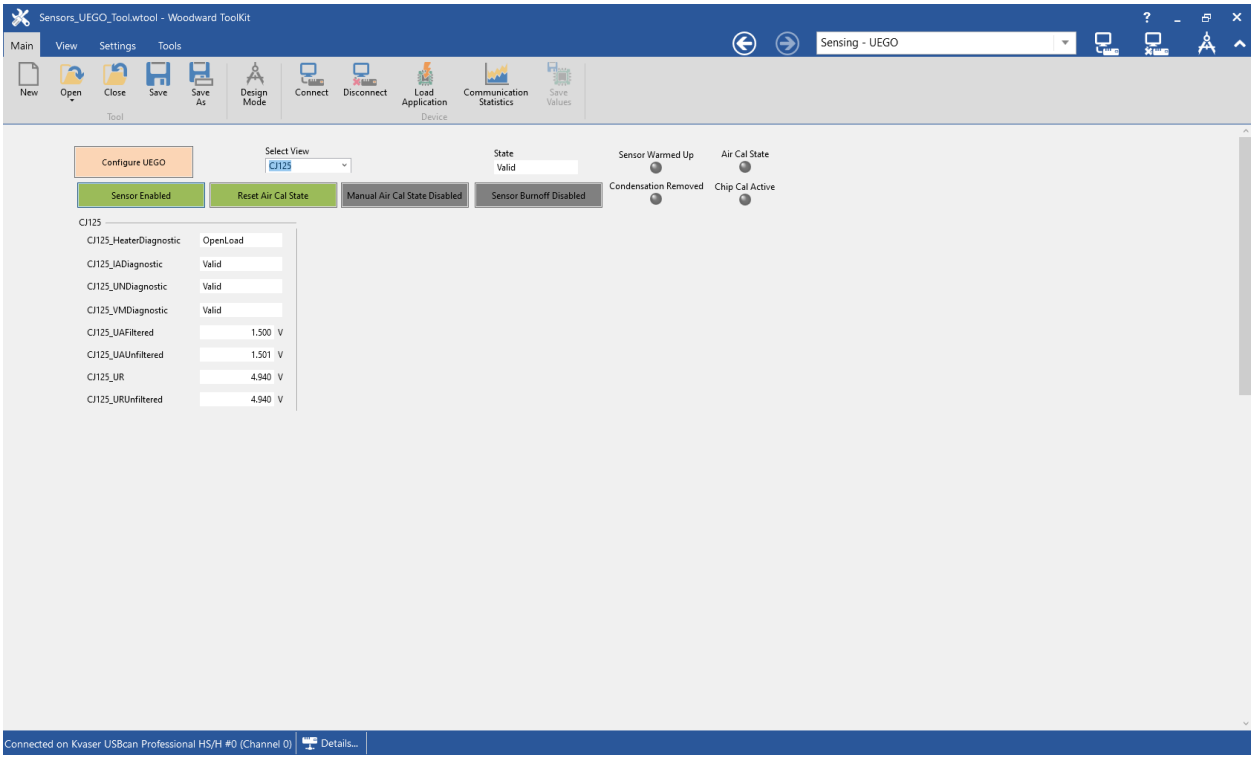
Below are a few of the Toolkit pages that will contain most of the defined parameters in the example model. Of course, the real sensors and data must be attached to complete it for the application that is being worked on. For that reason, red X's appear on some pages. Also, due to the diagnostics setup in the application, some values may display a red X.

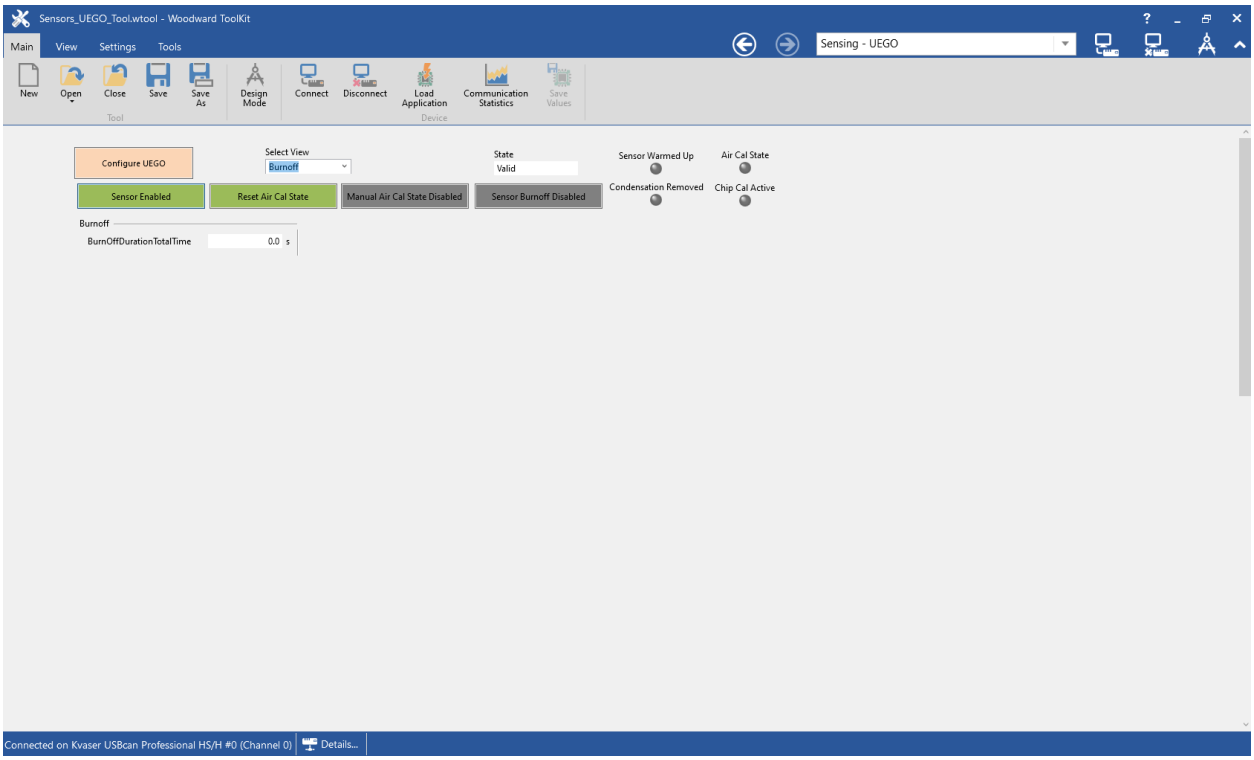

Sensors_UEGO_Tool.wtool - Woodward ToolKit
Main
View
Settings
Tools
Sensing - UEGO
New
Open
Close
Save
Save As
Design Mode
Connect
Disconnect
Load Application
Communication Statistics
Save Values
Configure UEGO
Select View
Diagnostics
Overview
Air Calibration
Chip Calibration
CJ125
Burnoff
Heater Control
Diagnostics
State
Valid
Sensor Warmed Up
Air Cal State
Condensation Removed
Chip Cal Active
Sensor Enabled
Reset Air Cal State
Manual Air Cal State Disabled
Sensor Burnoff Disabled
Sensor Diagnostic
Cause
Error
Active
Conditional Status
Inhibited
Calibration Monitor
Fault Actions
Allow Air Cal
Disable Air Cal
Use Previous Correction Factor
Use VEff & Hold Heater PID
Sensor Not Warmed Up
None
Air Cal Correction Factor Disabled
HMI Screen
Sensor Diagnostic Faults
CJ125 Heater Fault Status
CJ125 IA Fault Status
CJ125 UN Fault Status
CJ125 VM Fault Status
UA Range High Fault Status
UA Range Low Fault Status
UR Range High Fault Status
UR Range Low Fault Status
Sensor Inactive Fault Status
CJ125 Heater Fault Inhibit Status
UA Range Fault Inhibit Status
UR Range Fault Inhibit Status
UN-VM-IA Fault Inhibit Status
Sensor Inactive Fault Inhibit Status
Heater Diagnostic Faults
Heater Range High Fault Status
Heater Range Low Fault Status
Heater Control Fault Status
Heater Control Fault Inhibit Status
Heater Range High Fault Inhibit Status
Heater Range Low Fault Inhibit Status
Air Cal Limit Faults
Air Cal Lower Limit Fault Status
Air Cal Upper Limit Fault Status
Air Cal Limit Inhibit Status
Connected on Kvaser USBcan Professional HS/H #0 (Channel 0)
Details...

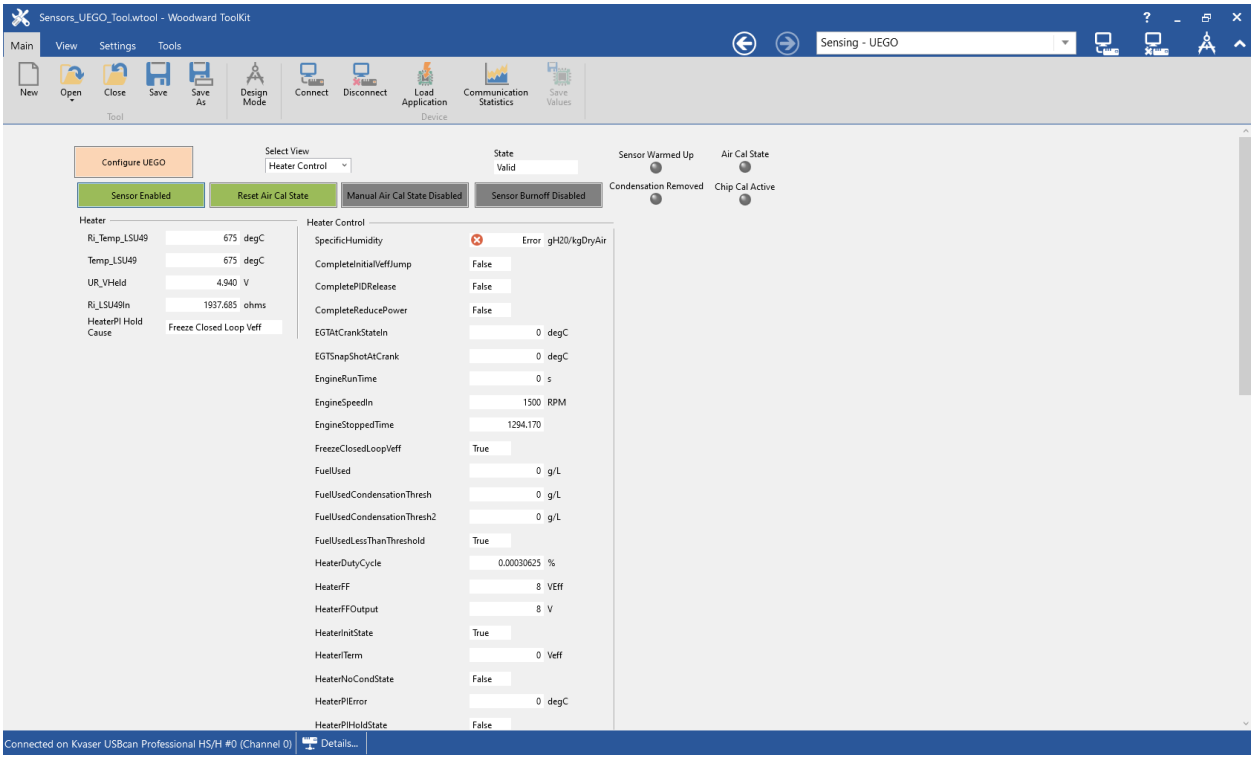

Sensors_UEGO_Tool.wtool - Woodward ToolKit
Main
View
Settings
Tools
Sensing - UEGO
Configure UEGO
Select View
Overview
State
Valid
Sensor Warmed Up
Air Cal State
Condensation Removed
Chip Cal Active
Sensor Enabled
Reset Air Cal State
Manual Air Cal State Disabled
Sensor Burnoff Disabled
Phi
Lambda
% O2
Ri Temp
Pumping Current
Heater Duty Cycle
UEGO Fault Causes
Sensor Fault
Heater Fault
Calibration Fault
AirCal Correction
Air Cal Counter
Start
Live
In
Out
Full
Defaults
Properties
Export
Clear
Select or enter a config name:
Save
Control
Zoom
Tools
Configurations
20 seconds
Show
Bold
Name
Value
Units
Scale Low
Scale High
Y1
Y2
Y Delta
Min
Max
Avg
Change/sec
Scale Position
Phi
% O2
Ri Temp
Chip Calibration Active
Connected on Kvaser USBcan Professional HS/H #0 (Channel 0)
Details...

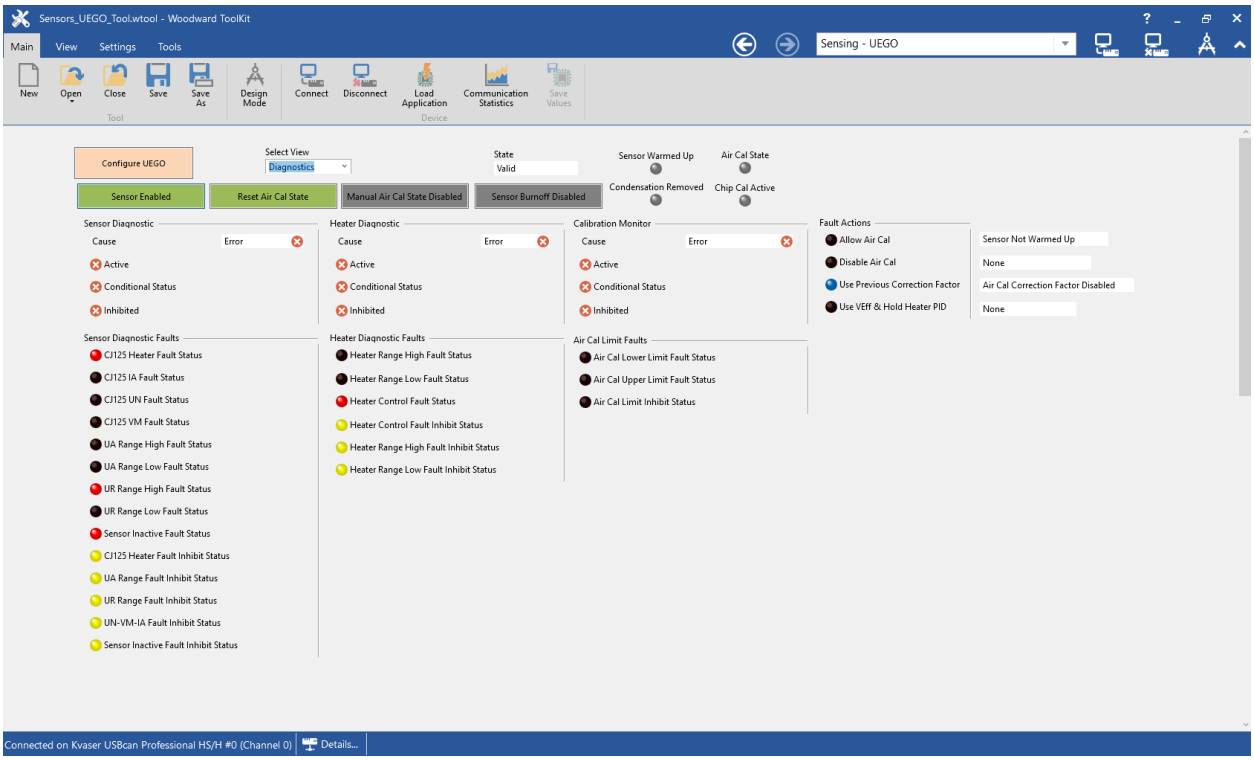

Sensors_UEGO_Tool.wtool - Woodward ToolKit
Main
View
Settings
Tools
Sensing - UEGO
New
Open
Close
Save
Save As
Design Mode
Connect
Disconnect
Load Application
Communication Statistics
Save Values
Tool
Device
Configure UEGO
Select View
Air Calibration
State
Valid
Sensor Warmed Up
Air Cal State
Condensation Removed
Chip Cal Active
Sensor Enabled
Reset Air Cal State
Manual Air Cal State Disabled
Sensor Burnoff Disabled
Enable
AirCal_Inhibitors
Manual Air Cal Conditions Not Valid
AirCalFSOTimer
0 sec
AirCalState
False
ManAirCalEnabled
False
ManAirCalTimeRemaining
3 s
ManAirCalTriggerState
False
Correction
AirCalCorrFactor
1
AirCalCorrPreLimit
0
AirCalCorrRaw
0
AirCalLearnLwrLimit
0.85
AirCalLearnUprLimit
1.15
Ip_AirIdealCorr
2.36729884 mA
Auto-Tuned Parameters
Ri_Gain
0.00240
Ri_Ofst
0.28907
Ip_Ofst
1.50088 V
Connected on Kvaser USBcan Professional HS/H #0 (Channel 0)
Details...


Sensors_UEGO_Tool.wtool - Woodward ToolKit
Main
View
Settings
Tools
Sensing - UEGO
New
Open
Close
Save
Save As
Design Mode
Connect
Disconnect
Load Application
Communication Statistics
Save Values
Tool
Device
Configure UEGO
Select View
Chip Calibration
State
Valid
Sensor Warmed Up
Air Cal State
Condensation Removed
Chip Cal Active
Sensor Enabled
Reset Air Cal State
Manual Air Cal State Disabled
Sensor Burnoff Disabled
Chip Calibration
ChipCalDwellTime
100 s
ChipCalDwellTimeIn
100 s
F3KState
True
HoldsFromRi
False
Ip_Ofst
1.50087476 V
Ip_OfstEnableTimeRemaining
1 s
IpOfstEnabled
False
IpOfstUAHold
False
IpOfstUAHoldTimeRemaining
0.2 s
LAState
False
PAState
False
RAState
False
Ri_GainEnableTimeRemaining
3 s
Ri_Ofst
0.2890721
Ri_OfstEnableTimeRemaining
1 s
RiGainEnabled
False
RiGainURHold
False
RiGainURHoldTimeRemaining
0.2 s
RiOfstEnabled
False
RiOfstURHold
False
RiOfstURHoldTimeRemaining
0.2 s
Connected on Kvaser USBcan Professional HS/H #0 (Channel 0)
Details...


Sensors_UEGO_Tool.wtool - Woodward ToolKit
Main
View
Settings
Tools
Sensing - UEGO
New
Open
Close
Save
Save As
Design Mode
Connect
Disconnect
Load Application
Communication Statistics
Save Values
Tool
Device
Configure UEGO
Select View
CJ125
State
Valid
Sensor Warmed Up
Air Cal State
Condensation Removed
Chip Cal Active
Sensor Enabled
Reset Air Cal State
Manual Air Cal State Disabled
Sensor Burnoff Disabled
CJ125
CJ125_HeaterDiagnostic
OpenLoad
CJ125_IADiagnostic
Valid
CJ125_UNDiagnostic
Valid
CJ125_VMDiagnostic
Valid
CJ125_UAFiltered
1.500 V
CJ125_UAUnfiltered
1.501 V
CJ125_UR
4.940 V
CJ125_URUnfiltered
4.940 V
Connected on Kvaser USBcan Professional HS/H #0 (Channel 0)
Details...


Sensors_UEGO_Tool.wtool - Woodward ToolKit
Main
View
Settings
Tools
Sensing - UEGO
New
Open
Close
Save
Save As
Design Mode
Connect
Disconnect
Load Application
Communication Statistics
Save Values
Tool
Device
Configure UEGO
Select View
Burnoff
State
Valid
Sensor Warmed Up
Air Cal State
Condensation Removed
Chip Cal Active
Sensor Enabled
Reset Air Cal State
Manual Air Cal State Disabled
Sensor Burnoff Disabled
Burnoff
BurnOffDurationTotalTime
0.0 s
Connected on Kvaser USBcan Professional HS/H #0 (Channel 0)
Details...


Sensors_UEGO_Tool.wtool - Woodward ToolKit
Main
View
Settings
Tools
Sensing - UEGO
New
Open
Close
Save
Save As
Design Mode
Connect
Disconnect
Load Application
Communication Statistics
Save Values
Configure UEGO
Select View
Heater Control
State
Valid
Sensor Warmed Up
Air Cal State
Condensation Removed
Chip Cal Active
Sensor Enabled
Reset Air Cal State
Manual Air Cal State Disabled
Sensor Burnoff Disabled
Heater
Ri_Temp_LSU49 675 degC
Temp_LSU49 675 degC
UR_VHeld 4.940 V
Ri_LSU49In 1937.685 ohms
HeaterPI Hold Cause Freeze Closed Loop Veff
Heater Control
SpecificHumidity Error gH20/kgDryAir
CompleteInitialVeffJump False
CompletePIDRelease False
CompleteReducePower False
EGTAtCrankStateIn 0 degC
EGTSnapShotAtCrank 0 degC
EngineRunTime 0 s
EngineSpeedIn 1500 RPM
EngineStoppedTime 1294.170
FreezeClosedLoopVeff True
FuelUsed 0 g/L
FuelUsedCondensationThresh 0 g/L
FuelUsedCondensationThresh2 0 g/L
FuelUsedLessThanThreshold True
HeaterDutyCycle 0.00030625 %
HeaterFF 8 VEff
HeaterFFOutput 8 V
HeaterInitState True
HeaterITerm 0 Veff
HeaterNoCondState False
HeaterPIError 0 degC
HeaterPIHoldState False
Connected on Kvaser USBcan Professional HS/H #0 (Channel 0)
Details...


Sensors_UEGO_Tool.wtool - Woodward ToolKit
Main
View
Settings
Tools
Sensing - UEGO
New
Open
Close
Save
Save As
Design Mode
Connect
Disconnect
Load Application
Communication Statistics
Save Values
Configure UEGO
Select View
Diagnostics
State
Valid
Sensor Warmed Up
Air Cal State
Condensation Removed
Chip Cal Active
Sensor Enabled
Reset Air Cal State
Manual Air Cal State Disabled
Sensor Burnoff Disabled
Sensor Diagnostic
Cause Error
Active
Conditional Status
Inhibited
Heater Diagnostic
Cause Error
Active
Conditional Status
Inhibited
Calibration Monitor
Cause Error
Active
Conditional Status
Inhibited
Fault Actions
Allow Air Cal
Disable Air Cal
Use Previous Correction Factor
Use VEff & Hold Heater PID
Sensor Not Warmed Up
None
Air Cal Correction Factor Disabled
None
Sensor Diagnostic Faults
CJ125 Heater Fault Status
CJ125 IA Fault Status
CJ125 UN Fault Status
CJ125 VM Fault Status
UA Range High Fault Status
UA Range Low Fault Status
UR Range High Fault Status
UR Range Low Fault Status
Sensor Inactive Fault Status
CJ125 Heater Fault Inhibit Status
UA Range Fault Inhibit Status
UR Range Fault Inhibit Status
UN-VM-IA Fault Inhibit Status
Sensor Inactive Fault Inhibit Status
Heater Diagnostic Faults
Heater Range High Fault Status
Heater Range Low Fault Status
Heater Control Fault Status
Heater Control Fault Inhibit Status
Heater Range High Fault Inhibit Status
Heater Range Low Fault Inhibit Status
Air Cal Limit Faults
Air Cal Lower Limit Fault Status
Air Cal Upper Limit Fault Status
Air Cal Limit Inhibit Status
Connected on Kvaser USBcan Professional HS/H #0 (Channel 0)
Details...

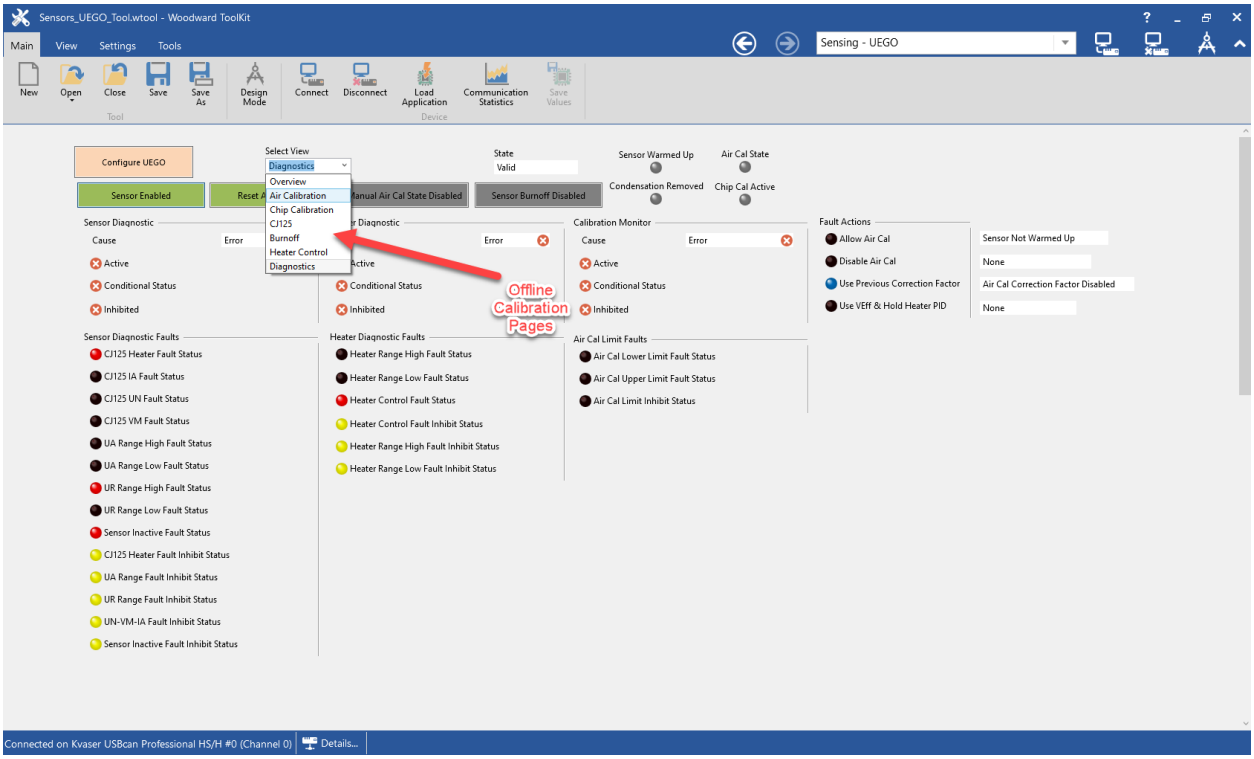


Figure 12-5. Examples of the Sensors-UEGO Toolkit Pages

Functional Description – UEGO Sensor

Introduction

The LESP Model Reference software logic supports the Bosch UEGO LSU 4.9 sensor. This universal exhaust gas oxygen (UEGO) sensor measures the oxygen content of the exhaust gas enabling the correct fuel-air ratio in the engine under all conditions. It has the ability to measure very rich to very lean fuel-air mixtures. In this application, it is used to feedback the fuel-air ratio to a closed loop AFR control logic done with the throttle valve. The following describes more details on this feature and how to set it up.

Toolkit Setup Page

The Toolkit tool can be used to setup the UEGO sensor. A special page contains the setup and logic that are required for correct UEGO sensor operation and performance.

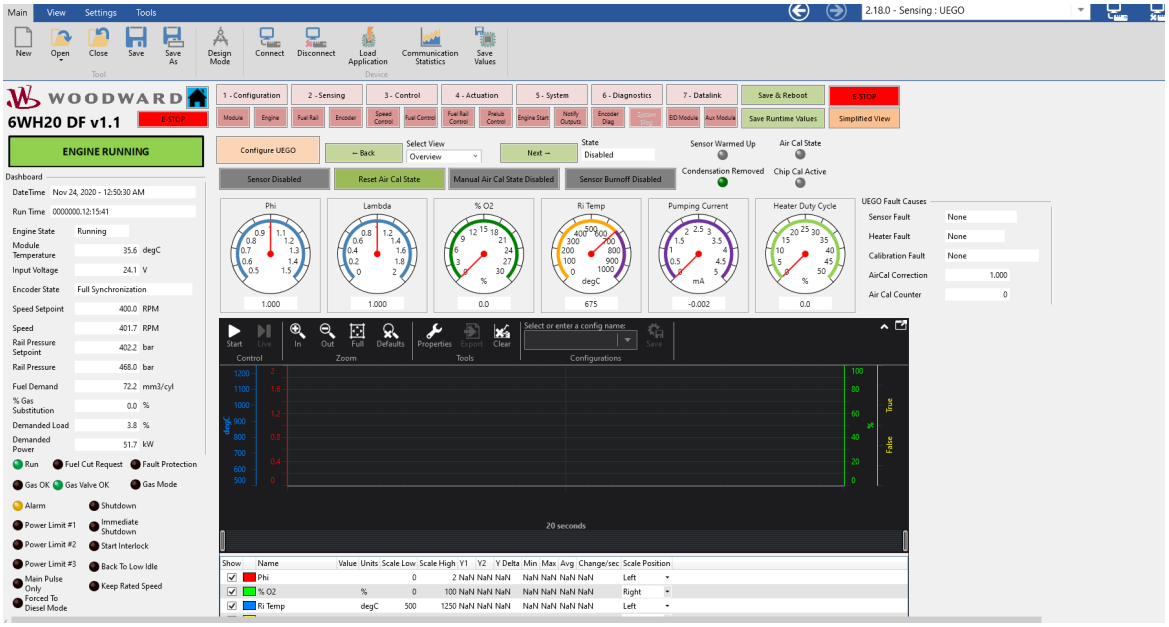


Figure 12-6. UEGO Sensor Toolkit Setup Page

By default, the sensor is disabled, but can be activated simply by pushing the enable button. When disabled, the sensor will not function at all.

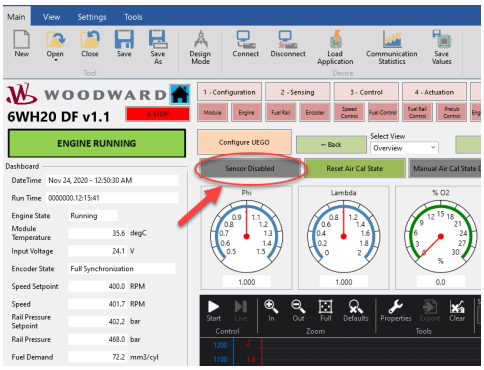


Figure 12-7. UEGO Sensor Activation / Deactivation

Once enabled while the engine is running, the sensor heater will be activated and controlled at the desired operating temperature of 780 degC. As long as the engine continues running, the sensor will remain active.

Once the engine is stopped, the heater will be disabled automatically, and the sensor will be switched off. If the engine is started again, the heater will be started, and the sensor will activate once the operating temperature has been reached.

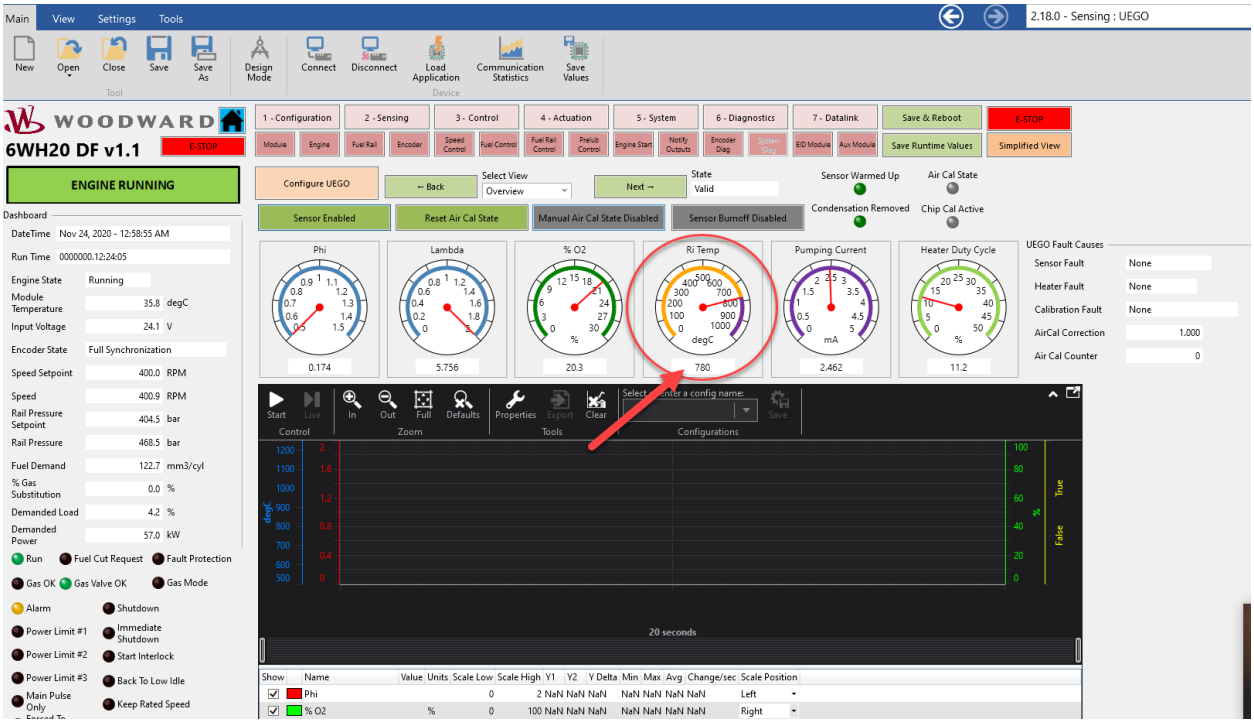


Figure 12-8. UEGO Sensor - Enabled

The example above shows the sensor being active in a simulator setup, where no exhaust gasses are available. This is the why the oxygen is at 20.3 % and no valid Phi / Lambda is displayed.

Manual Air Calibration

Over time, it is possible that the sensor reading may shift. This is a result of contamination that builds up on the element, increasing the amount of pump current required to pump oxygen. This causes the ECM to think the fuel-air mixture is too rich. To compensate, the ECM will reduce the amount of injected fuel. The reduction in fuel by the ECM could lead to higher NOx emissions and lean misfire. To avoid this situation, it is necessary to perform an air calibration of the sensor at regular intervals. Woodward recommends that an air calibration be performed at every oil change for maximum engine performance. An air calibration requires that the sensor be removed from the exhaust system and exposed to air. The environment must be free from exhaust gas or fuel vapors. Activation of manual air calibration is performed through the Woodward Service Tool. The sensor is heated by the ECM and adapted to the amount of oxygen in the air. Once complete, re-install the UEGO sensor in the exhaust system using the proper installation torque.

NOTICE

To prevent damage to the LECM or UEGO sensor, do not disconnect the sensor from the wiring harness until the LECM has shut down completely. This occurs approximately 30 seconds after switching the key to the off position.

Sensor Burn-Off

Once the air calibration adaptation has reached 10%, Woodward strongly recommends performing a burn-off of the sensor. If the air calibration adaptation is ≥ 15%, a burn-off procedure is mandatory. The burn-off feature raises the operating temperature of the sensor to burn off the contaminants that collect on the sensor over time. Burn-off is performed using the Woodward Service Tool. The sensor must be removed from the exhaust system and be allowed to hang freely in the air. The sensor will become very hot and should not contact anything. Burn-off occurs at 870°C for 6 to 15 minutes. During this procedure, it is important that the ramp rate for the effective heater voltage (Veff) not exceed 0.3 V/s. Higher gradients can damage the sensor and provides no advantage to the operator. After burn-off is complete, an air calibration must be performed so that a new air calibration factor can be learned. Once this is complete, the sensor can go back into operation.

⚠CAUTION

During air calibration and burn-off, the sensor element becomes very hot. Keep the sensor away from skin and components that can be damaged by heat. Use the proper personal protective equipment when handling the sensor.

Troubleshooting

The LECM performs active checks of the sensor depending upon the type of system. The LESP OBD system performs the most advanced checks including:

- Heater open
- Heater short
- Heater temperature lower-than-expected
- Heater temperature higher-than-expected
- Heater temperature control failure
- Sensor internal faults
- Air calibration failure
- Air calibration at upper-limit
- Air calibration at lower-limit

UEGO sensor faults may be system wiring problems and not sensor problems. Therefore, it is important to check the wiring harness between the ECM and the connector of the LSU. There are no serviceable parts in the LSU4.9. If any part of the sensor is damaged, the whole sensor must be replaced. If any of the wiring between the sensor and the sensor connector is damaged, the whole sensor must be replaced. The wire insulation is designed for high temperature environments and heat shrink or electrical tape will fail since they are not designed for high temperatures.

If the heater in the sensor is damaged, it will not reach the correct temperature for the sensor to function. If heater failure is suspected, check the resistance of the sensor by using an ohmmeter across pins 3 and 4. If the ohmmeter shows that the circuit is open, the heater is defective, and the entire sensor must be replaced. At room temperature (20 °C to 25 °C), the resistance should measure 3.2 ± 0.8 Ω. If the sensor is still warm from operation, the resistance will measure higher. Wait until the sensor has fully cooled and re-check the resistance to see if it is within specification.

Chapter 13.
Product Support and Service Options

Product Support Options

If you are experiencing problems with the installation, or unsatisfactory performance of a Woodward product, the following options are available:

1. Consult the troubleshooting guide in the manual.
2. Contact the **OE Manufacturer or Packager** of your system.
3. Contact the **Woodward Business Partner** serving your area.
4. Contact Woodward technical assistance via email (**EngineHelpDesk@Woodward.com**) with detailed information on the product, application, and symptoms. Your email will be forwarded to an appropriate expert on the product and application to respond by telephone or return email.
5. If the issue cannot be resolved, you can select a further course of action to pursue based on the available services listed in this chapter.

OEM or Packager Support: Many Woodward controls and control devices are installed into the equipment system and programmed by an Original Equipment Manufacturer (OEM) or Equipment Packager at their factory. In some cases, the programming is password-protected by the OEM or packager, and they are the best source for product service and support. Warranty service for Woodward products shipped with an equipment system should also be handled through the OEM or Packager. Please review your equipment system documentation for details.

Woodward Business Partner Support: Woodward works with and supports a global network of independent business partners whose mission is to serve the users of Woodward controls, as described here:

- A **Full-Service Distributor** has the primary responsibility for sales, service, system integration solutions, technical desk support, and aftermarket marketing of standard Woodward products within a specific geographic area and market segment.
- An **Authorized Independent Service Facility (AISF)** provides authorized service that includes repairs, repair parts, and warranty service on Woodward's behalf. Service (not new unit sales) is an AISF's primary mission.
- A **Recognized Engine Retrofitter (RER)** is an independent company that does retrofits and upgrades on reciprocating gas engines and dual-fuel conversions, and can provide the full line of Woodward systems and components for the retrofits and overhauls, emission compliance upgrades, long term service contracts, emergency repairs, etc.

A current list of Woodward Business Partners is available at www.woodward.com/local-partner.

Product Service Options

Depending on the type of product, the following options for servicing Woodward products may be available through your local Full-Service Distributor or the OEM or Packager of the equipment system.

- Replacement/Exchange (24-hour service)
- Flat Rate Repair
- Flat Rate Remanufacture

Replacement/Exchange: Replacement/Exchange is a premium program designed for the user who is in need of immediate service. It allows you to request and receive a like-new replacement unit in minimum time (usually within 24 hours of the request), providing a suitable unit is available at the time of the request, thereby minimizing costly downtime.

This option allows you to call your Full-Service Distributor in the event of an unexpected outage, or in advance of a scheduled outage, to request a replacement control unit. If the unit is available at the time of the call, it can usually be shipped out within 24 hours. You replace your field control unit with the like-new replacement and return the field unit to the Full-Service Distributor.

Flat Rate Repair: Flat Rate Repair is available for many of the standard mechanical products and some of the electronic products in the field. This program offers you repair service for your products with the advantage of knowing in advance what the cost will be.

Flat Rate Remanufacture: Flat Rate Remanufacture is very similar to the Flat Rate Repair option, with the exception that the unit will be returned to you in “like-new” condition. This option is applicable to mechanical products only.

Returning Equipment for Repair

If a control (or any part of an electronic control) is to be returned for repair, please contact your Full-Service Distributor in advance to obtain Return Authorization and shipping instructions.

When shipping the item(s), attach a tag with the following information:
- Return number
- Name and location where the control is installed
- Name and phone number of contact person
- Complete Woodward part number(s) and serial number(s)
- Description of the problem
- Instructions describing the desired type of repair

Packing a Control

Use the following materials when returning a complete control:
- Protective caps on any connectors
- Antistatic protective bags on all electronic modules
- Packing materials that will not damage the surface of the unit
- At least 100 mm (4 inches) of tightly packed, industry-approved packing material
- A packing carton with double walls
- A strong tape around the outside of the carton for increased strength

NOTICE

To prevent damage to electronic components caused by improper handling, read and observe the precautions in Woodward manual 82715, *Guide for Handling and Protection of Electronic Controls, Printed Circuit Boards, and Modules*.

Replacement Parts

When ordering replacement parts for controls, include the following information:
- the part number(s) (XXXX-XXXX) that is on the enclosure nameplate;
- the unit serial number, which is also on the nameplate.

Engineering Services

Woodward's Full-Service Distributors offer various Engineering Services for our products. For these services, you can contact the Distributor by telephone or by email.

- Technical Support
- Product Training
- Field Service

Technical Support is available from your equipment system supplier, your local Full-Service Distributor, or from many of Woodward's worldwide locations, depending upon the product and application. This service can assist you with technical questions or problem solving during the normal business hours of the Woodward location you contact.

Product Training is available as standard classes at many Distributor locations. Customized classes are also available, which can be tailored to your needs and held at one of our Distributor locations or at your site. This training, conducted by experienced personnel, will assure that you will be able to maintain system reliability and availability.

Field Service engineering on-site support is available, depending on the product and location, from one of our Full-Service Distributors. The field engineers are experienced both on Woodward products as well as on much of the non-Woodward equipment with which our products interface.

For information on these services, please contact one of the Full-Service Distributors listed at www.woodward.com/local-partner.

Contacting Woodward's Support Organization

For the name of your nearest Woodward Full-Service Distributor or service facility, please consult our worldwide directory at www.woodward.com/support, where you may also find the most current product support and contact information.

You can also contact the Woodward Customer Service Department at one of the following Woodward facilities to obtain the address and phone number of the nearest facility at which you can obtain information and service.

Products Used in Electrical Power Systems

Facility	Phone Number
Brazil	+55 (19) 3708 4800
China	+86 (512) 8818 5515
Germany:	+49 (711) 78954-510
India	+91 (124) 4399500
Japan	+81 (43) 213-2191
Korea	+82 (32) 422-5551
Poland	+48 (12) 295 13 00
United States	+1 (970) 482-5811

Products Used in Engine Systems

Facility	Phone Number
Brazil	+55 (19) 3708 4800
China	+86 (512) 8818 5515
Germany	+49 (711) 78954-510
India	+91 (124) 4399500
Japan	+81 (43) 213-2191
Korea	+ 82 (32) 422-5551
The Netherlands-	+31 (23) 5661111
United States	+1 (970) 482-5811

Products Used in Industrial Turbomachinery Systems

Facility	Phone Number
Brazil	+55 (19) 3708 4800
China	+86 (512) 8818 5515
India	+91 (124) 4399500
Japan	+81 (43) 213-2191
Korea	+ 82 (32) 422-5551
The Netherlands-	+31 (23) 5661111
Poland	+48 (12) 295 13 00
United States	+1 (970) 482-5811

Technical Assistance

If you need to contact technical assistance, you will need to provide the following information. Please write it down here before contacting the Engine OEM, the Packager, a Woodward Business Partner, or the Woodward factory:

General

Your Name _____

Site Location _____

Phone Number _____

Fax Number _____

Prime Mover Information

Manufacturer _____

Engine Model Number _____

Number of Cylinders _____

Type of Fuel (gas, gaseous, diesel, dual-fuel, etc.) _____

Power Output Rating _____

Application (power generation, marine, etc.) _____

Control/Governor Information

Control/Governor #1

Woodward Part Number & Rev. Letter _____

Control Description or Governor Type _____

Serial Number _____

Control/Governor #2

Woodward Part Number & Rev. Letter _____

Control Description or Governor Type _____

Serial Number _____

Control/Governor #3

Woodward Part Number & Rev. Letter _____

Control Description or Governor Type _____

Serial Number _____

Symptoms

Description _____

_____

If you have an electronic or programmable control, please have the adjustment setting positions or the menu settings written down and with you at the time of the call.

Technical Specifications

Revision History

Changes in Revision -

- New manual

Declarations

We appreciate your comments about the content of our publications.

Send comments to: industrial.support@woodward.com

Please reference publication 35203V2.

B 3 5 2 0 3 V 2 : -

PO Box 1519, Fort Collins CO 80522-1519, USA
1041 Woodward Way, Fort Collins CO 80524, USA
Phone +1 (970) 482-5811

Email and Website—www.woodward.com

Woodward has company-owned plants, subsidiaries, and branches, as well as authorized distributors and other authorized service and sales facilities throughout the world.

Complete address / phone / fax / email information for all locations is available on our website.