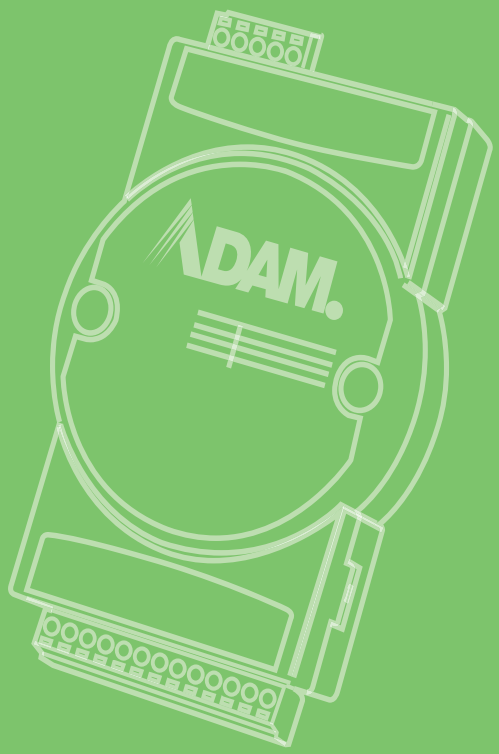
User Manual



ADAM-6000 Series

Ethernet-based Data Acquisition and Control Modules

ADVANTECH

Enabling an Intelligent Planet

Copyright

The documentation and the software included with this product are copyrighted 2022 by Advantech Co., Ltd. All rights are reserved. Advantech Co., Ltd. reserves the right to make improvements in the products described in this manual at any time without notice. No part of this manual may be reproduced, copied, translated, or transmitted in any form or by any means without the prior written permission of Advantech Co., Ltd. The information provided in this manual is intended to be accurate and reliable. However, Advantech Co., Ltd. assumes no responsibility for its use, nor for any infringements of the rights of third parties that may result from its use.

Acknowledgments

Intel and Pentium are trademarks of Intel Corporation.

Microsoft Windows and MS-DOS are registered trademarks of Microsoft Corp.

All other product names or trademarks are properties of their respective owners.

Product Warranty

Advantech warrants the original purchaser that each of its products will be free from defects in materials and workmanship for two years from the date of purchase.

This warranty does not apply to any products that have been repaired or altered by persons other than repair personnel authorized by Advantech, or products that have been subject to misuse, abuse, accident, or improper installation. Advantech assumes no liability under the terms of this warranty as a consequence of such events.

Because of Advantech's high quality-control standards and rigorous testing, most customers never need to use our repair service. If an Advantech product is defective, it will be repaired or replaced free of charge during the warranty period. For out-of-warranty repairs, customers will be billed according to the cost of replacement materials, service time, and freight. Please consult your dealer for more details.

If you believe your product to be defective, follow the steps outlined below.

1. Collect all the information about the problem encountered. (For example, CPU speed, Advantech products used, other hardware and software used, etc.) Note anything abnormal and list any onscreen messages displayed when the problem occurs.
2. Call your dealer and describe the problem. Please have your manual, product, and any helpful information readily available.
3. If your product is diagnosed as defective, obtain a return merchandize authorization (RMA) number from your dealer. This allows us to process your return more quickly.
4. Carefully pack the defective product, a completed Repair and Replacement Order Card, and a proof of purchase date (such as a photocopy of your sales receipt) into a shippable container. Products returned without a proof of purchase date are not eligible for warranty service.
5. Write the RMA number clearly on the outside of the package and ship the package prepaid to your dealer.

Part No. 2003600006 Edition 12

Printed in Taiwan November 2022

Declaration of Conformity

CE

This product has passed the CE test for environmental specifications when shielded cables are used for external wiring. We recommend the use of shielded cables. This type of cable is available from Advantech. Please contact your local supplier for ordering information.

Test conditions for passing also include the equipment being operated within an industrial enclosure. In order to protect the product from damage caused by electrostatic discharge (ESD) and EMI leakage, we strongly recommend the use of CE-compliant industrial enclosure products.

FCC Class A

This equipment has been tested and found to comply with the limits for a Class A digital device, pursuant to part 15 of the FCC Rules. These limits are designed to provide reasonable protection against harmful interference when the equipment is operated in a commercial environment. This equipment generates, uses, and can radiate radio frequency energy and, if not installed and used in accordance with the instruction manual, may cause harmful interference to radio communications. Operation of this equipment in a residential area is likely to cause harmful interference. In this event, users are required to correct the interference at their own expense.

Technical Support and Assistance

1. Visit the Advantech website at www.advantech.com/support to obtain the latest product information.
2. Contact your distributor, sales representative, or Advantech's customer service center for technical support if you need additional assistance. Please have the following information ready before calling:
   - Product name and serial number
   - Description of your peripheral attachments
   - Description of your software (operating system, version, application software, etc.)
   - A complete description of the problem
   - The exact wording of any error messages

Safety Instructions

1. Read these safety instructions carefully.
2. Retain this user manual for future reference.
3. Disconnect the equipment from all power outlets before cleaning. Use only a damp cloth for cleaning. Do not use liquid or spray detergents.
4. For pluggable equipment, the power outlet socket must be located near the equipment and easily accessible.
5. Protect the equipment from **humidity**.
6. Place the equipment on a reliable surface during installation. Dropping or letting the equipment fall may cause damage.
7. The openings on the enclosure are for air convection. Protect the equipment from overheating. Do not cover the openings.
8. Ensure that the voltage of the power source is correct before connecting the equipment to a power outlet.
9. Position the power cord away from high-traffic areas. Do not place anything over the power cord.
10. All cautions and warnings on the equipment should be noted.
11. If the equipment is not used for a long time, disconnect it from the power source to avoid damage from transient overvoltage.
12. Never pour liquid into an opening. This may cause fire or electrical shock.
13. Never open the equipment. For safety reasons, the equipment should be opened only by qualified service personnel.
14. If any of the following occurs, have the equipment checked by service personnel:
    - The power cord or plug is damaged.
    - Liquid has penetrated the equipment.
    - The equipment has been exposed to moisture.
    - The equipment is malfunctioning, or does not operate according to the user manual.
    - The equipment has been dropped and damaged.
    - The equipment shows obvious signs of breakage.
15. Do not leave the equipment in an environment with a storage temperature of below -20 °C (-4 °F) or above 60 °C (140 °F) as this may damage the components. The equipment should be kept in a controlled environment.
16. CAUTION: Batteries are at risk of exploding if incorrectly replaced. Replace only with the same or equivalent type as recommended by the manufacturer. Discard used batteries according to the manufacturer's instructions.
17. In accordance with IEC 704-1:1982 specifications, the sound pressure level at the operator's position does not exceed 70 dB (A).

DISCLAIMER: These instructions are provided according to IEC 704-1 standards. Advantech disclaims all responsibility for the accuracy of any statements contained herein.

Contents

Chapter 1 Understanding Your System 1

1.1 Introduction 2
Figure 1.1 ADAM-6000 Module System Architecture 2
1.2 Major Features 2
1.3 Specifications 4
1.4 Dimensions 4
Figure 1.2 ADAM-6000 Module Dimensions 4
1.5 LED Status 5

Chapter 2 Hardware Selection Guidelines 7

2.1 Selecting an I/O Module 8
Table 2.1: I/O Selection Guidelines 8
2.2 Selecting a Link Terminal and Cable 9
Figure 2.1 Connecting ADAM-6000 Modules to an Ethernet Terminal via Cable 9
Table 2.2: Ethernet RJ-45 Port Pin Assignment Chart 9
2.3 Selecting an Operator Interface 10

Chapter 3 Hardware Installation Guide 11

3.1 Interface Introduction 12
3.2 Mounting Options 13
3.2.1 Panel Mounting 13
Figure 3.1 Panel Mounting Bracket Dimensions 13
Figure 3.2 How to Fix a Module on the Mounting Bracket 13
3.2.2 DIN Rail Mounting 14
Figure 3.3 How to Fix a Module on the DIN Rail Adapter 14
Figure 3.4 How to Secure a Module to a DIN Rail 14
3.3 Wiring and Connections 15
3.3.1 Power Supply Wiring 15
Figure 3.5 How to Connect the Module Power Wires 15
3.3.2 I/O Module Wiring 16

Chapter 4 Introduction to Analog ADAM-6000 I/O Modules17

4.1 Analog Input Modules 18
4.2 ADAM-6015 7-ch Isolated RTD Input Module 18
4.2.1 Specifications 18
4.2.2 Application Wiring 20
Figure 4.1 ADAM-6015 RTD Input Wiring 20
4.2.3 Address Assignment 20
4.3 ADAM-6017 8-ch Analog Input/2-ch Digital Output Module 20
4.3.1 Specifications 20
Figure 4.2 ADAM-6017 Jumper Switches 22
4.3.2 Application Wiring 23
Figure 4.3 ADAM-6017 Analog Input Wiring 23
Figure 4.4 ADAM-6017 Analog Input Type Setting 23
Figure 4.5 ADAM-6017 Digital Output Wiring 23
4.3.3 Address Assignment 24

4.4 ADAM-6018 Isolated Thermocouple Input/8-ch Digital Output Module .. 24
Figure 4.6 ADAM-6018 8-ch Thermocouple Input 24
4.4.1 Specifications 25
4.4.2 Application Wiring 26
Figure 4.7 ADAM-6018 Thermocouple Input Wiring 26
Figure 4.8 ADAM-6018 Digital Output Wiring 26
4.4.3 Address Assignment 26
4.5 ADAM-6018+ 8-ch Isolated Thermocouple Input module 27
4.5.1 Specifications 27
4.5.2 Application Wiring 28
Figure 4.9 ADAM-6018+ thermocouple wiring 28
4.6 ADAM-6024 12-ch Isolated Universal I/O Module 28
4.6.1 Specifications 28
Figure 4.10ADAM-6024-A1E Jumper Settings 30
Figure 4.11ADAM-6024-D Jumper Settings 30
4.6.2 Application Wiring 31
Figure 4.12ADAM-6024 Analog I/O Wiring 31
Figure 4.13ADAM-6024 Digital Input Wiring 31
Figure 4.14ADAM-6024 Digital Output Wiring 32
4.6.3 Address Assignment 32

Chapter 5 Introduction to Digital ADAM-6000 I/O Modules33

5.1 Digital I/O and Relay Modules 34
5.2 ADAM-6050 18-ch Isolated Digital I/O Module 34
5.2.1 Specifications 34
5.2.2 Application Wiring 35
Figure 5.1 ADAM-6050 Digital Input Wiring 35
Figure 5.2 ADAM-6050 Digital Output Wiring 35
5.2.3 Address Assignment 36
5.3 ADAM-6051 14-ch Isolated Digital I/O Module w/2-ch Counter 36
5.3.1 Specifications 36
5.3.2 Application Wiring 37
Figure 5.3 ADAM-6051 Digital Input Wiring 37
Figure 5.4 ADAM-6051 Counter (Frequency) Input 38
Figure 5.5 ADAM-6051 Digital Output Wiring 38
5.3.3 Address Assignment 38
5.4 ADAM-6052 16-ch Source-Type Isolated Digital I/O Module 38
5.4.1 Specifications 39
Figure 5.6 ADAM-6052 Jumper Settings 40
5.4.2 Application Wiring 41
Figure 5.7 ADAM-6052 Digital Input Wiring 41
Figure 5.8 ADAM-6052 Digital Output Wiring 42
5.4.3 Address Assignment 42
5.5 ADAM-6060 6-ch Digital Input/6-ch Relay Module 42
5.5.1 Specifications 42
5.5.2 Application Wiring 44
Figure 5.9 ADAM-6060 Digital Input Wiring 44
Figure 5.10ADAM-6060 Relay Output Wiring 44
5.5.3 Address Assignment 44
5.6 ADAM-6066 6-ch Digital Input/6-ch Power Relay Module 45
5.6.1 Specifications: 45
5.6.2 Application Wiring 46
Figure 5.11ADAM-6066 Digital Input Wiring 46
Figure 5.12ADAM-6066 Relay Output Wiring 46
5.7 Digital Output Diagnostic Function 47
5.7.1 How to Obtain the Digital Output Diagnostic Status 48
Figure 5.13Abnormal DO Diagnostic Status 48

Figure 5.14 Normal DO Diagnostic Status 49

Chapter 6 System Configuration Guide 51

6.1 System Requirements 52
6.2 Installing Adam/Apax .NET Utility 52
6.3 Adam/Apax .NET Utility Overview 52
Figure 6.1 Adam/Apax .NET Utility Operation Window 53
6.3.1 Menu Bar 53
6.3.2 Toolbar 54
Figure 6.2 Adam/Apax .NET Utility Toolbar 54
6.3.3 Module Tree Display Area 55
Figure 6.3 Adam/Apax .NET Utility Module Display Area 55
6.3.4 Status Display Area 55
6.3.5 Configuration of ADAM-6000 Modules 56
Figure 6.4 Adam/Apax .NET Utility - Searching for Devices 56
6.3.6 Group Configuration 64
6.3.7 I/O Configuration 66
Figure 6.5 All-Channel, Individual Channel, and GCL Configuration Controls 66
6.4 Analog Input Modules (ADAM-6015, ADAM-6017, and ADAM-6018, ADAM-6018+) 67
6.4.1 All-Channel Configuration 67
Figure 6.6 Channels Range Configuration Area 67
Figure 6.7 Analog Input Trend Log 70
6.4.2 Individual Channel Configuration 72
Figure 6.8 Analog Input Alarm Mode Configuration 72
6.5 Universal I/O Modules (ADAM-6024) 74
6.5.1 All-Channel Configuration 74
Figure 6.9 ADAM-6024 Channel Configuration 74
Figure 6.10 ADAM-6024 Output Tab 75
6.6 Universal Digital I/O Modules (ADAM-6050, ADAM-6051- ADAM-6052, ADAM-6060, ADAM-6066) 75
6.6.1 All-Channel Configuration 75
6.6.2 Individual Channel Configuration 77
Figure 6.11 Digital Input Modes 77
Figure 6.12 Digital Output Modes 80
Figure 6.13 Graph Explaining Low to High Delay Output Mode 82
Figure 6.14 Graph Explaining Low to High Delay Output Mode 83
6.7 Introduction to P2P Functions 84
6.7.1 P2P Communication Modes 84
Figure 6.15 Basic Mode for P2P 84
Figure 6.16 Advanced mode for P2P 85
6.7.2 P2P Communication Methods 85
6.7.3 P2P Event Triggers 86
6.8 How to Configure P2P Functions 86
Figure 6.17 Peer to Peer/Event Tab 86
6.8.1 Basic Mode Configuration 87
Figure 6.18 P2P Basic Mode Configuration 87
6.8.2 Advanced Mode Configuration 88
Figure 6.19 P2P Advanced Mode Configuration 88
Figure 6.20 Copy One Setting to Other Channels 89
6.9 ADAM-6000 Web Server 90
6.9.1 HTML 5 90

Chapter 7 Planning Your Application Program 93

7.1 Introduction 94
7.2 ADAM .NET Class Library 94

Figure 7.1 Modifying ADAM-6050 .NET 95
Figure 7.2 Execute the sample code and configure your ADAM module 95
7.3 Modbus Protocol for ADAM-6000 Modules 96
7.3.1 Modbus Protocol Structure 96
7.3.2 Modbus Function Code Introductions 96
7.4 ASCII Commands for ADAM-6000 Modules 101
7.4.1 ASCII Syntax 101
7.4.2 System Command Set 101
7.4.3 Analog Input Command Set 106
7.4.4 Analog Input Alarm Command Set 117
7.4.5 Universal I/O Command Set 122
7.4.6 Digital I/O Command Set 129
7.5 SNMP for ADAM-6000 Modules 132
7.5.1 ADAM MIB file 132
7.5.2 SNMP Trap Configuration 132
Figure 7.3 Trap Configuration Using Adam/Apax .NET Utility 133
7.5.3 SNMP OID Value 134
7.6 MQTT for ADAM-6000 modules 135
7.6.1 Introduction of MQTT 135
7.6.2 MQTT Format for ADAM module 135
7.6.3 MQTT Configuration 140
7.6.4 How to Start MQTT with ADAM-6000 Modules 145
7.6.5 Real-Time Clock 147
7.6.6 SNTP Configuration Using Adam/Apax .NET Utility 147
7.6.7 SNTP Configuration Using ASCII Commands 148

Chapter 8 Graphic Condition Logic (GCL) 149

8.1 Overview 150
8.2 GCL Configuration Environment 150
Figure 8.1 GCL Configuration Environment 151
Figure 8.2 Four Stages for One Logic Rule 152
8.3 Configuring the Four Stages of a Logic Rule 153
8.3.1 Input Condition Stage 153
Figure 8.3 Input Condition Stage Configuration 154
Figure 8.4 Scaling Function of Analog Input Mode 154
Figure 8.5 Engineer Unit and Current Value 155
8.3.2 Logic Stage 157
Figure 8.6 Logic Stage Configuration 157
8.3.3 Execution Stage 158
Figure 8.7 Execution Stage Configuration 159
Figure 8.8 Send to Next Rule Function 160
Figure 8.9 The Next Logic Rule 160
8.3.4 Output Stage 161
Figure 8.10 Output Stage Configuration 161
Figure 8.11 Remote Message Output 164
8.4 Internal Flag for Logic Cascade and Feedback 166
8.4.1 Logic Cascade 166
Figure 8.12 Local Logic Cascade Architecture 166
Figure 8.13 Configuration of Logic Rule 1 166
Figure 8.14 Configuration of Logic Rule 2 167
Figure 8.15 Configuration of Logic Rule 3 167
Figure 8.16 Distributed Logic Cascade 168
Figure 8.17 Configuration of Logic Rule 1 168
Figure 8.18 Configuration of Logic Rule 2 168
Figure 8.19 Configuration of Logic Rule 3 168
8.4.2 Feedback 169
Figure 8.20 Building Logic Feedback 169

8.5 Logic Download and Online Monitoring 169
Figure 8.21 Online Monitoring Function 170
Figure 8.22 GCL Execution Sequence 171
8.6 Typical Applications with GCL 171
Figure 8.23 Ladder Diagram for On/Off Control 172
Figure 8.24 GCL Logic for On/Off Control 172
Figure 8.25 Time Chart for Sequence Control 173
Figure 8.26 GCL Logic for Sequence Control (Turns On in Sequence and Remains On) 173
Figure 8.27 Time Chart for 12 Digital Inputs to 1 Digital Output 174
Figure 8.28 GCL Logic for 12 Digital Inputs to 1 Digital Output . 174
Figure 8.29 Time Chart for Flicker Applications 175
Figure 8.30 GCL Logic for Flicker 175
Figure 8.31 Time Chart for Rising Edge 175
Figure 8.32 Ladder Diagram for Rising Edge 176
Figure 8.33 GCL Logic for Rising Edge 176
Figure 8.34 Time Chart for Falling Edge 177
Figure 8.35 Ladder Diagram for Falling Edge 177
Figure 8.36 GCL Logic for Falling Edge 177
Figure 8.37 Time Chart for Sequence Control (Continuously Turn On and Off in Sequence) 178
Figure 8.38 GCL Logic for Sequence Control (Continuously Turn On and Off in Sequence) 178
Figure 8.39 GCL Logic for Event Trigger (Only Occurs Once) .. 179
Figure 8.40 Event Trigger Configuration (Only Occurs Once) ... 179

Appendix A Design Worksheets 181

Appendix B Data Formats and I/O Ranges 185

B.1 ADAM-6000 Command Data Formats 186
Figure B.1 Request Comment Structure 186
Figure B.2 Response Comment Structure 186
B.2 ADAM-6000 I/O Modbus Mapping Tables 191

Appendix C Grounding Reference 221

C.1 Field Grounding and Shielding Application 222
C.1.1 Overview 222
C.2 Grounding 222
C.2.1 The Earth as a Reference 222
Figure C.1 Thinking of the Earth as a Ground 222
C.2.2 Frame Grounds and Grounding Bars 223
Figure C.2 Grounding Bar 223
Figure C.3 Figure C.3: Normal and Common Mode 223
C.2.3 Normal Mode and Common Mode 223
Figure C.4 Normal and Common Mode 224
C.2.4 Wire impedance 224
Figure C.5 High Voltage Transmission 224
Figure C.6 Wire Impedance 225
C.2.5 Single-Point Grounding 225
Figure C.7 Single-Point Grounding 225
Figure C.8 Single point grounding 225
C.3 Shielding 226
C.3.1 Cable Shield 226
Figure C.9 Single Isolation Cable 226
Figure C.10 Double Isolation Cable 226

C.3.2 System Shielding 227
Figure C.11System Shielding 227
Figure C.12The Characteristics of the Cable 227
Figure C.13System Shielding (1) 228
Figure C.14System Shielding (2) 228
C.4 Noise Reduction Techniques 228
Figure C.15Noise Reduction Techniques 229
C.5 Checklist 229

Appendix D REST for ADAM-6000 231

D.1 REST Introduction 232
D.2 REST Resources for ADAM 232
D.2.1 Analoginput 232
D.2.2 Analogoutput 233
D.2.3 Digitalinput 234
D.2.4 Digitaloutput 235
D.2.5 Counter 236

Chapter 1

Understanding Your System

1.1 Introduction

ADAM-6000 series Ethernet-based data acquisition and control (DA&C) modules provide I/O, data acquisition, and networking capabilities in one module, allowing you to build a cost-effective distributed monitoring and control solution for a wide variety of applications. Through a standard Ethernet network, ADAM-6000 modules can retrieve I/O values from sensors and can publish them as real-time I/O values to networking nodes via LAN, intranet, or the Internet. With Ethernet-enabled technology, ADAM-6000 modules allow you to build up a cost-effective DA&C system for building automation, environmental monitoring, facility management, and e-manufacturing applications. Figure 1-1 gives a brief overview of a system architecture that can be adopted for ADAM-6000 modules.

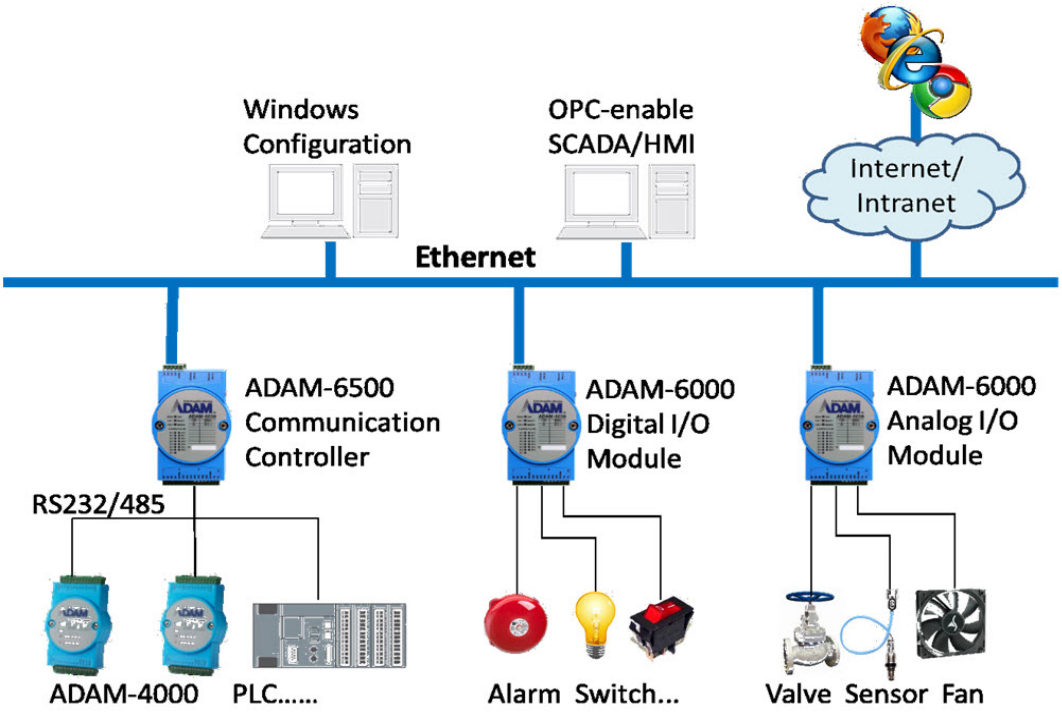


Figure 1.1 ADAM-6000 Module System Architecture

1.2 Major Features

Ethernet-Enabled DA&C I/O Modules

ADAM-6000 modules are based on the widely utilized Ethernet networking standard, which is employed in most business environments. You can easily add ADAM-6000 series I/O modules to existing Ethernet networks or use them in new Ethernet-enabled e-manufacturing networks. This series of modules supports 10/100 Mbps Ethernet and Modbus/TCP over TCP/IP for data connectivity, and UDP over Ethernet. With UDP/IP, ADAM-6000 series I/O modules can actively transmit data streams to up to eight Ethernet nodes. Through Ethernet networking, HMI/SCADA systems, and controllers, you can access and acquire real-time data from ADAM-6000 Ethernet-enabled DA&C modules. The data can then be integrated with business systems to derive valuable business information.

Note! *Some intelligent functions are only provided with the ADAM-6000-CE version. See Appendix F for further details.*

Intelligent I/O Modules

Upgraded from traditional I/O modules, all ADAM-6000 series modules have pre-built intelligent functions that can enhance system capabilities. For example, the digital input modules provide counter and totalizer functions; the digital output modules provide pulse output and delay output functions; the analog input modules provide descriptive statistical data calculations (e.g., min., max., and mean); and the analog output modules provide a PID loop control function.

Mixed I/O for All Applications

The ADAM-6000 series' mixed I/O design provides a cost-effective I/O option for application systems. The most commonly used I/O types for single-function units are available in a single module. This design concept not only saves I/O usage as well as costs, but it also speeds up I/O operations. For small DA&C system or standalone control units in medium-large systems, the ADAM-6000 series' mixed I/O design can easily fit your application needs with only one or two modules. With additional embedded control modules, these modules can be used to easily create a localized, less complex, and more distributed I/O architecture.

Remote Monitoring and Diagnosis

Previous differences in communication modes and data formats made it difficult to implement automation control and monitoring in IT-based infrastructure. In particular, users had to convert data to transform I/O datastreams from SCADA systems before transfer to a database or IT management system.

ADAM-6000 modules integrate the latest web language (HTML 5) and web-based architectural style (REST) with basic authentication for users to remotely acquire I/O data in any smart device web service without routing from the SCADA system. As an example, a smartphone web browser can now be used to remotely access an I/O module via HTTP.

Each ADAM-6000 module features a pre-built I/O module web page for displaying real-time I/O data, alarms, and module status via LAN or the Internet. Using any popular Internet browser, you can perform monitoring from both local and remote sites. Furthermore, web-enabled monitoring can be completed immediately without requiring any programming.

Modbus/TCP Protocol

ADAM-6000 modules support the widely used industry standard Modbus/TCP protocol, enabling you to connect with any Ethernet controllers or HMI/SCADA software that supports Modbus/TCP. Advantech also provides an OPC server for Modbus/TCP so that ADAM-6000 I/O module datastreams can be integrated with OPC client-enabled software, thus freeing you from having to develop new drivers.

Customized Web Page

Since ADAM-6000 modules have a default built-in web page, you can monitor and control the I/O status from any location by using Internet Explorer. Moreover, customized web pages can be uploaded to ADAM-6000 modules for individual applications. Advantech provides sample code in JavaScript* as a reference for you to design your own operator interface and then upload it to the specific ADAM-6000 modules via Adam/Apax .NET Utility.

Modbus/TCP Software Support

The firmware for ADAM-6000 modules has a built-in Modbus/TCP server. Advantech provides the ADAM .NET Class Library and Adam/Apax .NET Utility for module con-

figuration and customization. You can configure ADAM modules using this utility, and it can be integrated with any human-machine (HMI) software that supports Modbus/TCP. You can also purchase Advantech OPC Server to configure the Modbus/TCP settings.

1.3 Specifications

Ethernet	10/100BASE-T
Wiring	UTP (Cat 5 or later)
Bus Connection	RJ45 modular jack
Comm. Protocol	Modbus/TCP on TCP/IP and UDP
Data Transfer Rate	Up to 100 Mbps Unregulated 10 to 30 V_{DC}
Status Indicator	Power, CPU, Communication (Link, Collide, 10/100 Mbps, Tx, Rx)
Case	PC with captive mounting hardware
Screw Terminal Block	Accepts wire size #14-28 AWG, stripped length: 6.5 mm

Note! *Although the equipment is designed to operate below 30% humidity, static electricity problems are more common at lower humidity levels. Ensure you take adequate precautions when handling the equipment. We recommend using grounding straps, anti-static floor coverings, and other protection measures if you use the equipment in low-humidity environments.*

1.4 Dimensions

The following dimensions are given in millimeters. These dimensions are common for all ADAM-6000 modules.

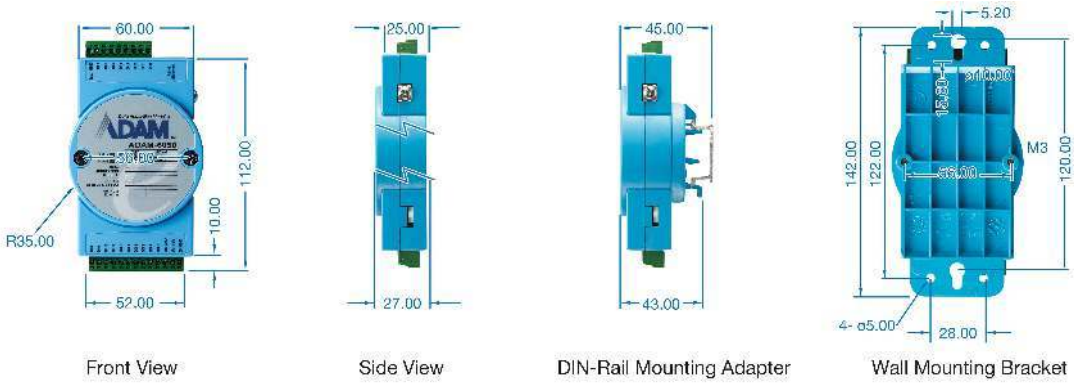


Figure 1.2 ADAM-6000 Module Dimensions

1.5 LED Status

There are two LEDs on the front panel of ADAM-6000 modules. Each LED has two indicators to represent system status:

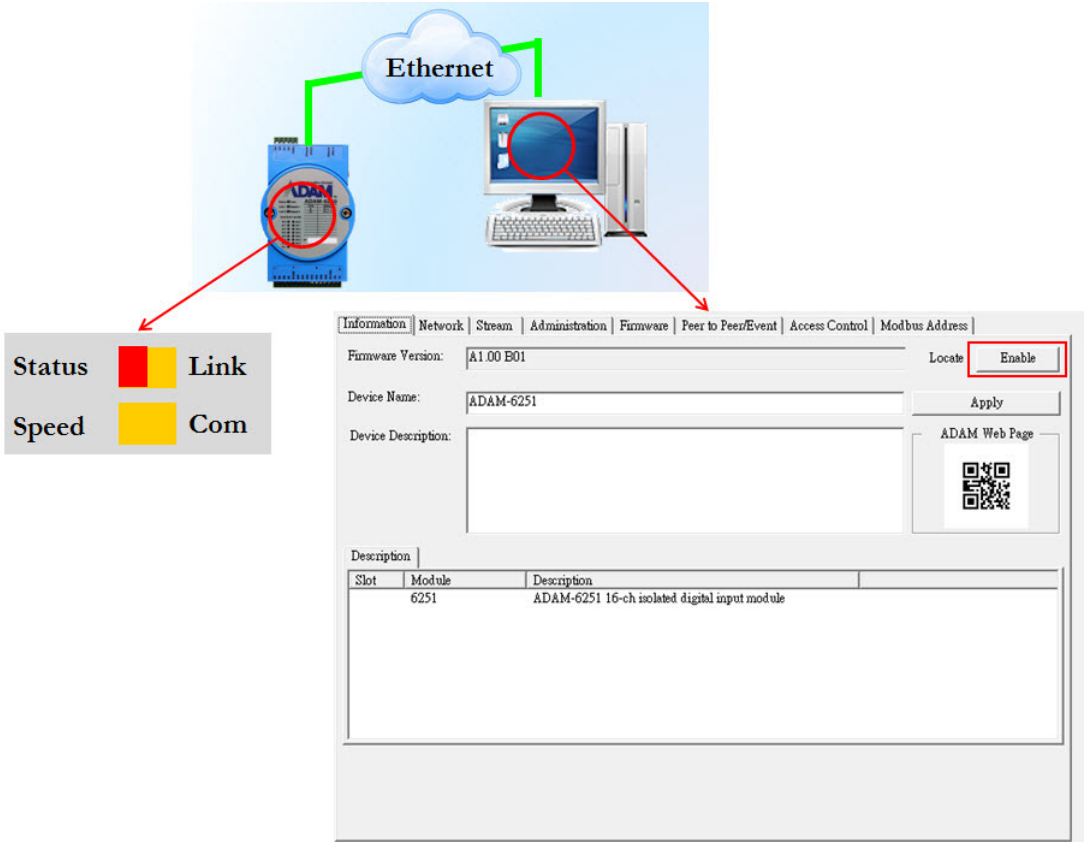
LED	Color	Color	Indication	Behavior
Status	Orange (when Status and Link are on at the same time)		Red Blink	Module is normally running
			Red On for 30s	When user enable LOCATE function
Link			Green On	Ethernet speed is 100 Mbps
Speed	Orange (when speed and COM are on at the same time)		Red On	Ethernet speed is 100 Mbps
COM			Green Blink	Module is transmitting or receiving data

For ADAM-6050-D1 and ADAM-6060-D1, the LED:

LED	Color	Indication	Behavior
Status		Red Blink	Module is normally running
		Red On for 30s	When user enable LOCATE function
Link		Green On	Ethernet speed is 100 Mbps
Speed		Red On	Ethernet speed is 100 Mbps
COM		Green Blink	Module is transmitting or receiving data

How to Locate Your Module

ADAM-6000 modules also have a locate function to help you physically identify a specific module that you may be looking for. When this function is enabled, the Status LED will remain red for 30 s. In Adam/Apax .NET Utility, you can enable the locate function by clicking **Enable** in the **Information** tab.



Chapter 2

Hardware Selection Guidelines

2.1 Selecting an I/O Module

To organize an ADAM-6000 remote DA&C system, you will need to select I/O modules to act as an interface between the host PC and field devices or sensors. The following should be considered when deciding which I/O modules to select.

- What types of I/O signals does your system use?
- How many inputs and outputs does your system require?
- How many modules are required for distributed I/O point arrangement?
- How will you arrange the modules to handle I/O points in individual areas of the installation site?
- How many hubs will you require to connect all of the modules?
- What is the required voltage range for each I/O module?
- What isolation environment is required for each I/O module?
- What are the noise and distance limitations for each I/O module?

Examples of I/O module selection considerations are detailed in Table 2.1.

Table 2.1: I/O Selection Guidelines

Type of I/O Module	Example Operations	Explanation
Discrete input module and block I/O module	Selector switches, push buttons, photoelectric eyes, limit switches, circuit breakers, proximity switches, level switches, motor starter contacts, relay contacts, and thumbwheel switches	Input modules sense ON/OFF or OPENED/CLOSED signals
Discrete output module and block I/O module	Alarms, control relays, fans, lights, horns, valves, motor starters, and solenoids	Output module signals interface with ON/OFF or OPENED/CLOSED devices
Analog input module	Thermocouple signals, RTD signals, temperature transducers, pressure transducers, load cell transducers, humidity transducers, flow transducers, potentiometers.	Convert continuous analog signals into input values for a host device
Analog output module	Analog valves, actuators, chart recorders, electric motor drives, analog meters	Set a host device's output to analog signals (generally through transducers) for field devices

2.2 Selecting a Link Terminal and Cable

Use an RJ-45 connector to connect the Ethernet port of ADAM-6000 modules to a hub. The cable employed for the connection should be a Cat 3 (10 Mbps) or Cat 5 (100 Mbps) UTP/STP cable, both of which comply with EIA/TIA 586 specifications. The maximum length between a hub and any ADAM-6000 module is 100 m (approx. 330 ft).

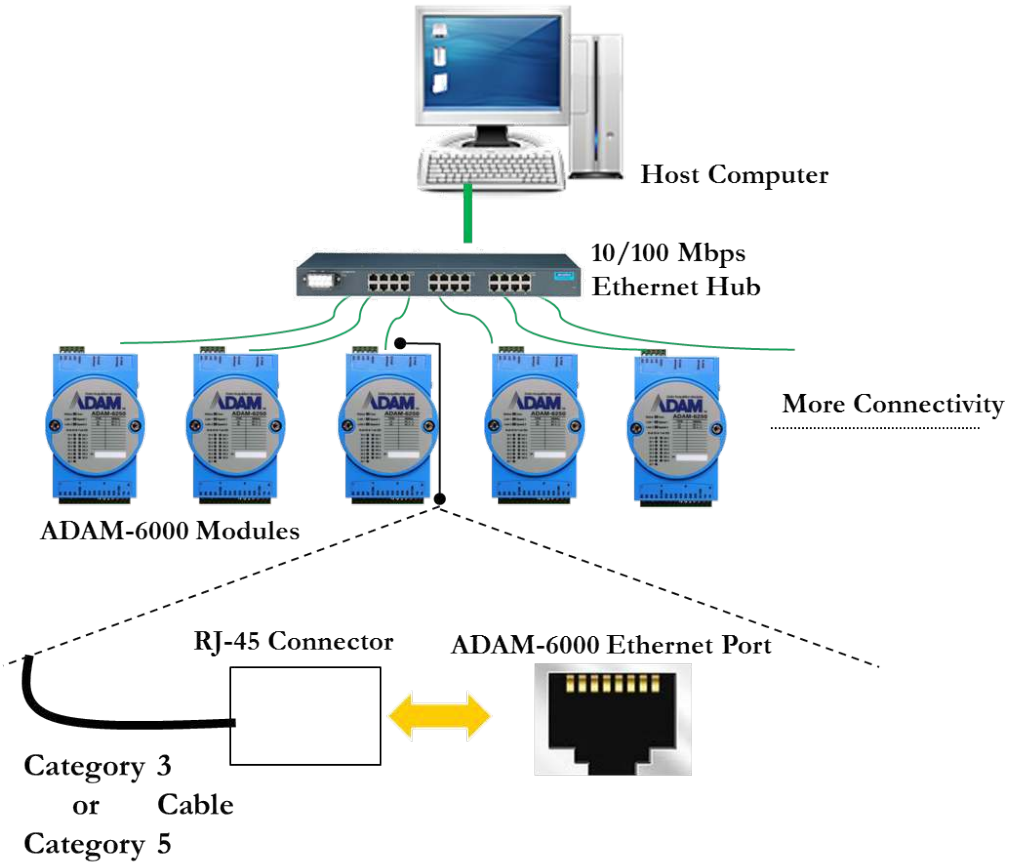


Figure 2.1 Connecting ADAM-6000 Modules to an Ethernet Terminal via Cable

Table 2.2: Ethernet RJ-45 Port Pin Assignment Chart

PIN Number	Signal	Function
1	RD+	Receive (+)
2	RD-	Receive (-)
3	TD+	Transmit (+)
4	(Not used)	-
5	(Not used)	-
6	TD-	Transmit (-)
7	(Not used)	-
8	(Not used)	-

2.3 Selecting an Operator Interface

To complete your DA&C system, it is necessary to select an operator interface. Supporting the Modbus/TCP protocol, ADAM-6000 modules can easily be integrated into different systems for various applications.

The real-time status of ADAM-6000 modules can be read from a web page using the following browsers:

- Microsoft Internet Explorer (version 9 or later)
- Google Chrome (version 30 or later)
- Safari (version 6 or later)
- Mozilla Firefox (version 25 or later)

If you want to integrate ADAM-6000 modules with HMI software in a SCADA system, HMI software packages that support Modbus/TCP can be used. Examples are as follows:

- Advantech PM Designer
- Wonderware InTouch
- Any software that supports the Modbus/TCP protocol

You can also purchase Advantech OPC Server, a highly user-friendly data exchange tool. Any HMI software designed with OPC Client can be employed to access ADAM-6000 modules.

To develop your own applications, the Adam .NET Class Library is ideal for building up user interfaces.

With these ready-to-go software packages, tasks such as remote data acquisition, process control, historical trending, and data analysis require only a few keystrokes to utilize.

Chapter 3

Hardware Installation Guide

3.1 Interface Introduction

Package Contents and System Requirements

Prior to installing ADAM-6000 modules, please check the following.

The package should contain the following contents:

- ADAM-6000 module with one bracket and DIN-rail adapter
- ADAM-6000 module user manual

The minimum specifications for the host computer are listed as follows:

- Microsoft Windows XP/7
- 32 MB RAM
- 20 MB of hard disk space
- VGA color monitor
- CD-ROM drive
- Mouse or other pointing device
- 10/100 Mbps Ethernet card

The following equipment will also be required to complete the installation:

- Ethernet hub (at least 2 ports)
- Two Ethernet cables with an RJ-45 connector
- Power supply for the ADAM-6000 module (+10 to 30 V, unregulated)
- Grounding Connector

3.2 Mounting Options

ADAM-6000 modules are compact units that can be installed with a panel mounting bracket or a DIN rail mounting bracket.

3.2.1 Panel Mounting

Before installing the ADAM-6000 module, you should determine the optimal placement in a panel or cabinet by referring the bracket dimensions shown in Figure 3.1. First, fix the bracket and then fix the ADAM-6000 module on the bracket, as shown in Figure 3.2.

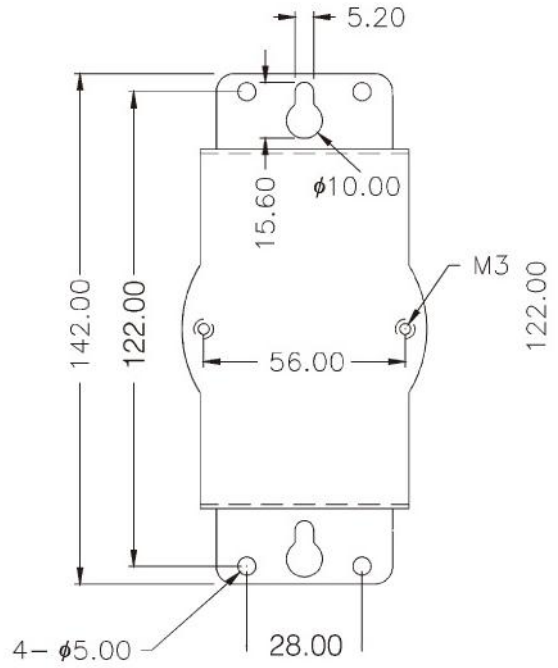


Figure 3.1 Panel Mounting Bracket Dimensions

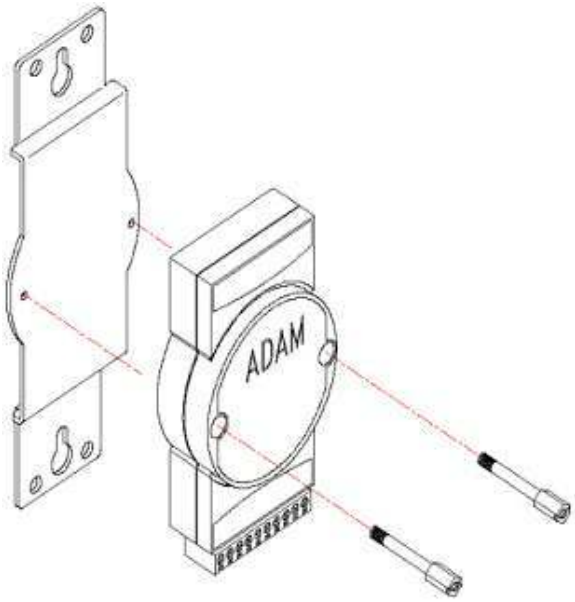


Figure 3.2 How to Fix a Module on the Mounting Bracket

3.2.2 DIN Rail Mounting

The ADAM-6000 module can also be secured to a cabinet by using DIN rails. First, fix the ADAM-6000 module to the DIN rail adapter (Figure 3-3) and then secure it on the DIN rail (Figure 3-4). When mounting the module on the rail, you should consider using end brackets at each end of the rail in order to prevent the module from sliding.

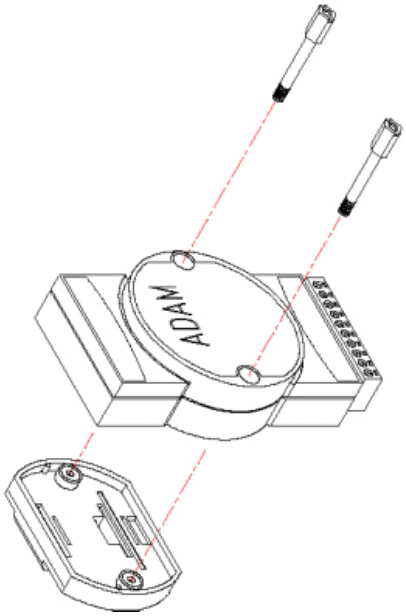


Figure 3.3 How to Fix a Module on the DIN Rail Adapter

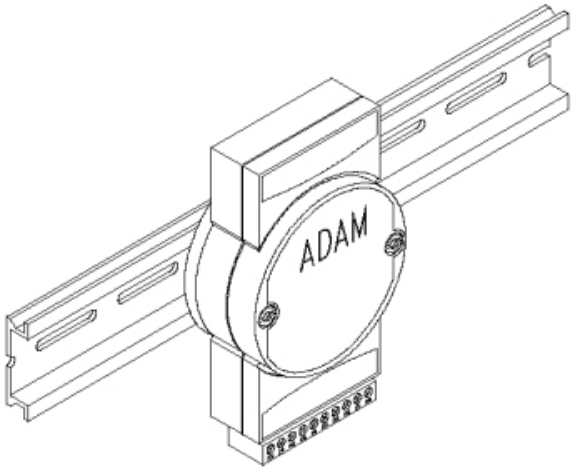
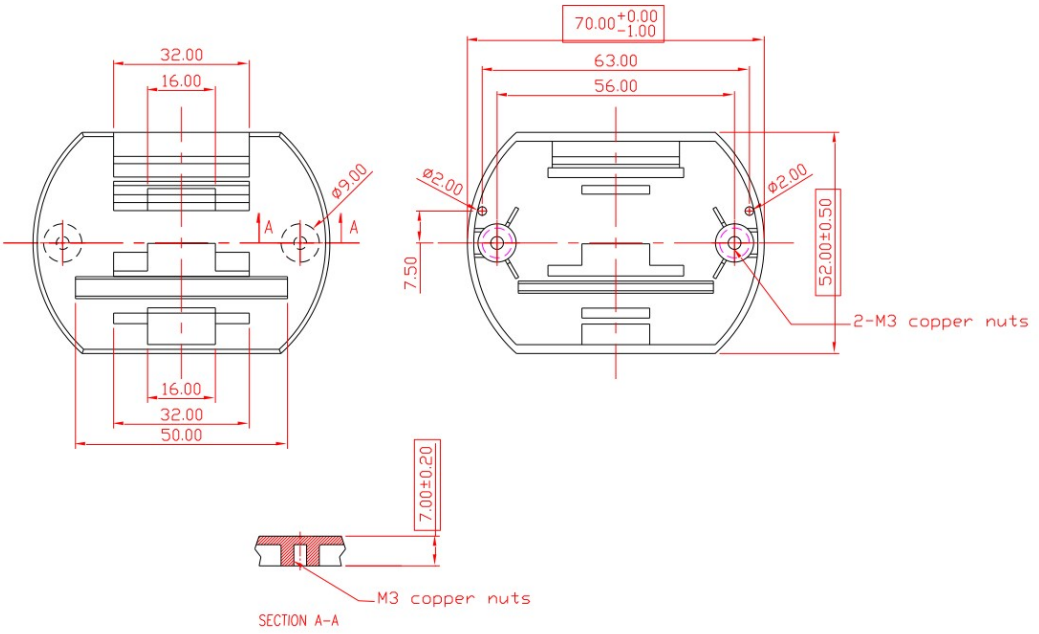


Figure 3.4 How to Secure a Module to a DIN Rail



3.3 Wiring and Connections

This section provides basic information on wiring the power supply, I/O units, and network connection.

3.3.1 Power Supply Wiring

Although ADAM-6000/TCP systems are designed for a standard industrial unregulated 24 V_{DC} power supply, they accept any power unit that supplies input power within the range of +10 to 30 V_{DC}. Power supply ripple must be limited to 200 mV peak-to-peak, and the immediate ripple voltage should be maintained between +10 and 30 V_{DC}. Screw terminals +Vs and GND are for wiring the power supply.

Note! *The wires should be at least 2 mm in diameter.*

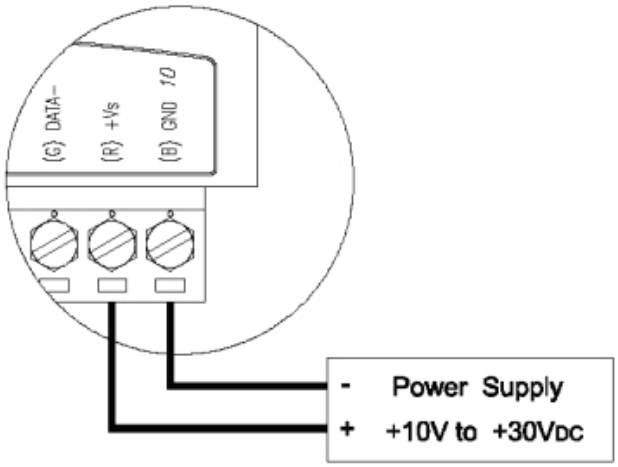


Figure 3.5 How to Connect the Module Power Wires

We advise using the following standard colors (which are also indicated on the modules) for the power lines:

+Vs (R) Red

GND (B)Black

3.3.2 I/O Module Wiring

A plug-in screw terminal block is used for the interface between I/O modules and field devices. The following information must be considered when connecting electrical devices to I/O modules.

- The terminal block accepts Wire Size #14~28 AWG (stripped length: 6.5 mm)
- Always use a continuous length of wire; do not combine wires
- Use the shortest possible wire length
- Use wire trays for routing where possible
- Avoid running wires near high-energy wiring
- Avoid running input wiring proximal to output wiring
- Avoid creating sharp bends in the wires

Chapter 4

Introduction to Analog ADAM-6000 I/O Modules

4.1 Analog Input Modules

Analog input modules use an A/D converter to convert sensor voltage, current, thermocouple, and RTD signals into data, which are then translated into engineering units. When prompted by the host computer, the data are sent via standard 10/100BASE-T Ethernet. The current status can then be read using a pre-built webpage or any HMI software that supports Modbus/TCP. Analog input modules protect your equipment from ground loops and power surges by providing opto-isolation of the A/D input as well as transformer-based isolation.

4.2 ADAM-6015 7-ch Isolated RTD Input Module

The ADAM-6015 is a 16-bit, 7-ch RTD input module with programmable input ranges on all channels. It accepts various RTD inputs (PT100, PT1000, Balco 500, and Ni), and data are transmitted to the host computer in engineering units (°C). Each analog channel can be configured to an independent range, thus allowing individual channels to be used simultaneously in different applications.

4.2.1 Specifications

- Communication: 10/100BASE-T Ethernet
- Supported protocols: Modbus/TCP,TCP/IP, UDP, HTTP, ICMP, DHCP, and ARP
- Supports P2P and GCL (see Section 6.7 and Chapter 8)

High-Speed Mode (DE Version Only)

In high-speed mode, the maximum total sample rate is 1 kHz (i.e., if 7 channels are used, then the sampling rate will be 1000/7, which is approximately 142 Hz per channel). This will be influenced by the number of connected Modbus clients and the Ethernet quality. To maximize performance in high-speed mode, any channels that are not in use should be disabled; otherwise, the accuracy may be affected.

Note! *When using a calibrator to simulate resistors in high-speed mode, no more than one channel should be enabled.*

Analog Input

- Channels: 7 (differential)
- Input impedance: >10 MΩ
- Input connections: 2- or 3-wire
- Input types: Pt 100/1000, Balco 500, and Ni 518 RTD
- RTD types and temperature range:
  - Pt 100: -50~150°C
    0~100°C
    0~200°C
    0~400°C
    -200~200°C
    IEC RTD 100 Ω (α = 0.0385)
    JIS RTD 100 Ω (α = 0.0392)
  - Pt 1000: -40~160°C
  - Balco 500: -30~120°C
  - Ni 518: -80~100°C
    0~100°C
- Accuracy:
  - ±0.1% or better
  - ±0.5% or better (high-speed mode)
    (measured by 3-wire RTD)
- Span drift: ±25 ppm/°C
- Zero drift: ±6 mV/°C
- Resolution: 16-bit
- Sample rate (total):
  - 10 Hz
  - 1 kHz (high-speed mode; DE version only)
- CMR @ 50/60 Hz: 90 dB (not supported in high-speed mode)
- NMR @ 50/60 Hz: 60 dB (not supported in high-speed mode)
- Wire burnout detection
- Overvoltage protection: ±35 V_{DC}
- Built-in TVS/ESD protection

General

- Built-in watchdog timer
- Isolation protection: 2000 V_{DC}
- Power input: Unregulated 10~30 V_{DC}
- Power consumption: 2.5 W @ 24 V_{DC}
- Power reversal protection
- Operating humidity: 20~95% RH (non-condensing)
- Storage humidity: 0~95% RH (non-condensing)
- Operating temperature: -10~70°C
- Storage temperature: -20~80°C

4.2.2 Application Wiring

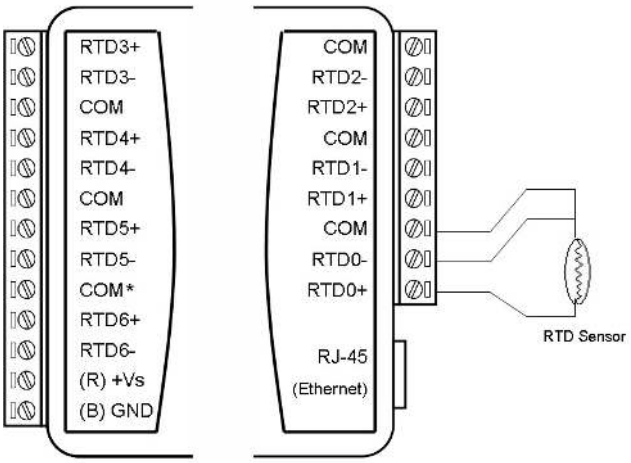


Figure 4.1 ADAM-6015 RTD Input Wiring

Note: * RTD6 and RTD5 share the same COM.

4.2.3 Address Assignment

Based on the Modbus/TCP standard, the addresses of ADAM-6000 module I/O channels in the system are defined by a simple rule. See Appendix B.2.1 for information on mapping the I/O addresses.

4.3 ADAM-6017 8-ch Analog Input/2-ch Digital Output Module

The ADAM-6017 is a 16-bit, 8-ch analog differential input module with programmable input ranges on all channels. The module has been designed with eight analog inputs and two digital outputs. The accepted input types are millivolt (±150, ±500, 0~150, 0~500 mV), voltage (±1, ±5, ±10, 0~1, 0~5, 0~10 V), and current (0~20, 4~20, ±20 mA) signals, and data are transmitted to the host computer in engineering units (mV, V, or mA). Each analog channel can be configured to an independent range, thus allowing individual channels to be used simultaneously in different applications.

4.3.1 Specifications

- Communication: 10/100BASE-T Ethernet
- Protocols: MQTT, SNMP, Modbus/TCP, TCP/IP, UDP, HTTP, ICMP, DHCP, ARP
- Supports P2P and GCL (see Section 6.7 and Chapter 8)

Analog Input

- Channels: 8 (differential)
- Input impedance: >10 MΩ (voltage), 120 Ω (current)
- Input type: mV, V, mA
- Input range: ±150 mV, ±500 mV, ±1 V, ±5 V, ±10 V, 0~150 mV, 0~500 mV, 0~1 V, 0~5 V, 0~10 V, 0~20 mA, 4~20 mA, ±20 mA
- Accuracy:
  - ±0.1% of FSR (voltage) @ 25°C
  - ±0.2% of FSR (current) @ 25°C
- Span drift: ±25 ppm/°C
- Zero drift: ±6 mV/°C
- Resolution: 16-bit
- Sample rate (total):
  - 10 Hz
  - 100 Hz
- CMR @ 50/60 Hz: 90 dB
- NMR @ 50/60 Hz: 67 dB
- Calibration: Auto calibration
- Burnout detection (4~20 mA only)
- Common-mode voltage: 350 V_{DC}

Digital Output

- Channels: 2
- Sink type: Open collector to 30 V, 100 mA (max. load)
- Power dissipation: 300 mW for each module
- Output-delay on: 100 μs
- Output-delay off: 150 μs
- Overvoltage protection (max.): 42 V_{DC}
- Overcurrent protection (max.): 2 A
- Leakage current: 200 μA (max.) for D version

General

- Isolation protection: 2000 V_{DC}
- Power input: 10~30 V_{DC}
- Power consumption: 2.7 W @ 24 V_{DC}
- Power reversal protection
- Operating humidity: 20~95% RH (non-condensing)
- Storage humidity: 0~95% RH (non-condensing)
- Operating temp (exclusive of RTC function): -20~70°C (-40~70°C for D version)
- Storage temp (exclusive of RTC function): -30~80°C (-40~85°C for D version)
- Watchdog timer (system): 1.6 s
- RTC (D version only): ISO8601 format

Note! *The operation/storage temperature for the RTC function is -30~70°C.*

Jumper Settings

ADAM-6017-CE		ADAM-6017-AE&BE	
Channel Number	**Select Jumper**	**Channel Number**	**Select Jumper**
CH0	CN3	CH0	JP6
CH1	CN4	CH1	JP7
CH2	CN5	CH2	JP8
CH3	CN6	CH3	JP1
CH4	CN7	CH4	JP2
CH5	CN8	CH5	JP3
CH6	CN9	CH6	JP4
CH7	CN10	CH7	JP5

To simplify the jumper settings, for the ADAM-6017 (D version), you can set the analog input type to voltage or current by adjusting the switch without opening the case.

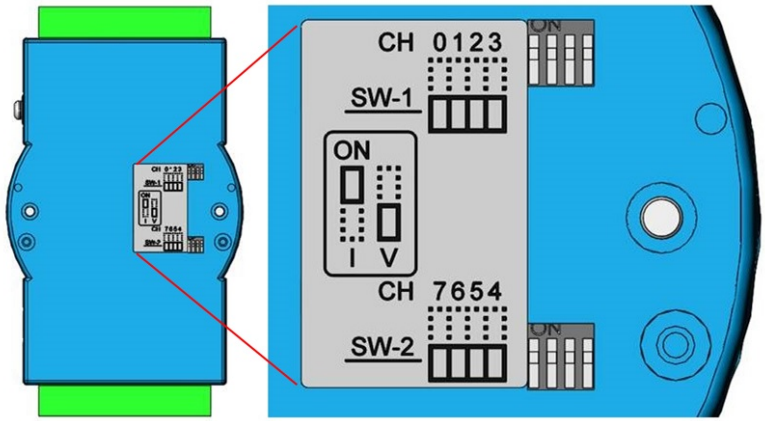
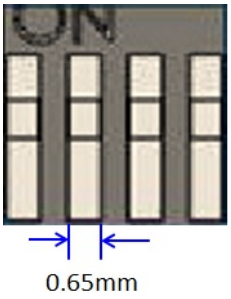


Figure 4.2 ADAM-6017 Jumper Switches

Switch	**SW1**				**SW2**			
Analog input channel	CH0	CH1	CH2	CH3	CH4	CH5	CH6	CH7
Switch ON	Current input mode							
Switch OFF (default)	Voltage input mode							

Note! *Using tools wider than 0.65 mm to adjust the switch will result in damage to the switch.*



4.3.2 Application Wiring

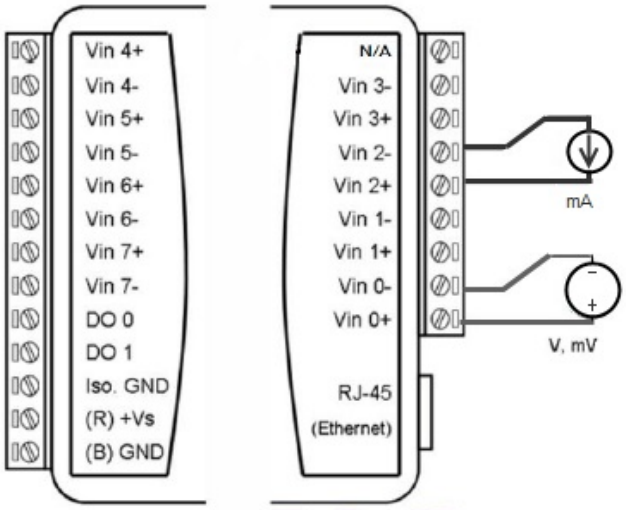


Figure 4.3 ADAM-6017 Analog Input Wiring

The ADAM-6017 has a 120-Ω resistor built in to each channel; thus, no additional resistors need to be added for current input measurements. Simply adjust the jumper setting according to the input type you require. Figure 4.3 shows the jumpers for setting the inputs to voltage mode or current mode.

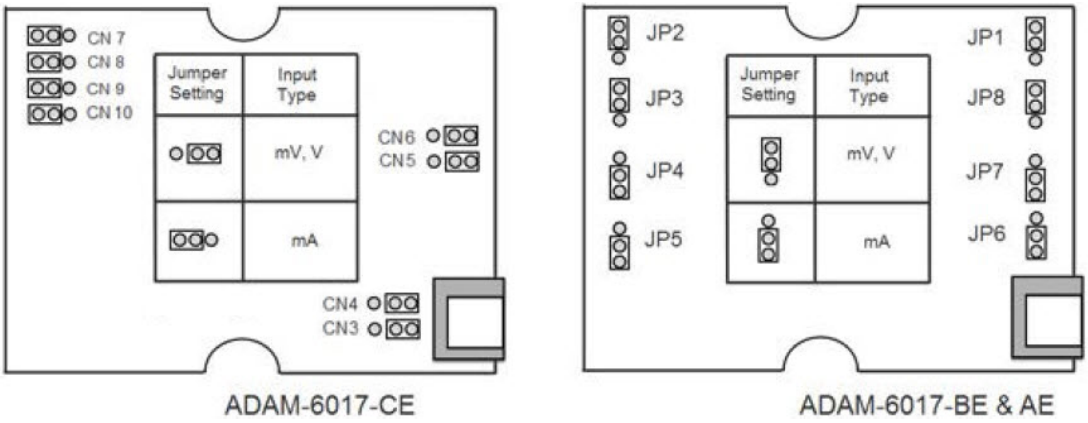


Figure 4.4 ADAM-6017 Analog Input Type Setting

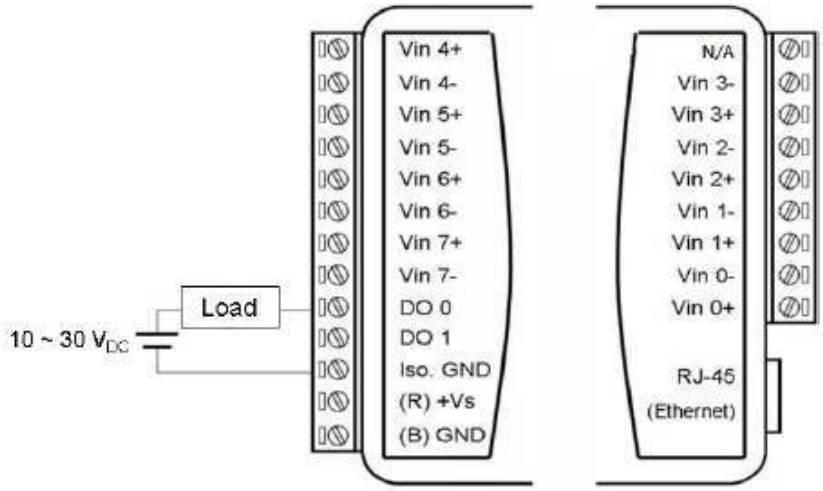


Figure 4.5 ADAM-6017 Digital Output Wiring

4.3.3 Address Assignment

Based on the Modbus/TCP standard, the addresses of ADAM-6000 I/O channels you place in the system are defined by a simple rule. See Appendix B.2.2 for information on mapping the I/O addresses.

4.4 ADAM-6018 Isolated Thermocouple Input/8-ch Digital Output Module

The ADAM-6018 is a 16-bit, 8-ch thermocouple input module with programmable input ranges on all channels. The module has eight thermocouple inputs (Types J, K, T, E, R, S, and B) and eight digital outputs. Each input can be configured to an independent range, thus allowing individual channels to be used simultaneously in different applications.

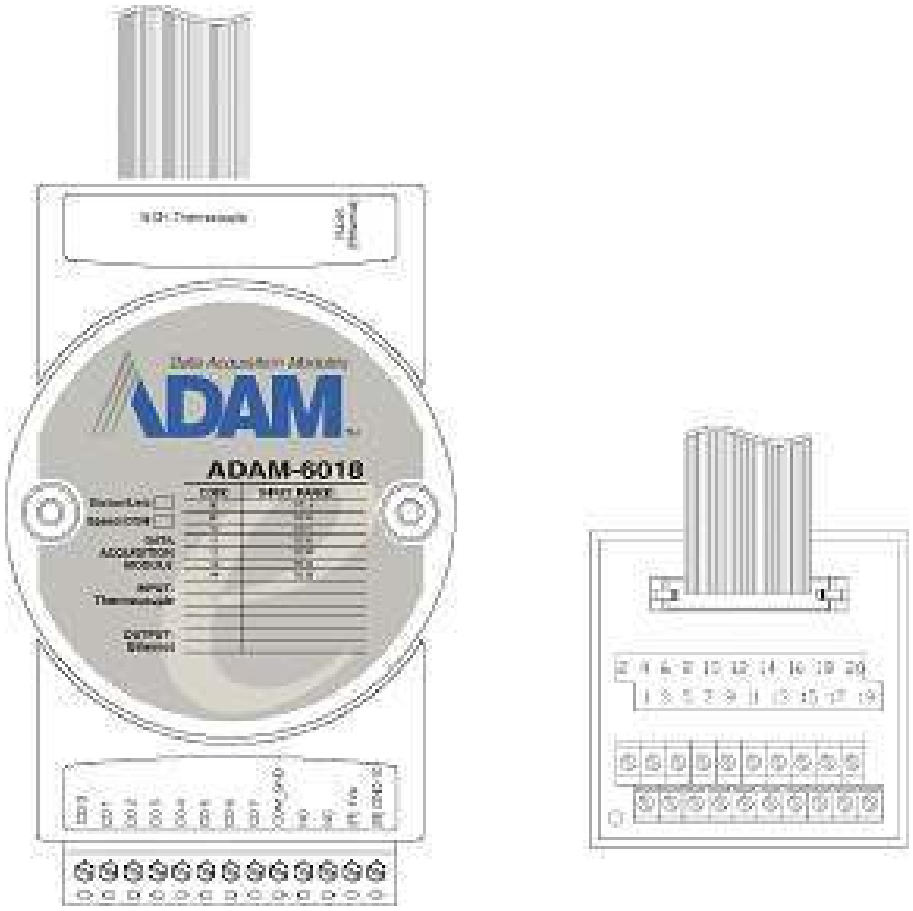


Figure 4.6 ADAM-6018 8-ch Thermocouple Input

4.4.1 Specifications

- Communication: 10/100BASE-T Ethernet
- Supported protocols: Modbus/TCP, TCP/IP, UDP, HTTP, ICMP, DHCP, and ARP
- Supports P2P and GCL (see Section 6.7 and Chapter 8)

Analog Input

- Channels: 8 (differential)
- Input impedance: >10 MΩ
- Input type: Thermocouple
- Thermocouple type and range:
  - Type J: 0~760°C
  - Type K: 0~1370°C
  - Type T: -100~400°C
  - Type E: 0~1000°C
  - Type R: 500~1750°C
  - Type S: 500~1750°C
  - Type B: 500~1800°C
- Accuracy: ±0.1% or better
- Span drift: ±25 ppm/°C
- Zero drift: ±6 mV/°C
- Resolution: 16-bit
- Sample rate: 10 Hz
- CMR @ 50/60 Hz: 90 dB
- NMR @ 50/60 Hz: 60 dB
- Overvoltage protection ±35 V_{DC}
- Built-in TVS/ESD protection
- Wire burnout detection

Digital Output

- Channels: 8
- Sink type: Open collector to 30 V, 100 mA (max. load)
- Power dissipation: 300 mW for each module

General

- Built-in watchdog timer
- Isolation protection: 2000 V_{DC}
- Power input: Unregulated 10~30 V_{DC}
- Power consumption: 2 W @ 24 V_{DC}
- Power reversal protection
- Operating humidity: 20~95% RH (non-condensing)
- Storage humidity: 0~95% RH (non-condensing)
- Operating temperature: -10~70°C
- Storage temperature: -20~80°C

4.4.2 Application Wiring

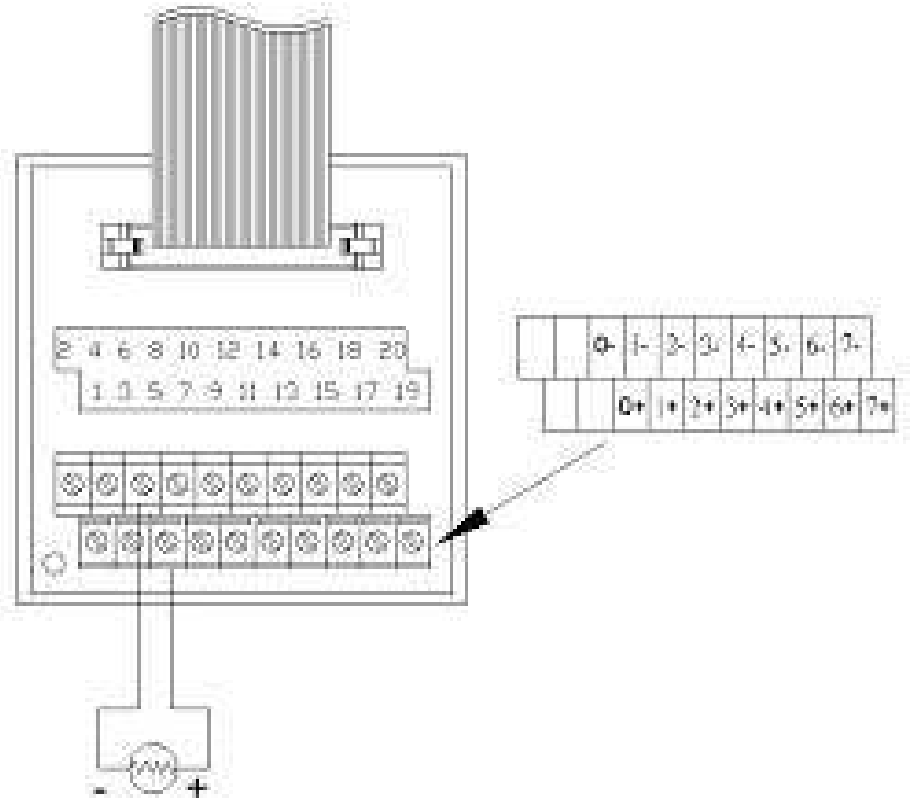


Figure 4.7 ADAM-6018 Thermocouple Input Wiring

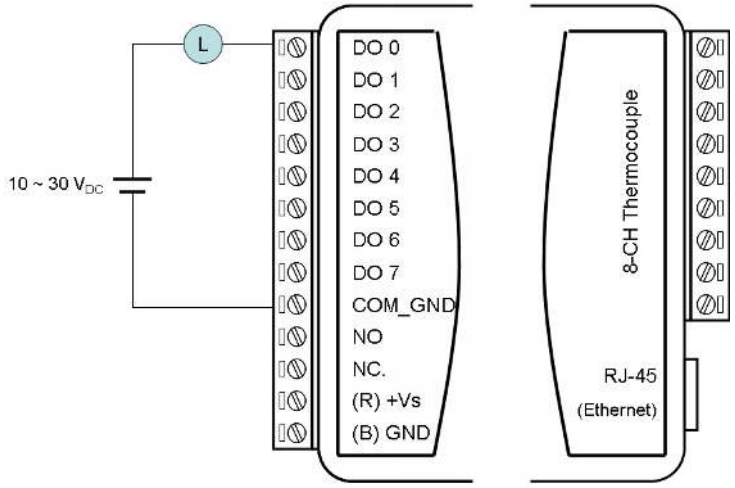


Figure 4.8 ADAM-6018 Digital Output Wiring

4.4.3 Address Assignment

Based on the Modbus/TCP standard, the addresses of ADAM-6000 I/O channels you place in the system are defined by a simple rule. See Appendix B.2.3 for information on mapping the I/O addresses.

4.5 ADAM-6018+ 8-ch Isolated Thermocouple Input module

The ADAM-6018+ is a 16-bit, 8-ch thermocouple input module with programmable input ranges on all channels. The module has eight thermocouple inputs (Types J, K, T, E, R, S, and B) and eight digital outputs. Each input can be configured to an independent range, thus allowing individual channels to be used simultaneously in different applications.

4.5.1 Specifications

- Communication: 10/100BASE-T Ethernet
- Supported protocols: Modbus/TCP, TCP/IP, UDP, HTTP, ICMP, DHCP, ARP, MQTT and SNMP
- Supports P2P and GCL (see Section 6.7 and Chapter 8)

Thermocouple Input

- Channels: 8 (differential)
- Input type: Thermocouple
- Thermocouple type and range:
  - Type J: 0~760°C
  - Type K: 0~1370°C
  - Type T: -100~400°C
  - Type E: 0~1000°C
  - Type R: 500~1750°C
  - Type S: 500~1750°C
  - Type B: 500~1800°C
- Accuracy@25°C: (mount in vertical direction as fig 3.4 shown)
  - Type J,K,E,R,S: ±0.1% FSR Max
  - Type B: ±0.15% FSR Max
  - Type T: ±0.2% FSR Max
- Span drift: ±25 ppm/°C
- Zero drift: ±6 mV/°C
- Resolution: 16-bit
- Sample rate: 10 Hz
- Overvoltage protection ±35 VDC
- Built-in TVS/ESD protection
- Wire burnout detection

General

- Built-in watchdog timer
- Isolation protection: 2000 VDC
- Power input: Unregulated 10~30 VDC
- Power consumption: 1 W @ 24 VDC
- Power reversal protection
- Operating humidity: 20~95% RH (non-condensing)
- Storage humidity: 0~95% RH (non-condensing)
- Operating temperature: -40~70°C
- Storage temperature: -40~85°C

4.5.2 Application Wiring

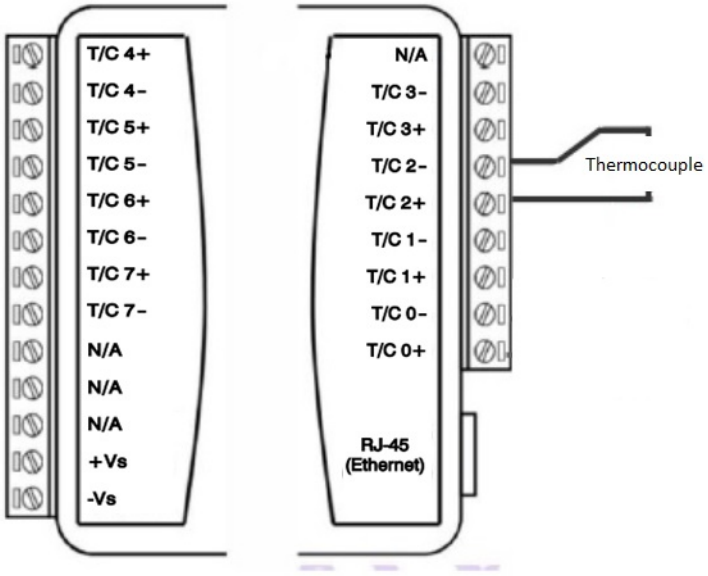


Figure 4.9 ADAM-6018+ thermocouple wiring

4.6 ADAM-6024 12-ch Isolated Universal I/O Module

The ADAM-6024 is a 12-ch universal I/O module with programmable input ranges on all channels. The module has six analog inputs, two analog outputs, two digital inputs, and two digital outputs. The analog input channels are 16-bit universal signal inputs, accepting voltage (±10 V) and current (0~20, 4~20 mA) signals. The analog output channels are 12-bit outputs for volts (0~10 V) and current (0~20 mA, 4~20 mA). Each analog channel can be configured to an independent range, thus allowing individual channels to be used simultaneously in different applications.

4.6.1 Specifications

- Communication: 10/100BASE-T Ethernet
- Supported protocols: Modbus/TCP,TCP/IP, UDP, HTTP, ICMP, DHCP and ARP
- Receives data from other modules that support P2P and GCL functionality, and generates analog output signals (see Section 6.7 and Chapter 8 for more detail about P2P and GCL)

Analog Input

- Channels: 6 (differential)
- Range: ±10 V_{DC}, 0~20 mA, 4~20 mA
- Input impedance: 20 MΩ
- Accuracy: ±0.1% of FSR at 25°C
- Resolution: 16-bit
- CMR @ 50/60 Hz: 90 dB
- NMR @ 50/60 Hz: 60 dB
- Span drift: ±25 ppm/°C
- Zero drift: ±6 uV/°C
- Isolation protection: 2000 V_{DC}

Analog Output

- Channels: 2
- Range: 0~10 V_{DC}, 0~20 mA, 4~20 mA
- Accuracy: ±0.1% of FSR at 25°C
- Resolution: 12-bit
- Current load resistor: 500 Ω (max.)
- Voltage load resistor: 1 kΩ (min.)
- Isolation protection: 2000 V_{DC}
- Drift: ±50 ppm/°C

Digital Input

- Channels: 2
- Dry contact:
  - Logic level 0: close to GND
  - Logic level 1: open
- Wet contact:
  - Logic level 0: 0~3 V_{DC}
  - Logic level 1: 10~30 V_{DC}

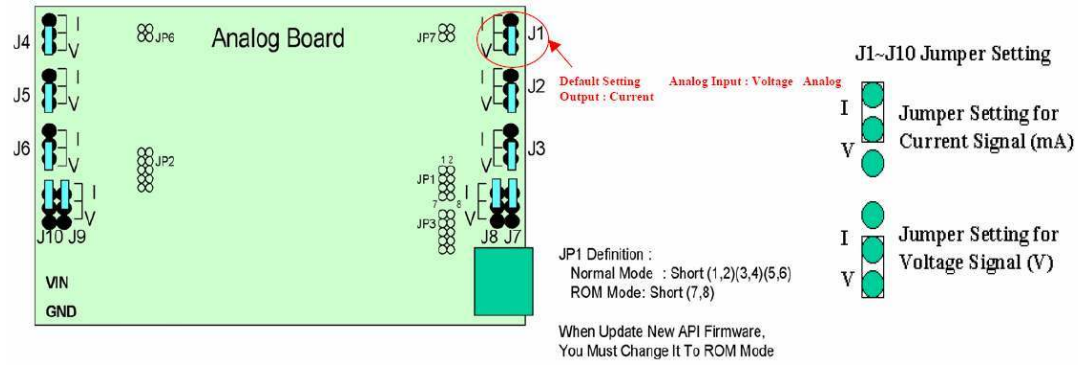
Digital Output

- Channels: 2
- Sink type: Open collector to 30 V, 100 mA (max.)
- Power dissipation: 300 mW for each module

General

- Built-in watchdog timer
- Isolation protection: 2000 V_{DC}
- Power input: Unregulated 10~30 V_{DC}
- Power consumption: 4W@24 V_{DC}
- Power reversal protection
- Operating humidity: 20~95% RH (non-condensing)
- Storage humidity: 0~95% RH (non-condensing)
- Operating temperature: -40~70°C
- Storage temperature: -40~85°C

Jumper Settings (ADAM-6024-A1E)



Channel	Jumper	Current	Voltage
AI0	J1	I	V
AI1	J2	I	V
AI2	J3	I	V
AI3	J4	I	V
AI4	J5	I	V
AI5	J6	I	V
AO0	J7	I	V
	J8	I	V
AO1	J9	I	V
	J10	I	V

Figure 4.10 ADAM-6024-A1E Jumper Settings

Jumper Settings (ADAM-6024-D)

The jumpers are for changing AI channel type between Current Mode and Voltage Mode (Default)

Channel	Jumper	Current Mode	Voltage Mode (Default)
AI0	JAI1	1-2 Short	2-3 Short
AI1	JAI2	1-2 Short	2-3 Short
AI2	JAI3	1-2 Short	2-3 Short
AI3	JAI4	1-2 Short	2-3 Short
AI4	JAI5	1-2 Short	2-3 Short
AI5	JAI6	1-2 Short	2-3 Short

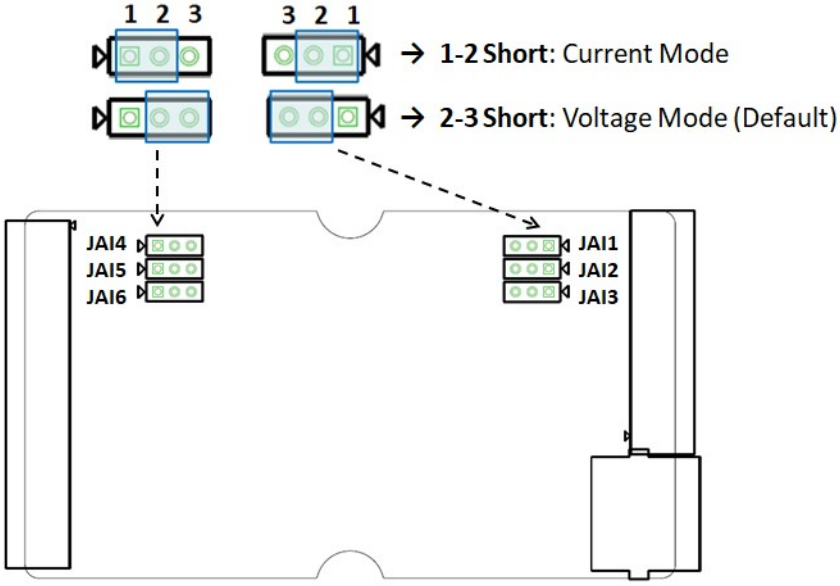


Figure 4.11 ADAM-6024-D Jumper Settings

Note! *Analog output of ADAM-6024-D doesn't need jump setting.*

4.6.2 Application Wiring

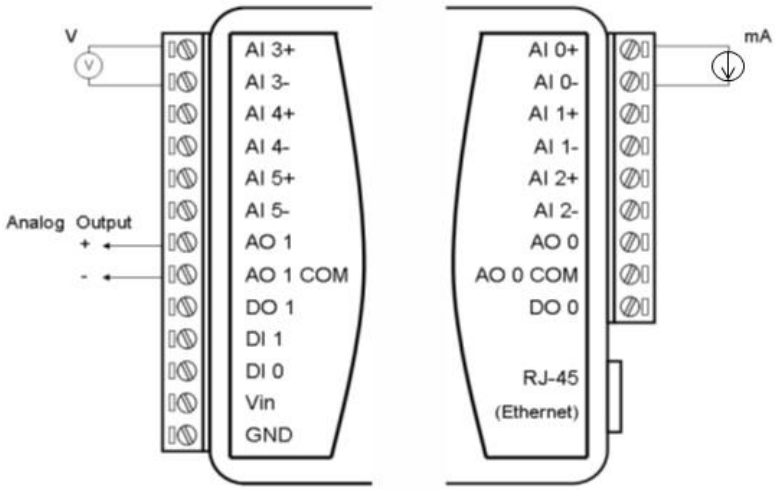


Figure 4.12 ADAM-6024 Analog I/O Wiring

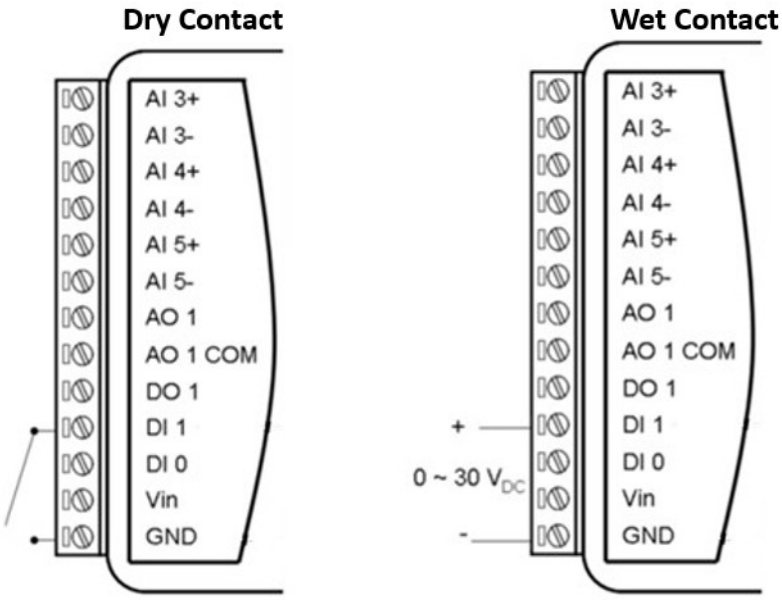


Figure 4.13 ADAM-6024 Digital Input Wiring

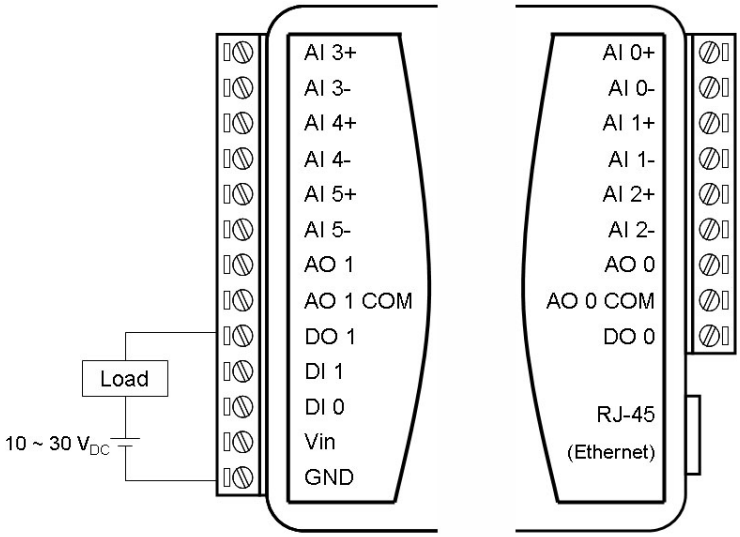


Figure 4.14 ADAM-6024 Digital Output Wiring

4.6.3 Address Assignment

Based on the Modbus/TCP standard, the addresses of ADAM-6000 I/O channels you place in the system are defined by a simple rule. See Appendix B.2.4 for information on mapping the I/O addresses.

Chapter 5

Introduction to Digital ADAM-6000 I/O Modules

5.1 Digital I/O and Relay Modules

Digital I/O modules can be connected to digital sensors and actuators. These modules support both dry and wet contact for different applications. Relays, on the other hand, are electrically operated switches. Relay modules are typically employed to control a circuit by using a low-power signal. When prompted by the host computer, data are sent through a standard 10/100BASE-T Ethernet or IEEE 802.11b WLAN. You can read/set the digital I/O status via a pre-built web page or HMI software that supports the Modbus/TCP protocol.

5.2 ADAM-6050 18-ch Isolated Digital I/O Module

The ADAM-6050 is a high-density I/O module with a built-in 10/100BASE-T interface for seamless Ethernet connectivity. The module has 12 digital inputs and 6 digital outputs with 2000 V_{DC} isolation protection. All inputs have a latch function for handling important signal handling, and they can be used as 3-kHz counter and frequency input channels. The outputs support pulse output.

5.2.1 Specifications

- Communication: 10/100BASE-T Ethernet
- Supported protocols: MQTT, SNMP, Modbus/TCP, TCP/IP, UDP, HTTP, ICMP, DHCP, and ARP
- Supports P2P and GCL (see Section 6.7 and Chapter 8)

Digital Input

- Channels: 12
- Dry contact:
  - Logic level 0: Close to GND
  - Logic level 1: Open
- Wet contact:
  - Logic level 0: 0~3 V_{DC}
  - Logic level 1: 10~30 V_{DC}
- Supports 3-kHz counter input (32-bit with overflow flag)
- Frequency input range: 0.2~3 kHz
- Supports inverted digital input status

Digital Output

- Channels: 6
- Sink type: Open collector to 30 V, 100 mA (max. load)
- Supports 5-kHz pulse output
- Supports high-to-low and low-to-high delay output
- Leakage current: 200 μA (max.) (D version)
- 250 uA (max.) (D1 version)

General

- Built-in watchdog timer
- Isolation protection: 2000 V_{DC}
- Power input: Unregulated 10~30 V_{DC}
- Power consumption: 2 W (max.) @ 24 V_{DC}
- Power reversal protection
- Operating humidity: 20~95% RH (non-condensing)
- Storage humidity: 0~95% RH (non-condensing)
- Operating temperature: -20~70°C (D version: -40~70°C)
- Storage temperature: -30~80°C (D version: -40~85°C)

5.2.2 Application Wiring

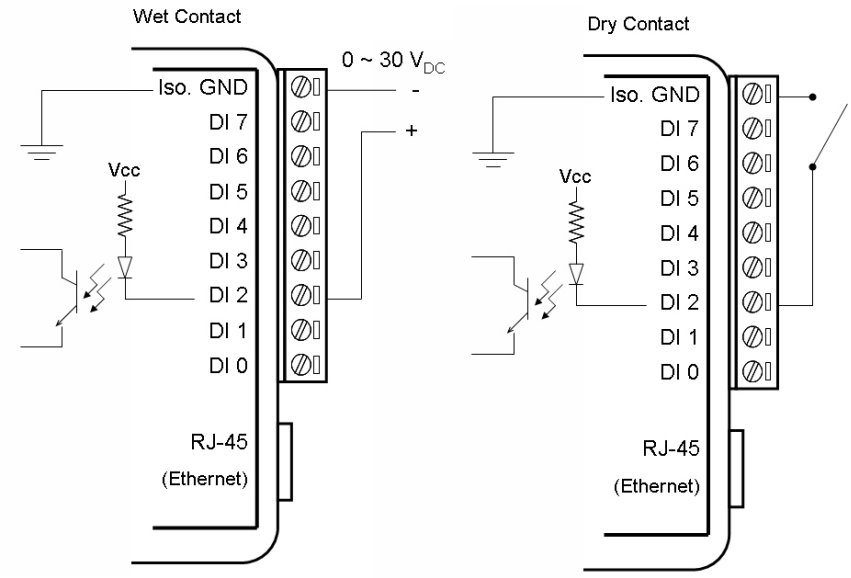


Figure 5.1 ADAM-6050 Digital Input Wiring

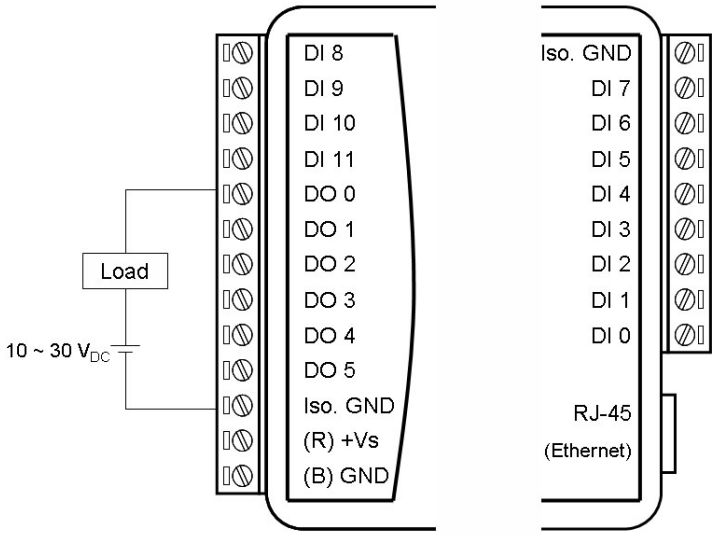


Figure 5.2 ADAM-6050 Digital Output Wiring

5.2.3 Address Assignment

Based on the Modbus/TCP standard, the addresses of ADAM-6000 I/O channels you place in the system are defined by a simple rule. See Appendix B.2.5 for information on mapping the I/O addresses. All inputs in the ADAM-6050 can be configured to be used as 32-bit counters (each counter has two addresses: a low word and a high word) by using Windows Utility (see Section 6.3).

5.3 ADAM-6051 14-ch Isolated Digital I/O Module w/ 2-ch Counter

The ADAM-6051 is a high-density digital I/O module with a built-in 10/100BASE-T interface for seamless Ethernet connectivity. The module has 12 digital inputs, 2 counter channels, and 2 digital outputs with 2000 V_{DC} isolation protection. All digital inputs have a latch function for important signal handling and can be used as 3-kHz counter and frequency input channels. The digital outputs support pulse output.

5.3.1 Specifications

- Communication: 10/100BASE-T Ethernet
- Supported protocols: MQTT, SNMP, Modbus/TCP, TCP/IP, UDP, HTTP, ICMP, DHCP, and ARP
- Supports P2P and GCL (see Section 6.7 and Chapter 8)

Digital Input

- Channels: 12
- Dry contact:
  - Logic level 0: Close to GND
  - Logic level 1: Open
- Wet contact:
  - Logic level 0: 0~3 V_{DC}
  - Logic level 1: 10~30 V_{DC}
- Supports 3-kHz counter input (32-bit with overflow flag)
- Supports 3-kHz frequency input
- Supports inverted digital input status

Counter Input

- Channels: 2 (32-bit with overflow flag)
- Maximum count: 4,294,967,295
- Frequency range:
  - 0.2~4500 Hz (frequency mode)
  - 0~4500 Hz (counter mode)
- Wet contact:
  - Logic level 0: 0~3 V_{DC}
  - Logic level 1: 10~30 V_{DC}

Digital Output

- Channels: 2
- Sink type: Open Collector to 30 V, 100 mA (maximum load)
- Supports 5-kHz pulse output

- Supports high-to-low and low-to-high delay output
- Leakage current: 200 μA (D version)

General

- Built-in watchdog timer
- Isolation protection: 2000 V_{DC}
- Power input: Unregulated 10~30 V_{DC}
- Power consumption: 3 W @ 24 V_{DC}
- Power reversal protection
- Operating humidity: 20~95% RH (non-condensing)
- Storage humidity: 0~95% RH (non-condensing)
- Operating temperature: -20~70°C (D version: -40~70°C)
- Storage temperature: -30~80°C (D version: -40~85°C)

5.3.2 Application Wiring

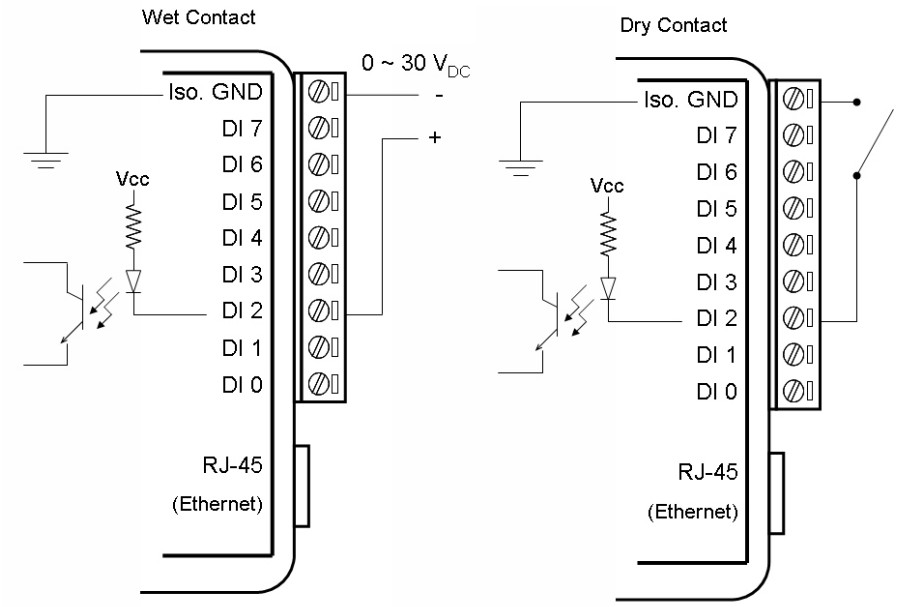


Figure 5.3 ADAM-6051 Digital Input Wiring

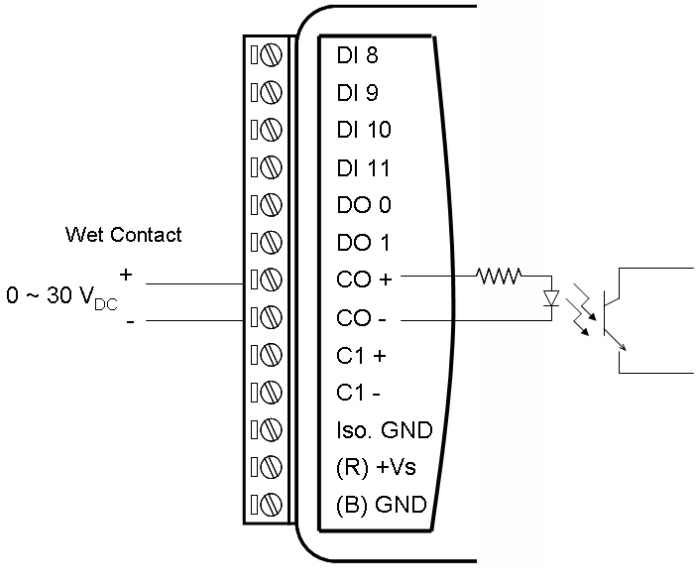


Figure 5.4 ADAM-6051 Counter (Frequency) Input

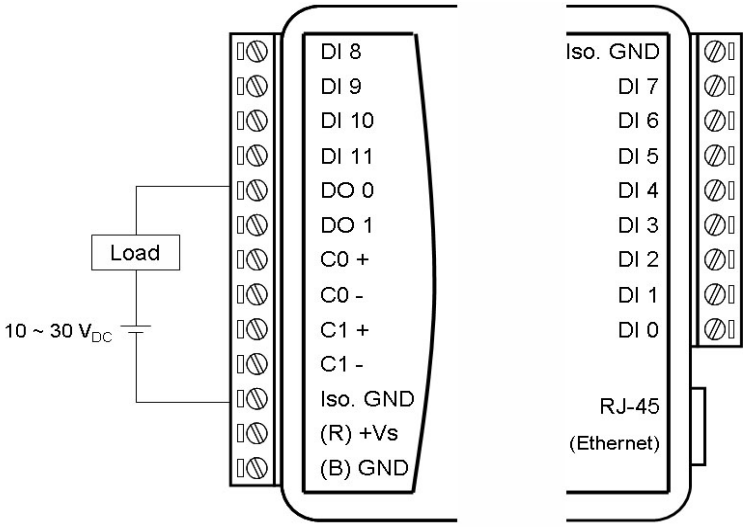


Figure 5.5 ADAM-6051 Digital Output Wiring

5.3.3 Address Assignment

Based on the Modbus/TCP standard, the addresses of ADAM-6000 module I/O channels you place in the system are defined by a simple rule. Please refer to Appendix B.2.6 for information on mapping the I/O addresses. All digital inputs in the ADAM-6051 can be configured to be used as 32-bit counters (each counter has two addresses: a low word and a high word) by using Windows Utility (see Section 6.3).

5.4 ADAM-6052 16-ch Source-Type Isolated Digital I/O Module

The ADAM-6052 is a high-density digital I/O module with a built-in 10/100BASE-T interface for seamless Ethernet connectivity. The module has 8 digital inputs and 8 digital outputs. All inputs have a latch function and can be used as 3-kHz counter and frequency input channels. The outputs support source-type and pulse output.

5.4.1 Specifications

- Communication: 10/100BASE-T Ethernet
- Supported protocols: MQTT, SNMP, Modbus/TCP, TCP/IP, UDP, HTTP, ICMP, DHCP, and ARP
- Supports P2P and GCL (see Section 6.7 and Chapter 8)

Digital Input

- Channels: 8
- Dry contact:
  - Logic level 0: Open
  - Logic level 1: Close to Ground
- Wet contact:
  - Logic level 0: 0~3 V_{DC}
  - Logic level 1: 10~30 V_{DC}
- Supports 3-kHz counter input (32-bit with overflow flag)
- Supports 3-kHz frequency input
- Supports inverted digital input status

Digital Output

- Channels: 8
- Source type: 10~35 V_{DC}, 1 A (per channel)
- Note: When operating at 70°C, the maximum total current for DO0~DO3 and DO4~DO7 is recommended to be less than 3 A
- Supports 5-kHz pulse output
- Supports high-to-low and low-to-high delay output

General

- Built-in watchdog timer
- Isolation protection: 2000 V_{DC}
- Power input: Unregulated 10~30 V_{DC}
- Power consumption: 2 W @ 24 V_{DC}
- Power reversal protection
- Operating humidity: 20~95% RH (non-condensing)
- Storage humidity: 0~95% RH (non-condensing)
- Operating temperature: -20~70°C (D version: -40~70°C)
- Storage temperature: -30~80°C (D version: -40~85°C)

Jumper Settings

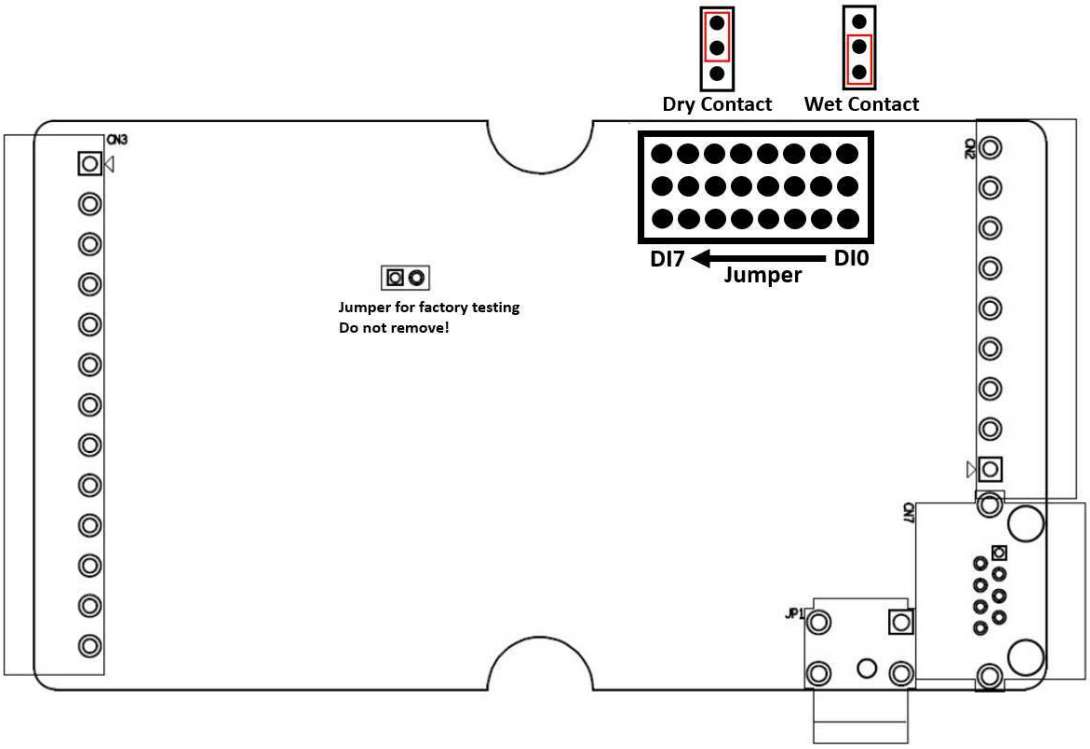


Figure 5.6 ADAM-6052 Jumper Settings

5.4.2 Application Wiring

The ADAM-6052 supports both dry and wet contact for the inputs. You can change between dry and wet contact mode by adjusting the jumper.

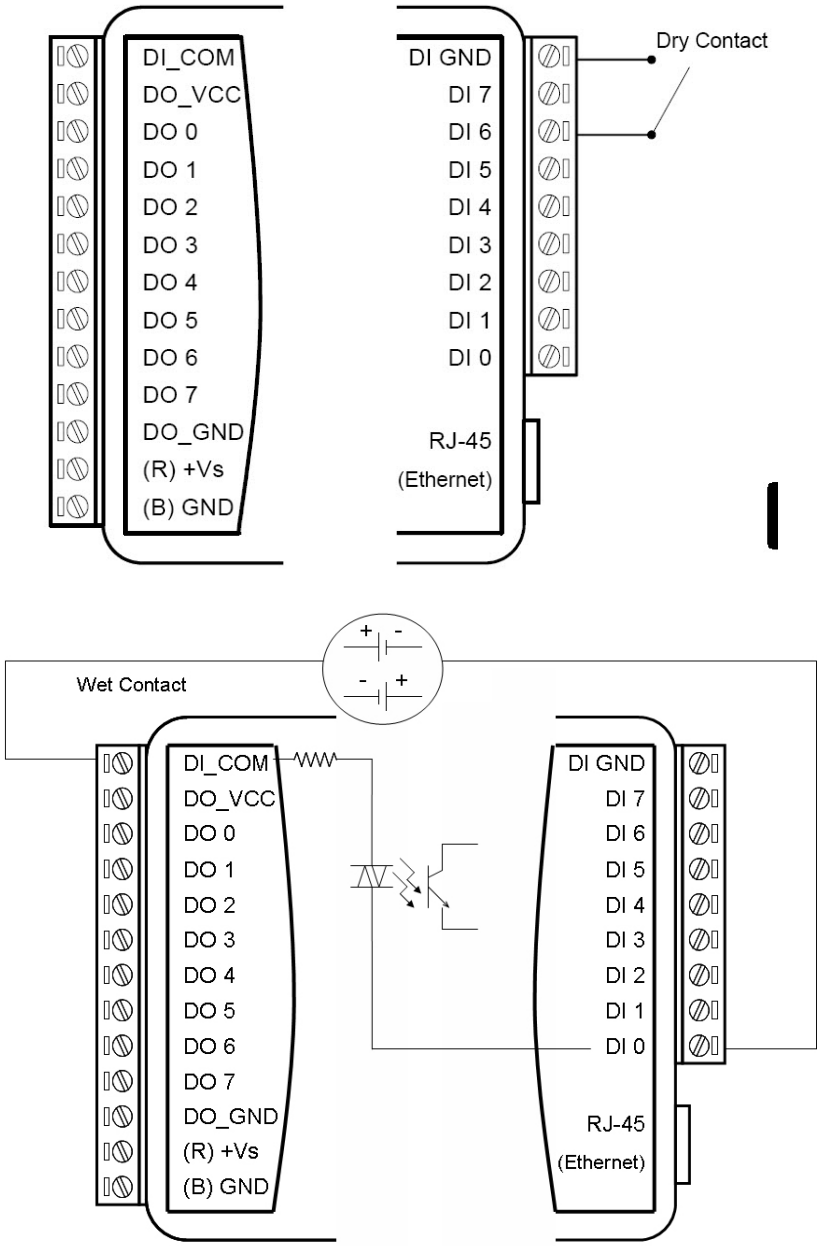


Figure 5.7 ADAM-6052 Digital Input Wiring

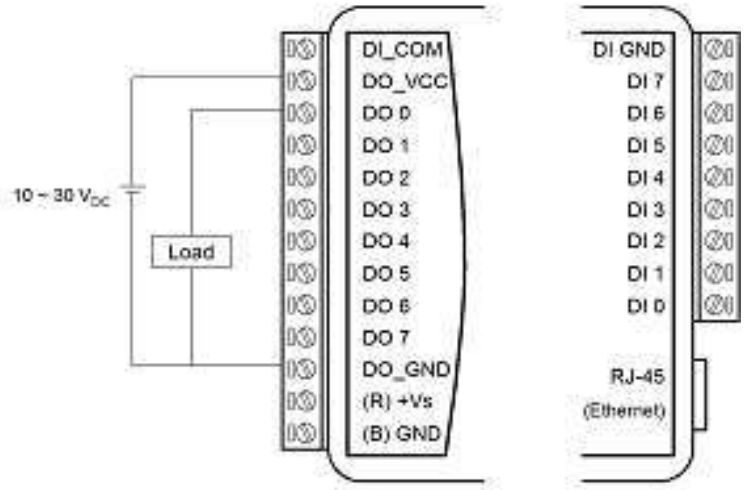


Figure 5.8 ADAM-6052 Digital Output Wiring

5.4.3 Address Assignment

Based on the Modbus/TCP standard, the addresses of ADAM-6000 module I/O channels are defined by a simple rule. Please refer to Appendix B.2.7 for information on mapping the I/O addresses. ADAM-6052 inputs can be configured as 32-bit counters (each counter has two addresses: a low word and high word) by using Adam/Apax .NET Utility (see Section 6.3).

5.5 ADAM-6060 6-ch Digital Input/6-ch Relay Module

The ADAM-6060 is a high-density I/O module with a 10/100BASE-T interface. Bonding with an Ethernet port and web page, the module provides 6 digital inputs and 6 relay outputs (Form A) and has a contact rating of 120 V_{AC} @ 0.5 A and 30 V_{DC} @ 1 A. All inputs have a latch for important signal handling and can be used as 3-kHz counter and frequency input channels. The outputs support pulse output.

5.5.1 Specifications

- Communication: 10/100BASE-T Ethernet
- Supported protocols: MQTT, SNMP, Modbus/TCP, TCP/IP, UDP, HTTP, ICMP, DHCP, and ARP
- Supports P2P and GCL (see Section 6.7 and Chapter 8)

Digital Input

- Channels: 6
- Dry contact:
  - Logic level 0: Close to GND
  - Logic level 1: Open
- Wet contact:
  - Logic level 0: 0~3 V_{DC}
  - Logic level 1: 10~30 V_{DC}
- Supports 3-kHz counter input (32-bit with overflow flag)
- Frequency input range: 0.2 Hz~3 kHz
- Support inverted digital input status
- Keep/discard counter value when powered off

Relay Output

- Channels: 6 (Form A)
- Contact rating (Resistive):
  - 120 V_{AC} @ 0.5 A
  - 30 V_{DC} @ 1 A
- Breakdown voltage: 500 V_{AC} (50/60 Hz)
- Relay-on time: 7 ms
- Relay-off time: 3 ms
- Total switching time: 10 ms
- Insulation resistance: 1 GΩ (min.) @ 500 V_{DC}
- Maximum switching rate: 20 operations/min (at rated load)
- Electrical endurance
  - At 12 V/10 mA: 5 x 10^7 operations (typical)
  - At 6 V/100 mA: 1 x 10^7 operations (typical)
  - At 60 V/500 mA: 5 x 10^5 operations (typical)
  - At 30 V/1000 mA: 1 x 10^6 operations (typical)
  - At 30 V/2000 mA: 2 x 10^5 operations (typical)
- Mechanical endurance
  - 10^8 operations (typical)
- Supports pulse output (max. 3 Hz)

General

- Built-in watchdog timer
- Isolation protection: 2000 V_{DC}
- Power input: Unregulated 10~30 V_{DC}
- Power consumption: 3 W (max.) @ 24 V_{DC}
- Power reversal protection
- Operating humidity: 20~95% RH (non-condensing)
- Storage humidity: 0~95% RH (non-condensing)
- Operating temperature: -20~70°C (D version: -40~70°C)
- Storage temperature: -30~80° C (D version: -40~85°C)

5.5.2 Application Wiring

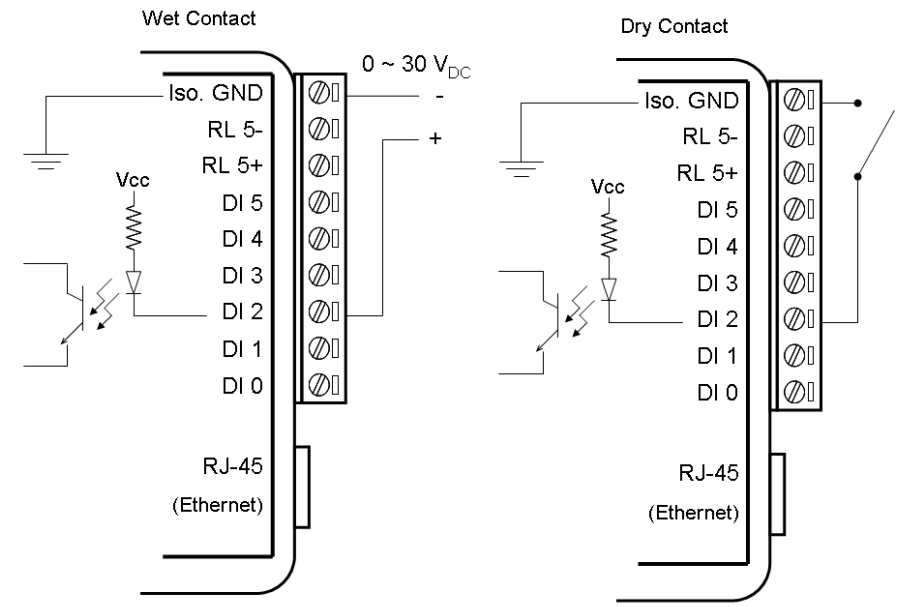


Figure 5.9 ADAM-6060 Digital Input Wiring

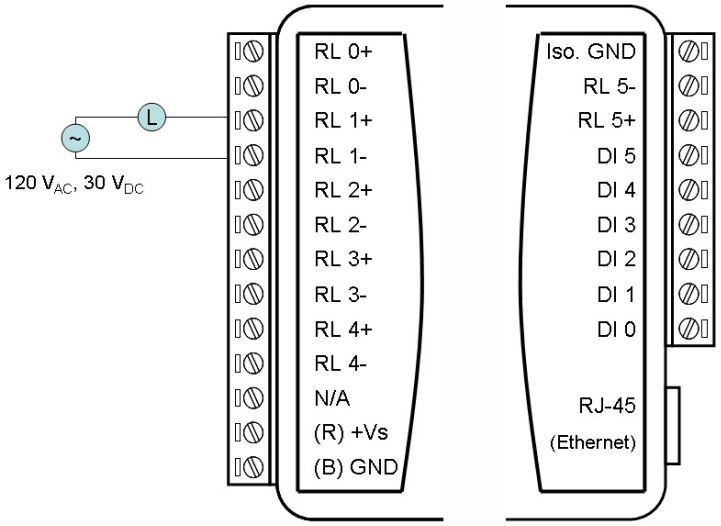


Figure 5.10 ADAM-6060 Relay Output Wiring

5.5.3 Address Assignment

Based on the Modbus/TCP standard, the addresses of ADAM-6000 module I/O channels you place in the system are defined by a simple rule. Refer to Appendix B.2.8 for information on mapping the I/O addresses. All inputs in the ADAM-6060 can be configured to be used as 32-bit counters (each counter consists of two addresses: a low word and a high word) by using Windows Utility (see Section 6.3).

5.6 ADAM-6066 6-ch Digital Input/6-ch Power Relay Module

The ADAM-6066 is a high-density I/O module with a 10/100BASE-T interface for seamless Ethernet connectivity. It has 6 digital inputs and 6 high-voltage relay outputs (Form A). The module has a contact rating of 250 V_{AC} @5A and 30 V_{DC} @ 3 A. All inputs have a latch function for important signal handling and can be used as 3-kHz counter and frequency input channels. The outputs support pulse output.

5.6.1 Specifications:

- Communication: 10/100BASE-T Ethernet
- Supported protocols: MQTT,SNMP,Modbus/TCP, TCP/IP, UDP, HTTP, ICMP, DHCP, and ARP Modbus/TCP,SNMP,TCP/IP, UDP, HTTP, ICMP, DHCP and ARP
- Supports P2P and GCL (see Section 6.7 and Chapter 8)

Digital Input

- Channels: 6
- Dry contact:
  - Logic level 0: Close to GND
  - Logic level 1: Open
- Wet contact:
  - Logic level 0: 0~3 V_{DC}
  - Logic level 1: 10~30 V_{DC}
- Supports 3-kHz counter input (32-bit with overflow flag)
- Supports 3-kHz frequency input
- Supports inverted digital input status

Relay Output

- Channels: 6 (Form A)
- Contact rating (Resistive):
  - 250 V_{AC} @ 5 A
  - 30 V_{DC} @ 3 A
- Breakdown voltage: 500 V_{AC} (50/60 Hz)
- Relay on time: 7 ms
- Relay off time: 3 ms
- Total switching time: 10 ms
- Insulation resistance: 1 GΩ (min.) at 500 V_{DC}
- Maximum switching rate: 20 operations/min (at rated load)
- Electrical endurance: 1 x 10^5 operations
- Mechanical endurance
  - 2 x 10^7 operations (typical)
  - (Under no load at an operating frequency of 180 operations/min)
- Supports pulse output (max. 3 Hz)

General

- Built-in watchdog timer
- Isolation protection: 2000 V_{DC}
- Power input: Unregulated 10~30 V_{DC}
- Power consumption: 2.5 W @ 24 V_{DC}
- Power reversal protection
- Operating humidity: 20~95% RH (non-condensing)
- Storage humidity: 0~95% RH (non-condensing)
- Operating temperature: -20~70°C (D version: -40~70°C)
- Storage temperature: -30~80°C (D version:-40~85°C)

5.6.2 Application Wiring

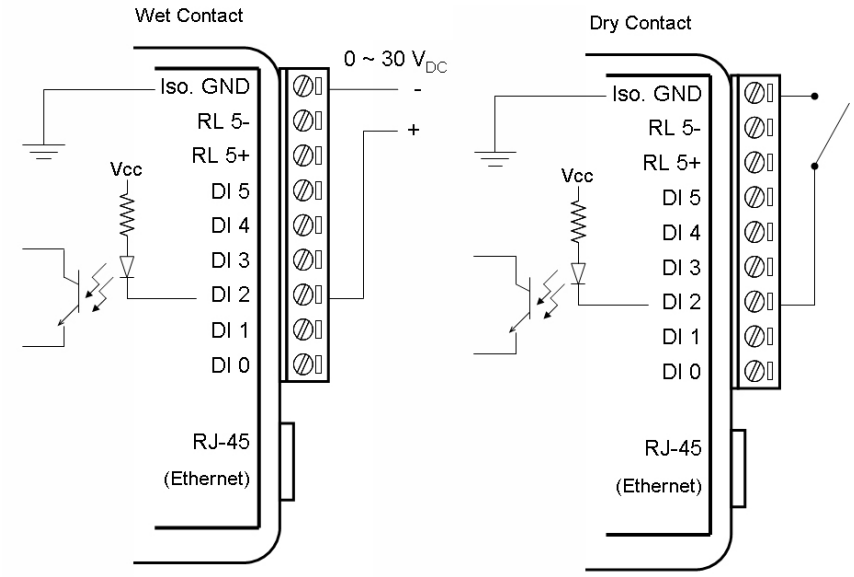


Figure 5.11 ADAM-6066 Digital Input Wiring

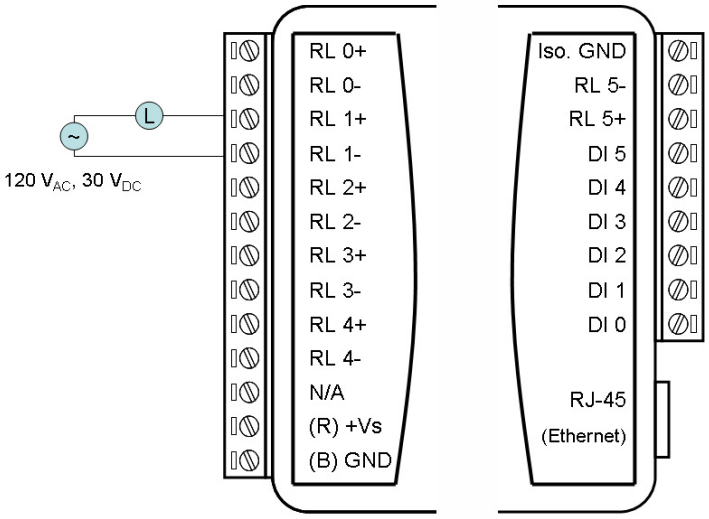


Figure 5.12 ADAM-6066 Relay Output Wiring

5.7 Digital Output Diagnostic Function

When a digital output is active, a circuit wire break or short to ground will cause the output to fail. To help clarify such a situation quickly, ADAM-6000 modules (all D versions) have a digital output diagnostic function that can detect abnormalities in the digital output and issue a notification. The diagnostic status is given according to the following groups:

Module	Diagnostic Group	Output Channel
ADAM-6017	Group 0	DO0, DO1
ADAM-6050	Group 0	DO0, DO1
	Group 1	DO2, DO3
	Group 2	DO4, DO5
ADAM-6050-D1	Group 0	DO 0
	Group 1	DO 1
	Group 2	DO 2
	Group 3	DO 3
	Group 4	DO 4
	Group 5	DO 5
	Group 6	DO 6
	Group 7	DO 7
ADAM-6051	Group 0	DO0, DO1
ADAM-6052	Group 0	DO0
	Group 1	DO1
	Group 2	DO2
	Group 3	DO3
	Group 4	DO4
	Group 5	DO5
	Group 6	DO6
	Group 7	DO7

Note that for the ADAM-6050 and ADAM-6051, each group corresponds to a pair of digital outputs, whereas for the ADAM-6052, each group corresponds to an individual channel. When an error occurs with one or both channels in a group, the diagnostic status for that group will change. Possible reasons for an abnormality are given as follows.

For the ADAM-6017, ADAM-6050, and ADAM-6051

When the digital output is not active:

- The digital output circuit wire break has occurred (open load)
- The digital output connection is short to ground

When the digital output is active:

- The output has been exposed to an overcurrent (>1 A)

Note: To ensure that the digital outputs and diagnostic function operate normally, each digital output should be configured within the specification for individual channels: 30 V, 100 mA (max.).

For the ADAM-6052

When the digital output is active:

- The digital output connection is short to ground
- The output has been exposed to an overcurrent (>1 A, typical)

5.7.1 How to Obtain the Digital Output Diagnostic Status

The digital output diagnostic status can be obtained using Adam/Apax .Net Utility, the Modbus address, or an ASCII command.

Obtaining the Digital Output Diagnostic Status With Adam/Apax .NET Utility

Since Version 2.05.10B08, Adam/Apax .NET Utility has had a digital output diagnostic function. In the example shown in Figure 5.13, the digital output diagnostic status is abnormal for Group 1 (DO2~DO3) and Group2 (DO4~DO5) (note that individual groups will appear in this field only if their status is abnormal; thus, Group 0 does not appear in this example). This indicates that a problem has occurred with one of these output channels.

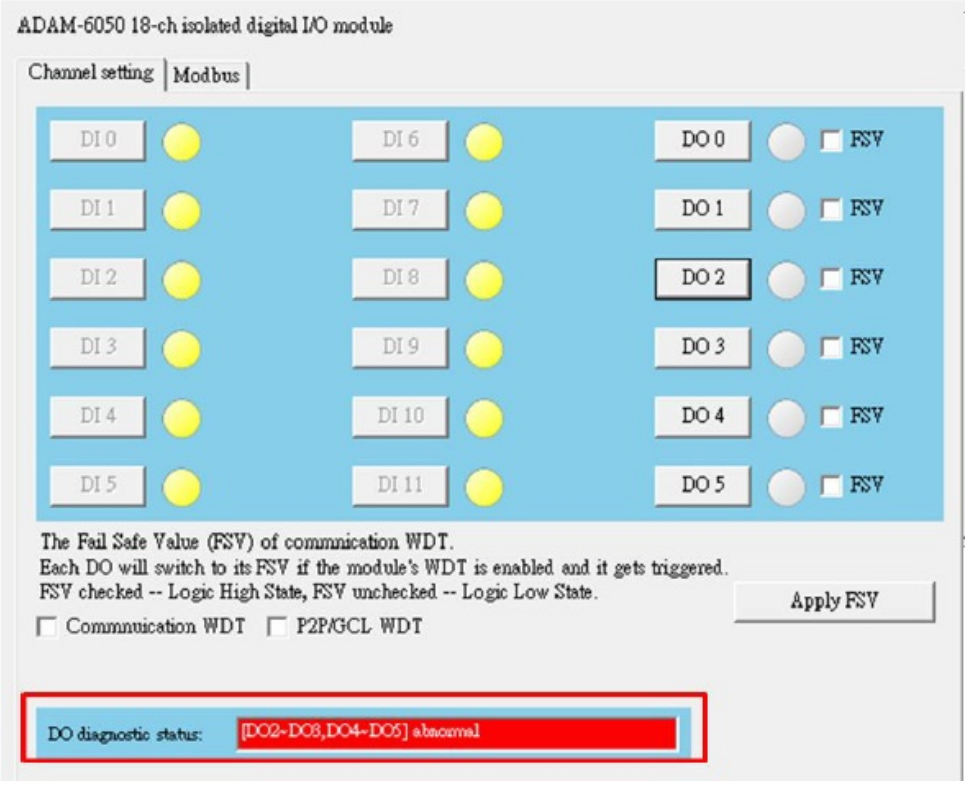


Figure 5.13 Abnormal DO Diagnostic Status

In Figure 5.14 the digital output diagnostic status is "All normal" meaning that all channels in the group are connected correctly (no wire break or short to ground) before the digital outputs are activated.

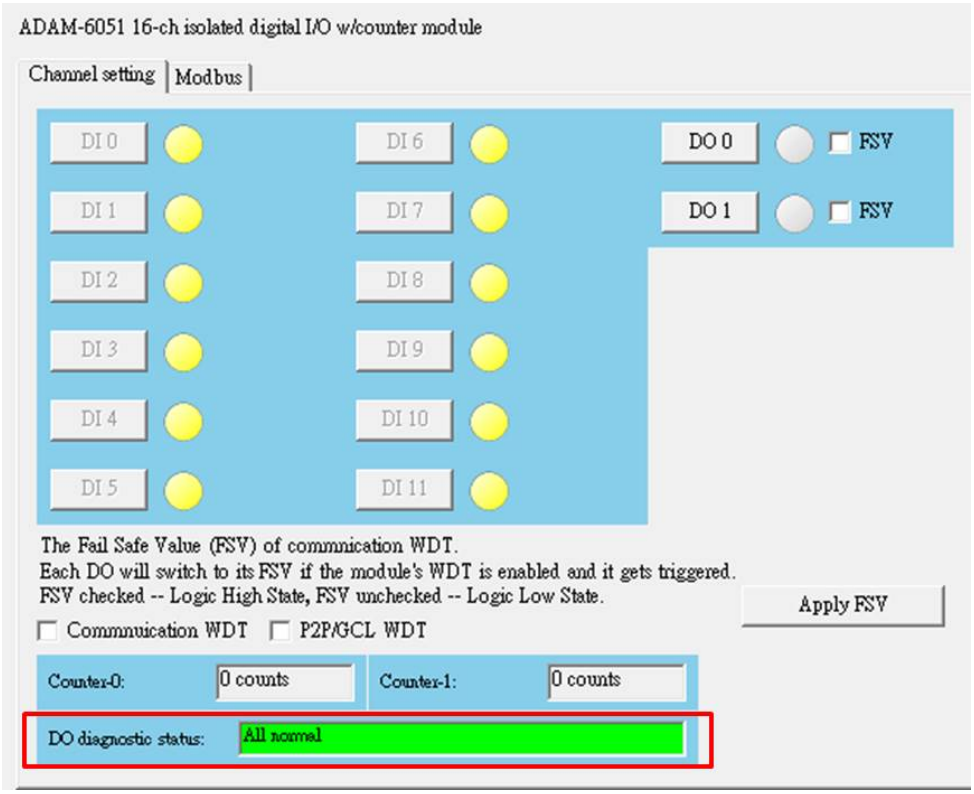


Figure 5.14 Normal DO Diagnostic Status

Obtaining the Digital Output Diagnostic Status With a Modbus Address Value

Address (4X)	Channels	Description	Attribute
40307	All	Digital output diagnostic status (for D version)	Read

The following table shows the bit positions relative to the groups for the ADAM-6050, ADAM-6051, and ADAM-6052. The status of the groups can thus be interpreted according to the value shown in each bit position. The group status values will be displayed as binary values, with Bit 1 being the right-most bit position and Bit 8 being the left-most bit position.

Bit Position for Modbus Address 40307	Relative Group for Interpreting the Digital Output Diagnostic Status Value		
	ADAM-6050	ADAM-6051	ADAM-6052/ADAM-6050-D1
Bit 1	Group 0	Group 0	Group 0
Bit 2	Group 1	Reserved	Group 1
Bit 3	Group 2		Group 2
Bit 4	Reserved		Group 3
Bit 5			Group 4
Bit 6			Group 5
Bit 7			Group 6
Bit 8			Group 7

Example (ADAM-6050): In the case of the previous example shown in Figure 5.13, the group status values would be "xxxxx110". Here, Bits 1, 2, and 3 indicate the digital output diagnostic status of Groups 0, 1, and 2, respectively. The group status can thus be interpreted as follows:

- Group 0 = 0 (Normal)
- Group 1 = 1 (Abnormal)
- Group2 = 1 (Abnormal)

Obtaining the Digital Output Diagnostic Status With an ASCII Command

This example shows the ASCII command and response for requesting the status of digital outputs.

Syntax	$017	
Response	!01(Group#n)…(Group #1)(Group#0)(cr)	
Example	Command:	$017
	Response:	!01110

Because the ADAM-6050 has three digital output groups for the diagnostic status, the bit positions from right to left indicate the status of Groups 0~2 are as follows:

- Group 0 = 0 (Normal)
- Group 1 = 1 (Abnormal)
- Group 2 = 1 (Abnormal)

For ADAM-6050-D1, each digital output channel has its own diagnostic status.

Syntax	$017	
Response	!01(DO#0)…(DO #n)(DO#0)(cr)	
Example	Command:	$017
	Response:	!01101110

- DO0 = 1 (Abnormal) DO1 = 0 (Abnormal)
- DO2 = 1 (Abnormal) DO3 = 1 (Abnormal)
- DO4 = 1 (Abnormal) DO5 = 0 (Normal)

Chapter 6

System Configuration Guide

6.1 System Requirements

Host Computer

- Microsoft Windows XP/7
- 32 MB RAM
- 20 MB hard disk space
- VGA color monitor
- Mouse or other pointing device
- 10/100-Mbps Ethernet Card

Communication Interface

- 10/1000-Mbps Ethernet hub (min. 2 ports)
- Two Ethernet cables (RJ-45)
- Crossover Ethernet cable (RJ-45)

6.2 Installing Adam/Apax .NET Utility

Adam/Apax .NET Utility is an application provided by Advantech for the configuration and operation of ADAM modules. The installation file can be found on the companion CD with your ADAM module, and it is also available for download for free at http://www.advantech.com (click on Download Area under Service & Support for the latest version). Once installed, a shortcut to the utility will appear on your desktop.

Note: Before installing Adam/Apax .NET Utility, you will need to install .NET Framework 2.0 or later.

6.3 Adam/Apax .NET Utility Overview

Adam/Apax .NET Utility is a graphical interface for configuring and operating ADAM modules. It is also a convenient tool for testing and monitoring remote DA&C systems. The following text instructions describe how to use the utility.

To start Adam/Apax .NET Utility, double-click the shortcut on the desktop or click the icon in the start menu folder. When the program is first opened, the main window will appear as shown in Figure 6.1.

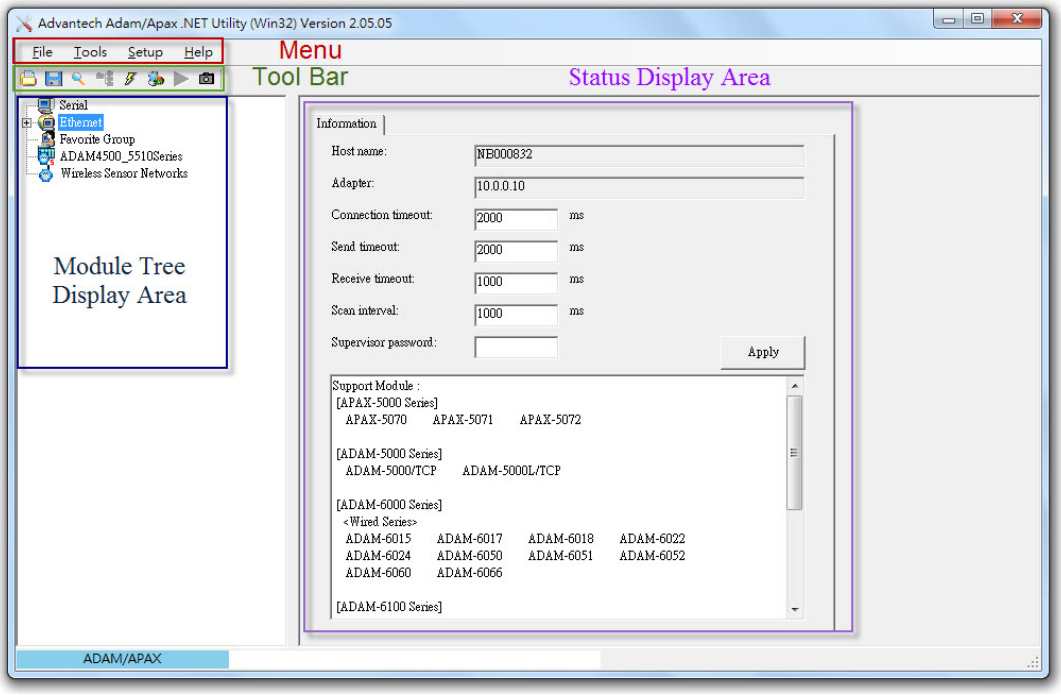


Figure 6.1 Adam/Apax .NET Utility Operation Window

As shown in the figure, this window has four main areas: 1) the Menu Bar, 2) the Toolbar, 3) the Module Tree Display Area, and 4) the Status Display Area.

6.3.1 Menu Bar

The menu bar comprises four menus: File, Tools, Setup, and Help. The items under each menu are described as follows:

File Menu

Open Favorite Group	Allows you to load a saved configuration file for a favorite group
Save Favorite Group	Allows you to save a favorite group into a configuration file
Auto-Initial Group	Checking this option will load the same favorite group configuration next time you launch Adam/Apax .NET Utility
Exit	Exit Adam/Apax .NET Utility

Tools Menu

Search Device	Search for all ADAM modules connected to the host PC (see Section 6.3.5)
Add Devices to Group	Adds ADAM modules to the favorite group; only selected devices in the Module Tree Display Area will be added to the group
Group Configuration	This item is for updating the firmware, configuration, and HTML files of a single module or multiple modules. The configuration file includes settings on device information, general information, P2P and streaming, GCL, and Modbus address XML files. The configuration file can be exported as a Cfg file from the **Firmware** tab in the Status Display Area.
Terminal for Command Testing	Launches a terminal for communicating with ADAM modules via ASCII command and Modbus/TCP (see Sections 7.3 and 7.4 for more information)

Print Screen	Exports the Adam/Apax .NET Utility screen as an image file
Monitor Stream/Event Data	ADAM modules support a datastream function. This allows you to define the host (such as a PC) by IP, and ADAM modules will then periodically transmit their I/O status to the host. The IP address and transmission period can be configured from the **Stream** tab in the Status Display Area. The **Stream** tab is introduced in Section 6.3.5.
Monitor Peer to Peer	Select this option to receive messages from ADAM modules that have the P2P (event trigger) function enabled
Monitor GCL IO Data Message	Select this option to receive I/O data messages from ADAM modules that have the GCL function enabled.

Note! *When you enable the GCL function, the datastream function will automatically be disabled until you disable the GLC function.*

Setup Menu

Favorite Group	This is for configuring your Favorite group, including adding devices, modifying or deleting current devices, sorting current devices, and diagnosing device connections
Refresh Serial and Ethernet	This will cause Adam/Apax .NET Utility to refresh the serial and LAN network connection
Add COM Ports	This is for adding serial COM ports to Adam/Apax .NET Utility (this does not apply to ADAM-6000 modules)
Show TreeView	Clicking on this item shows the Module Tree Display Area
Allow Calibration	Select this to enable/disable module calibration

Help Menu

Check Up-to-Date on the Web	Connect to the Advantech download website and checks for the latest version of the utility.
About	This shows information on the version of Adam/Apax .NET Utility currently installed on your computer

6.3.2 Toolbar

The toolbar (Figure 6.2) contains icons for the most commonly used menu items.

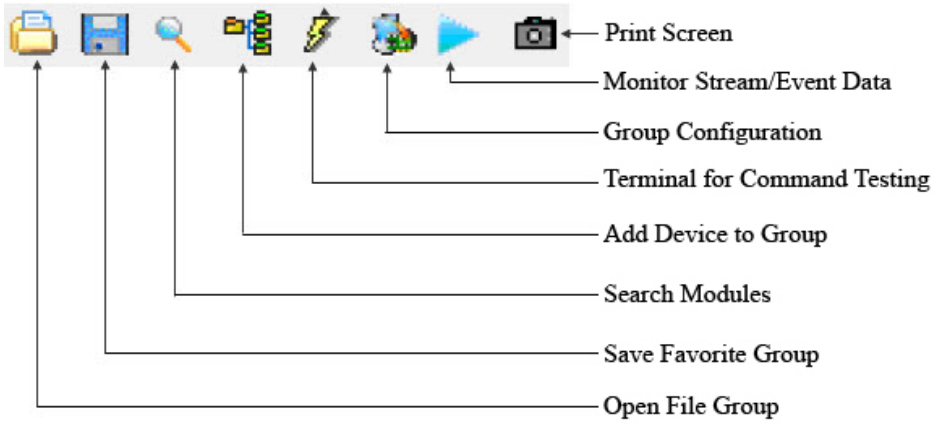


Figure 6.2 Adam/Apax .NET Utility Toolbar

6.3.3 Module Tree Display Area

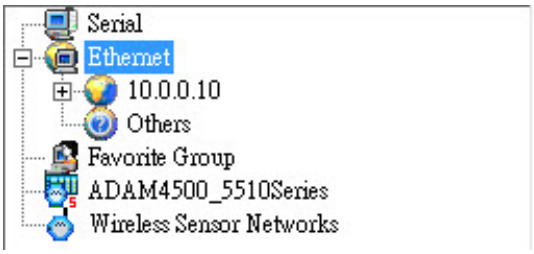


Figure 6.3 Adam/Apax .NET Utility Module Display Area

The Module Tree Display Area is the left part of the main window. There are five major categories in the display area, some of which will be visible only when you have certain modules connected:

Serial	All serial I/O modules (ADAM-4000, ADAM-4100, and ADAM-5000 RS-485 modules) connected to the host PC will be listed in this category.
Ethernet	All Ethernet I/O Modules (ADAM-5000, ADAM-6000, and ADAM-6100 TCP modules) connected to the host PC will be listed in this category.
Favorite Group	Devices you have added to your personal favorite group are listed under this category, making it easier for you to locate specific modules. The favorite group can contain multiple groups. To create a new group, right-click on **Favorite Group** and select **Add New Group**. You will then be prompted to enter a name for the group. To add devices to that group, right-click on the group you have created and select **Add New Device**. You will then be prompted to give the new device a name and select the module type from either the **Serial Device** tab or the **Ethernet Device** tab. You can also enter the device parameters here. In addition to modifying the group (select **Modify Group**) and deleting the group (select **Delete Group**), you can also select diagnose the connection for a group (select **Diagnose Connection**) by right-clicking on the group name.
ADAM-4500_5510Series	Any DOS-based remote controllers (e.g., ADAM-4500 and ADAM-5510 series) will be listed under this category.
Wireless Sensor Networks	Any wireless modules (e.g., WISE-4000 series) connected to the host PC will be listed under this category.

6.3.4 Status Display Area

The Status Display Area is the main window that you will interact with. All configuration and testing is performed here. The content of this window will vary depending on which items you select in the Module Tree Display Area.

6.3.5 Configuration of ADAM-6000 Modules

Once an ADAM-6000 module has been connected to the host PC and you have searched for it, you will find it listed in the Module Tree Display Area under the Ethernet category. Select the Ethernet category on the Module Tree Display Area and click the **Search Modules** icon on the Toolbar. Adam/Apax .NET Utility will then search for all ADAM-6000 modules on the Ethernet network. If this is the first time you have connected the module, its IP will be 10.0.0.1 by default and it will appear under Others in the Module Tree Display Area.

Note! *If a network firewall is enabled, you might not be able to connect to your ADAM-6000 module. You may need to add an exception for Adam/Apax .NET Utility in Windows Firewall via Windows Control Panel.*

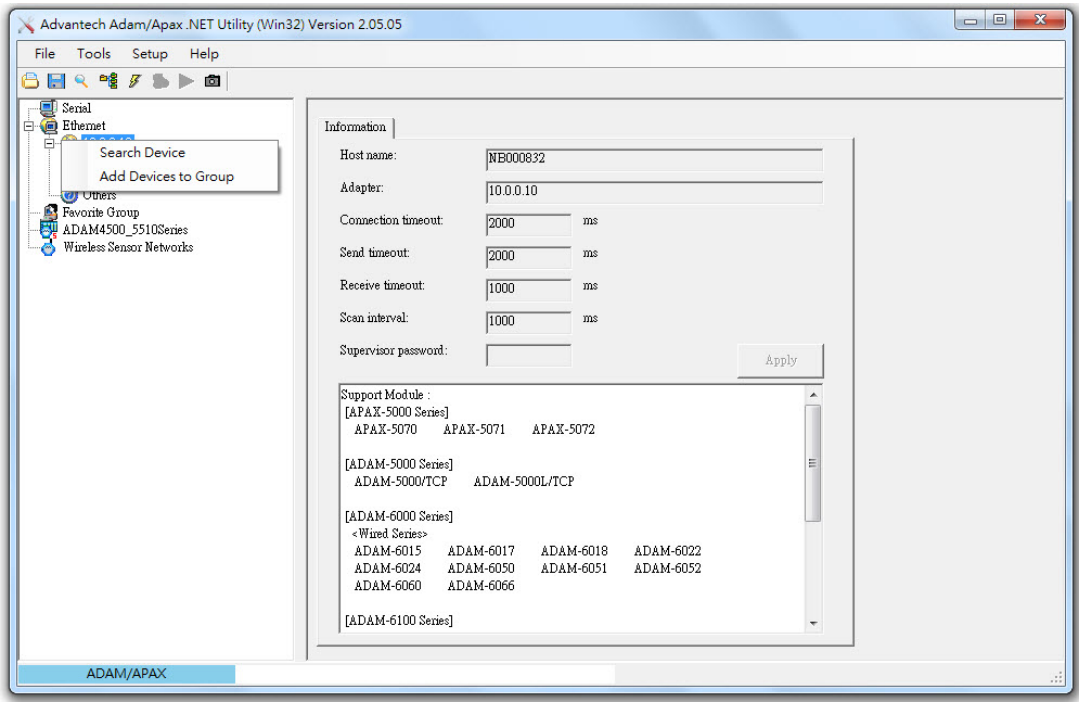
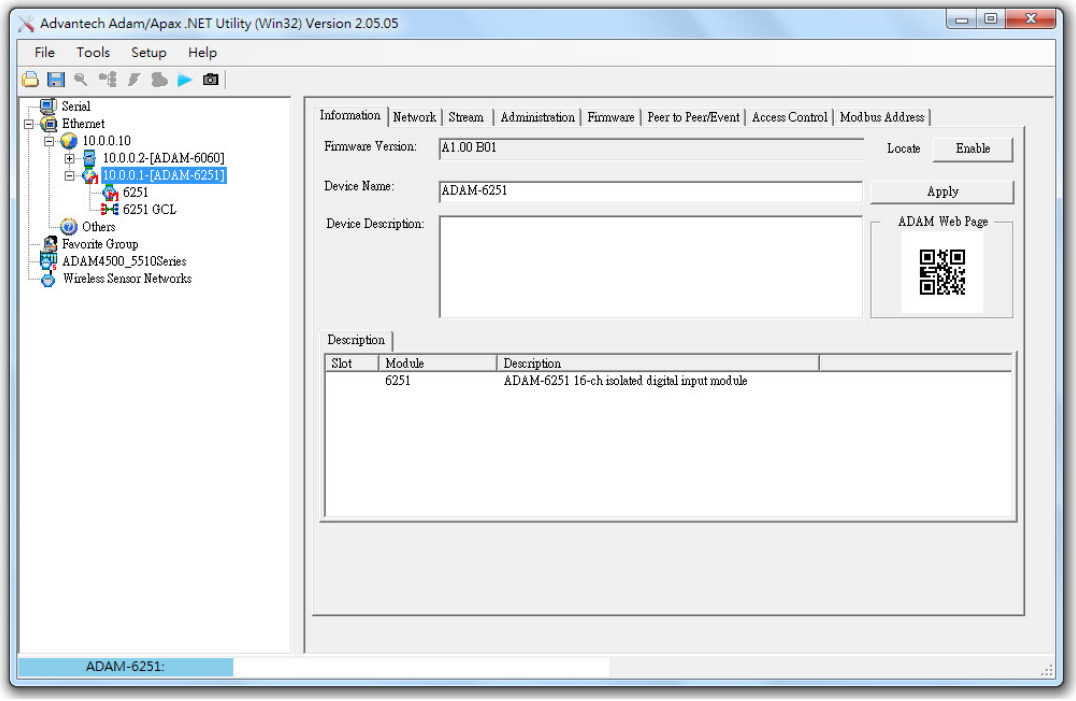


Figure 6.4 Adam/Apax .NET Utility - Searching for Devices

You need to change the IP address of the ADAM-6000 module so that it is the same subnet as the host PC. Enter the correct IP address, subnet address, and default gateway on the Status Display Area and then click **Apply Change**. A dialog box will appear asking you to enter the password. The default password of ADAM-6000 modules is "00000000" (without quotation marks). After you have entered the correct password, the ADAM-6000 module will be under IP of your host PC. Note that you can change the password later.

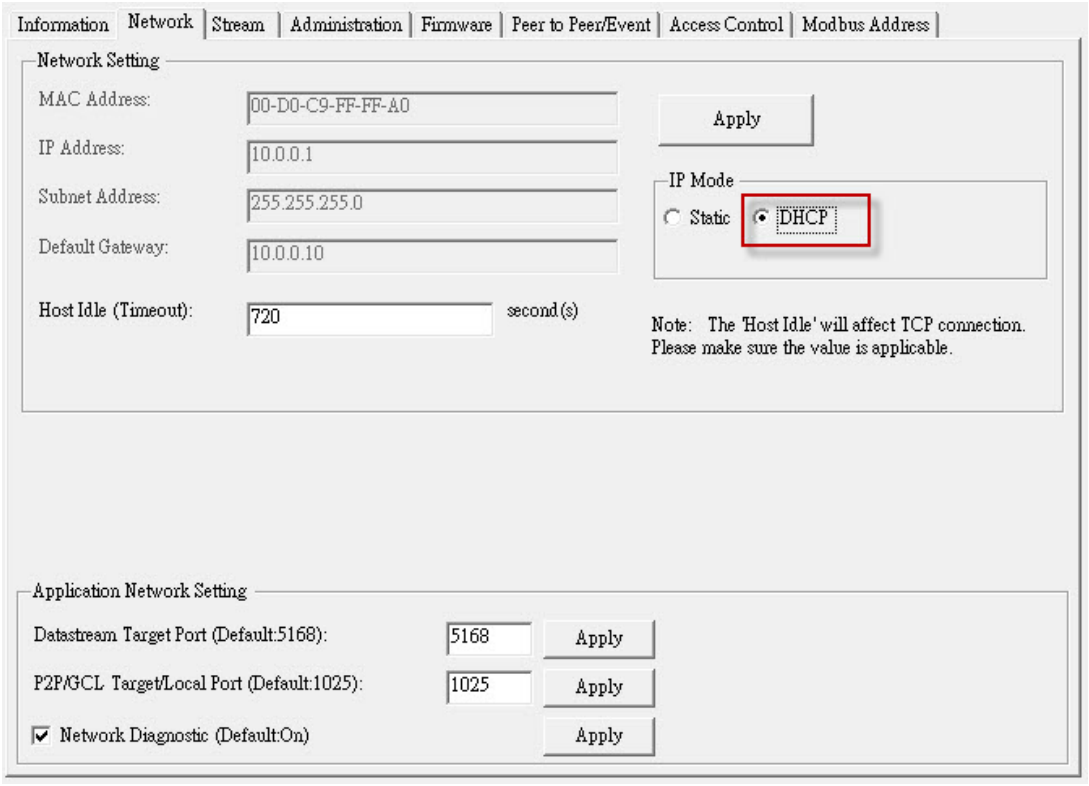
When you select the IP address of the ADAM-6000 modules you want use in Module Tree Display Area, eight tabs will become available in the Status Display Area. These tabs are for the general configuration of that module. Once you have changed any settings, remember to click **Apply** or **Apply Change**. These eight tabs are detailed in the following sections.

The Information Tab



This tab shows the firmware version as well as the device name and device description, both of which can be modified from here. Giving your modules a specific name and description can be useful for when several ADAM-6000 modules are connected to the same network. You can also enable/disable the locate function, which is intended to help you to physically locate the selected module (basically, when you click Enable, the module's Status/Link LED indicator will be red for 30 s; see Section 1.5 for a description of the LED status). The tab also shows a QR code that will be generated for the URL of the selected module's web server. Note that individual module configurations can be saved/loaded from the **Firmware** tab (explained later in this section). The configuration file contains settings of network, stream/event data, access control, and I/O configuration.

The Network Tab



This tab contains two main panels: the **Network Settings** panel and the **Application Network Settings** panel. The content of these panels is described in the following text.

The Network Setting Panel

This panel is for adjusting typical network configuration settings for ADAM modules. Here, you can set the network connection protocol (DHCP or static IP), IP address, subnet address, default gateway, and host idle time (timeout).

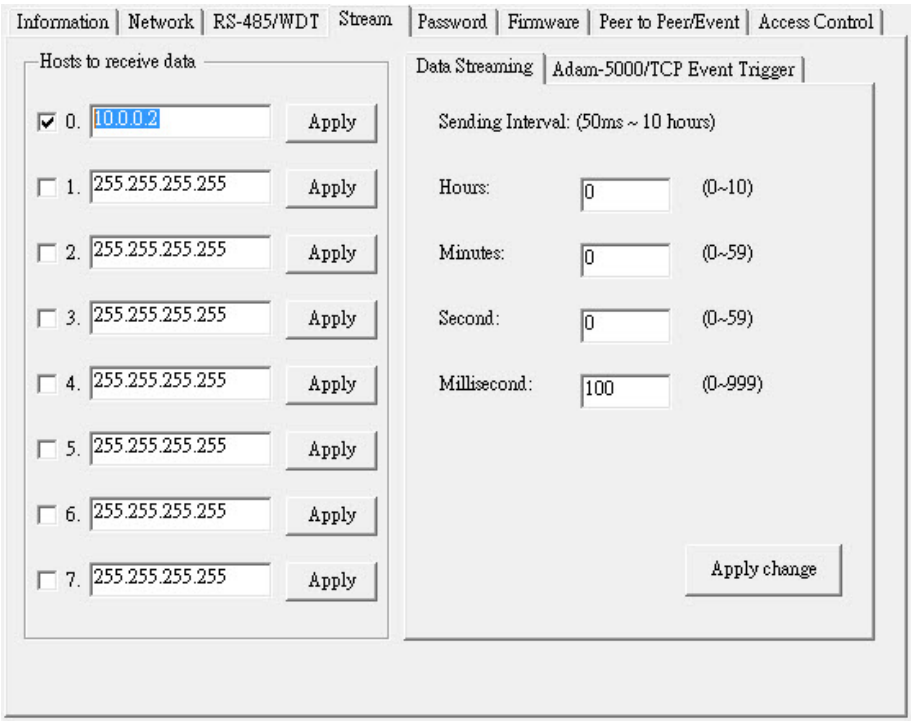
The Application Network Settings Panel

This panel is for configuring the datastream and P2P/GCL port. When Network Diagnostic is selected, the ADAM module will periodically monitor and diagnose the Ethernet switch. If the Ethernet port is not used for communication, this function should be disabled.

Note 1: When a web browser is used to open the web page on an ADAM-6000 module, a Java virtual machine (JVM) will use several TCP connections to download a Jar file. These connections will be released after the Jar file has been downloaded.

Note 2: After the GCL/P2P port settings have been modified, the module will reboot automatically (connection recovery time: 3 s).

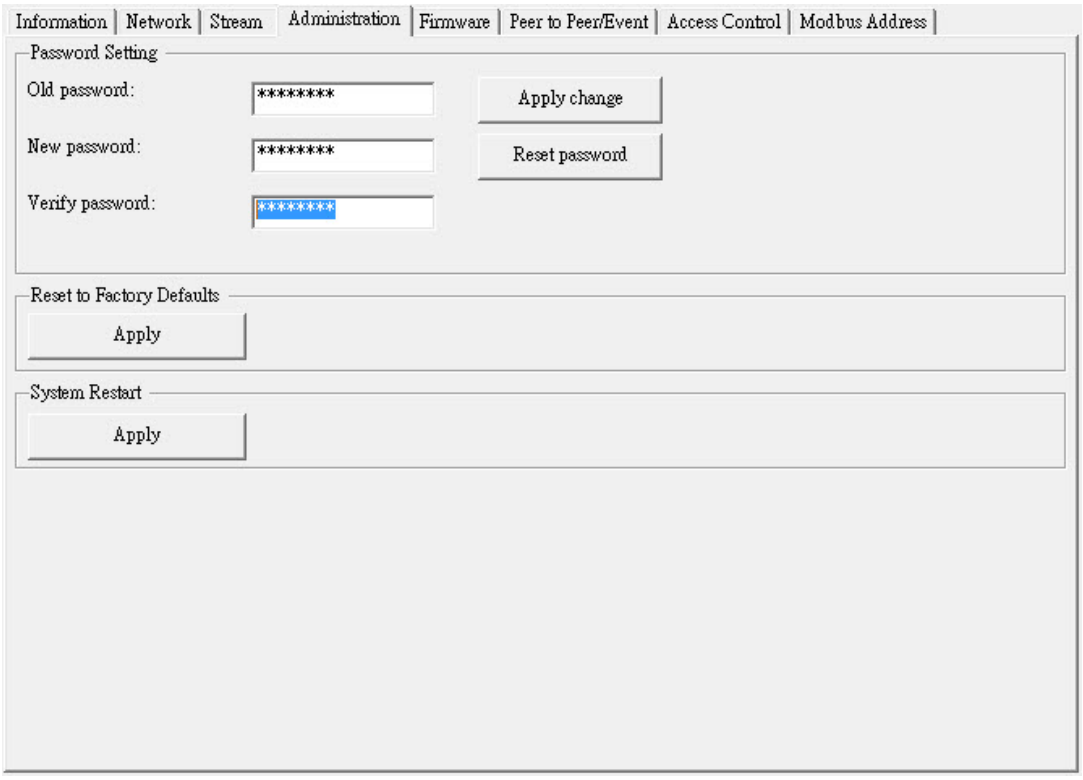
The Stream Tab



ADAM-6000 modules can be configured to periodically transmit data to up to eight hosts. This sequence of signals is called a datastream. On the **Stream** tab, the **Hosts to receive data** panel allows you to define the IP addresses of hosts that will receive data from ADAM-6000 modules. On the **Data Streaming** tab (right-hand side), you can also set the intervals at which ADAM-6000 modules will transmit data to the Hosts.

Note! *In the above image, the **ADAM-5000/TCP Event Trigger** tab is specifically for the ADAM-5000.*

The Administration Tab



Note! *The default password is "00000000"*

The **Administration** tab allows you to set the password for the selected ADAM-6000 module. To change the password, you will need to enter the current password in the Old password box and then enter the new password in the New password and Verify password boxes. The password is required for many configurations and operations, so setting your own password can help ensure system security. You can also apply the factory default settings and restart the module from the tab.

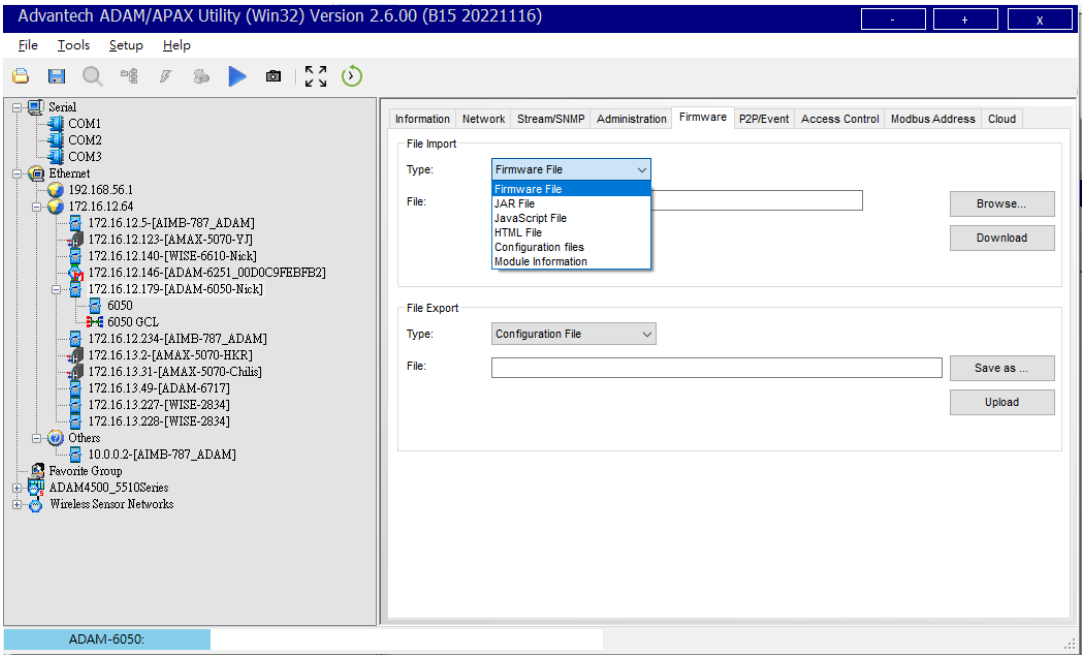
Reset to Default Factory Settings

Click Apply on this panel to clear the system configuration of the selected ADAM-6000 module and restore the factory default settings.

System Restart

Click **Apply** to restart the module. Note that your current settings will be preserved.

The Firmware Tab



Advantech will occasionally release new firmware versions to add or improve the functionality of ADAM-6000 modules. Visit http://www.advantech.com to check for the latest firmware downloads. Firmware downloads will contain four file types: Bin, Html, Xml, and Jar. The Bin file is the actual firmware file and the Html and Jar files are for the web server on the ADAM-6000 module.

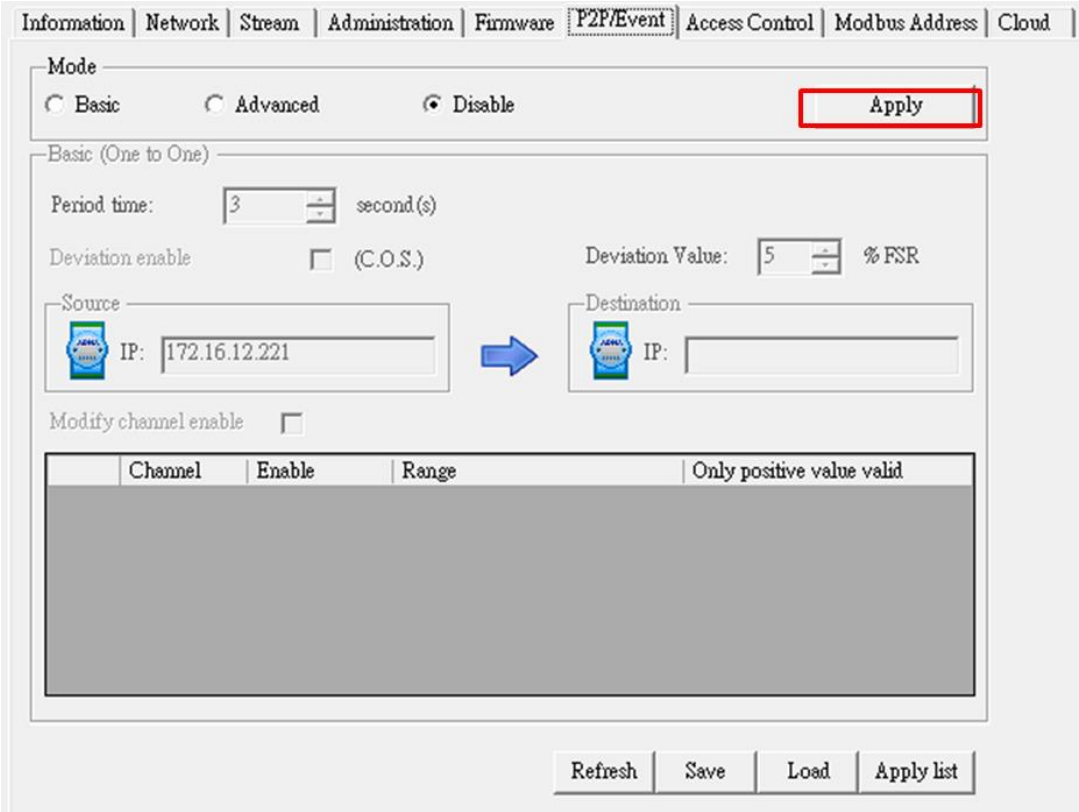
The File Import Panel

This is where you can import firmware to your ADAM-6000 module. Click Browse to select the three firmware files on your computer. Then, click Download to install the new firmware on the ADAM-6000 module.

The File Export Panel

This is panel is where you can export an ADAM module configuration file. Click Save As… and choose the destination file path. Then, click Upload to save the configuration file.

The Peer to Peer/Event Tab

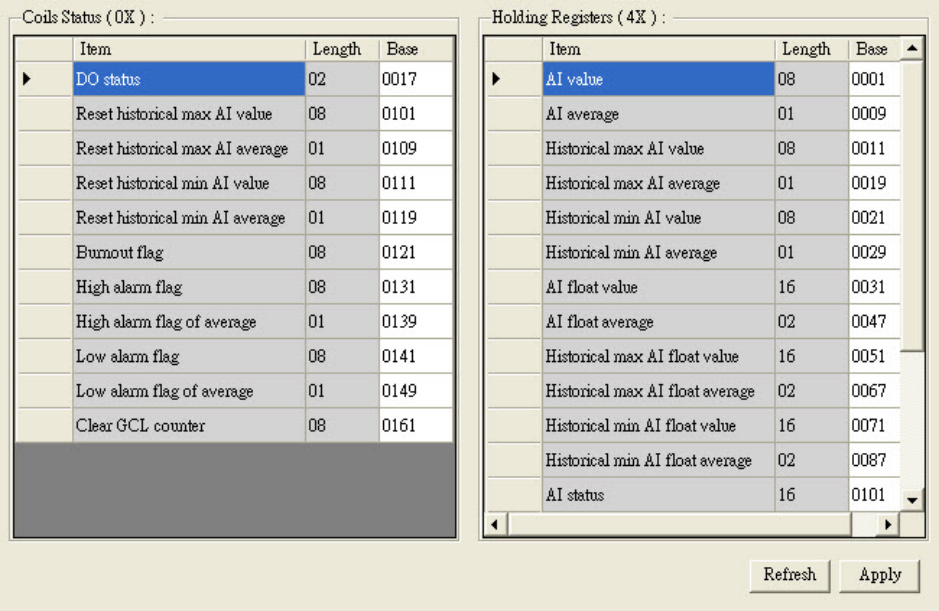


You can enable and configure the P2P (event) function in this tab. For more details about the P2P (event) function, see Section 6.7.

The Access Control Tab

This tab is for setting which computers/devices can control the selected ADAM-6000 module. First, select either the IP address or MAC address in the **Controlled By** panel and then click **Apply**. Then, in the **Security IP/MAC Setting** panel, you will need to select the **Enable/Disable** check box and then directly enter the IP or MAC address of the authorized computers/devices. Finally, click **Apply** to apply the changes to a single IP/MAC address or click **Apply all** to apply all changes. In the above image, only IP Addresses 172.18.3.52 and 172.18.3.116 are authorized to control the selected ADAM-6000 module. If no check boxes were selected, then any computer/device would be able to control the selected module.

The Modbus Address Tab



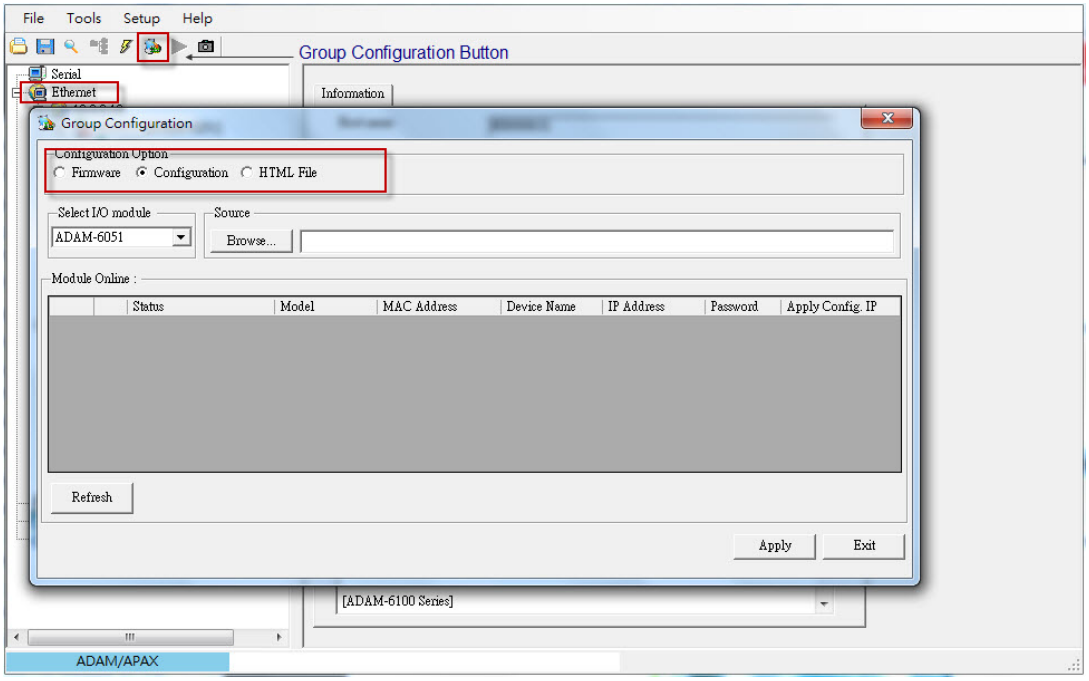
To provide greater flexibility and scalability in deploying ADAM modules, the limitations of Modbus address settings have been removed to make the modules as configurable as possible. Basically, there are two types of Modbus address section (0X and 4X) for configuring each function. For example, the above image shows the Modbus address settings for the ADAM-6017.

6.3.6 Group Configuration

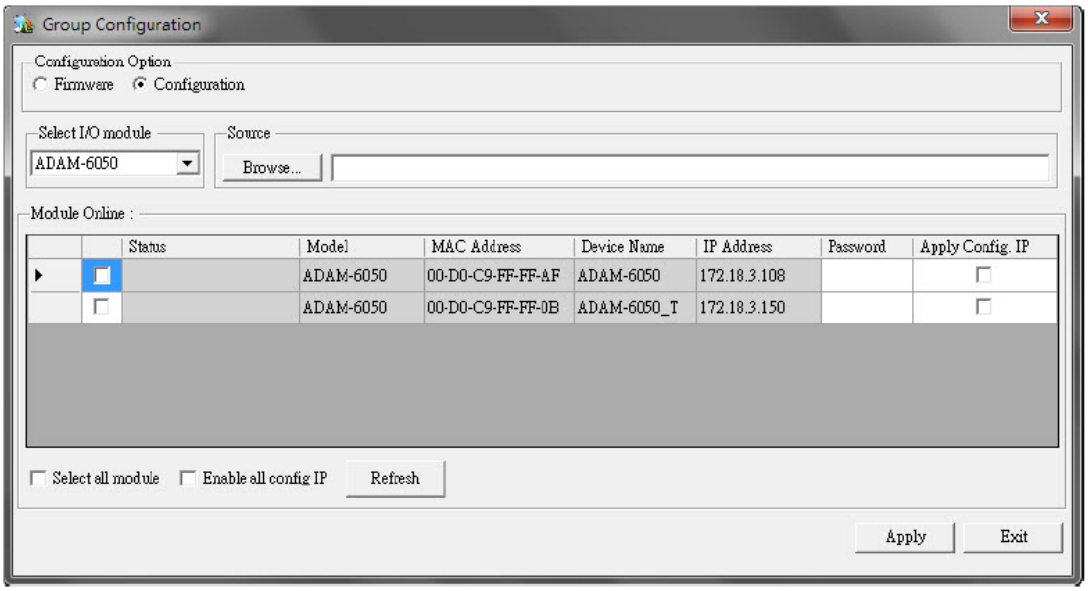
In certain applications, it is necessary to apply the same settings to multiple modules because they are performing the same tasks at different sites. Previously, users would have to configure each module individually prior to on-site deployment. After the modules are installed and the system is running, it will still require repetitive effort to carry out firmware updates.

To overcome this, ADAM-6000 modules are equipped with a group configuration function to reduce repetition and accelerate the configuration of multiple modules; this includes firmware upgrades, configuration files, and HTML 5 files, all of which can be updated in a single process. Follow these instructions to open the Group Configuration window:

1. Click on **Ethernet** on the Module Tree Display Area.
2. Click the **Group Configuration** icon on the toolbar or select **Group Configuration** under the **Tools** menu. This will open the following window:



3. Select **Firmware**, **Configuration**, or **HTML File** (depending on what you wish to update; in this example, **Configuration** is selected)
4. Select the type of I/O module you wish to apply the update to (this will select all modules of this type on the network)
5. Click **Browse** and you will be prompted to select the firmware/configuration file you wish to use
6. Choose which modules you wish to reconfigure/update for and enter the password; note that the default password is "00000000" (without the quotation marks)



7. Click **Apply** to apply the changes, and then you will see the operating progress on the Status Display Area.

Note! *Do not remove the power from your module when the group configuration function is processing. Otherwise, the module will probably crash.*

6.3.7 I/O Configuration

After you have completed the general configuration of the selected ADAM-6000 module (as described in the previous section), you will need to configure the I/O channels (e.g., channel range, calibration, and alarm settings). At the same time, you can see input channel value and set value of output channel in the **Status** panel. Refer to the Module Tree Display Area shown in Figure 6.5. When you click on the IP address of the ADAM-6000 module you wish to configure, you will see two items below the IP address: the module number (for all-channel configuration) and the module number followed by "GCL" (for GCL configuration). When you click on the plus and minus control beside the module number, you will be prompted to enter the password for the selected module. Once you have entered the correct password, a list of individual channels (for individual channel configuration) will appear below the module number.

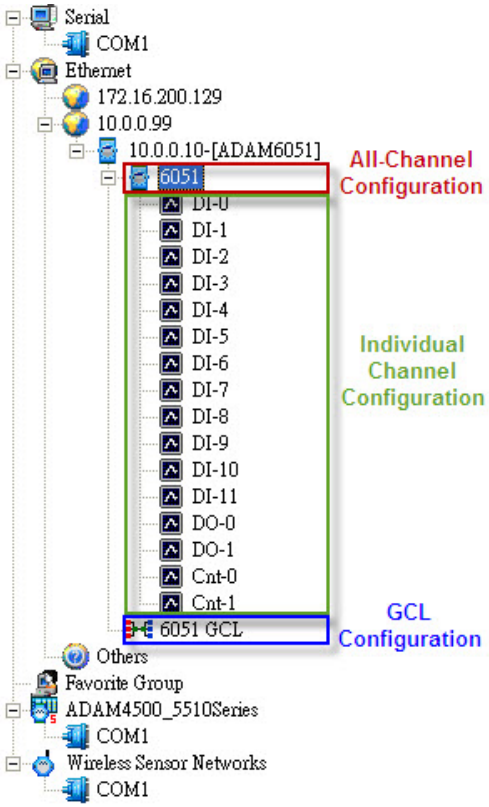


Figure 6.5 All-Channel, Individual Channel, and GCL Configuration Controls

When you click on the module number, the analog input value and configuration settings for all channels will be shown in the Status Display Area. When you click on one of the individual channel items, the values and configuration settings for only the specified channel will be shown in the Status Display Area. The following sections describe the all-channel configuration and individual channel configuration in more detail.

6.4 Analog Input Modules (ADAM-6015, ADAM-6017, and ADAM-6018, ADAM-6018+)

6.4.1 All-Channel Configuration

For these ADAM-6000 modules, when you click an all-channel configuration item in the Module Tree Display Area, the four main parts of interest in the Status Display Area will be the **Input Range**, **Integration Time**, **Calibration**, and **Channel Information** panels.

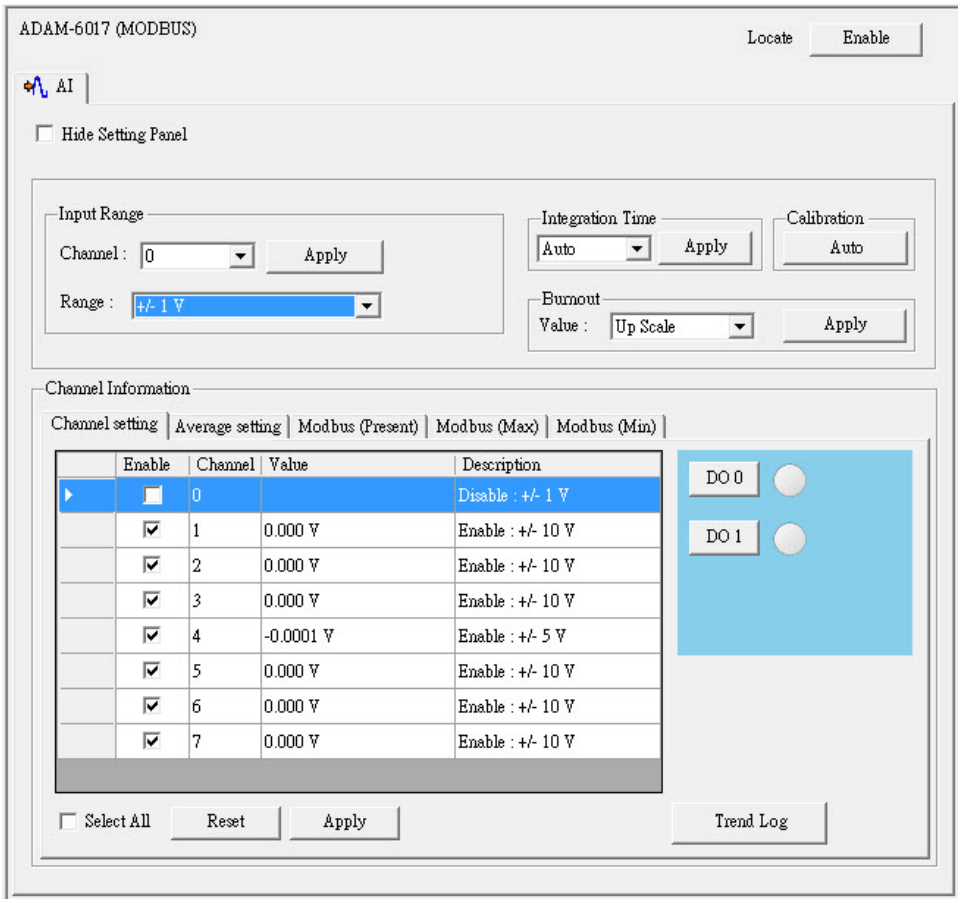


Figure 6.6 Channels Range Configuration Area

Input Range

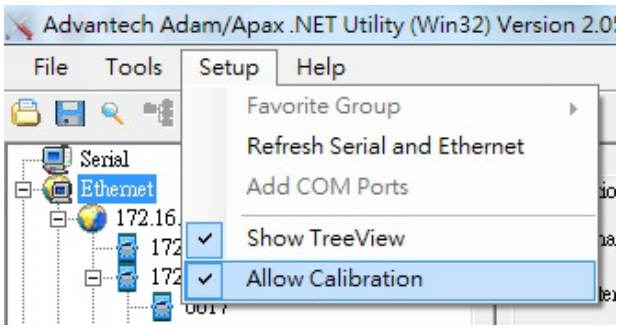
This panel allows you to set a different range for each channel. To do this, select the channel number from the **Channel** box, select the range from the **Range** box, and then click **Apply** to accept the changes.

Integration Time

To remove noise from the power supply, ADAM-6000 series analog input modules feature a built-in filter (50 and 60 Hz). For this setting, select the filter you wish to apply from the **Integration Time** box. Then, click **Apply** to accept the changes.

Calibration

Before you can adjust the calibration settings, you will need to enable the calibration function. To do this, click **Allow Calibration** under the **Setup** menu.



For the ADAM-6015, ADAM-6018, and ADAM-6024, follow these steps to perform calibration:

Zero Calibration

1. Click **Zero** in the **Calibration** panel
2. Connect a signal with the minimum value of the full scale range (e.g., 0 V) to the channel requiring calibration
3. Once you have completed the wiring, click **Apply** to start the calibration

Span Calibration

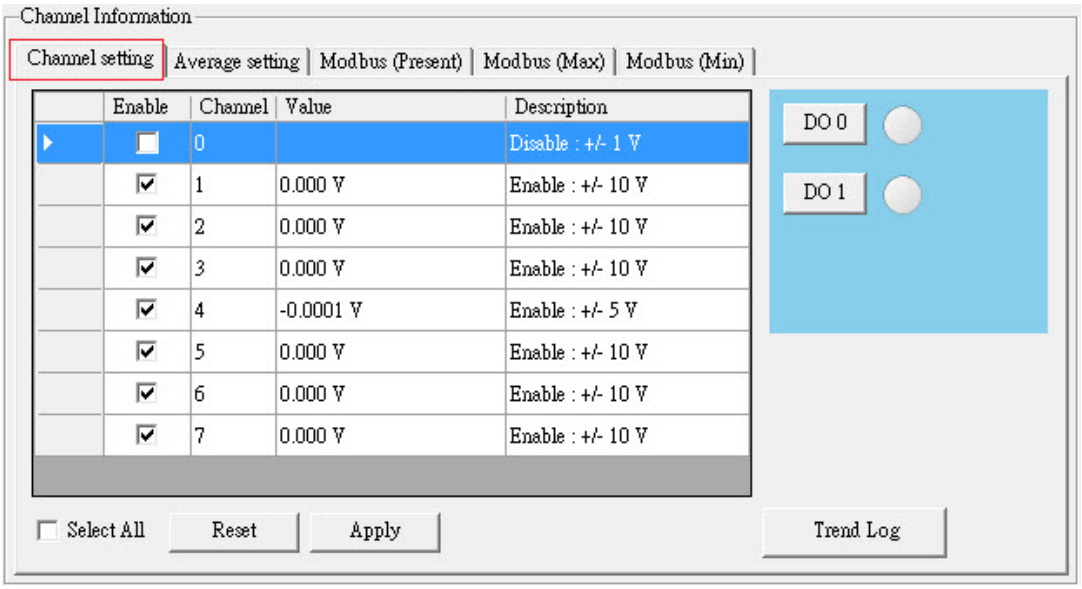
1. Click **Span** in the **Calibration** panel
2. Connect a signal with the maximum value of the full scale range (e.g., 10 V) to the channel requiring calibration.
3. Once you have completed the wiring, click **Apply** to start the calibration

For the ADAM-6017, you can perform auto calibration instead of manual calibration. To do this, click **Auto** in the **Calibration** panel.

Channel Information

This panel contains five tabs for viewing and configuring the analog input value of all channels: the **Channel setting** tab, the **Average setting** tab, the **Modbus (Present)** tab, the **Modbus (Max)** tab, and the **Modbus (Min)** tab.

The Channel Setting Tab



This tab displays the current values of the analog input channels. For the ADAM-6017 and ADAM-6018, the values of digital input channels are also displayed in this tab. Simply select the channels you want to monitor and click **Apply**.

You can also view historical trends for the selected channels by clicking **Trend Log**. As shown in Figure 6.7, you can select which channels you wish to log by checking them in the **Channel setting** panel on the right side of the screen and clicking **Apply**. Then, click **Start** and the data logging will commence, thus allowing you to view real-time historical trends. You can then click **Stop** and then **Save to file** to save the trend data to your computer.

Clicking **Show History** and **Clear History** will show/clear the historical trend data, whereas clicking **Save History** will allow you to save the data as a Csv file. To clear the chart, click **Clear Graph**. On the right-hand side of the screen you can enter the number of data points you wish to collect in the **BufferSize** box, and you can also set the data polling interval in the **PollingInterval** box.

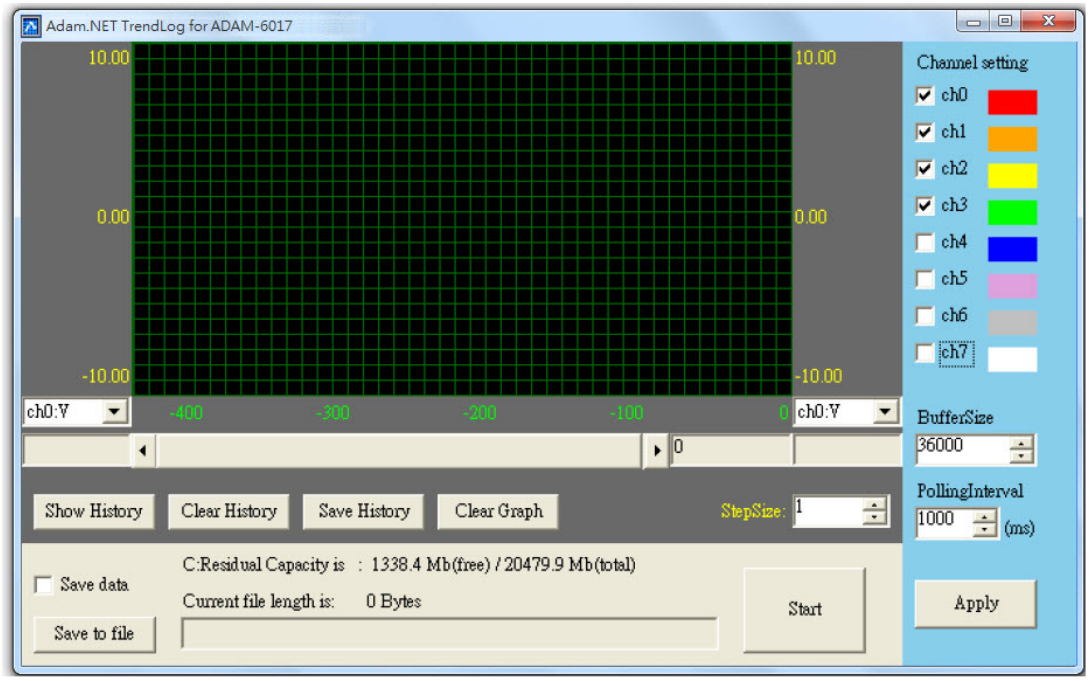
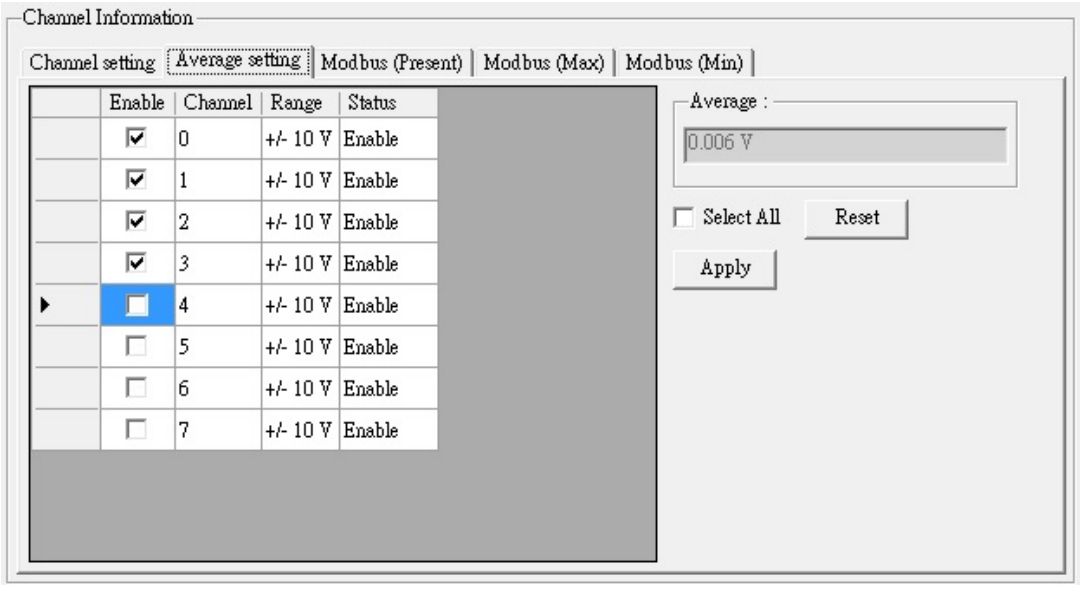


Figure 6.7 Analog Input Trend Log

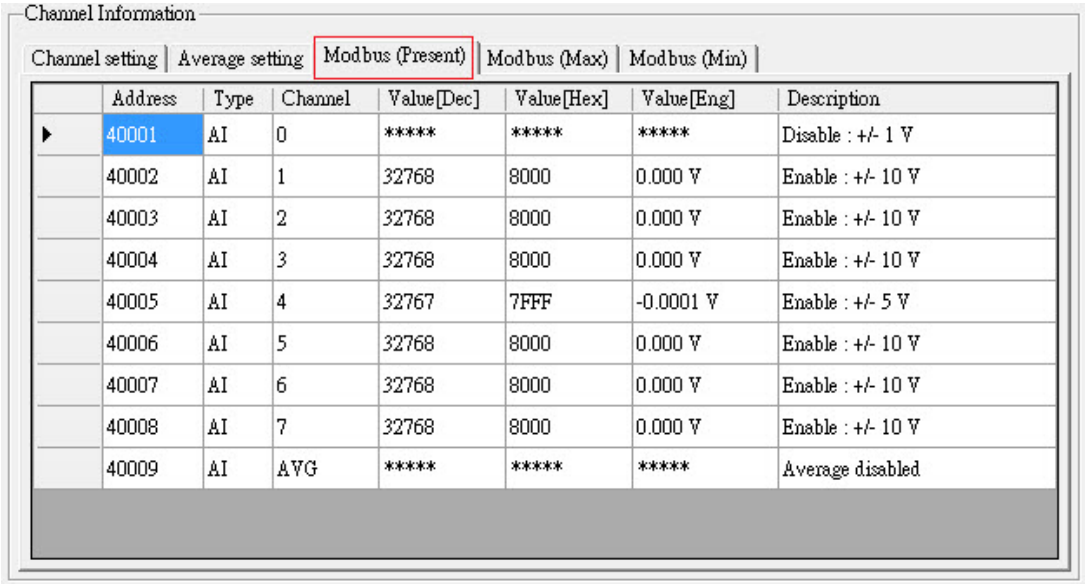
With the wire burnout detection function of the ADAM-6015 and ADAM-6018, if there is no sensor connected to an input channel, you will see the message "Burn out" appear in the **Information** box of the related channel.

The Average Setting Tab



The ADAM-6015, ADAM-6017, and ADAM-6018 feature an averaging function that is executed by the built-in processor. To use this function, simply check the channels of interest in the **Average setting** tab. In the example above, the averaging function has been enabled for four channels (Channels 0~3). Once enabled, the average value of the selected five channels will be displayed in the **Average** box.

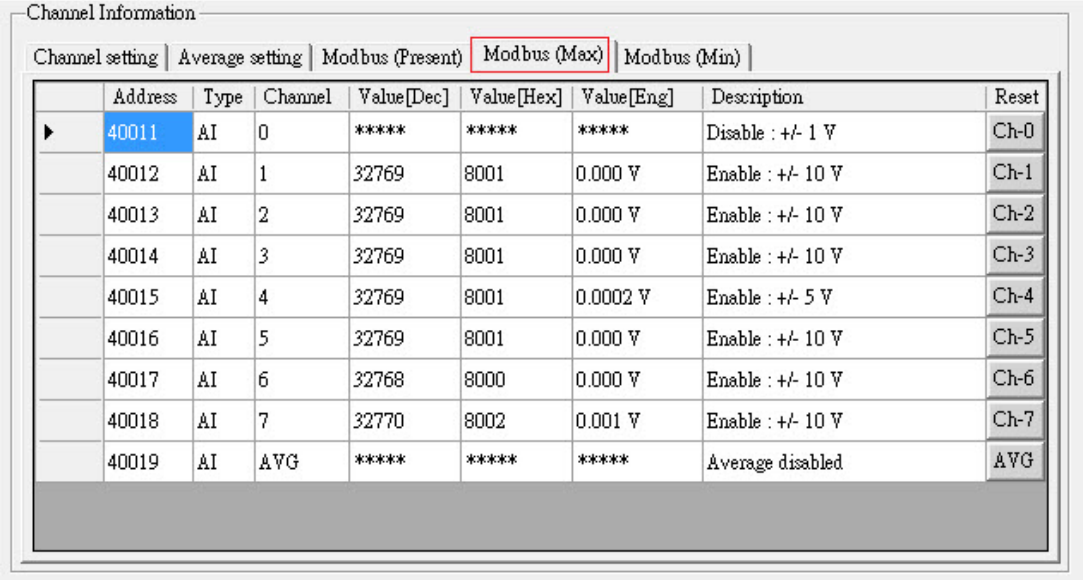
The Modbus (Present) Tab



Address	Type	Channel	Value[Dec]	Value[Hex]	Value[Eng]	Description
40001	AI	0	*****	*****	*****	Disable : +/- 1 V
40002	AI	1	32768	8000	0.000 V	Enable : +/- 10 V
40003	AI	2	32768	8000	0.000 V	Enable : +/- 10 V
40004	AI	3	32768	8000	0.000 V	Enable : +/- 10 V
40005	AI	4	32767	7FFF	-0.0001 V	Enable : +/- 5 V
40006	AI	5	32768	8000	0.000 V	Enable : +/- 10 V
40007	AI	6	32768	8000	0.000 V	Enable : +/- 10 V
40008	AI	7	32768	8000	0.000 V	Enable : +/- 10 V
40009	AI	AVG	*****	*****	*****	Average disabled

This tab shows the current analog input values decimal, hex, and engineering units for all related Modbus addresses.

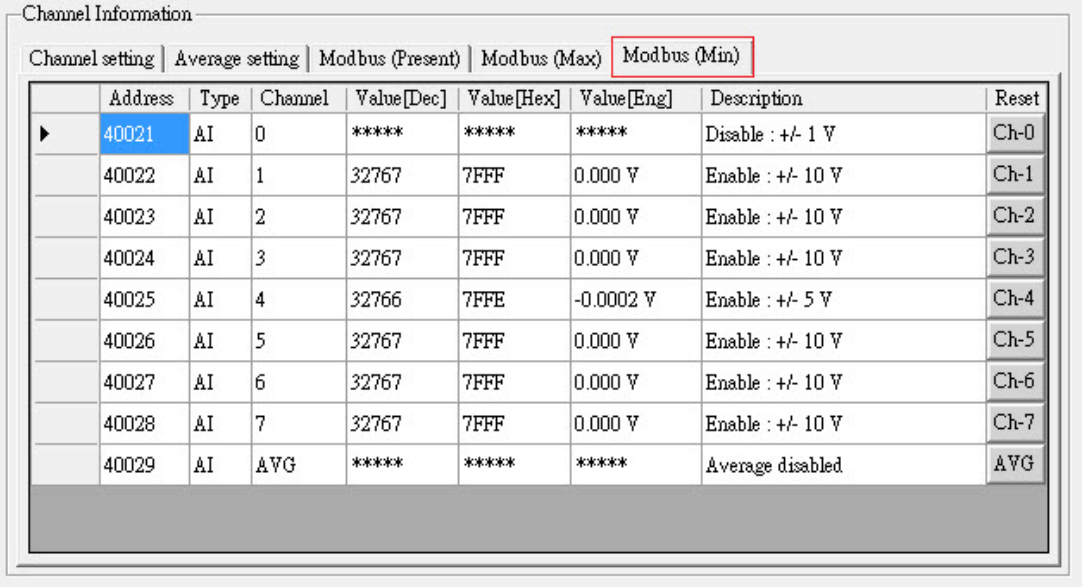
The Modbus (Max) Tab



Address	Type	Channel	Value[Dec]	Value[Hex]	Value[Eng]	Description	Reset
40011	AI	0	*****	*****	*****	Disable : +/- 1 V	Ch-0
40012	AI	1	32769	8001	0.000 V	Enable : +/- 10 V	Ch-1
40013	AI	2	32769	8001	0.000 V	Enable : +/- 10 V	Ch-2
40014	AI	3	32769	8001	0.000 V	Enable : +/- 10 V	Ch-3
40015	AI	4	32769	8001	0.0002 V	Enable : +/- 5 V	Ch-4
40016	AI	5	32769	8001	0.000 V	Enable : +/- 10 V	Ch-5
40017	AI	6	32768	8000	0.000 V	Enable : +/- 10 V	Ch-6
40018	AI	7	32770	8002	0.001 V	Enable : +/- 10 V	Ch-7
40019	AI	AVG	*****	*****	*****	Average disabled	AVG

The ADAM-6015, ADAM-6017, and ADAM-6018 feature a historical maximum value log. You can view the historical maximum analog input values in decimal, hex, and engineering unit for all related Modbus address. To re-initialize the log, click the corresponding channel buttons in the **Reset** column.

The Modbus (Min) Tab



	Address	Type	Channel	Value[Dec]	Value[Hex]	Value[Eng]	Description	Reset
▶	40021	AI	0	*****	*****	*****	Disable : +/- 1 V	Ch-0
	40022	AI	1	32767	7FFF	0.000 V	Enable : +/- 10 V	Ch-1
	40023	AI	2	32767	7FFF	0.000 V	Enable : +/- 10 V	Ch-2
	40024	AI	3	32767	7FFF	0.000 V	Enable : +/- 10 V	Ch-3
	40025	AI	4	32766	7FFE	-0.0002 V	Enable : +/- 5 V	Ch-4
	40026	AI	5	32767	7FFF	0.000 V	Enable : +/- 10 V	Ch-5
	40027	AI	6	32767	7FFF	0.000 V	Enable : +/- 10 V	Ch-6
	40028	AI	7	32767	7FFF	0.000 V	Enable : +/- 10 V	Ch-7
	40029	AI	AVG	*****	*****	*****	Average disabled	AVG

The ADAM-6015, ADAM-6017, and ADAM-6018 feature a historical minimum value log. You can view historical minimum analog input values in decimal, hex, and engineering units for all related Modbus addresses. To re-initialize the log, click the corresponding channel buttons in the **Reset** column.

6.4.2 Individual Channel Configuration

You can view the analog input value and configure the settings for each channel by clicking on one of the individual channel configuration items (note that the average you set in the **Average setting** tab will also be displayed here). The upper part of the Status Display Area will show the current analog input value and the defined range for the selected channel, as shown in Figure 6.8.

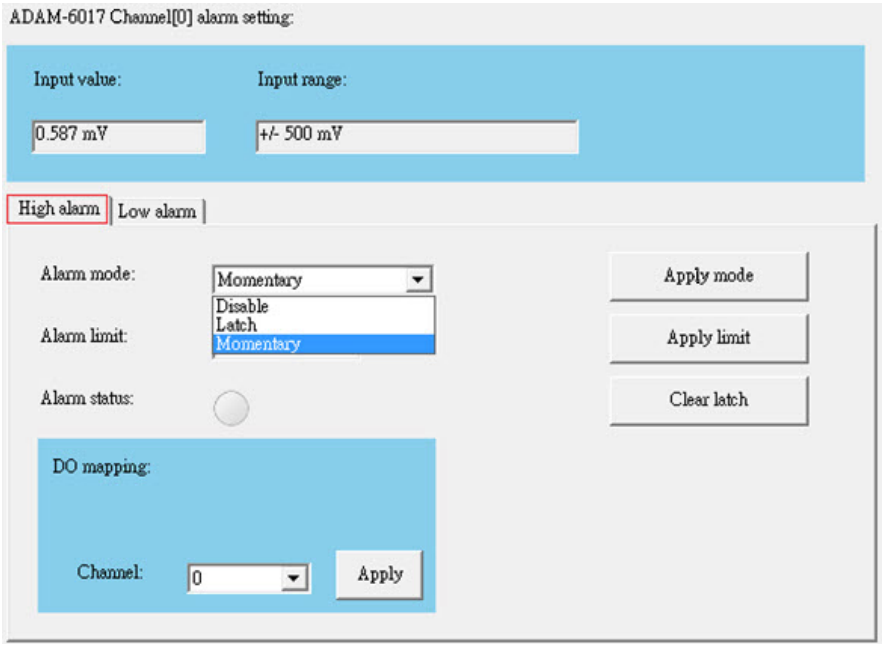


Figure 6.8 Analog Input Alarm Mode Configuration

For the ADAM-6015, ADAM-6017, and ADAM-6018, this screen allows you to configure the built-in alarm function. Two tabs for configuring the high and low alarms for the selected channel are at the lower part of the Status Display Area.

For both the high and low alarms, you can select one of three alarm modes from the **Alarm mode** box:

- **Disable:** The alarm is disabled, meaning that when the alarm condition occurs, nothing will happen.
- **Latch:** Once the alarm condition occurs, the alarm status will be set to logic high and the **Alarm status** LED will continuously be lit; these will remain until the alarm is cleared manually. For the ADAM-6017 and ADAM-6018, the output channel specified in the **DO mapping** panel will continuously generate logic high value. Click **Clear latch** to clear the alarm.
- **Momentary:** The alarm status will change dynamically depending on whether the alarm condition occurs. If the alarm condition occurs, the alarm status will be logic high; when the alarm condition disappears, the alarm status will change to logic low. Under this option, the **Alarm status** LED and the digital output channel will change according to the alarm condition.

After you choose the alarm mode, click **Apply mode** to apply the changes.

You can then define the high or low alarm value by entering the value in the **Alarm limit** box and then clicking **Apply limit**. When the analog input value is more than the high alarm value or less than the low alarm value, the alarm condition will be met and the alarm status will then be set to logic high. For the ADAM-6015, ADAM-6017, and ADAM-6018, the alarm status will be shown by the **Alarm status** LED. Finally, to map the alarm to a specific a digital output channel, select the channel of interest from the **Channel** box and then click **Apply**.

6.5 Universal I/O Modules (ADAM-6024)

6.5.1 All-Channel Configuration

The ADAM-6024 features analog I/O and digital I/O channels. Click the all-channel configuration item in the Module Tree Display Area and there will be two tabs in the Status Display Area: the **Input** tab and the **Output** tab. On the **Input** tab, there are four main areas of importance in the Status Display Area, similar to the pages for the ADAM-6015, ADAM-6017, and ADAM-6018. All the configurations in the **Input Range** and **Integration Time** panels are the same as those for these three modules. However, unlike these modules, the ADAM-6024 does not feature averaging, max., and min. functions. Thus, the **Channel Information** panel for the ADAM-6024 contains only two tabs: the **Channel setting** tab and the **Modbus (Present)** tab.

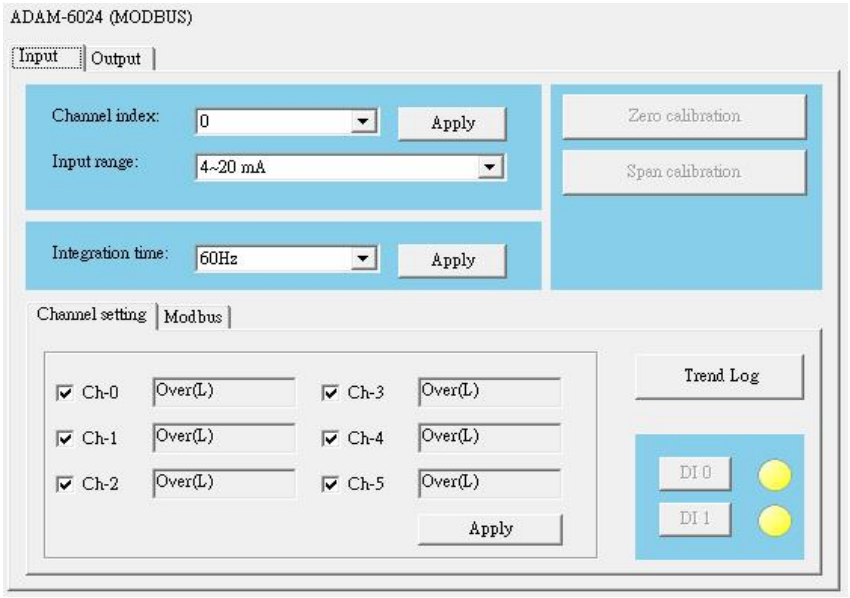


Figure 6.9 ADAM-6024 Channel Configuration

The Input Tab

This tab shows the current values of the analog input channels. Select the analog input channels you want to monitor by checking the box in the **Enable** column and then click **Apply**. If the analog input value is out of the input range, you will see "Over(L)" in the box for the corresponding channel. At the right side, you can see the current digital input value by **DI 0** and **DI 1** LED display. You also can view the graphical historical trend of analog input channel by clicking the **Trend Log** button. All the operations for trend logging are the same as those for the ADAM-6015, ADAM-6017, and ADAM-6018.

The Output Tab

This tab shows the current analog input values in decimal and hex format for all related Modbus address. From the **Output** tab, you can set the value of an analog or digital output channel as well as configure all related settings.

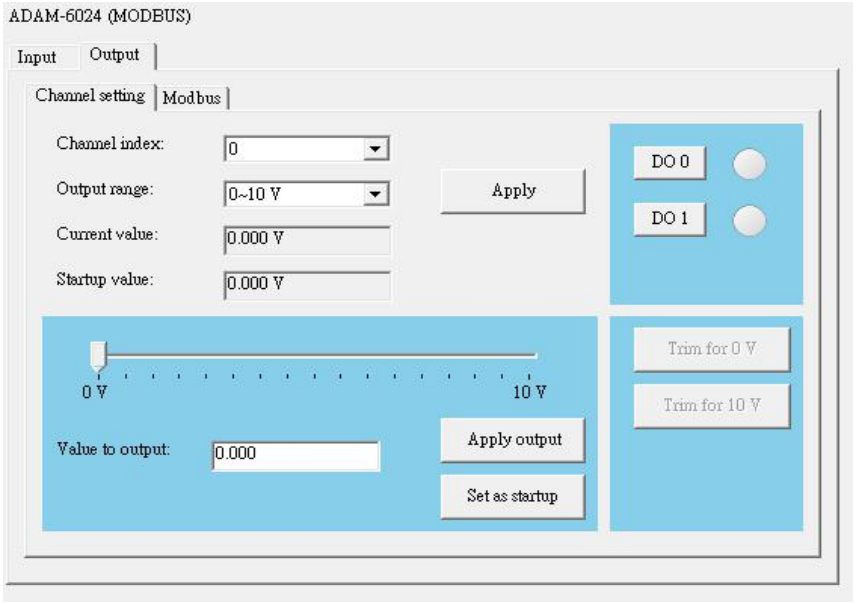


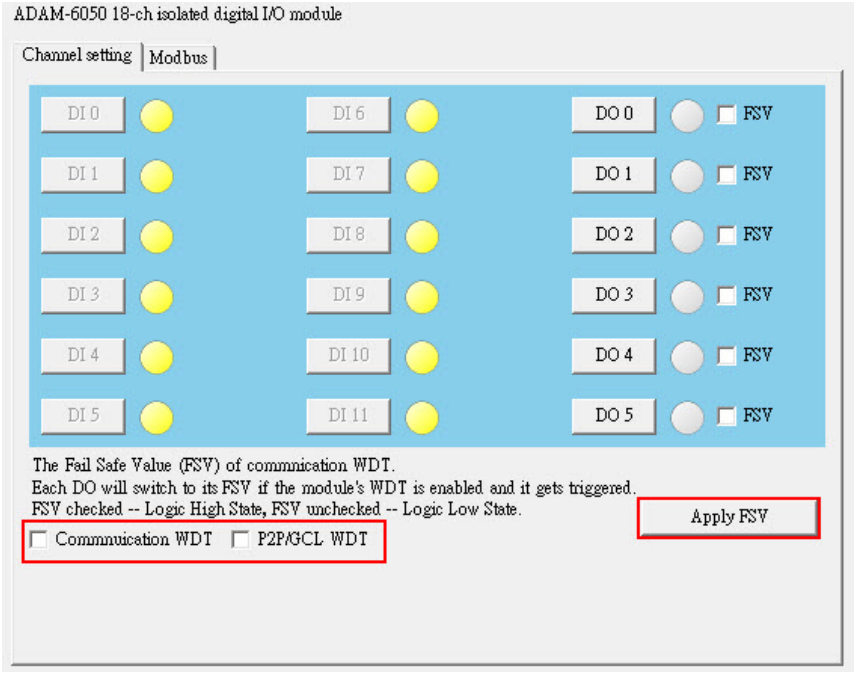
Figure 6.10 ADAM-6024 Output Tab

6.6 Universal Digital I/O Modules (ADAM-6050, ADAM-6051- ADAM-6052, ADAM-6060, ADAM-6066)

6.6.1 All-Channel Configuration

When you click the all-channel configuration item in the Module Tree Display Area, two tabs will be visible in the Status Display Area: the **Channel setting** tab and the **Modbus** tab. In the following text, the ADAM-6050 is used as an example.

The Channel Setting Tab



From this tab, you can view the status of all digital input channels from the LED beside each channel button. You can also control the statuses of all digital output channels by clicking the corresponding button.

Fail-Safe Value Configuration

When communication between the host PC and an ADAM-6000 digital module is broken, the digital output channels can generate a predefined value, which is referred to as a fail-safe value (FSV).

If the **FSV** box beside a channel is checked, it means that the module will set that output channel to logic high when a WDT timeout occurs. There are two applications for this. After all changes have been made, click **Apply FSV** for the changes to take effect.

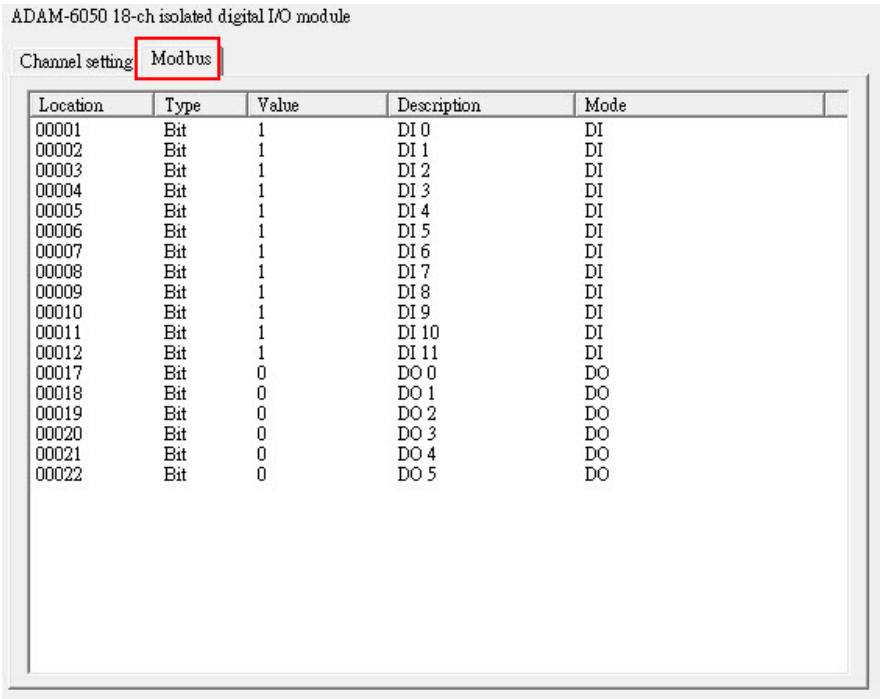
Communication WDT

When the module has not received any TCP network packet from the client in some time, the module will automatically set the FSV to output if the waiting time is greater than the host idle time.

P2P/GCL WDT

When the module has not received P2P/GCL network packets in some time, this means that the waiting time is greater than the idle time you have entered; the module will automatically send the FSV to the host PC if you have enabled this function.

The Modbus Tab



Location	Type	Value	Description	Mode
00001	Bit	1	DI 0	DI
00002	Bit	1	DI 1	DI
00003	Bit	1	DI 2	DI
00004	Bit	1	DI 3	DI
00005	Bit	1	DI 4	DI
00006	Bit	1	DI 5	DI
00007	Bit	1	DI 6	DI
00008	Bit	1	DI 7	DI
00009	Bit	1	DI 8	DI
00010	Bit	1	DI 9	DI
00011	Bit	1	DI 10	DI
00012	Bit	1	DI 11	DI
00017	Bit	0	DO 0	DO
00018	Bit	0	DO 1	DO
00019	Bit	0	DO 2	DO
00020	Bit	0	DO 3	DO
00021	Bit	0	DO 4	DO
00022	Bit	0	DO 5	DO

From this tab, you can view current digital I/O output values for all related Modbus addresses.

6.6.2 Individual Channel Configuration

To view the values and configure the settings of digital I/O channels, simply click the channel of interest in the list of individual channel configuration items.

Digital Input Mode

If you choose a digital input channel from the list of individual channel configuration items, the Status Display Area will appear as shown in Figure 6.11.

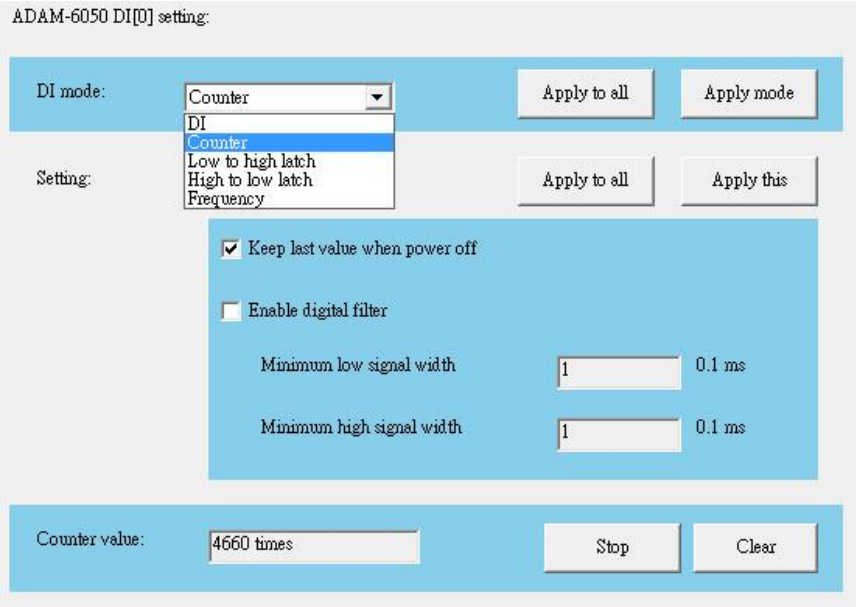
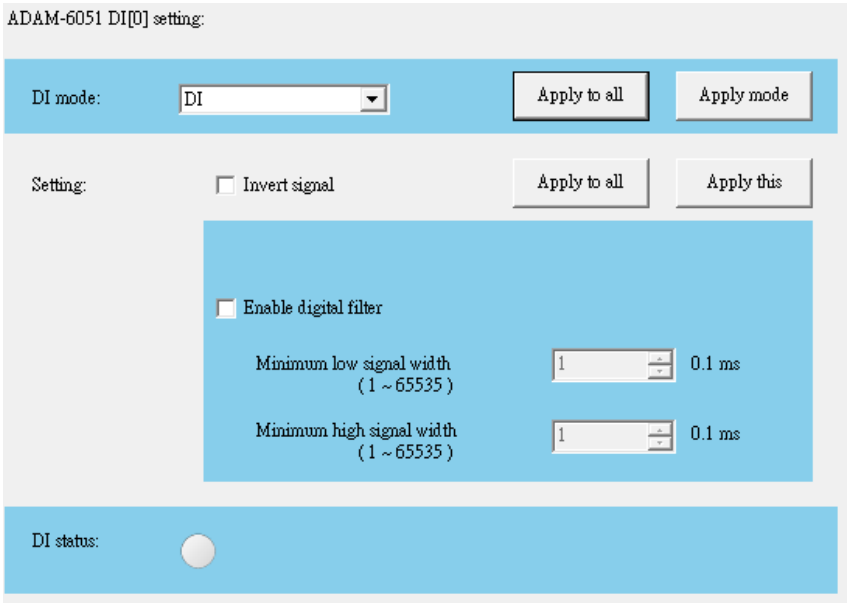


Figure 6.11 Digital Input Modes

You can choose different input modes for the selected digital input channel from the **DI mode** box (the option you select will depend on the hardware specification). After you have selected the mode, click **Apply mode** to save the changes. The five modes you can choose from are detailed in the following text.

DI Mode: DI

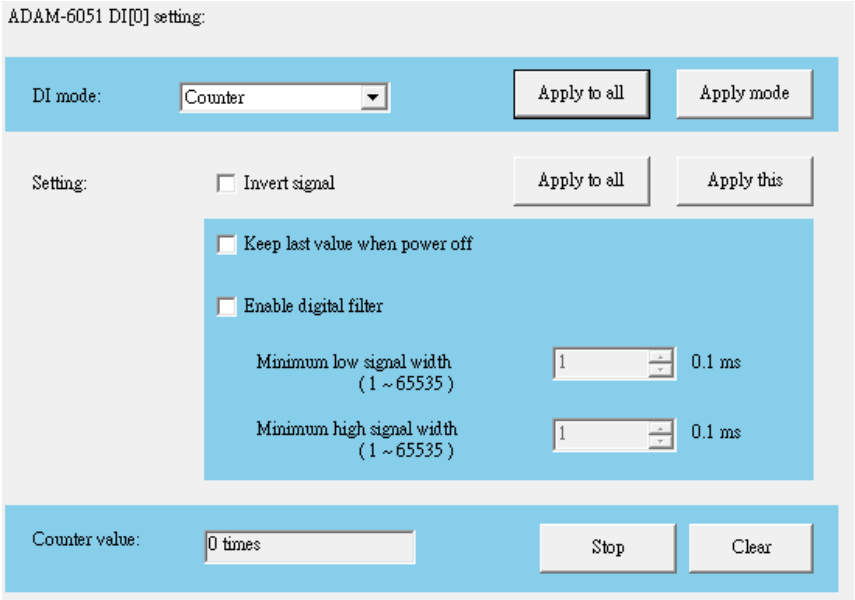


In this mode, you can see the digital input value by clicking the **DI status** LED.

Some digital modules support an invert digital input status function. When you enable this function, the module will automatically inverse the digital input value. For example, if the actual external signal value is logic low, then the **DI status** LED will be lit (normally, it is lit only when the signal is logic high). If your module supports this function, an **Invert signal** box will be visible in the **Setting** panel. Simply select/clear this box to enable/disable this function and then click **Apply to all** (for all channels) or **Apply** (for the selected channel) to complete the configuration.

All ADAM-6000 digital modules support a digital filter for removing high- and low-frequency noise. You can enable/disable the filter by selecting/clearing the **Enable digital filter** box. When the filter is enabled, you can define the minimum and maximum acceptable signal width from the **Minimum low signal width** and **Minimum high signal width** boxes (unit: ms). Remember to click **Apply to all** (for all channels) or **Apply** (for the selected channel) to complete the configuration.

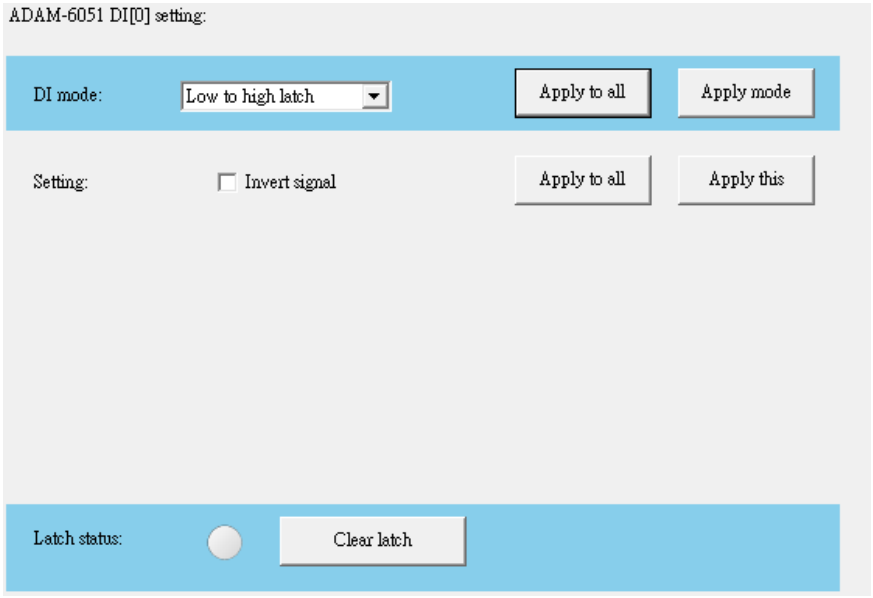
DI Mode: Counter



A counter counts the number of pulse numbers of a digital signal from the selected channel and then records that in a register. When **Counter** is selected from the **DI mode** box, the Status Display Area will appear similar to when **DI** is selected. Under this mode, the current count value of the selected channel will be displayed in the **Counter value** box. You can start or stop the counter by clicking **Start**/**Stop** next to the **Counter value** box, and you can also reset the counter (the value in the register will also be initialized to zero) by clicking **Clear**.

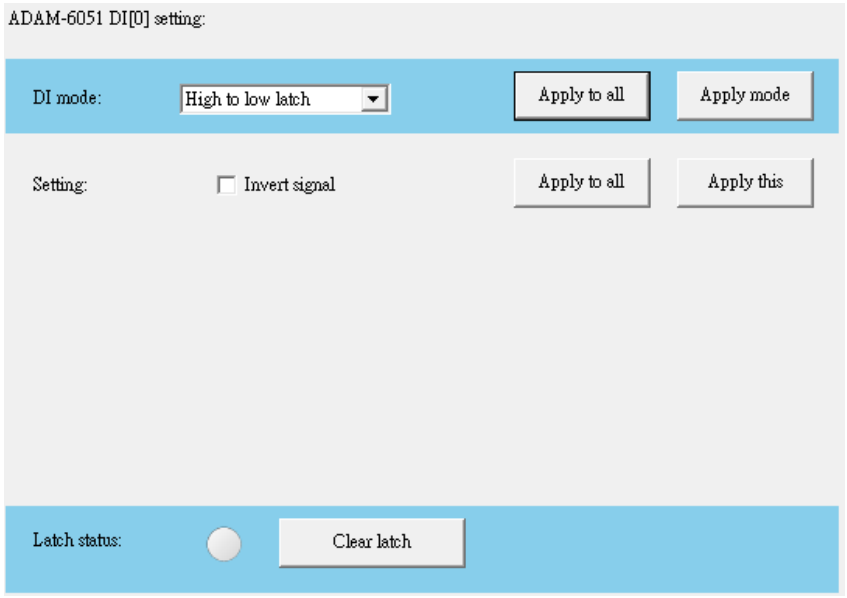
Similar to when **DI** is selected from the **DI mode** box, you can enable/disable the invert digital input status function and digital filter in the **Setting** panel. One additional setting, however, is that you can define whether the counter should keep the last value when the module is powered off; when the module is powered on again, the counter will continue counting from that stored value. Otherwise, the counter will be reset to zero when the module is powered on. You can enable/disable this function by selecting/clearing the **Keep last value when power off** box and then clicking **Apply to all** (for all channels) or **Apply this** (for the selected channel) to complete the configuration.

DI Mode: Low-to-High Latch



Low-to-high latch mode means that once the digital input channel detects a logic level change from low to high, the logic status will remain logic high until you clear latch manually, which will return the logic status to logic low. The logic status can be seen by the **Latch status** LED. The latch can be cleared by clicking **Clear latch**. This mode also supports the invert digital input status function, which can be enabled/disabled by checking/clearing the **Invert signal** box and then clicking **Apply to all** (for all channels) or **Apply this** (for the selected channel) to complete the configuration.

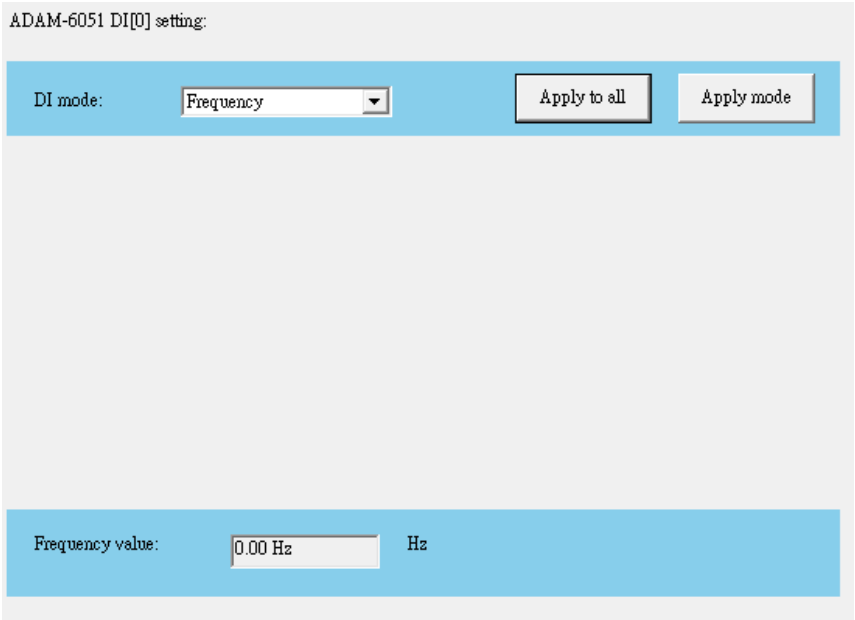
DI Mode: High-to-Low Latch



High-to-low latch mode means that once the digital input channel detects a logic level change from high to low, the logic status will remain as logic high until you clear latch manually, which will return the logic status to logic low. The logic status can be seen by the **Latch status** LED. The latch can be cleared by clicking **Clear latch**. This mode also supports the invert digital input status function, which can be enabled/dis-

abled by selecting/clearing the **Invert signal** box and then clicking **Apply to all** (for all channels) or **Apply this** (for the selected channel) to complete the configuration.

DI Mode: Frequency



When **Frequency** is selected from the **DI mode** box, the module will calculate the frequency of the digital input signal for the selected channel. This value will be displayed in the **Frequency value** box.

Digital Output Mode

If you choose a digital output channel from the list of individual channel configuration items, the Status Display Area will appear as shown in Figure 6.12.

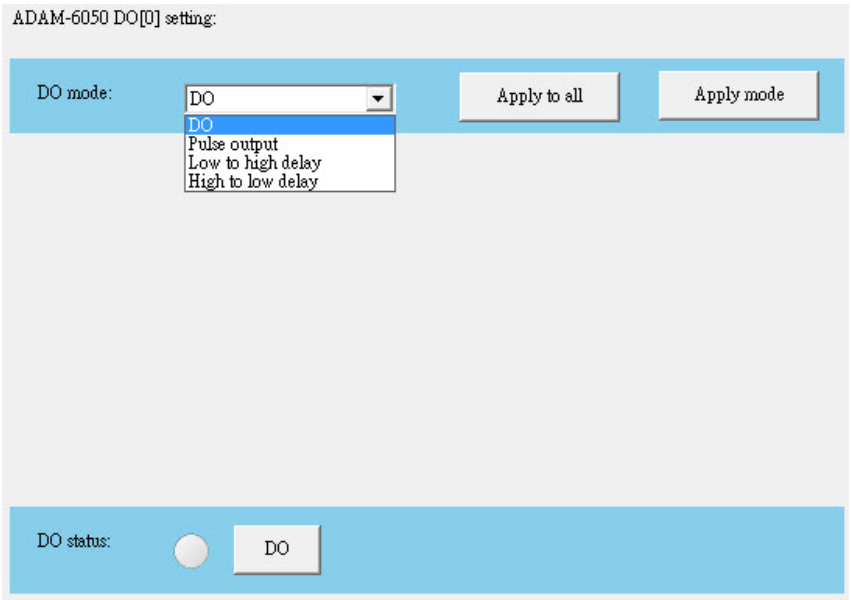
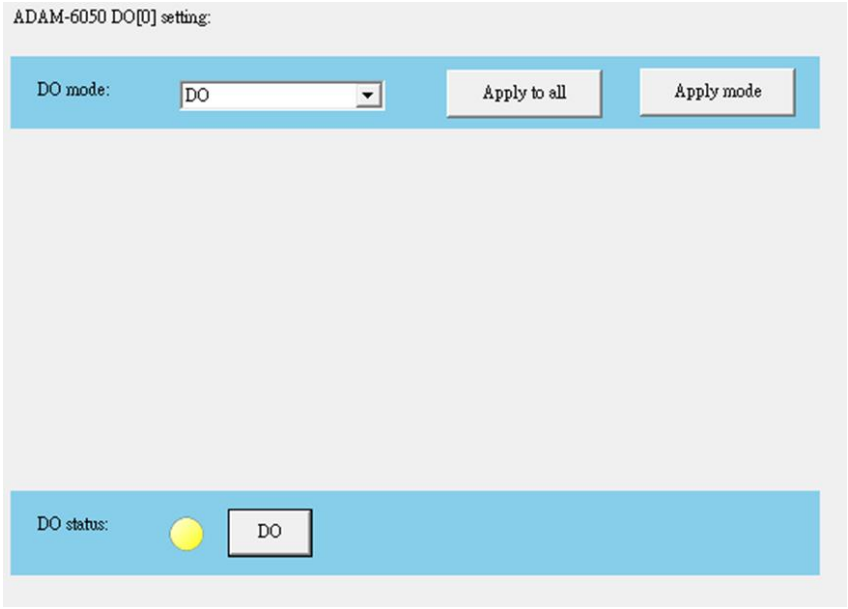


Figure 6.12 Digital Output Modes

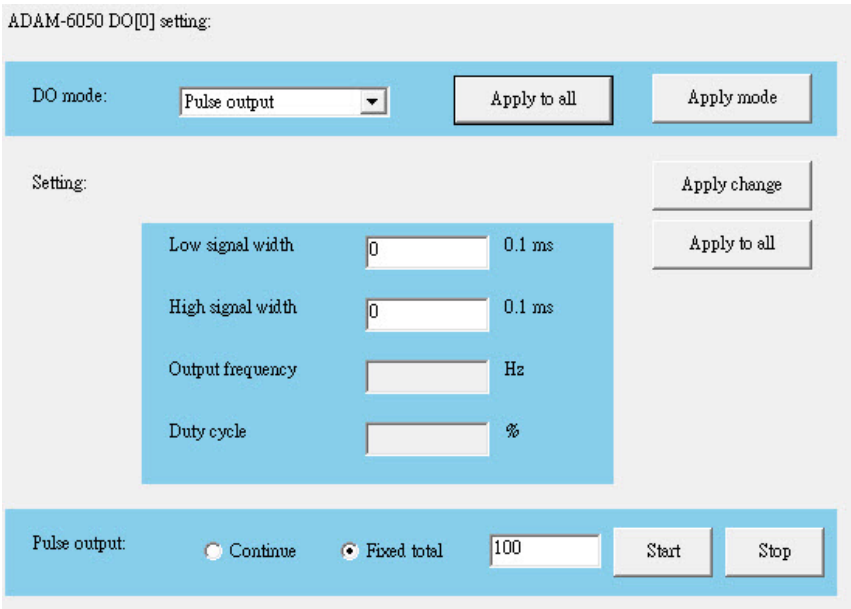
You can choose different output modes for the selected digital output channel from the **DO mode** box (the option you select will depend on the hardware specifications). After you have selected the mode, click **Apply mode** to save the changes. There four modes you can choose from, as detailed in the following text.

DO Mode: DO



This mode allows you to control the digital output value of the selected channel, which can be adjusted by clicking **DO**. The current digital output value will be shown by the **DO status** LED.

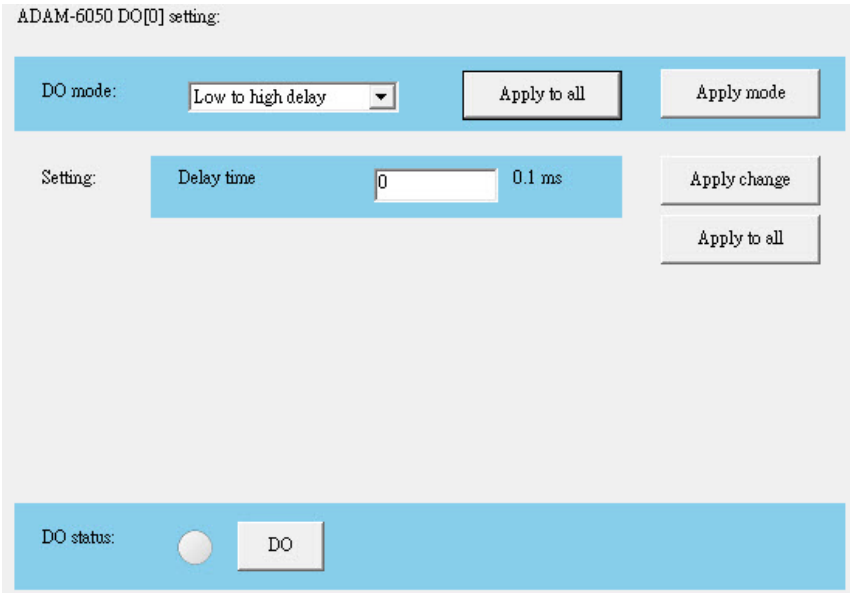
DO Mode: Pulse Output



When **Pulse output** is selected from the **DO mode** box, the selected digital output channel will generate a continuous pulse train or a finite number of pulses. You can define the pulse width in the **Low signal width** and **High signal width** boxes in the

Setting panel (unit: 0.1 ms). The frequency and duty cycle of the pulse output signal will be calculated automatically and displayed in the **Output frequency** and **Duty cycle** boxes. After you have completed the settings, click **Apply mode** (for individual channels) or **Apply to all** (for all channels). You can then choose to generate a continuous pulse train or finite number of pulses by selecting **Continue** (for a pulse train) or **Fixed total** (for a finite number of pulses). When you selected **Fixed total**, you will need to enter how many pulses you want to generate. After the pulse output mode has been selected, click **Start**/**Stop** to generate/stop the pulse output.

DO Mode: Low-to-High Delay



When you choose **Low to high delay** from the **DO mode** box, it is the same as selecting **DO** except that there will be a delay before the output value changes from logic low to logic high, as depicted in Figure 6.13.

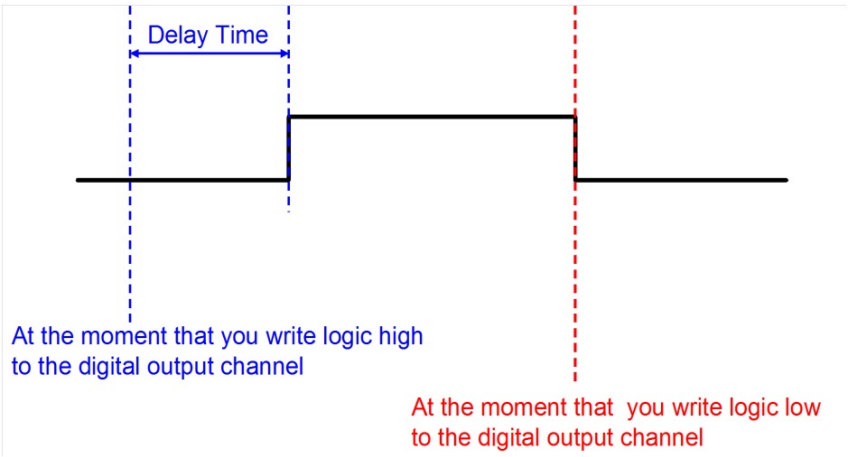


Figure 6.13 Graph Explaining Low to High Delay Output Mode

To define the delay time, simply enter the value in the **Delay time** box and then click **Apply** to complete the configuration. You can then control the digital output value by clicking **DO** and you can determine its current value from the **DO status** LED.

DO Mode: High-to-Low Delay

When you choose **High to low delay** from the **DOmode** box, it is the same as selecting **DO** except that there will be a delay before the output value changes from logic high to logic low, as depicted in Figure 6.14.

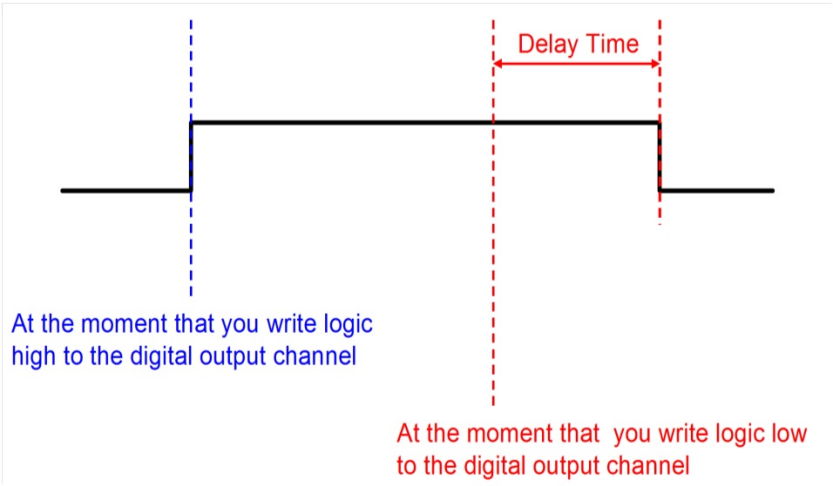


Figure 6.14 Graph Explaining Low to High Delay Output Mode

To define the delay time, simply enter the value in the **Delay time** box and then click **Apply** to complete the configuration. You can then control the digital output value by clicking **DO** and you can determine its current value from the **DO status** LED.

6.7 Introduction to P2P Functions

When you want to send a signal from one module to another module, P2P is the ideal solution. With the P2P function enabled, ADAM-6000 modules can actively update their input values to other devices such as PCs or other ADAM-6000 modules. A typical application is using a pair of ADAM-6000 modules, in which the value of an input channel on one module will be automatically updated to output channel on another module. The data will be transferred automatically as long as the connection between the two ADAM-6000 modules is already established, and no controller is needed to handle the communication.

Note! *1. Please use an Ethernet switch between a pair of P2P modules (do not use an Ethernet hub) in order to prevent data packet collisions.*

2. ADAM-6000 modules support two functions: P2P (Event) and GCL (see Chapter 8). You cannot enable both of these two features at the same time. Thus, if GCL is enabled and want to use P2P, you will need to disable GCL first (see Section 8.2 for instructions on how to disable GCL).

3. To utilize the P2P function, you will need to upgrade the firmware version of your ADAM-6000 module to 3.x or later.

6.7.1 P2P Communication Modes

All ADAM-6000 modules feature two types of P2P function: 1) basic mode and 2) advanced mode.

Basic Mode

For basic mode, there will be only one target device (Module B) receiving data from the source module (Module A). Usually, Module B is another ADAM-6000 module. The input channels of Module A will be mapped to the output channels of Module B, so that the values of all Module A inputs channels are automatically updated to the Module B outputs. You can also define a mask to disconnect a relationship between some inputs and outputs.

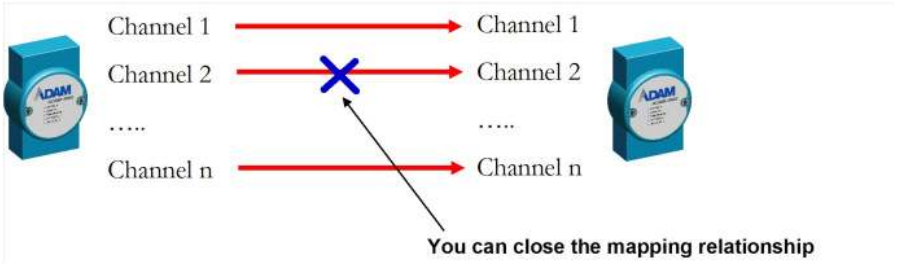


Figure 6.15 Basic Mode for P2P

Advanced Mode

For advanced mode, there will be multiple target devices (Module B, Module C, etc.) receiving data from the source module (Module A). You can define different target devices by assigning different IP address to each channel of Module A. For example, you can map Input Channel 1 of Module A to Output Channel 3 of Module B, while Input Channel 2 of Module A is mapped to Output Channel 4 of Module C. Refer to Figure 6.16.

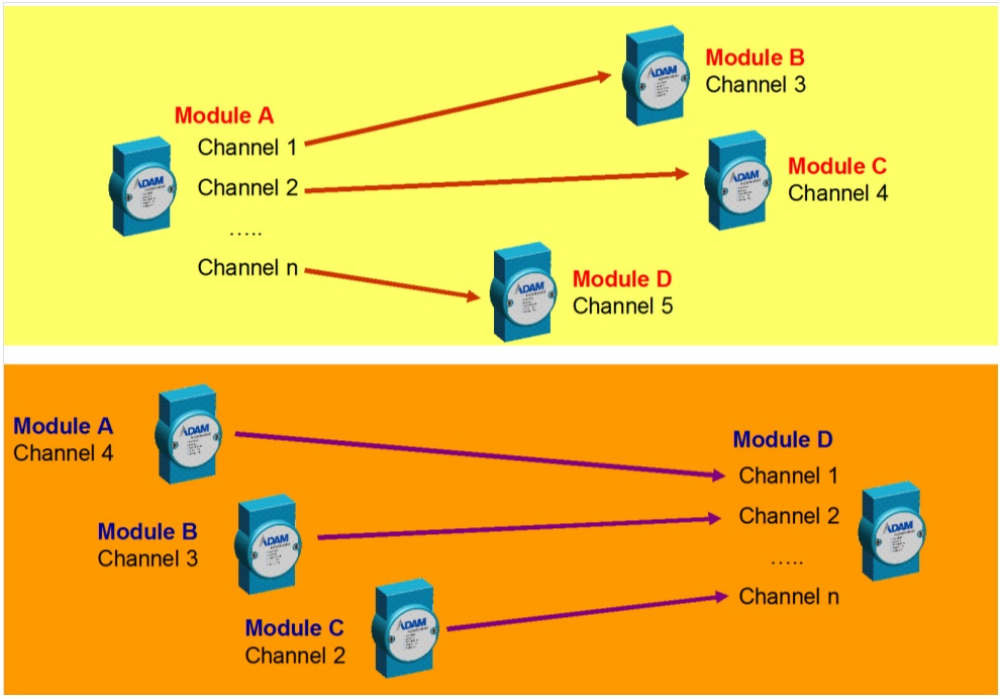


Figure 6.16 Advanced mode for P2P

6.7.2 P2P Communication Methods

As for when the data will be updated from a source module to the target devices, there are two options to choose from: 1) period time and 2) period time + change-of-status (COS).

Period Time

With this function, the value of the input channel will be updated to the target devices at the defined period.

Period Time + COS

This option still causes the value of the input channel to be updated to the target devices at the defined period, but when a COS occurs (i.e., a change in the analog input value greater than a specified deviation or a digital input status change), the value of the input channel will immediately update to the target devices.

6.7.3 P2P Event Triggers

In many applications, data will only be sent to a host computer when a specific event occurs, such as when a digital or analog signal changes. In this type of application, the P2P function is ideal. The target P2P device can be a computer, for which you would simply need to enter its IP address and select basic mode as the communication mode and period time + COS as the communication method.

There should be one program running on the host computer to receive the data, and we provide an example C program (VC++ 6.0) on the companion CD. Although ADAM-6000 modules will send data to the host computer periodically (for the sake of communication security), you can still distinguish whether messages have been sent via the period time or COS function. The message contains information on which channels have changed. Thus, if the message indicates no change in all channels, then no event has occurred.

Note! *There is invariably some level of uncertainty in network communication. Sometimes, there may be packet loss when an event occurs. This is why we provide the period time + COS function (no COS function alone). When an event occurs, even if a packet is lost, the data will be sent again at the next period. This improves system reliability.*

6.8 How to Configure P2P Functions

Select the IP address of an ADAM-6000 module from the Module Tree Display Area and click the **Peer to Peer/Event** tab. The screen will appear as shown in Figure 6.17.

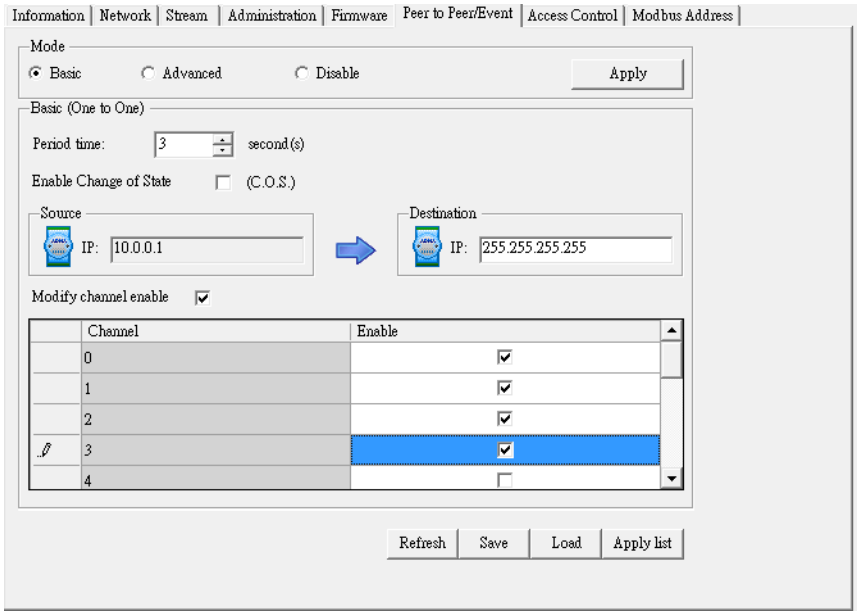


Figure 6.17 Peer to Peer/Event Tab

By default, the P2P function is disabled. You can enable it by selecting **Basic** or **Advanced** in the **Mode** panel and then clicking **Apply**. ADAM-6000 modules support both P2P and GCL functionality (see Chapter 8 for information on GCL); however, only one of them can be enabled at one time. If GCL is already enabled and you choose to enable P2P, an alert will appear asking you to first disable GCL (see Section 8.2 for how to disable GCL). After GCL has been disabled, you can then select **Basic** or **Advanced** to enable P2P.

6.8.1 Basic Mode Configuration

In basic mode, the Status Display Area will appear as shown in Figure 6.18. You can define the target device by entering its IP address in the **Destination** box in the **Basic (One to One)** panel.

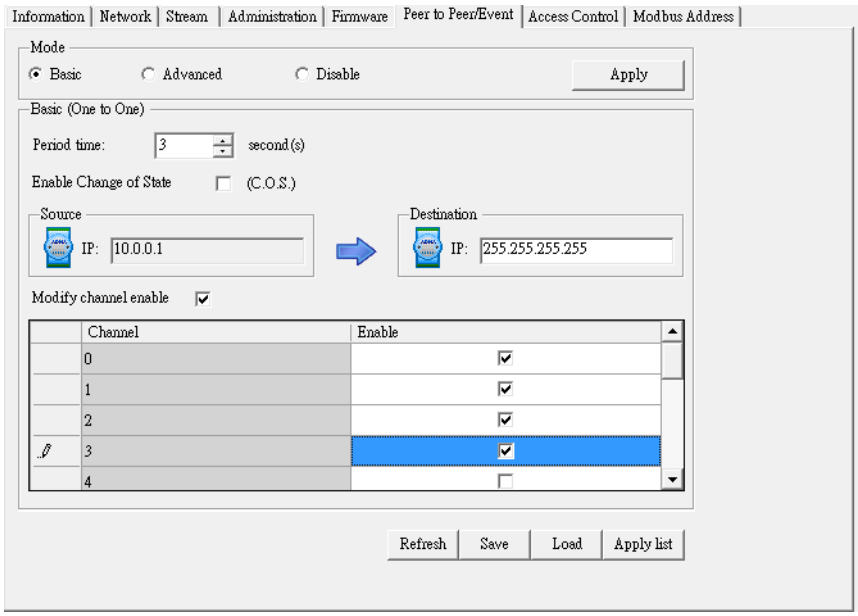


Figure 6.18 P2P Basic Mode Configuration

Note that when you select basic mode, the default communication method is period time; to select **period time + COS**, you will need to select the **Deviation Enable (C.O.S)** box (for analog modules; not shown in this example) or the **Enable Change of State** box (for digital modules). If you do not select this box, the communication method will be period time.

The period to transfer data from the source module to the destination module can be set in the **Period time** box in the **Basic (One to One)** panel. You can define the deviation for analog input by the **Deviation Rate** numeric control (the value is a percentage and represents the change value divided by the total range).

By default, all input channels of the source module will all be mapped to all output channels of the destination module. However, you can manually define which channels are mapped by clicking the **Modify channel enable** box. This will allow you to choose which input channels to map to the corresponding output channels by selecting the channel in the **Enable** column and then clicking **Apply list**. In Figure 6.18, the values of Input Channels 0~3 of the source module will update to Output Channels 0~3 of the destination module. You can save the current mapping relation into a configuration file by clicking **Save**. You can also load a mapping configuration file by clicking **Load**. Click **Refresh** will show the current mapping configuration on the source module in the table.

6.8.2 Advanced Mode Configuration

In advanced mode, the Status Display Area will appear as shown in Figure 6.19. The mapping relationship is configured using controls in the **Source** and **Destination** panels.

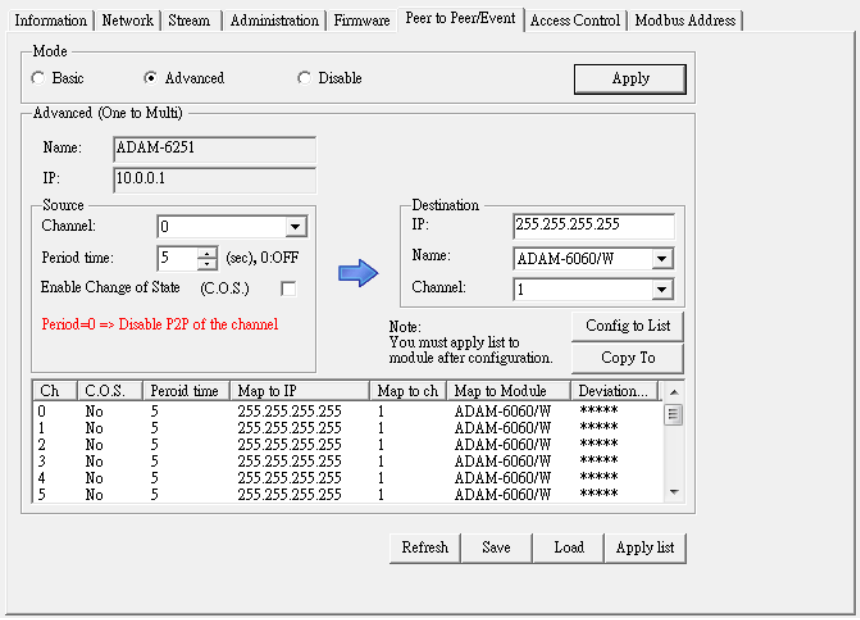


Figure 6.19 P2P Advanced Mode Configuration

Follow these steps to define the mapping relationship:

1. Select the input channel from the **Channel** box in the **Source** panel
2. Use **Period time**, the **Deviation enable (C.O.S)** box (for analog modules) or **Change of State (C.O.S)** box (for digital modules), and **Deviation Rate** in the **Source** panel to define when to transfer the data for that channel
3. Enter the IP address of the target module in the **IP** box in the **Destination** panel
4. Select the name of the target module from the **Name** box
5. From the **Channel** box, select the output channel on the target module that will receive the data
6. Click **Config to list**

Once you have completed these steps, the configuration for that channel will be displayed in the mapping table at the bottom of the **Advanced (One to Multi)** panel.

You will need to repeat Steps 1~4 for each input channel you wish to map. Once you have configured all the input channels, click **Apply list** to download the mapping configuration to the target module. You can save all configurations in the mapping table to a file by clicking **Save**. You can also load a previous configuration file by clicking **Load**. Clicking **Refresh** will show the current configuration of the source module in the mapping table.

Note! *It is suggested that you to download all channels mapping configuration together at one time instead of downloading one-channel setting many times. The reason is that this can reduce the number of times the flash memory on target module is used, thus helping to extend the flash memory life.*

To save time, you can copy one channel configuration to other channels and then make final adjustments to each channel. To do this, click **Copy To**, which will open the window shown in Figure 6.20.

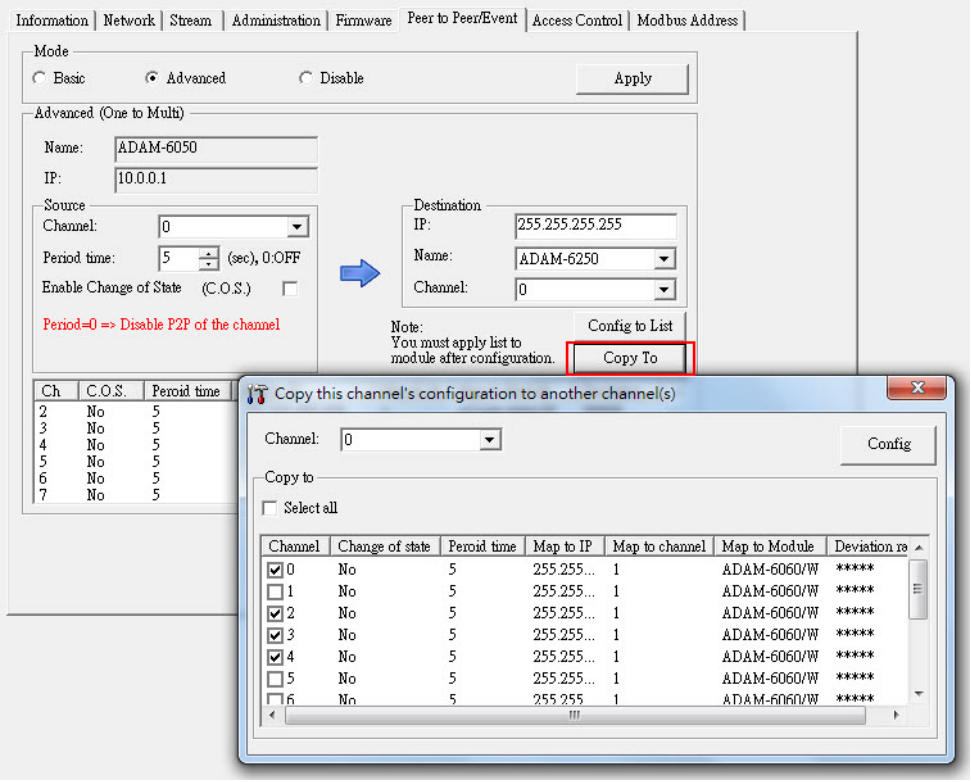


Figure 6.20 Copy One Setting to Other Channels

Here, you will need to choose the channel you wish to copy from the **Channel** box and then select the channels you want to copy the settings to by selecting them from the **Channel** column in the **Copy to** panel and then clicking **Config** (check **Select all** to copy to all channels). In this example, the settings of Channel 0 will be copied to Channels 0, 2, 3, and 4.

When you return to the **Peer to Peer/Event** tab, you will find that the settings of the channels you selected now appear in the mapping table. You can the select the individual channels you need to modify and change the parameters.

P2P Data Transfer Performance

Wired LAN Module

Condition: transfer data from one channel of an ADAM-6050 module to one channel of another ADAM-6050 module, via one Ethernet switch.

Data Transfer Time: <1.2 ms

6.9 ADAM-6000 Web Server

ADAM-6000 modules all feature a built-in web server that can be accessed using a standard web browser. The web service allows programmers to create powerful customized web pages by using web programming languages. Remote computers or devices can thus be used to monitor and control the I/O status of ADAM-6000 modules remotely via a web browser. ADAM-6000 modules come with a default built-in web page that you can modify using HTML5.

The default HTML settings on ADAM-6000 modules do not support HTML5; for HTML5 support, you will need to download a new file from the Advantech website.

To access the web server, simply type the IP address into your browser to connect to your ADAM-6000 module. You will be prompted to enter a user name (default: root) and password (default: 00000000). After you have entered the correct password, you can start monitoring/controlling the I/O channels on your ADAM-6000 module.

6.9.1 HTML 5

HTML 5 Introduction

HTML is the most widely used language in web content design. The latest version, HTML 5, enhances the syntax structure and incorporates multiple web technologies (e.g., CSS and JavaScript), thus allowing for the implementation of additional web services, APIs, and interactive applications in mobile communications.

Remote Monitoring and Control via the ADAM-6000 Web Server

This new feature will bring obvious benefits in terms of being able to perform field maintenance from anywhere over an Ethernet network. This section describes how to connect and configure your system and devices to perform remote monitoring and control, including from a PC, laptop, and portable devices such as a smartphone or pad.

Preset conditions:

1. Complete the installation and network configuration of your ADAM-6000 module
2. Ensure that your ADAM-6000 module is connected to your local Ethernet network

Note! ADAM-6000 modules are developed by public HTML 5 base, but for detailed indication and data transmission mode may be different on the web page of the operating system. The minimum browser requirements are as follows:

Smartphone Browser Requirements

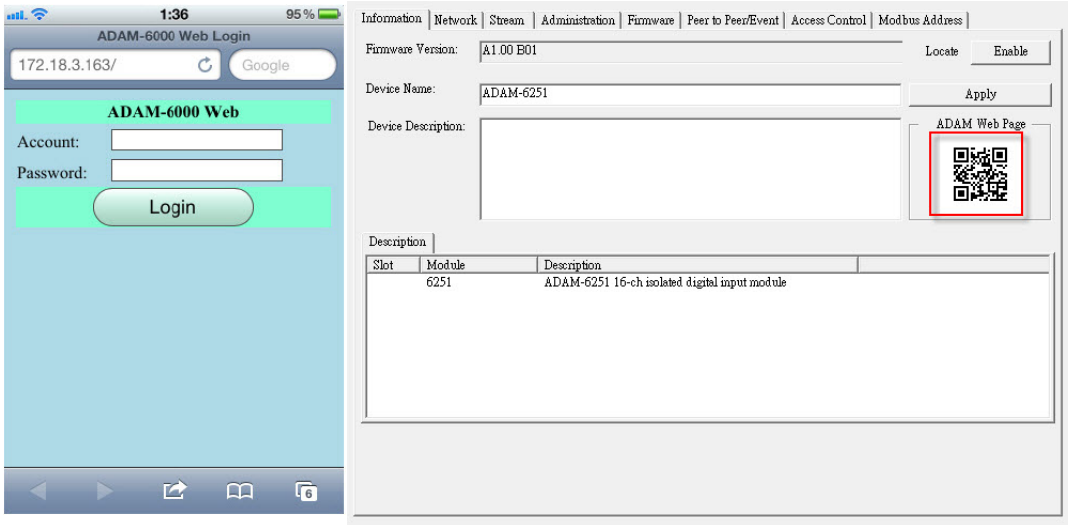
- Safari 5 for Apple iOS
- Web Browser for Google Android 4.0 (Ice Cream Sandwich)
- Chrome for Google Android 4.0 (Ice Cream Sandwich)

PC Browser Requirements

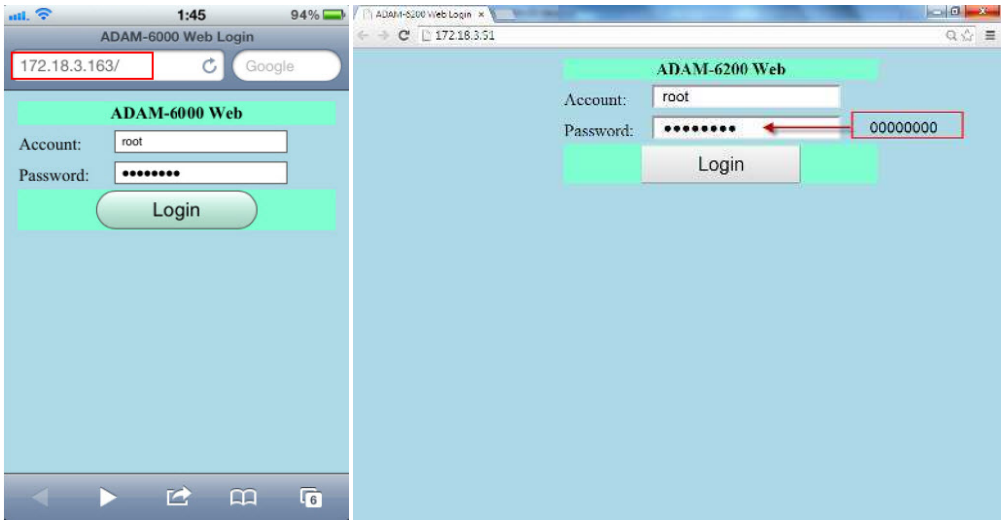
- Internet Explorer (Version 9)
- Google Chrome (Version 8)
- Mozilla Firefox (Version 10)

Operating Steps for Smartphone

1. Connect your smartphone to your local Ethernet network and then open your browser
2. Using your smartphone, you can scan the QR code shown in the utility or enter the module's IP address in the address bar of your smartphone browser to access the login page.



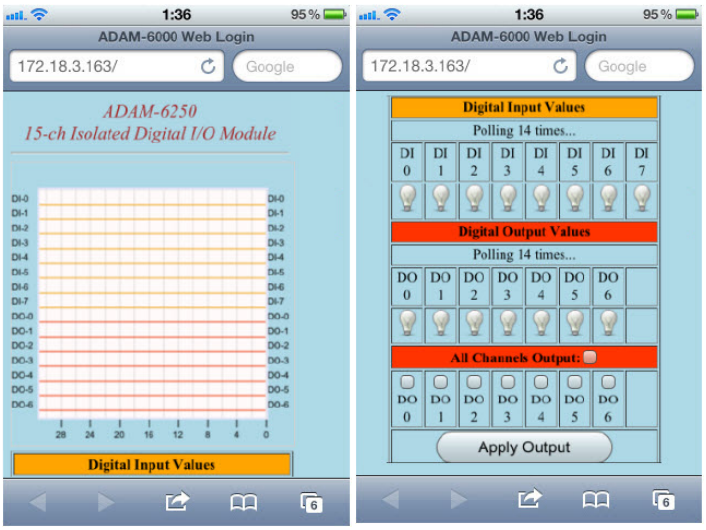
3. Enter the account and password and click **Login**.

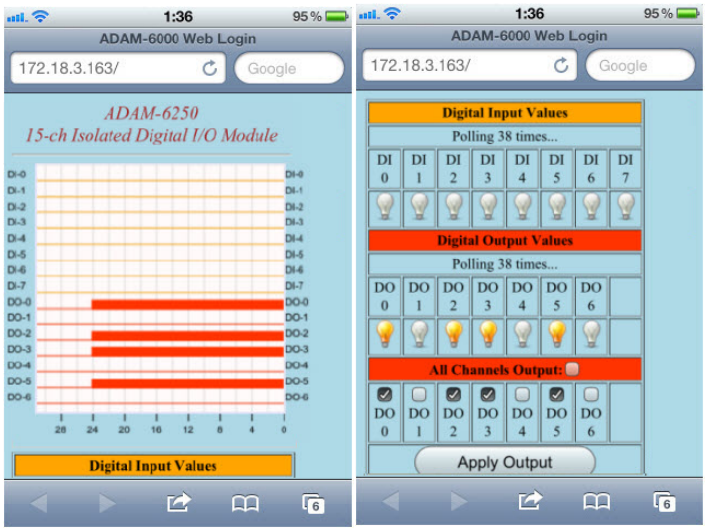


Note! *1. The default account is "root" and the default password is "00000000" (without quotation marks)*

2. To log in via a computer, simply type the IP address into your browser.

4. After you log in, the operation page will appear. This page allows you to monitor the I/O status (trend log) and enable output logging for individual channels. As an example, the bottom images show that if you check **DO 0**, **DO 2**, **DO 3**, and **DO 5**, and then click **Apply Output**, the bulbs for the selected channels will be lit and the trend log will also change.





Chapter 7

Planning Your Application Program

7.1 Introduction

After completing the system configuration, you can begin to plan the application program. This chapter introduces two programming tools for you to implement a DA&C system. The DLL drivers and command sets provide a user-friendly for you to interface your applications and ADAM-6000 I/O modules.

7.2 ADAM .NET Class Library

The Advantech Adam .NET Class Library allows you to quickly and easily develop application programs that can support the .NET Framework 2.0. The Adam .NET Class Library includes all necessary functions to utilize ADAM-6000 modules.

Before installing the Adam .NET Class Library, ensure that your PC supports the .NET Framework 2.0. You can download it from http://support.advantech.com/support/. To install the Adam .NET Class Library, run the setup file Adam.NET Class Library.msi.

After you complete the installation of the Adam .NET Class Library, the Win32 Class Library will be installed at the following path:

- Program Files\Advantech\AdamApax.NET Class Library\Class Library\Win32
  Additionally, the WinCE Class Library will be installed at the following path:
- Program Files\Advantech\AdamApax.NET Class Library\Class Library\WinCE
  To help you become familiar with developing your application program in a short time, there are many example programs that can be found at the following path:
- Program Files\Advantech\AdamApax.NET Class Library\Sample Code
  These examples will provide a basic guide on how to write program code for controlling ADAM-6000 modules. In many cases, you can simple compile and execute programs after modifying the IP address and module name.
  The following example is for the ADAM-6052. The code is written in Visual Basic in Microsoft Visual Studio. The code for this example (ADAM 60xxxDIO.sln) can be found at the following path:
- Program Files\Advantech\AdamApax.NET Class Library\Sample Code\ADAM\Win32\VB\ADAM-6000 Series\Adam60XXDIO.
  Simply open the source code and modify the IP address and module name according to your needs (remember to remove the apostrophe so that the line is no longer ignored as a comment).

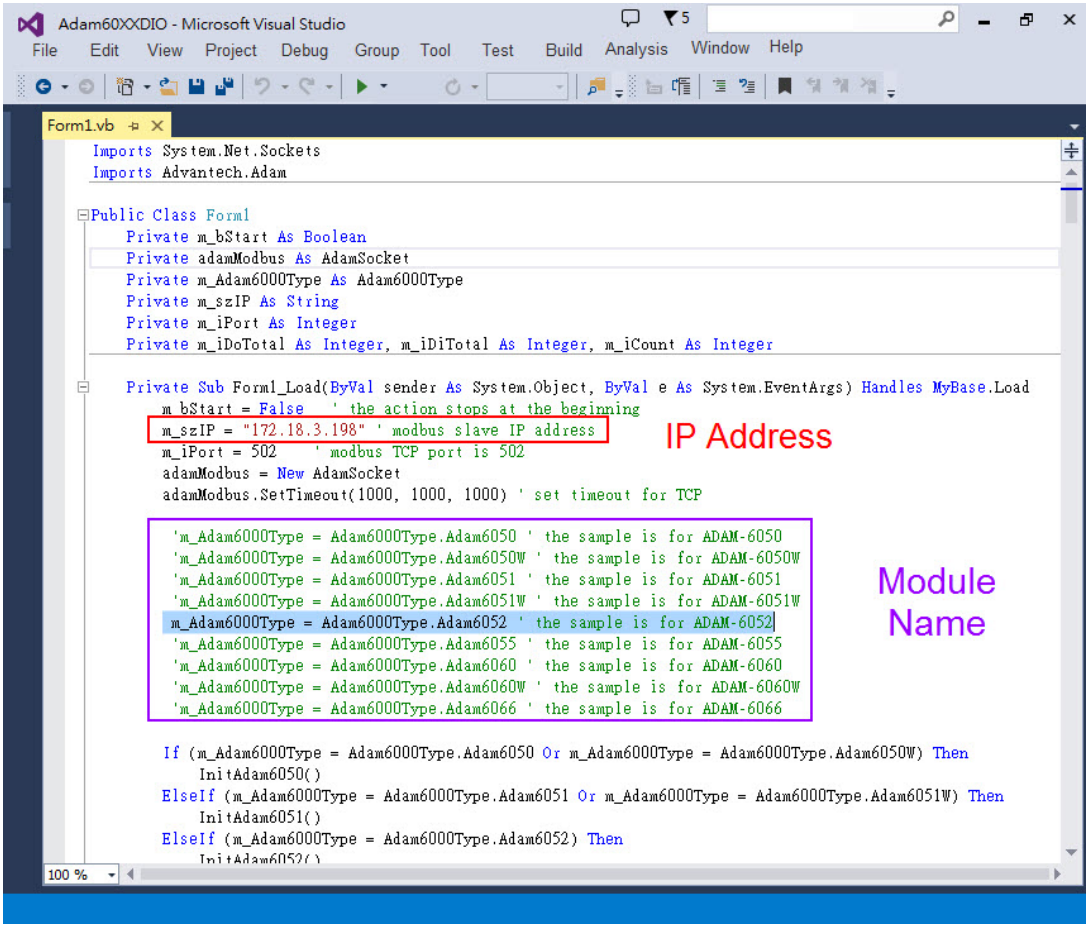


Figure 7.1 Modifying ADAM-6050 .NET

After you have modified the code, you can compile the program and execute it to start the application in order to configure your module.

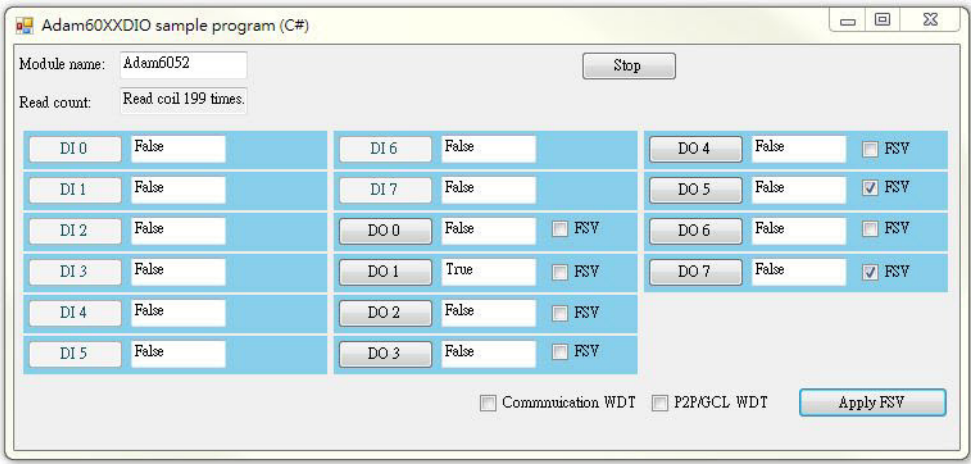


Figure 7.2 Execute the sample code and configure your ADAM module

The ADAM .NET Class Library contains many functions. Documentation is provided to help you understand these functions. This can be accessed from the **Start** menu folder.

7.3 Modbus Protocol for ADAM-6000 Modules

ADAM-6000 modules can communicate with the host PC in the command-response form. When data are not being transmitted, the modules will be in listen mode. Each module will be assigned a unique address. When issuing a command to a system, the host PC will use these addresses to communicate with specific modules and then wait for a response. If none is detected, a timeout will occur, the sequence will be aborted, and control will be returned to the host.

This remainder of this chapter explains the structure of relevant Modbus/TCP commands. Example code is provided in the ADAM .NET Class Library to aid you with reading/writing Modbus/TCP addresses for ADAM-6000 modules. WinCE and Win32 examples can be found at the following path:

- Program Files\Advantech\AdamApax.NET Class Library\Sample Code\ADAM

Note: See Appendix B.2 for the Modbus/TCP addresses of ADAM-6000 modules.

7.3.1 Modbus Protocol Structure

It is important to understand the encapsulation of a Modbus request or response carried on the Modbus/TCP network. A complete command consists of a "command head" (i.e., Modbus application protocol header) and "command body" (i.e., protocol data unit). The command head is prefixed by six bytes and follows the Modbus data packet format; the command body defines the target device and requested action. The examples given in the following section will help you to understand this structure.

7.3.2 Modbus Function Code Introductions

The following function codes are given as a reference to assist with understanding the programming requirements:

Code (Hex)	Name	Usage
01	Read coil status	Read discrete output bit
02	Read input status	Read discrete input bit
03	Read holding registers	Read 16-bit register; used to read integer or floating point process data
04	Read input registers	
05	Force single coil	Write data to force coil On/Off
06	Preset single register	Write data in 16-bit format
08	Loopback diagnosis	Diagnostic testing of the communication port
15	Force multiple coils	Write multiple data to force coil On/Off
16	Preset multiple registers	Write data in 16-bit format

Function Code 01

Reads the discrete output ON/OFF status of an ADAM-6000 module in a binary format.

Request message format:

Command Body					
Station Address	Function Code	Start Address High Byte	Start Address Low Byte	Requested Number of Coil High Byte	Requested Number of Coil Low Byte

Example: Read Coils 1~8 (Addresses 00017~00024) from an ADAM-6000 module.

01 01 00 17 00 08

Response message format:

Command Body				
Station Address	Function Code	Byte Count	Data	Data

Example: Coils 2~7 are on, all others are off.

01 01 01 42

In the response, the status of Coils 1~8 is shown as the byte value 42 (hex), which is equivalent to 0100 0010 in binary format.

Function Code 02

Reads the discrete input ON/OFF status of an ADAM-6000 module in a binary format.

Request message format:

Command Body					
Station Address	Function Code	Start Address High Byte	Start Address Low Byte	Requested Number of Input High Byte	Requested Number of Input Low Byte

Example: Read Coils 1~8 (Addresses 00001~00008) from an ADAM-6000 module.

01 02 00 01 00 08

Response message format:

Command Body				
Station Address	Function Code	Byte Count	Data	Data

Example: Inputs 2 and 3 are on, all others are off.

01 02 01 60

In the response, the status of Inputs 1~8 is shown as the byte value 60 (hex), which is equivalent to 0110 0000 in binary format.

Function Codes 03 and 04

Reads the binary content of input registers.

Request message format:

Command Body					
Station Address	Function Code	Start Address High Byte	Start Address Low Byte	Requested Number of Register High Byte	Requested Number of Register Low Byte

Example: Read Analog Inputs 1 and 2 at Addresses 40001~40002 as a floating point value from an ADAM-6017 module.

01 04 00 01 00 02

Response message format:

Command Body				
Station Address	Function Code	Byte Count	Data	Data

Example: The raw data of Analog Input 1 = 17096 and Analog Input 2 = 0

When input range is set to 0~10 V, the voltages for these inputs are as follows:

Analog Input 1 = (17097/65535) * 10 V = 2.608 V

Analog Input 2 = (0/65535)*10 V = 0 V

01 04 04 42 C8 00 00

Function Code 05

Forces a single coil to either ON or OFF. The requested ON/OFF state is specified by a constant in the query data field. A value of FF 00 (hex) requests it to be ON; a value of 00 00 (hex) requests it to be OFF; a value of FF FF (hex) requests the forced value to be released.

Request message format:

Command Body					
Station Address	Function Code	Coil Address High Byte	Coil Address Low Byte	Force Data High Byte	Force Data Low Byte

Example: Force Coil 3 (Address 00003) to ON in an ADAM-6000 module.

01 05 00 03 FF 00

Response message format:

Command Body					
Station Address	Function Code	Coil Address High Byte	Coil Address Low Byte	Force Data High Byte	Force Data Low Byte

The normal response is an echo of the query, returned after the coil state has been forced.

Function Code 06

Presets an integer value into a single register.

Request message format:

Command Body					
Station Address	Function Code	Register Address High Byte	Register Address Low Byte	Preset Data High Byte	Preset Data Low Byte

Example: Preset Register 40002 to 00 04 (hex) in an ADAM-6000 module.

01 06 00 02 00 04

Response message format:

Command Body					
Station Address	Function Code	Register Address High Byte	Register Address Low Byte	Preset Data High Byte	Preset Data Low Byte

The normal response is an echo of the query, returned after the coil state has been preset.

Function Code 08

Echoes a received query message. Messages can be any length up to half the length of the data buffer minus 8 bytes.

Request message format:

Command Body		
Station Address	Function Code	Any data, length limited to approximately half the length of the data buffer

Example: 01 08 00 02 00 04

Response message format:

Command Body		
Station Address	Function Code	Data bytes received

Example: 01 08 00 02 00 04

Function Code 15 ("0F" in hex)

Forces each coil in a sequence of coils to either an ON or OFF state.

Request message format:

Command Body								
Station Address	Function Code	Start Address High Byte	Start Address Low Byte	Requested Number of Coil High Byte	Requested Number of Coil Low Byte	Byte Count	Force Data High Byte	Force Data Low Byte

Example: Request to force a series of 10 coils starting at Address 00017 ("11" in hex) in an ADAM-6000 module.

01 0F 00 11 00 0A 02 CD 01

The query data contents are two bytes: CD 01 (hex), equivalent to 1100 1101 0000 0001 in binary format. The binary bits are mapped to the addresses in the following manner.

Bit:	1	1	0	0	1	1	0	1	0	0	0	0	0	0	0	1
Address (000XX):	24	23	22	21	20	19	18	17	-	-	-	-	-	-	26	25

Response message format:

Command Body					
Station Address	Function Code	Start Address High Byte	Start Address Low Byte	Requested Number of Coil High Byte	Requested Number of Coil Low Byte

A normal response returns the station address, function code, start address, and requested number of the forced coil.

Example: 01 0F 00 11 00 0A

Function Code 16 ("10" in hex)

Applies a preset value in a sequence of holding registers.

Request message format:

Command Body							
Station Address	Function Code	Start Address High Byte	Start Address Low Byte	Requested Number of Register High Byte	Requested Number of Register Low Byte	Byte Count	Data

Example: Preset Constant 1 (Address 40009) to 100.0 in an ADAM-6000 module.

01 10 00 09 00 02 04 42 C8 00 00

Response message format:

Command Body					
Station Address	Function Code	Start Address High Byte	Start Address Low Byte	Requested Number of Register High Byte	Requested Number of Register Low Byte

A normal response returns the station address, function code, start address, and requested number of preset registers.

Example: 01 10 00 09 00 02

7.4 ASCII Commands for ADAM-6000 Modules

In case you are unfamiliar with the Modbus protocol, Advantech offers a function library as a protocol translator, integrating ASCII commands into the Modbus/TCP structure. Therefore, if you are familiar with ASCII commands, you can use them to access an ADAM-6000 module.

7.4.1 ASCII Syntax

Note: All commands must be written in UPPERCASE!

The command set is divided into the following five categories:

- System command set
- Analog input command set
- Analog input alarm command set
- Universal I/O command set
- Digital I/O command set

These categories are summarized in the following text, followed by information on the individual commands. Although some commands share the same format, their effect varies from module to module. Therefore, the full command sets for each type of modules are listed along with a description of the effect the command has on the given module.

7.4.2 System Command Set

Command Syntax	Command Name	Description
$aaM	Read module name	Returns the name of a specific module
$aaF	Read firmware version	Returns the firmware version of a specific module
#aaVdbbbbddddddd	Write GCL internal flag values	Writes one or more GCL internal flag values to a specific module
$aaVd	Read GCL internal flag values	Returns all GCL internal flag values of a specific module
$aa7	Get DO diagnostic status	Digital output diagnostic status 0=normal, 1=abnormal

Note: $aaM and $aaF are supported by all ADAM-6000 I/O modules.

#aaVdbbbbddddddd is supported by the ADAM-6050, ADAM-6051, ADAM-6052, ADAM-6060, and ADAM-6066.

#aaVd is supported by the ADAM-6050, ADAM-6051, ADAM-6052, ADAM-6060, and ADAM-6066.

ADAM-6000 CE supports additional commands to those listed here. See Appendix F for more details.

$aaM

Name Read module name

Description Returns the name of a specific module

Syntax
$aaM(cr)
$ is a delimiter
aa (range 00~FF) is the 2-character hex slave address of the specified module (always 01)
M refers to the read model number command
(cr) is the terminating character, carriage return (0Dh)

Response
!aa60bb(cr) if the command is valid
?aa(cr) if an invalid command was entered
There is no response if the module detects a syntax error, communication error, or if the address does not exist
! is a delimiter indicating a valid command was received
? is a delimiter indicating the command was invalid
aa (range 00~FF) represents the 2-character hex slave address of the specified module
bb (range 00~FF) represents the 2-character model number (hex) of the specified module
(cr) is the terminating character, carriage return (0Dh)

Example
command: $01M(cr)
response: !016050(cr)

This command requests the module at Address 01h to return its model number. The module responds with "6050," indicating that the module is an ADAM-6050.

$aaF

Name Read firmware version

Description Returns the firmware version of a specific module

Syntax
$aaF(cr)
$ is a delimiter
aa (range 00~FF) is the 2-character hex slave address of the specified module (always 01)
F refers to the read firmware version command
(cr) is the terminating character, carriage return (0Dh)

Response
!aa(version)(cr) if the command is valid
?aa(cr) if an invalid command was entered
There is no response if the module detects a syntax error, communication error, or if the address does not exist
! is a delimiter indicating a valid command was received
? is a delimiter indicating the command was invalid
aa (range 00~FF) represents the 2-character hex slave address of the specified module
(version) represents the firmware version of the module
(cr) is the terminating character, carriage return (0Dh)

Example
command: $01F(cr)
response: !01 1.01(cr)

This command requests the module at Address 01h to return its firmware version. The response indicates that it is version 1.01.

#aaVdbbbbdddddddd

Name	Write GCL internal flag values (auxiliary flags)
Description	Writes one or more GCL internal flag values to a specific module (see Sections 8.3.1 and 8.3.4 for a definition of GCL internal flags)
Syntax	#aaVdbbbbdddddddd(cr) # is a delimiter aa (range 00~FF) is the 2-character hex slave address of the specified module (always 01) Vd refers to the write GCL internal flag values command bbbb indicates which GCL internal flag(s) to set To write to all GCL internal flags: 0000 To write to a single GCL internal flag: First character is 1, and Characters 2~4 indicate the GCL internal flag number which can range from 0h to Fh dddddddd is the hex representation of the GCL internal flag value(s) Each character represents 4 GCL internal flag values (cr) is the terminating character, carriage return (0Dh)
Response	>aa(cr) if the command is valid ?aa(cr) if an invalid command was entered There is no response if the module detects a syntax error, communication error, or if the address does not exist > is a delimiter indicating a valid command was received ? is a delimiter indicating the command was invalid aa (range 00~FF) represents the 2-character hex slave address of the responding module (cr) is the terminating character, carriage return (0Dh)
Example	command: #01VDO00000000000(cr) response: >01(cr) This command sets all GCL internal flags (Flag 0~15) to logic low.
Example	command: #01VDO0000000FFFF(cr) response: >01(cr) This command sets all GCL internal flags (Flag 0~15) to logic high.
Example	command: #01Vd100300000001(cr) response: >01(cr) This command sets one specific GCL internal flag (Flag 3) to logic high.
Example	command: #01Vd100E00000000(cr) Response: >01(cr) This command sets one specific GCL internal flag (Flag 14) to logic low.

$aa7

Name	Get DO diagnostic status	
Description Syntax	Get digital output diagnostic status 0=normal, 1=abnormal $aa7(cr) $ is a delimiter aa (range 00~FF) is the 2-character hex slave address of the specified module (always 01) 7 refers to the Get DO diagnostic status command (cr) is the terminating character, carriage return (0Dh)	
Response	!aa(DO0) (DO1)…(DO5) (cr) if the command is valid ?aa(cr) if an invalid command was entered There is no response if the module detects a syntax error, communication error, or if the address does not exist ! is a delimiter indicating a valid command was received ? is a delimiter indicating the command was invalid Aa (range 00~FF) represents the 2-character hex slave address of the specified module (DO0)(DO1)…(DO5)represents the firmware version of the module (cr) is the terminating character, carriage return (0Dh)	
Example	command:	$017
	Response:	!01101110

- DO0 = 1 (Abnormal) DO1 = 0 (Abnormal)
- DO2 = 1 (Abnormal) DO3 = 1 (Abnormal)
- DO4 = 1 (Abnormal) DO5 = 0 (Normal)

$aaVd

Name	Read GCL internal flag values (auxiliary flags)
Description	Reads all GCL internal flag values of a specific module (see Sections 8.3.1 and 8.3.4 for a definition of GCL internal flags)
Syntax	$aaVd(cr) $ is a delimiter aa (range 00~FF) is the 2-character hex slave address of the specified module (always 01) Vd refers to the read GCL internal flag values command (cr) is the terminating character, carriage return (0Dh)
Response	>aadddddddd(cr) if the command is valid ?aa(cr) if an invalid command was entered There is no response if the module detects a syntax error, communication error, or if the address does not exist > is a delimiter indicating a valid command was received ? is a delimiter indicating the command was invalid aa (range 00~FF) represents the 2-character hex slave address of the specified module dddddddd is the hex representation of the GCL internal flag value(s) Each character represents 4 GCL internal flag values (cr) is the terminating character, carriage return (0Dh)
Example	command: $01Vd (cr) response: >010000002B(cr) This command reads all GCL internal flags values. The character "B" (hex) = "1011" (binary) and the character "2" (hex) = "0010" (binary). Therefore, GCL Internal Flags 0, 1, 3, 5 are logic high while the other GCL internal flags are logic low.

7.4.3 Analog Input Command Set

(ADAM-6015, 6017, 6018, 6018+)

Command Syntax	Command Name	Description
#aan	Read single analog input	Returns the input value of a specific analog input channel
#aa	Read all analog inputs	Returns the input values of all analog input channels
$aa0	Span calibration	Calibrates a module to correct for gain errors
$aa1	Zero calibration	Calibrates a module to correct for offset errors
$aa6	Read channel enable/ disable statuses	Returns the enable/disable status of all analog input channels
$aa5mm	Set all channel enable/ disable statuses	Sets the enable/disable status of all analog input channels
#aaMH	Read all max. data values	Returns the max. data values of all analog input channels
#aaMHn	Read single max. data value	Returns the max. data value of a specific analog input channel
#aaML	Read all min. data values	Returns the min. data values of all analog input channels
#aaMLn	Read single min. data value	Returns the min. data of a specific analog input channel
#aaDnd	Set single digital output status	Sets the output status of a specific digital output channel
$aaBnn	Read single analog input range code	Returns the input range code of a specific analog input channel
$aaBRCnn	Read single analog input range code	Returns the input range code of a specific analog input channel (ADAM-6017 only)
$aaAccrr	Write single analog input code	Writes the input range code to a specific analog input channel
$aaAccrrrr	Write single analog input code	Writes the input range code to a specific analog input channel (ADAM-6017 only)

#aan

Name	Read single analog input
Description	Returns the input value of a specific analog input channel
Syntax	#aan(cr) # is a delimiter aa (range 00~FF) is the 2-character hex slave address of the specified module (always 01) n (range 0~8) refers to the specific channel you want to read the input data from (cr) is the terminating character, carriage return (0Dh)
Response	>(data)(cr) if the command is valid ?aa(cr) if an invalid command was entered There is no response if the module detects a syntax error, communication error, or if the address does not exist > is a delimiter indicating a valid command was received ? is a delimiter indicating the command was invalid (cr) is the terminating character, carriage return (0Dh)
Example	command: #012(cr) response: >+10.000 Channel 2 of the ADAM-6000 analog module at Address 01h responds with an input value of +10.000.

#aa

Name	Read all analog inputs
Description	Returns the input data of all analog input channels
Syntax	#aa(cr) # is a delimiter aa (range 00~FF) is the 2-character hex slave address of the specified module (always 01) (cr) is the terminating character, carriage return (0Dh)
Response	>(data)(data)(data)(data)(data)(data)(data)(data)(data)(cr) if the command is valid ?aa(cr) if an invalid command was entered There is no response if the module detects a syntax error, communication error, or if the address does not exist > is a delimiter indicating a valid command was received ? is a delimiter indicating the command was invalid (cr) is the terminating character, carriage return (0Dh)
Note:	The last data returned is the average value of the preset channels in this module
Example	command: #01(cr) response: >+10.000+10.000+10.000+10.000+10.000+10.000+10.000+10.000+10.000

$aa0

Name	Span calibration
Description	Calibrates a module to correct for gain errors
Syntax	$aa0(cr) $ is a delimiter aa (range 00~FF) is the 2-character hex slave address of the specified module (always 01) 0 refers to the span or auto calibration command (cr) is the terminating character, carriage return (0Dh)
Response	!aa(cr) if the command is valid ?aa(cr) if an invalid command was entered There is no response if the module detects a syntax error, communication error, or if the address does not exist ! is a delimiter indicating a valid command was received ? is a delimiter indicating the command was invalid aa (range 00~FF) is the 2-character hex slave address of the specified module (cr) is the terminating character, carriage return (0Dh)
Note:	To successfully calibrate an analog input module's input range, a reliable calibration input signal should be connected to the analog input module before and during the calibration.

$aa1

Name	Zero calibration
Description	Calibrates a module to correct for offset errors
Syntax	$aa1(cr) $ is a delimiter aa (range 00~FF) is the 2-character hex slave address of the specified module (always 01) 1 refers to the offset or auto calibration command (cr) is the terminating character, carriage return (0Dh)
Response	!aa(cr) if the command is valid ?aa(cr) if an invalid command was entered There is no response if the module detects a syntax error, communication error, or if the address does not exist ! is a delimiter indicating a valid command was received ? is a delimiter indicating the command was invalid aa (range 00~FF) represents the 2-character hex slave address of the specified module (cr) is the terminating character, carriage return (0Dh)
Note:	To successfully calibrate an analog input module's input range, a reliable calibration input signal should be connected to the analog input module before and during the calibration.

$aa6

Name	Read all channel enable/disable statuses
Description	Returns the enable/disable status of all analog input channels
Syntax	$aa6(cr) $ is a delimiter aa (range 00~FF) is the 2-character hex slave address of the specified module (always 01) 6 refers to the read channel enable/disable statuses command (cr) is the terminating character, carriage return (0Dh)
Response	!aamm(cr) if the command is valid ?aa(cr) if an invalid command was entered There is no response if the module detects a syntax error, communication error or if the address does not exist ! is a delimiter indicating a valid command was received ? is a delimiter indicating the command was invalid aa (range 00~FF) is the 2-character hex slave address of the specified module mm are two hexadecimal values (each value is interpreted as 4 bits). The first 4-bit value represents the status of channels 7-4 and the second value represents the status of Channels 3-0. A value of 0 means the channel is disabled, while a value of 1 means the channel is enabled. (cr) is the terminating character, carriage return (0Dh)
Example	command: $016(cr) response: !01FF(cr) This command requests the module at Address 01h to return the enable/disable status of all analog input channels. The module responds with "FF" (hex), which is equivalent to "1111" (binary) and "1111" (binary), meaning that all channels are enabled.

$aa5mm

Name	Set all channel enable/disable statuses
Description	Sets the enable/disable status of all analog input channels
Syntax	$aa5mm(cr) $ is a delimiter aa (range 00~FF) is the 2-character hex slave address of the specified module (always 01) 5 refers to the set all enable/disable statuses command mm are two hexadecimal values (each value is interpreted as 4 bits). The first 4-bit value represents the status of channels 7-4 and the second value represents the status of Channels 3-0. A value of 0 means the channel is disabled, while a value of 1 means the channel is enabled. (cr) is the terminating character, carriage return (0Dh)
Response	!aa(cr) if the command is valid ?aa(cr) if an invalid command was entered There is no response if the module detects a syntax error, communication error, or if the address does not exist ! is a delimiter indicating a valid command was received ? is a delimiter indicating the command was invalid aa (range 00~FF) is the 2-character hex slave address of the specified module (cr) is the terminating character, carriage return (0Dh)

Example

command: $01581(cr)

response: !01(cr)

This command enables/disables all channels of the analog input module at Address 01h. The character "8" (hex) = "1000" (binary), which enables Channel 7 and disables Channels 4~6; the character "1" (hex) = "0001" (binary), which disables Channels 1~3 and enables Channel 0.

#aaMH

Name

Read all max. data values

Description

Returns the max. data values of all analog input channels

Syntax

#aaMH(cr)
is a delimiter
aa (range 00~FF) is the 2-character hex slave address of the specified module (always 01)
MH refers to the read all max. data values command
(cr) is the terminating character, carriage return (0Dh)

Response

>(data)(data)(data)(data)(data)(data)(data)(data)(cr) if the command is valid
?aa(cr) if an invalid command was entered
There is no response if the module detects a syntax error, communication error, or if the address does not exist
> is a delimiter indicating a valid command was received
? is a delimiter indicating the command was invalid
aa (range 00~FF) is the 2-character hex slave address of the specified module
(cr) is the terminating character, carriage return (0Dh)

Example

command: #01MH(cr)

Response: +10.000+10.000+10.000+10.000+10.000+10.000
+10.000+10.000+10.000

This command requests the module at Address 01h to return the historic max. value from each analog input channel.

Note:

The last data returned is the average of the max. values from all channels

#aaMHn

Name	Read single max. data value
Description	Returns the max. data value of a specific analog input channel
Syntax	#aaMHn(cr) # is a delimiter aa (range 00~FF) is the 2-character hex slave address of the specified module (always 01) MH refers to the read single max. data value command n (range 0-8) represents the specified channel you want to read the input data from (cr) is the terminating character, carriage return (0Dh)
Response	>(data)(cr) if the command is valid ?aa(cr) if an invalid command was entered There is no response if the module detects a syntax error, communication error, or if the address does not exist > is a delimiter indicating a valid command was received ? is a delimiter indicating the command was invalid aa (range 00~FF) represents the 2-character hex slave address of the specified module (cr) is the terminating character, carriage return (0Dh)
Example	command: #01MH2(cr) response: >+10.000 This command requests the module at Address 01h to return the historic max. value from Analog Input Channel 2.

#aaML

Name	Read all min. data values
Description	Returns the min. data values of all analog input channels
Syntax	#aaML(cr) # is a delimiter aa (range 00~FF) is the 2-character hex slave address of the specified module (always 01) ML refers to the read all min. data values command (cr) is the terminating character, carriage return (0Dh)
Response	>(data)(data)(data)(data)(data)(data)(data)(data)(data)(cr) if the command is valid ?aa(cr) if an invalid command was entered There is no response if the module detects a syntax error, communication error, or if the address does not exist > is a delimiter indicating a valid command was received ? is a delimiter indicating the command was invalid aa (range 00~FF) represents the 2-character hex slave address of the specified module (cr) is the terminating character, carriage return (0Dh)
Example	command: #01ML(cr) response: >+10.000+10.000+10.000+10.000+10.000+10.000+10.000+10.000+10.000 This command requests the module at Address 01h to return the historic min. value from each analog input channel.
Note:	The last data returned is the average value of all channels.

#aaMLn

Name Read single min. data value

Description Reads the min. data value of a specific analog input channel

Syntax
#aaMLn(cr)
is a delimiter
aa (range 00~FF) is the 2-character hex slave address of the specified module (always 01)
ML refers to the read single min. data value command
n (range 0-8) represents the specified channel you want to read the input data from
(cr) is the terminating character, carriage return (0Dh)

Response
>(data)(cr) if the command is valid
?aa(cr) if an invalid command was entered
There is no response if the module detects a syntax error, communication error, or if the address does not exist
> is a delimiter indicating a valid command was received
? is a delimiter indicating the command was invalid
aa (range 00~FF) is the 2-character hex slave address of the specified module
(cr) is the terminating character, carriage return (0Dh)

Example
command: #01ML3(cr)
response: >+10.000
This command requests the module at Address 01h to return the historic min. value from Analog Input Channel 3.

#aaDnd

Name Set single digital output status

Description Sets the output status of a specific digital output channel

Syntax
#aaDnd(cr)
is a delimiter
aa (range 00~FF) is the 2-character hex slave address of the specified module (always 01)
D refers to the set digital output statuses command
n (range 0-1) represents the specific channel you want to set the output status of
d (range 0-1) represents the status you want to set to the specified channel
(cr) is the terminating character, carriage return (0Dh)

Response
!aa(cr) if the command is valid
?aa(cr) if an invalid command was entered
There is no response if the module detects a syntax error, communication error, or if the address does not exist
! is a delimiter indicating a valid command was received
? is a delimiter indicating the command was invalid
aa (range 00~FF) is the 2-character hex slave address of the specified module
(cr) is the terminating character, carriage return (0Dh)

Example
command: #01DO1(cr)
response: !01
This command set Digital Output Channel 0 to "ON" for the module at Address 01h.

$aaBnn

Name	Read single analog input range code
Description	Returns the range code of a specific analog input channel
Syntax	$aaBnn(cr) $ is a delimiter aa (range 00~FF) is the 2-character hex slave address of the specified module (always 01) B refers to the read single analog input range code command nn (range 00-07) is the channel you want to read the range code from (cr) is the terminating character, carriage return (0Dh)
Response	!aa(data)(code) if the command is valid ?aa(cr) if an invalid command was entered There is no response if the module detects a syntax error, communication error, or if the address does not exist ! is a delimiter indicating a valid command was received ? is a delimiter indicating the command was invalid (cr) is the terminating character, carriage return (0Dh) (code) is the range code (refer to the following tables)

ADAM-6017-BE Analog Input Channel Range Codes

Range Code (Hex)	Range Code (Decimal)	Range Description
08	8	+/-10 V
09	9	+/-5V
0A	10	+/-1V
0B	11	+/-500 mV
0C	12	+/-150 mV
0D	13	0~20 mA
07	7	4~20 mA
0x0148*	-	0~10 V
0x0147*	-	0~5 V
0x0145*	-	0~1 V
0x0106*	-	0~500 mV
0x0105*	-	0~150 mV
0x0181*	-	+/-20 mA

* these range codes are supported only by the ADAM-6017-CE

Note! *Use $aaBRCnn to read the single analog input range code of an ADAM-6017-CE.*

ADAM-6015 Analog Input Channel Range Code

Range Code (Hex)	Range Code (Decimal)	Range Description
20	32	Pt 100 (a=0.00385) -50~150°C
21	33	Pt 100 (a=0.00385) 0~100°C
22	34	Pt 100 (a=0.00385) 0~200°C
23	35	Pt 100 (a=0.00385) 0~400°C
24	36	Pt 100 (a=0.00385) -200~200°C
25	37	Pt 100 (a=0.00392) -50~150°C
26	38	Pt 100 (a=0.00392) 0~100°C
27	39	Pt 100 (a=0.00392) 0~200°C
28	40	Pt 100 (a=0.00392) 0~400°C
29	41	Pt 100 (a=0.00392) -200~200°C
2A	42	Pt 1000 -40~160°C
2B	43	Balco 500 -30~120°C
2C	44	Ni 518 -80~100°C
2D	45	Ni 518 0~100°C

ADAM-6018/6018+ Analog Input Channel Range Code

Range Code (Hex)	Range Code (Decimal)	Range Description
0E	14	Thermocouple J (0~760°C)
0F	15	Thermocouple K (0~1370°C)
10	16	Thermocouple T (-100~400°C)
11	17	Thermocouple E (0~1000°C)
12	18	Thermocouple R (500~1750°C)
13	19	Thermocouple S (500~1750°C)
14	20	Thermocouple B (500~1800°C)

$aaBRCnn

Name Read single analog input range code

Description Returns the range code of a specific analog input channel (ADAM-6017 only)

Syntax
$aaBRCnn
$ is a delimiter
aa (range 00~FF) represents the 2-character hex slave address of the specified module (always 01)
B refers to the read single analog input range code command
RC is the range code
nn (range 00-07) is the channel you want to read the range code from

Response
!aa(data)(code) if the command is valid
?aa(cr) indicates that an invalid command was entered
There is no response if the module detects a syntax error, communication error, or if the address does not exist
! is a delimiter indicating that a valid command was received
? is a delimiter indicating that the command was invalid
(cr) is a terminating character, carriage return (0Dh)
(code) is the range code (refer to the following tables)

ADAM-6017-CE & D Analog Input Channel Range Code

Range Code (Hex)	Range Description
0x0143	+/-10 V
0x0142	+/-5 V
0x0140	+/-1 V
0x0104	+/-500 mV
0x0103	+/-150 mV
0x0148	0~10 V
0x0147	0~5 V
0x0145	0~1 V
0x0106	0~500 mV
0x0105	0~150 mV
0x0182	0~20 mA
0x0180	4~20 mA
0x0181	+/-20 mA

$aaAccrr/$aaAccrrrr

Name Write single analog input range code

Description Writes the analog input range code to a specific analog input channel and responds whether the setting is successful

Syntax
$aaAccrr/$aaAccrrrr
$ is a delimiter
aa (range 00~FF) represents the 2-character hex slave address of the specified module (always 01)
A refers to the write single analog input range code command
Cc is the channel you want to set
rr/rrrr is the specified channel you want to write the range code to

Response
?aa(cr) indicates that an invalid command was entered
There is no response if the module detects a syntax error, communication error, or if the address does not exist
! is a delimiter indicating that a valid command was received
? is a delimiter indicating that the command was invalid
(cr) is the terminating character, carriage return (0Dh)
(code) is the range code (refer to the following tables)
Note: $aaAccrrrr is only for the ADAM-6017-CE

ADAM-6017 Analog Input Range

Range Description	CE & D Version Range Code	AE & BE Version Range Code
+/-10 V	0x0143	08
+/-5 V	0x0142	09
+/-1 V	0x0140	0A
+/-500 mV	0x0104	0B
+/-150 mV	0x0103	0C
0~10 V	0x0148	
0~5 V	0x0147	
0~1 V	0x0145	
0~500 mV	0x0106	
0~150 mV	0x0105	
0~20 mA	0x0182	0D
4~20 mA	0x0180	07
+/-20 mA	0x0181	

Note!

1. The ADAM-6017-AE and ADAM-6017-BE version of the range code is also applicable to the ADAM-6017-CE. However, we suggest that you use the new range code in our modules.

2. For the ADAM-6015, see the ADAM-6018 table above.

Example	command:	$01A010147(cr)
		Channel 1 is set to 0x0147, which means "0~5 V"
	response:	!010B(cr) - setting successful
		!010147(cr) - setting successful
		?01 - setting fail
Example	command:	$01A0109(cr)
		channel 1 set to 9, which means "+/-5 V"
	response:	!01(cr) - setting successful ?01-setting fail ?01AVG(cr) - setting fail, because the channel is set to "ON" of the average channel

7.4.4 Analog Input Alarm Command Set

(ADAM-6015, 6017, 6018, 6018+)

Command Syntax	Command Name	Description
$aaCjAhs	Set alarm mode	Sets the high/low alarm of a specific channel to either momentary or latching mode
$aaCjAh	Read alarm mode	Returns the alarm mode of a specific channel
$aaCjAhEs	Enable/disable high/low alarm	Enables/disables the high/low alarm of a specific channel
$aaCjCh	Clear latch alarm	Resets a latched alarm for a specific channel
$aaCjAhCCn	Set alarm connection (for ADAM-6018/6017)	Connects the high/low alarm of a specific analog input channel to interlock with a specific digital output channel
$aaCjRhC	Read alarm connection (for ADAM-6017/6018)	Returns the alarm configuration of a specific channel
$aaCjAhU	Set alarm limit	Sets the high/low alarm limit value of a specific channel
$aaCjRhU	Read alarm limit	Returns the high/low alarm limit value of a specific channel
$aaCjS	Read alarm status	Returns the alarm status of a specific channel

$aaCjAhs

Name
Set alarm mode

Description
Sets the high/low alarm of a specific channel to either latching or momentary mode

Syntax
$aaCjAhs(cr)
$ is a delimiter
aa (range 00~FF) is the 2-character hex slave address of the specified module (always 01)
Cj specifies the channel j (j: 0~7)
A refers to the set alarm mode command
h indicates the alarm type (H = high alarm, L = low alarm)
s indicates the alarm mode (M = momentary mode, L = latching mode)
(cr) is the terminating character, carriage return (0Dh)

Response
!aa(cr) if the command is valid
?aa(cr) if an invalid command was entered
There is no response if the system detects a syntax error, communication error, or if the address does not exist
! is a delimiter indicating a valid command was received
aa is the 2-character hexadecimal slave address of the specified module
(cr) is the terminating character, carriage return (0Dh)

Example
command: $01C1AHL(cr)
response: !01(cr)
Channel 1 of the module at Address 01h is instructed to set its high alarm to latching mode. The module confirms that the command has been received.

$aaCjAh

Name Read alarm mode

Description Returns the alarm mode of a specific channel

Syntax
$aaCjAh(cr)
$ is a delimiter
aa (range 00~FF) is the 2-character hex slave address of the specified module (always 01)
Cj specifies the channel j (j: 0~7)
A refers to the read alarm mode command
h indicates the alarm type (H = high alarm, L = low alarm)
(cr) is the terminating character, carriage return (0Dh)

Response
!aas(cr) if the command is valid
?aa(cr) if an invalid command was entered
There is no response if the system detects a syntax error, communication error, or if the address does not exist
! is a delimiter indicating a valid command was received
aa is the 2-character hexadecimal slave address of the specified module
s indicates the alarm mode (M = momentary mode, L = latching mode)
(cr) is the terminating character, carriage return (0Dh)

Example
command: $01C1AL(cr)
response: !01M(cr)
Channel 1 of the module at Address 01h is instructed to return its low alarm mode. The module responds that it is in momentary mode.

$aaCjAhEs

Name Enable/disable high/low alarm

Description Enables/disables the high/low alarm of a specific channel

Syntax
$aaCjAhEs(cr)
$ is a delimiter
aa (range 00~FF) is the 2-character hex slave address of the specified module (always 01)
Cj specifies the channel j (j: 0~7)
AhEs refers to the enable/disable high/low alarm command
h indicates the alarm type (H = high alarm, L = low alarm)
s indicates the alarm enable/disable status (E = enable, D = disable)
(cr) is the terminating character, carriage return (0Dh)

Response
!aa(cr) if the command was valid
?aa(cr) if an invalid command is entered
There is no response if the system detects a syntax error, communication error, or if the address does not exist
! is a delimiter indicating a valid command was received
aa is the 2-character hexadecimal slave address of the specified module
(cr) is the terminating character, carriage return (0Dh)

Example
command: $01C1ALEE(cr)
response: !01(cr)
Channel 1 of the module at Address 01h is instructed to enable its low alarm function. The module confirms that its low alarm function has been enabled.

Note: An analog input module requires up to 2 s after it receives an enable/disable alarm command for the setting take effect.

$aaCjCh

Name Clear latch alarm

Description Resets a latched alarm for a specific channel

Syntax
$aaCjCh(cr)
$ is a delimiter
aa (range 00~FF) is the 2-character hex slave address of the specified module (always 01)
Cj specifies the channel j (j: 0~7)
Ch refers to the clear latch alarm command
h indicates the alarm type (H = high alarm, L = low alarm)
(cr) is the terminating character, carriage return (0Dh)

Response
!aa(cr) if the command is valid
?aa(cr) if an invalid command was entered
There is no response if the system detects a syntax error, communication error, or if the address does not exist
! is a delimiter indicating a valid command was received
aa is the 2-character hexadecimal slave address of the specified module
(cr) is the terminating character, carriage return (0Dh)

Example
command: $01C1CL(cr)
response: !01(cr)
Channel 1 of the module at Address 01h is instructed to set its low alarm state to OFF. The module confirms that it has done so.

$aaCjAhCCn

Name Set alarm connection

Description Connects the high/low alarm of a specific analog input channel to interlock with a specific digital output channel

Syntax
$aaCjAhCCn(cr)
$ is a delimiter
aa (range 00~FF) is the 2-character hex slave address of the specified module (always 01)
Cj specifies the analog input channel j (j: 0~7)
AhC refers to the set alarm connection command
h indicates the alarm type (H = high alarm, L = low alarm)
Cn specifies the digital output channel n (n: 0~1)
To disconnect the digital output, n should be set as ?*?
(cr) is the terminating character, carriage return (0Dh)

Response
!aa(cr) if the command is valid
?aa(cr) if an invalid command was entered
There is no response if the system detects a syntax error, communication error, or if the address does not exist
! is a delimiter indicating a valid command was received
aa is the 2-character hexadecimal slave address of the specified module
(cr) is the terminating character, carriage return (0Dh)

Example
command: $01C1ALCC0(cr)
response: !01(cr)
Channel 1 of the module at Address 01h is instructed to connect its low alarm to the digital output of Channel 0. The system confirms that it has done so.

$aaCjRhC

Name Read alarm connection

Description Returns the high/low alarm limit value of a specific channel

Syntax
$aaCjRhC(cr)
$ is a delimiter
aa (range 00~FF) is the 2-character hex slave address of the specified module (always 01)
Cj specifies the analog input channel j (j: 0~7)
RhC refers to the read alarm connection command
h indicates the alarm type (H = high alarm, L = low alarm)
(cr) is the terminating character, carriage return (0Dh)

Response
!aaCn(cr) if the command is valid
?aa(cr) if an invalid command was entered
There is no response if the system detects a syntax error, communication error, or if the address does not exist
! is a delimiter indicating a valid command was received
aa is the 2-character hexadecimal slave address of the specified module
Cn identifies whether the desired digital output channel n (n: 0~1) interlocks with the alarm of the specific analog input channel. If the values of n are "*", then the analog input has no connection with the digital output point.
(cr) is the terminating character, carriage return (0Dh)

Example
command: $01C1RLC(cr)
response: !01C0(cr)
Channel 1 of the module at Address 01h is instructed to return its low alarm output connection. The system responds that the low alarm output connects to the digital output at Channel 0.

$aaCjAhU

Name Set alarm limit

Description Sets the high/low alarm limit value of a specific channel

Syntax
$aaCjAhU(data)(cr)
$ is a delimiter
aa (range 00~FF) is the 2-character hex slave address of the specified module (always 01)
Cj specifies the analog input channel j (j: 0~7)
AhU refers to the set alarm limit command
h indicates the alarm type (H = high alarm, L = low alarm)
(data) specifies the alarm limit setting
The format is always in engineering units
(cr) is the terminating character, carriage return (0Dh)

Response
!aa(cr) if the command is valid
?aa(cr) if an invalid command was entered
There is no response if the system detects a syntax error, communication error, or if the address does not exist
! is a delimiter indicating a valid command was received
aa is the 2-character hexadecimal slave address of the specified module
(cr) represents terminating character, carriage return (0Dh)

Example
command: $01C1AHU+080.00(cr)
response: !01(cr)
The high alarm limit of Channel 1 in the module at Address 01h has been set +80. The system confirms that the command has been received.

Note: An analog input module requires up to 2 s after receiving a set alarm limit command for the settings to take effect.

$aaCjRhU

Name Read alarm limit

Description Returns the high/low alarm limit value of a specific channel

Syntax
$aaCjRhU(cr)
$ is a delimiter
aa (range 00~FF) is the 2-character hex slave address of the specified module (always 01)
Cj specifies the analog input channel j (j: 0~7)
RhU refers to the read alarm limit command
h indicates the alarm type (H = high alarm, L = low alarm)
(cr) is the terminating character, carriage return (0Dh)

Response
!aa(data)(cr) if the command is valid
?aa(cr) if an invalid command was entered
There is no response if the system detects a syntax error, communication error, or if the address does not exist
! is a delimiter indicating a valid command was received
aa is the 2-character hexadecimal slave address of the specified module
(data) represents the alarm limit setting (in engineering units)
(cr) is the terminating character, carriage return (0Dh)

Example
command: $01C1RHU(cr)
response: !01+2.0500(cr)
Channel 1 of the module at Address 01h has been configured to accept a 5 V input. The command instructs the system to return the high alarm limit value for that channel. The system responds that the high alarm limit value for this channel is 2.0500 V.

$aaCjS

Name Read alarm status

Description Returns the alarm status of a specific channel

Syntax
$aaCjS(cr)
$ is a delimiter
aa (range 00~FF) is the 2-character hex slave address of the specified module (always 01)
Cj specifies the analog input channel j (j: 0~7)
S refers to the read alarm status command
(cr) is the terminating character, carriage return (0Dh)

Response
!aahl(cr) if the command is valid
?aa(cr) if an invalid command was entered
There is no response if the system detects a syntax error, communication error, or if the address does not exist
! is a delimiter indicating that a valid command was received
aa is the 2-character hexadecimal slave address of the specified module
h represents the high alarm status, where "1" = the high alarm has been triggered, "0" = the high alarm has not been triggered
l represents the low alarm status, where "1" = the low alarm has been triggered, "0" = the low alarm has not been triggered
(cr) is the terminating character, carriage return (0Dh)

Example
command: $01C1S(cr)
response: !0101(cr)
The command requests the module at Address 01h to return its alarm status for Channel 1. The system responds that a high alarm has not been triggered, but the low alarm has been triggered.

7.4.5 Universal I/O Command Set

(ADAM-6024)

Command Syntax	Command Name	Description
$aa5mm	Set all enable/disable statuses	Sets the enable/disable status of all analog input channels
$aa6	Read all enable/disable statuses	Returns the enable/disable status of all analog input channels
#aa	Read all analog input values	Returns the input values of all analog input channels
#aacc	Read single analog input value	Returns the input value of a specific analog input channel
$aaDcc	Read single analog output startup value	Returns the startup output value of a specific analog output channel
$aaDcchhh	Set single analog output startup value	Sets the startup output value of a specific analog output channel
#aaccdd.ddd	Write single analog output value	Writes a value to a specific analog output channel
#aa7	Read all digital input statuses	Returns the statuses of all digital input channels
#aaccdd	Write digital output values	Writes a value to one or all digital output channels
$aa2	Read digital output values	Reads the value of all digital output channels
$aaBcc	Read single analog input range code	Returns the channel range code of a specific analog input channel
$aaCcc	Read single analog output range code	Returns the channel range code from the specified analog output channel

$aa5mm

Name	Set all enable/disable statuses
Description	Sets the enable/disable statuses of all analog input channels
Syntax	$aa5mm(cr) $ is a delimiter aa (range 00~FF) is the 2-character hex slave address of the specified module (always 01) 5 refers to the set all enable/disable statuses command mm (range 00~FF) are two hexadecimal characters (each character is interpreted as 4 bits). The first 4-bit value represents the status of Channels 5~4; the second 4-bit value represents the status of Channels 3~0. A value of 0 means that the channel is disabled, whereas a value of 1 means that the channel is enabled. (cr) is the terminating character, carriage return (0Dh)
Response	!aa(cr) if the command is valid ?aa(cr) if an invalid command was entered There is no response if the module detects a syntax error, communication error, or if the address does not exist ! is a delimiter indicating a valid command was received ? is a delimiter indicating the command was invalid aa (range 00~FF) is the 2-character hex slave address of the specified module (cr) is the terminating character, carriage return (0Dh)

Example command: $01521(cr)
response: !01(cr)
The command enables/disables all analog input channels of the module at Address 01h. A value of "2" (hex) = "0010" (binary), which enables Channel 5 and disables Channel 4; a value of 1 (hex) = "0001" (binary), which disables Channels 1~3 and enables Channel 0.

$aa6

Name Read all enable/disable statuses

Description Returns the enable/disable statuses of all analog input channels

Syntax $aa6(cr)
$ is a delimiter
aa (range 00~FF) is the 2-character hex slave address of the specified module (always 01)
6 refers to the read all enable/disable statuses command
(cr) is the terminating character, carriage return (0Dh)

Response !aamm(cr) if the command is valid
?aa(cr) if an invalid command was entered
There is no response if the module detects a syntax error, communication error, or if the address does not exist
! is a delimiter indicating a valid command was received
? is a delimiter indicating the command was invalid
aa (range 00~FF) is the 2-character hex slave address of the specified module
mm are two hex values (each value is interpreted as 4 bits). The first 4-bit value represents the status of Channels 7~4 and the second value represents the status of Channels 3~0. A value of 0 means that the channel is disabled, whereas a value of 1 means that the channel is enabled.
(cr) is the terminating character, carriage return (0Dh)

Example command: $016(cr)
response: !013F(cr)
The command requests the module at Address 01h to return the enable/disable statuses of all analog input channels. The value "3F" (hex) = "0011" and "1111" (binary), meaning that Channels 0~5 are all enabled.

#aa

Name Read all analog input values

Description Returns the input values of all analog input channels

Syntax #aa(cr)
is a delimiter
aa (range 00~FF) is the 2-character hex slave address of the specified module (always 01)
(cr) is the terminating character, carriage return (0Dh)

Response >(data)(data)(data)(data)(data)(data)(cr) if the command is valid
?aa(cr) if an invalid command was entered
There is no response if the module detects a syntax error, communication error, or if the address does not exist
> is a delimiter indicating a valid command was received
? is a delimiter indicating the command was invalid
(cr) is the terminating character, carriage return (0Dh)

Example command: #01(cr)
response: >+10.000+10.000+10.000+10.000+10.000+10.000

#aacc

Name	Read single analog input value
Description	Returns the input value of a specific analog input channel
Syntax	#aacc(cr) # is a delimiter aa (range 00~FF) is the 2-character hex slave address of the specified module (always 01) cc (range 00-05) specifies the channel you want to read the input data from (cr) is the terminating character, carriage return (0Dh)
Response	>(data)(cr) if the command is valid. ?aa(cr) if an invalid command is entered There is no response if the module detects a syntax error, communication error, or if the address does not exist > is a delimiter indicating a valid command was received ? is a delimiter indicating the command was invalid (cr) is the terminating character, carriage return (0Dh)
Example	command: #0103(cr) response: >+10.000 Analog Input Channel 3 of the module at Address 01h responds with an input value of +10.000.

$aaDcc

Name	Read single analog output startup value
Description	Returns the startup value of a specific analog output channel
Syntax	$aaDcc(cr) $ is a delimiter aa (range 00~FF) is the 2-character hex slave address of the specified module (always 01) D refers to the read single analog output startup value command cc (range 00-01) specifies the channel you want to read the startup value from (cr) is the terminating character, carriage return (0Dh)
Response	!aahhh(cr) if the command is valid ?aa(cr) if an invalid command was entered There is no response if the module detects a syntax error, communication error, or if the address does not exist ! is a delimiter indicating a valid command was received ? is a delimiter indicating the command was invalid aa (range 00~FF) is the 2-character hex slave address of the specified module (always 01) hhh (range 000~FFF) is the 3-character hex startup value of the specified analog output channel (cr) is the terminating character, carriage return (0Dh)
Example	command: $01DO1(cr) response: !01FFF(cr) Analog Output Channel 1 of the module at Address 01h responds with a startup value of +10.000 (i.e., the analog output range of Channel 1 is 0~10 V).

$aaDcchhh

Name Set single analog output startup value

Description Sets the startup value of a specific analog output channel

Syntax
$aaDcchhh(cr)
$ is a delimiter
aa (range 00~FF) is the 2-character hex slave address of the specified module (always 01)
D refers to the set single analog output startup value command
cc (range 00-01) specifies the channel you want to set the startup value of
hhh (range 000~FFF) is the 3-character hex startup value of the specified analog output channel
(cr) is the terminating character, carriage return (0Dh)

Response
!aa(cr) if the command is valid
?aa(cr) if an invalid command was entered
There is no response if the module detects a syntax error, communication error, or if the address does not exist
! is a delimiter indicating a valid command was received
? is a delimiter indicating the command was invalid
aa (range 00~FF) is the 2-character hex slave address of the specified module (always 01)
(cr) is the terminating character, carriage return (0Dh)

Example
command: $01DO1FFF(cr)
response: !01(cr)
The startup value of Analog Output Channel 1 of the module at Address 01h is set to +10.000 (i.e., the analog output range of Channel 1 is 0~10 V).

#aaccdd.ddd

Name Write single analog output value

Description Writes a value to a specific analog output channel

Syntax
#aaccdd.ddd(cr)
is a delimiter
aa (range 00~FF) is the 2-character hex slave address of the specified module (always 01)
cc (range 00-01) specifies the channel you want to write the output value to
dd.ddd (in engineering unit) is the analog output value of the specified analog output channel
(cr) is the terminating character, carriage return (0Dh)

Response
>(cr) if the command is valid
?aa(cr) if an invalid command was entered
There is no response if the module detects a syntax error, communication error, or if the address does not exist
> is a delimiter indicating a valid command was received
? is a delimiter indicating the command was invalid
aa (range 00~FF) is the 2-character hex slave address of the specified module (always 01)
(cr) is the terminating character, carriage return (0Dh)

Example
command: #010105.555(cr)
response: >(cr)
The value of Analog Output Channel 1 of the module at Address 01h is set to +05.555.

$aa7

Name	Read all digital input statuses
Description	Returns the values of all digital input channels
Syntax	$aa7(cr) $ is a delimiter aa (range 00~FF) is the 2-character hex slave address of the specified module (always 01) 7 refers to the read all digital input statuses command (cr) is the terminating character, carriage return (0Dh)
Response	!aa(data)(cr) if the command is valid ?aa(cr) if an invalid command was entered There is no response if the module detects a syntax error, communication error, or if the address does not exist ! is a delimiter indicating a valid command was received ? is a delimiter indicating the command was invalid aa (range 00~FF) is the 2-character hex slave address of the specified module (data) is a 2-character hexadecimal value representing the statuses of the digital input channels (cr) is the terminating character, carriage return (0Dh)
Example	command: $017(cr) response: !0101 (cr) This command requests the module at Address 01h to return the values of all digital input channels. Digital Input Channel 0 is ON and Channel 1 is OFF since the return value is 1 (01b).

#aaccdd

Name	Write digital output values
Description	Writes a value to one or all digital output channels
Syntax	#aaccdd(cr) # is a delimiter aa (range 00~FF) is the 2-character hex slave address of the specified module (always 01) cc specifies which channel(s) you want to set Write to all channels (byte): Both characters should be 0 Write to a single channel (bit): First character is 1 The second character indicates the channel number (0~1) dd is the hexadecimal representation of the digital output value(s) Write to all channels (byte): The decimal equivalent of these hex values represents the channel values Write to a single channel (bit): First character is always 0 The second character is either 0 or 1 (i.e., the digital output value)
Response	>(cr) if the command is valid ?aa(cr) if an invalid command was entered There is no response if the module detects a syntax error, communication error, or if the address does not exist > is a delimiter indicating a valid command was received ? is a delimiter indicating the command was invalid aa (range 00~FF) is the 2-character hex slave address of the responding module (cr) is the terminating character, carriage return (0Dh)

Example

command: #011101(cr)

response: >(cr)

An output byte with value 1 is sent to Digital Output Channel 1 of a module at Address 01h. Digital Output Channel 1 of the module is thus set to ON.

command: #010002(cr)

response: >(cr)

An output byte with value 02 (10b) is sent to the module at address 01h. Digital Output Channel 1 is set to ON; Channel 0 is set to OFF.

$aa2

Name

Read digital output values

Description

Read the value of all digital output channels

Syntax

$aa2(cr)
$is a delimiter
aa (range 00~FF) is the 2-character hex slave address of the specified module (always 01)
2 is a symbol of reading digital output

Response

!(cr) if the command is valid
?aa(cr) if an invalid command was entered
There is no response if the module detects a syntax error, communication error, or if the address does not exist
! is a delimiter indicating a valid command was received
? is a delimiter indicating the command was invalid
aa (range 00~FF) is the 2-character hex slave address of the responding module
(cr) is the terminating character, carriage return (0Dh)

Example

command: $aa2(cr)

response:!0101(cr)

An output byte with value 1(01b) is read from Digital Output Channel 0 of a module at Address 01h.
Digital Output Channel 0 is set to ON; Channel 1 is set to OFF. →

command: $aa2(cr)

response:!0102(cr)

An output byte with value 2 (10b) is read from the module at address 01h. Digital Output Channel 1 is set to ON; Channel 0 is set to OFF.

$aaBcc

Name	Read single analog input range code
Description	Returns the channel range code of a specific analog input channel
Syntax	$aaBnn(cr) $ is a delimiter aa (range 00~FF) is the 2-character hex slave address of the specified module (always 01) B refers to the single analog input range code command cc (range 00-07) specifies the channel you want to read the range code from (cr) is the terminating character, carriage return (0Dh)
Response	!aa(data)(code) if the command is valid ?aa(cr) if an invalid command was entered There is no response if the module detects a syntax error, communication error, or if the address does not exist ! is a delimiter indicating a valid command was received ? is a delimiter indicating the command was invalid (cr) is the terminating character, carriage return (0Dh) (code) is the range code (refer to the following tables)

ADAM-6024 Analog Input Channel Range Code

Range Code (Hex)	Range Code (Decimal)	Range Description
07	7	4~20 mA
08	8	±10 V
0D	13	0~20 mA

Example	command:	$01B01(cr)
	response:	!010D The range code of Channel 1 is 0D (hex), meaning "0~24 mA"

$aaCcc

Name	Read single analog output range code
Description	Returns the range code of a specific analog output channel
Syntax	$aaCnn(cr) $ is a delimiter aa (range 00~FF) is the 2-character hex slave address of the specified module (always 01) C refers to the read single analog output range code command cc (range 00-07) specifies the channel you want to read the range code from (cr) is the terminating character, carriage return (0Dh)
Response	!aa(data)(code) if the command is valid ?aa(cr) if an invalid command was entered There is no response if the module detects a syntax error, communication error, or if the address does not exist ! is a delimiter indicating a valid command was received ? is a delimiter indicating the command was invalid (cr) is the terminating character, carriage return (0Dh) (code) is the range code read (refer to the following)

ADAM-6024 Analog Output Channel Range Code

Range Code (Hex)	Range Code (Decimal)	Range Description
00	0	0~20 mA
01	1	4~20 mA
02	2	0~10 V

Example	command:	$01C01(cr)
	response:	!0102 The range code of Channel 1 is 02, meaning "0~10 V".

7.4.6 Digital I/O Command Set

(ADAM-6050, 6051, 6052, 6060, 6066)

Command Syntax	Command Name	Description
$aa6	Read all channel statuses	Returns the statuses of all channels
#aabb	Set digital output statuses	Sets the output status of one or all digital output channels
$aaJCFFFFssmm	Read digital input counter values	Returns the counter value of one or more digital input channels

$aa6

Name	Read all channel statuses
Description	Returns the statuses of all digital input channels
Syntax	$aa6(cr) $ is a delimiter aa (range 00~FF) is the 2-character hex slave address of the specified module (always 01) 6 refers to the read all channel statuses command (cr) is the terminating character, carriage return (0Dh)
Response	!aa00(data)(data)(data)(data)(cr) if the command is valid ?aa(cr) if an invalid command was entered There is no response if the module detects a syntax error, communication error, or if the address does not exist ! is a delimiter indicating a valid command was received ? is a delimiter indicating the command was invalid aa (range 00~FF) represents the 2-character hex slave address of the specified module (data) is a 2-character hex value representing the values of the digital input module (cr) is the terminating character, carriage return (0Dh)
Example	command: $016(cr) response: !01000FFD(cr) This command requests the module at Address 01h to return the values of all channels. The first 2-character portion of the response (excluding the "!" delimiter) indicates the module address; the second 2-character portion of the response is reserved (always 00). Characters 5~8 in the response give the values for Channels 15~12, 11~8, 7~4, and 3~0, respectively. In this example, the character "0" (hex) = "0000" (binary), meaning that Channels 15~12 are all OFF; the character "F" (hex) = "1111" (binary), meaning that Channels 11~8 and 7~4 are all ON; and the character "8" (hex) = "1101" (binary), meaning that Channel 0, 2, and 3 are ON whereas Channel 1 is OFF.

#aabb(data)

Name	Set digital output status
Description	This command sets a single or all digital output channels to the specific ADAM-6000 module.
Syntax	#aabb(data)(cr) # is a delimiter. aa (range 00~FF) is the 2-character hex slave address of the specified module (always 01) bb specifies which channel(s) you want to set Write to all channels (byte): Both characters should be equal to 0 (BB=00) Write to a single channel (bit): The first character is 1, the second character indicates the channel number (range 0h~Fh) (data) is the hex representation of the digital output value(s) Write to a single channel (bit): The first character is always 0, the second character is either 0 or 1 Write to all channels (byte): the binary equivalent of these hex values represents the channel values

Response >(cr) if the command is valid
?aa(cr) if an invalid command was entered
There is no response if the module detects a syntax error, communication error, or if the address does not exist
> is a delimiter indicating a valid command was received
? is a delimiter indicating the command was invalid
aa (range 00~FF) is the 2-character hex slave address of the responding module
(cr) is the terminating character, carriage return (0Dh)

Example command: #011201(cr)

response: >(cr)

An output bit with value 1 is sent to Channel 2 of the digital output module at Address 01h.

Channel 2 of the digital output module is thus set to ON.

command: #010012(cr)

response: >(cr)

An output byte with value 12h (00010010) is sent to the digital output module at Address 01h. Thus, Channels 1 and 4 will be set to ON, and all other channels will be set to OFF.

$aaJCFFFFssmm

Name Read digital input counter values

Description Returns the counter value of one or more digital input channels

Syntax $aaJCFFFFssmm(cr)
$ is a delimiter
aa (range 00~FF) is the 2-character hex slave address of the specified module (always 01)
JCFFFF refers to the read digital input counter values command
ss (range 00-07) specifies the start channel you want to read the counter value from
mm (range 00-07) specifies the number of channels you want to read the counter values from
(cr) is the terminating character, carriage return (0Dh)

Response >aa(data) if the command is valid
?aa(cr) if an invalid command was entered
There is no response if the module detects a syntax error, communication error, or if the address does not exist
> is a delimiter indicating a valid command was received
? is a delimiter indicating the command was invalid
(data) is the counter value
(cr) is the terminating character, carriage return (0Dh)

Example Command: $01JCFFFF0001(cr)

Response: >010000000A(cr)

This command requests the module at Address 01h to return the counter value from Channel 0 (the value "00" means that the first read channel is 0; the value "01" means that only one channel is read). The module returns the count value 0000000A(h) from Channel 0.

Example Command: $01JCFFFF0C02(cr)

Response: >010000000A00000001(cr)

This command requests the module at Address 01h to return the counter value from Channels 12 and 13 (the value "OC" means that the first read channel is 12; the value "02" means that two channels are read). That module return the count value 0000000A(h) from Channel 12 and 00000001(h) from Channel 13.

7.5 SNMP for ADAM-6000 Modules

ADAM-6000 (D version) supports SNMP v2c, which is useful for managing ADAM-6000 modules on your network. ADAM-6000 modules also support SNMP Trap v1. When the SNMP trap function has been enabled, ADAM-6000 modules will actively notify the management station of any I/O status changes by way of an SNMP message.

7.5.1 ADAM MIB file

Advantech offers ADAM MIB file which is designed based on standard SNMP format for ADAM modules and simplifies the integration with NMS (Network Management Software). By importing ADAM MIB file to NMS, ADAM-6000 can be monitored on the network easily.

Note: The object Identity(node) CounterObj for counter is only supported by the ADAM-6051 in counter mode.

7.5.2 SNMP Trap Configuration

SNMP trap functions can be configured using Adam/Apax .NET Utility or ASCII commands.

Host IP:	The management station IP where SNMP trap messages are sent to. Note: The trap function must be disabled before configuring the host IP.
Deadband:	To prevent traps from triggering excessively by noise, a deadband can be set as the minimum interval between the triggers of two traps.
Specific type:	Trap information contains details on the specific type of trap, which can be utilized to identify the event location. The specific type is shown in below table.

Specific Trap Types

Specific Type	Channel	Description
1	0	Digital Input Change
2	1	
3	2	
4	3	
5	4	
6	5	
7	6	
8	7	
17	0	Digital Output Change
18	1	
19	2	
20	3	
21	4	
22	5	
23	6	
24	7	

Specific Type	Channel	Description
131	0	Analog Input High Alarm
132	1	
133	2	
134	3	
135	4	
136	5	
137	6	
138	7	
141	0	Analog Input Low Alarm
142	1	
143	2	
144	3	
145	4	
146	5	
147	6	
148	7	

Configuration Using Adam/Apax .NET Utility

The host IP where SNMP trap messages are sent to can be configured using Adam/Apax .NET Utility V2.05.10 (B04) or later. Follow these steps to do so:

1. From the Module Tree Display Area, select the module you wish to configure
2. Click the **Stream/Trap** tab in the Status Display Area and then click the **SNMP Trap** tab
3. Ensure that the trap function is disabled (clear the check box beside the host IP)
4. Enter the host IP and click **Apply**
5. Enable the trap function (select the check box beside the host IP)
6. Set the deadband value and click **Apply change**

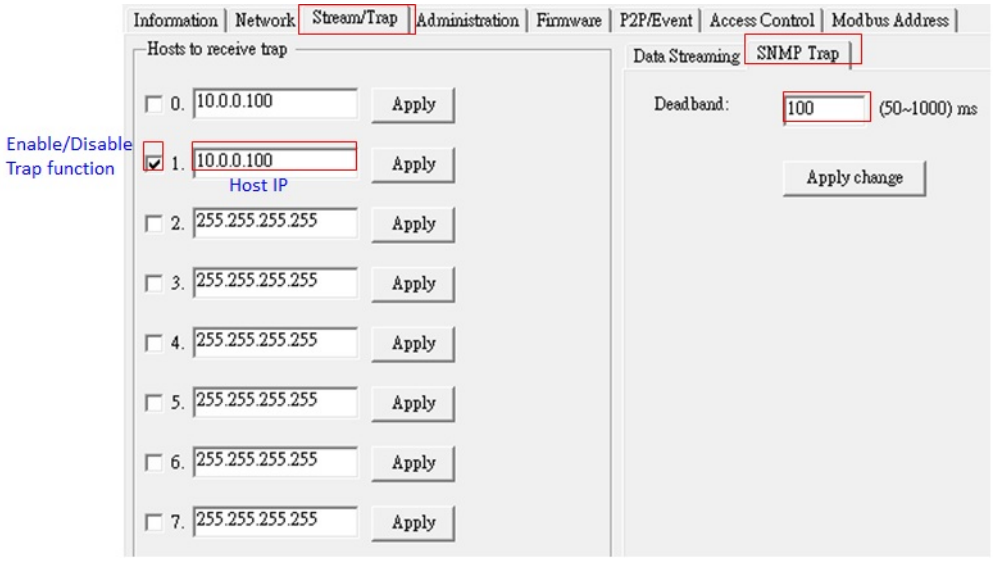


Figure 7.3 Trap Configuration Using Adam/Apax .NET Utility

Configuration Using ASCII Commands

Command Syntax	Description	Example
%01SETTRAPcR c: Host ID (0~7)	Enable trap	Situation: Enable Host 1 SNMP trap notifications Command: %01SETTRAP1R Response: >01
%01SETTRAPcP c: Host ID (0~7)	Disable trap	Situation: Disable host 1 SNMP trap notice Command: %01SETTRAP1P Response: >01
%01SETTRAPcaabbddee aabbddee: Host IP in hex c: Host ID (0~7)	Set the host IP	Situation: Set Host 1 IP to 10.0.0.100 Command: %01SETTRAP10A000064 Response: >01
%01SETTRAPST00000xxx xxx: ms (hex)	Set Trap deadband	Situation: Set Trap deadband to 100 ms Command: %01SETTRAPST00000064 Response: >01
%01GETTRAPc c: Host ID (0~7)	Get the IP of the host	Situation: Get IP address of host 1 Command: %01GETTRAP1 Response: !0A000064
%01GETTRAPST	Get trap deadband	Situation: Get trap deadband (100 ms) Command: %01GETTRAPST Response: !00000064
%01GETTRAPcSTU c: Host ID (0~7)	Get trap enable/disable flag	Situation: Get trap enable/disable status of Host 1 Command: %01GETTRAP1STU Response: !00 / !01 (00=disable,01=enable)

7.5.3 SNMP OID Value

By setting the value of the SNMP OID, you can set/get the information of the modules via SNMP, including module configuration, I/O value, and network information. You can get the setting information from the description in the MIB file.

aiRangeCode: OID for Setting the Analog Input Range

Value	Input range
143	±10 V
142	±5 V
140	±1 V
104	±500 mV
103	±150 mV
148	0~10 V
147	0~5 V
145	0~1 V
106	0~500 mV
105	0~150 mV
182	0~20 mA
180	4~20 mA
181	±20 mA

Note! *Please refer to the description in the MIB file regarding the settings for all the OIDs.*

7.6 MQTT for ADAM-6000 modules

7.6.1 Introduction of MQTT

The MQTT protocol is a lightweight messaging transport for IoT applications. Clients connect to a broker, which forwards MQTT messages. ADAM-6000 modules have several features that make them more flexible for IoT applications.

Feature 1: Actively Publish MQTT Message

ADAM-6000 modules can be set up to actively publish I/O data in the form of MQTT messages at user-defined intervals. This feature provides an efficient means for transmitting data and lowering the system loading.

Feature 2: Shorten Downtime by Active Alarm Event Notifications

Alarm events typically refer to a change in digital input status or when an analog input is within or exceeds a defined range. ADAM-6000 modules feature an alarm trigger mechanism that can issue immediate notifications. When an alarm condition is met, an MQTT message will be immediately published to the broker.

Feature 3: No Gateway is Needed to Handle the MQTT Protocol

In general applications, a gateway is needed to deal with MQTT protocol conversion, which can make the system structure unnecessarily complicated. ADAM-6000 modules feature built-in MQTT protocol handling capability and can transmit data directly to the broker via MQTT without a gateway, thus reducing the cost and effort involved in establishing MQTT applications.

7.6.2 MQTT Format for ADAM module

Digital I/O Modules: ADAM-6050-D, ADAM-6051-D, ADAM-6052-D, ADAM-6060-D, ADAM-6066-D

Description	MQTT Topic	JSON data	Firmware
Gets the I/O data of ADAM digital input/ output module	Advantech/MAC ID /data Example: Advantech/0013430C981C/ data	{"s":1,"t":0,"q":192,"c":1, "dix":DI status,"dox":DO status}	V6.01B11 or later
Set the value of a digital output of ADAM module	Advantech/MAC ID/ctl/dox Example: Advantech/0013430C981F/ctl/ do1	{"v":DO status},	V6.01B11 or later
Will Topic	Advantech/MACID/ Device_Status Example: Advantech/0013430C981F/ Device_Status	{ "status":"Device Status", "name":"Device Name","macid":"MACID" , "ipaddr":"IP Address"}	V6.01B13 or later

Note! *Refer to the following Index Settings section for an introduction to the items marked in blue.*

Analog Input modules: ADAM-6017-D

Description	MQTT Topic	JSON data	Firmware
Get the I/O data of an ADAM analog input module	Advantech/MAC ID /data Example: Advantech/0013430C981C/data	{"s":1,"t":time,"q":192,"c":1, "aix":AI value,"ai_stx":condition, "dox":DO status, "do_stx":condition}	V6.02B00 or later
Get an analog input range configuration	Advantech/MAC ID/cfg/sensor/aix Example: Advantech/0013430C981F/cfg/sensor/ai1	{"typ":"AI Range"}	V6.02B00 or later
Set the value of a digital output of an ADAM module	Advantech/MAC ID/ctl/dox Example: Advantech/0013430C981F/ctl/do1	{"v":DO status},	V6.02B00 or later
Set an analog input configuration	Advantech/MAC ID/set/sensor/aix Example: Advantech/00D0C9F94344/set/sensor/ai1	{"typ":"AI Range"}	V6.02B00 or later
Will topic	Advantech/MACID/Device_Status Example: Advantech/0013430C981F/Device_Status	{ "status":"Device Status", "name":"Device Name","macid":"MACID", "ipaddr":"IP Address"}	V6.02B00 or later

Thermocouple Input modules:ADAM-6018+-D

Description	MQTT Topic	JSON data	Firmware
Gets the I/O data of ADAM Analog input module[*1]	Advantech/MAC ID /data **Example:** Advantech/0013430C981C/data	{"s":1,"t":time,"q":192,"c":1, "aix":AI value,"ai_stx":condition, }	V6.01 B10 and later version
Set to trigger the I/O data update to broker (data update immediately)	Advantech/MACID/read/data **Example:** Advantech/0013430C981C/read/data	{"v":true}	V6.01 B10 and later version
Gets an analog input range configuration.	Advantech/MAC ID/cfg/sensor/aix **Example:** Advantech/0013430C981F/cfg/sensor/ai1	{"typ":"Thermocouple range"}	V6.01 B10 and later version

Set an analog input configuration	Advantech/MAC ID/set/sensor/aix **Example:** Advantech/00D0C9F94344/set/sensor/ai1	{"typ":"Thermocouple range"}	V6.01 B10 and later version
Will Topic	Advantech/MACID/Device_Status **Example:** Advantech/0013430C981F/Device_Status	{"status":"Device Status", "name":"Device Name","macid":"MACID", "ipaddr":"IP Address"}	V6.01 B10 and later version

Note! **1 Data updates to broker periodically according to the setting, so what you get from the broker with this topic is the data updated to the broker last time - may not be the same as current I/O data for modules. If you want to get the data updated immediately on broker. Use topic Advantech/MACID/read/data to trigger the module to update the current I/O data to broker.*

Universal I/O Module: ADAM-6024-D

Description	MQTT Topic	JSON data	Firmware
Gets the I/O data of ADAM digital input/output module	Advantech/MAC ID /data Example: Advantech/0013430C981C/data	{"s":1,"t":0,"q":192,"c":1, "dix":DI status,"dox":DO status}	V6.01 B06 or later
Set the value of a digital output of ADAM module	Advantech/MAC ID/ctl/dox Example: Advantech/0013430C981F/ctl/do1	{"v":DO status},	V6.01 B06 or later
Get the I/O data of an ADAM analog input module	Advantech/MAC ID /data Example: Advantech/0013430C981C/data	{"s":1,"t":time,"q":192," c":1, "aix":AI value,"ai_stx":condition, "dox":DO status, "do_stx":condition}	V6.01 B06 or later
Get an analog input range configuration	Advantech/MAC ID/cfg/sensor/aix Example: Advantech/0013430C981F/cfg/sensor/ai1	{"typ":"AI Range"}	V6.01 B06 or later
Set an analog input configuration	Advantech/MAC ID/set/sensor/aix Example: Advantech/00D0C9F94344/set/sensor/ai1	{"typ":"AI Range"}	V6.01 B06 or later
Get the I/O data of ADAM analog output module	Advantech/MAC ID /data Example: Advantech/0013430C981A/data	{"s":1,"t":0,"q":192,"c":1, "aox":AO value, "dix": DI status}	V6.01 B06 or later
Set the I/O data of ADAM analog output module	Advantech/MAC ID/ctl/aox Example: Advantech/0013430C981F/ctl/ao2	{"v":AO Value}	V6.01 B06 or later

Set an analog output configuration	Advantech/MAC ID/set/sensor/aox Example: Advantech/0013430C981F/set/sensor/ao2	{"typ":"AO Range"}	V6.01 B06 or later
Will Topic	Advantech/MACID/Device_Status Example: Advantech/0013430C981F/Device_Status	{ "status":"Device Status", "name":"Device Name","macid":"MACID", "ipaddr":"IP Address"}	V6.01 B06 or later

Index Settings

General

s:	Reserved for further use, default value = 1
t:	Trigger time Format: YYYY-MM-DDThh:mm:ss YYYY = year, MM = month, DD = date, hh = hour, mm = minute, ss = second Note: the function is not applied on ADAM-6050/6051/6052/6060/6066, t = 0
q:	Reserved for further use, default value 192
c:	Reserved for further use, default value 1
Device Status:	connect = module is connected disconnect = module is disconnected

Digital I/O Modules

dix:	Digital input status of channel (x-1) example: {"di2":true} means the status of Digital Input Channel 1 = true
dox:	Digital output status of channel (x-1) example: {"do2":true} means the status of Digital Output Channel 1 = true
DO status	true: on; false: off
DI status	true: on; false: off

Analog I/O Modules

aix:	Analog input value of channel (x-1) Note: If the analog input channel is disabled, the analog input value will be "9999.9999"
ai_stx:	Condition of analog input channel (x-1)

ai_stx value	**Condition**
0	Channel disable
1	Streaming, normal
2	High latch
3	High momentary
4	Low latch
5	Low momentary

do_stx

Condition of digital output channel (x-1) (ADAM-6017 only)

do_stx Value	Condition
1	Streaming, normal
2	DO change

Analog Input Range

The corresponding range values are for setting up the input range:

AI Range Command	Input Range
0-20mA	0~20 mA
4-20mA	4~20 mA
+-20mA	±20 mA
0-5V	0~5 V
1-5V	1~5 V
0-10V	0~10 V
0-1V	0~1 V
0-500mV	0~500 mV
0-150mV	0~150 mV
+-0V	±10 V
+-5V	±5 V
+-2.5V	±2.5 V
+-1V	±1 V
+-500mV	±500 mV

Thermocouple Range

The corresponding range values are for setting up the input range:

Thermocouple range command	Thermocouple range
K Type:0-1370C	Set type to K Type:0-1370°C
J Type:0-760C	Set type to K Type: 0-760°C
E Type: 0-1000C	Set type to K Type: 0-1000°C
T Type:-100-400C	Set type to K Type: -100-400°C
R Type:500-1750C	Set type to K Type: 500-1750°C
S Type:500-1750C	Set type to K Type: 500-1750°C
B Type:500-1800C	Set type to K Type: 500-1800°C

7.6.3 MQTT Configuration

The MQTT of ADAM-6000 modules can be configured using Adam/Apax.Net Utility (V2.05.11B17 or later) or ASCII commands.

Note: The MQTT function must be disabled before configuration and then enabled after the configuration has been completed.

Host (Broker IP)

You can set up the broker URL or IP address. ADAM-6000 modules connect to the broker via the standard MQTT protocol.

Heartbeat (Keep Alive)

The broker will regularly check the connection with ADAM-6000 modules at the interval defined by the heartbeat (Keep Alive) setting. The minimum interval is 5 s.

Deadband

A deadband is set to determine the minimum interval between publishing two MQTT messages. It is set to prevent MQTT message from being published excessively due to noise.

Retain Message

When the retain function is enabled, the broker will store the last message of the topic. If a new subscription for the topic is made, the message will be sent to the client. The client can get the last message immediately and does not need to wait until the next message is updated.

Will Topic

If the client subscribes to the topic for an ADAM-6000 module that is disconnected, the broker will inform the clients by sending the will message to whichever clients subscribe to the will topic

Will Topic of ADAM: Advantech/MACID/Device_Status

Will message: { "status":"Device Status", "name":"Device Name","macid":"MACID", "ipaddr":"IP Address"}

Example: {"status":"disconnect", "name":"ADAM6051", "macid": "00D0C9FEFFF5", "ipaddr": "10.0.0.1"}

QoS(Quality of Service)

Users can choose the QoS level of publish/subscribe. Three levels of QoS (Quality of Service) are defined in MQTT.

Level 0: broker/client deliver the message at most once

Level 1: broker/client deliver the message at least once

Level 2: broker/client deliver the message exactly once

Publish/Subscribe Topic

The MQTT message is forwarded by the broker based on the MQTT topic. Each message contains the data value. When client publish an MQTT message to the broker, the clients who subscribe the topic will receive the MQTT message accordingly.

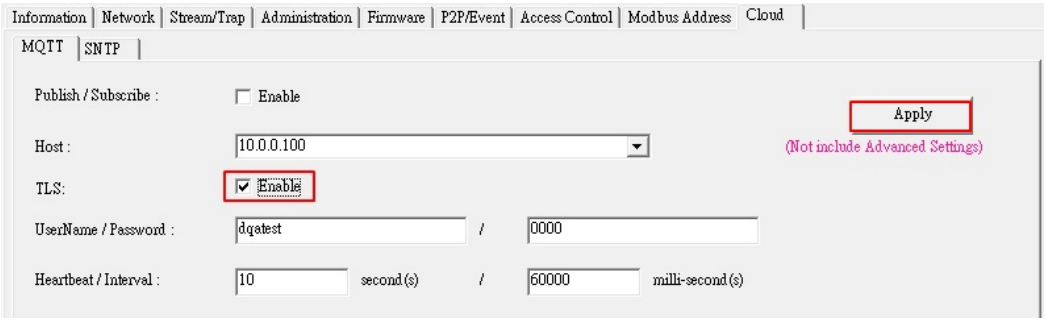
UserName/Password

For some applications that require authorization control, the broker will constrain the subscriber's authority to the data. For ADAM-6000 modules, the username/password can be set using Adam/Apax .NET utility. Then the MQTT message from ADAM-6000 modules will come with the username and password to access the broker

Model	Firmware version
ADAM-6050-D, ADAM-6051-D ADAM-6052-D, ADAM-6060-D ADAM-6066-D	V6.02B01 or later
ADAM-6017-D	V6.02B00 or later
ADAM-6018+-D	V6.01B10 or later

MQTT TLS Encryption

ADAM supports the TLS encryption during transmission. Enable the TLS function and click apply to implement the setting.

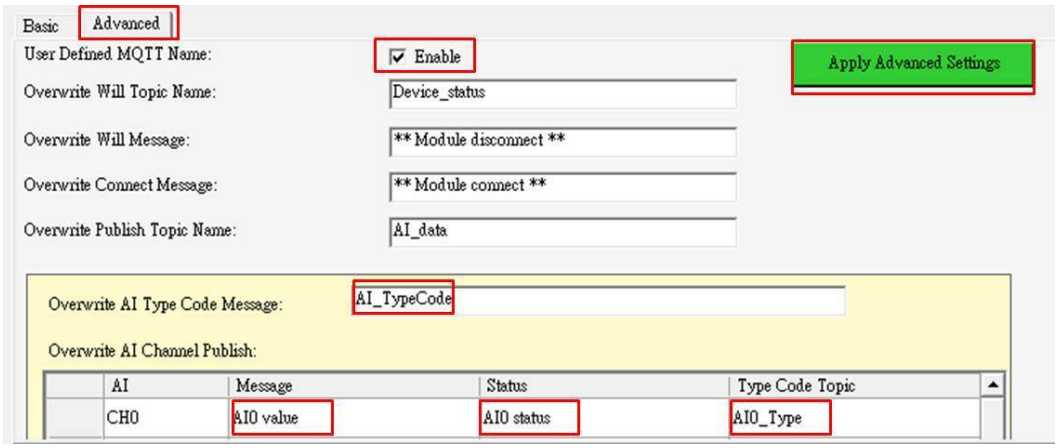


Model	Firmware version	Adam/Apax .NET utility version
ADAM-6017-D	V6.10 B07 or later version	2.05.11 B23 or later version
ADAM-6018+-D	V6.01 B10 or later version	2.05.11 B23 or later version
ADAM-6050-D, ADAM-6051-D, ADAM-6052- D,ADAM-6060-D, ADAM-6066-D	V6.10 B02 or later version	2.05.11 B23 or later version

User defined MQTT topic

ADAM supports the user defined topic, users can modify the default topic name by Adam/Apax .NET utility.

Model	Firmware version	Adam/Apax .NET utility version
ADAM-6017-D	V6.10 B07 or later version	2.05.11 B23 or later version
ADAM-6050-D, ADAM-6051-D, ADAM-6052- D,ADAM-6060-D, ADAM-6066-D	V6.10 B02 or later version	2.05.11 B23 or later version



Enable the user definded MQTT function

Overwrite Will Topic Name: change the will topic name

Overwrite Will Message: change the will message

Overwrite Connect Message: change the connect message when connecting

Overwrite Publish Topic Name: change the I/O value Topic

Overwrite AI Type code Message: change the AI type code title of analog input channel in MQTT message payload

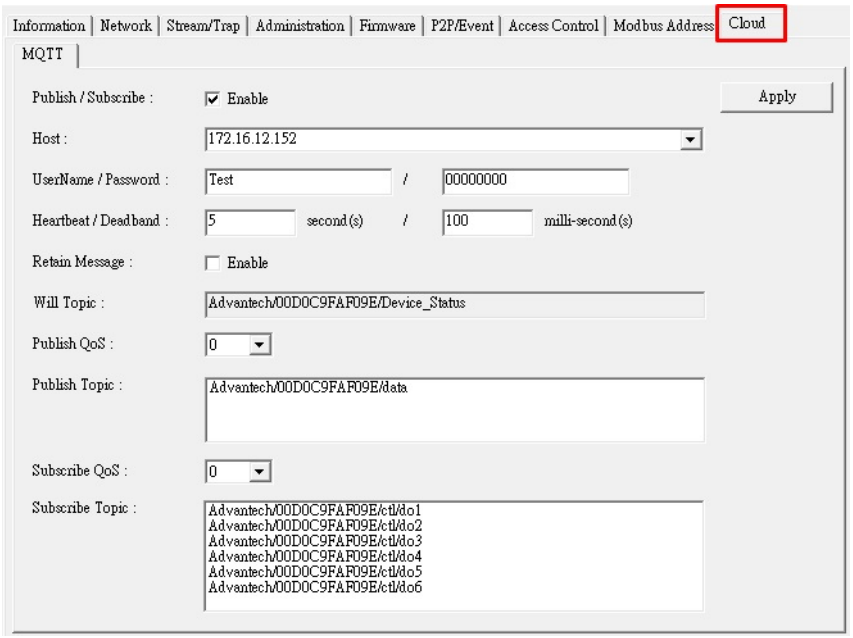
Overwrite the AI channel publish: Double click the field and change the I/O value Message, Status, Type Code Topic of each channel

Note!

1. Click the "Apply Advanced Settings" to make the setting take effect
2. Set all topic and message to default by disabling the user definded MQTT Function and click "Apply Advanced Settings"

Configuration Using Adam/Apax .NET Utility

Click the **Cloud** tab to configure the MQTT settings.



You can set up the broker URL or IP address in the **Host** box. Three public broker sources link are listed in the utility:

- iot.eclipse.org
- test.mosquitto.org
- broker.mqttdashboard.com

Configuration Using ASCII

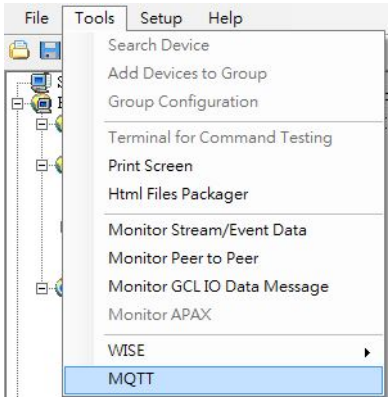
Command	Description	Remarks
%aaSETMQTTENxx	Set MQTT enable/disable aa: always 01 xx: 01 (enable) 00 (disable)	Return: >01 Error: ?01
%aaSETMQTTADxx…x	Set IP address of the broker aa: always 01 xx…x: IP address/domain (0~49 character)	Return: >01 Error: ?01
%aaSETMQTTHBxxxx	Set heartbeat interval aa: always 01 xxxx: heartbeat interval in second (0005~FFFF)	Return: >01 Error: ?01
%aaSETMQTTPDxxxx	Set publishing deadband aa: always 01 xxxx: publishing deadband in millisec-ond (0032~03E8)	Return: >01 Error: ?01
%aaSETMQTTPRxx	Set publishing retain enable/disable aa: always 01 xx: 01 (enable), 00 (disable)	Return: >01 Error: ?01
%aaSETMQTTPQxx	Set publishing Qos aa: always 01 (xx): publishing Qos (00~02)	Return: >01 Error: ?01

%aaSETMQTTSQxx	Set subscribing Qos aa: always 01 (xx): publishing Qos (00~02)	Return: >01 Error: ?01
ujhhhj%aaGETMQTTEN	Get MQTT enable/disable aa: always 01	Return: !01 (enable) !00 (disable) Error: ?01
%aaGETMQTTAD	Get IP address of the broker aa: always 01	Return: !IP Address/ Domain (IP Address/ DomainName) Error: ?01
%aaGETMQTTHB	Get heartbeat interval aa: always 01	Return: !xxxx (heartbeat interval in hex format) Error: ?01
%aaGETMQTTPD	Get publishing deadband aa: always 01	Return: !xxxx (deadband in hex format) Error: ?01
%aaGETMQTTPR	Get publishing retain enable/disable aa: always 01	Return: !00 (enable) !01 (disable) Error: ?01
%aaGETMQTTPQ	Get publishing Qos aa: always 01	Return: !xx (publishing Qos in hex format) Error: ?01
%aaGETMQTTSQ	Get subscribing Qos aa: always 01	Return: !xx (subscribing Qos in hex format) Error: ?01
%aaSETMQTTUNxx…x	Set MQTT user name aa: always 01 xx…x: user name, if set null module will disable the username and password function (0~49 character)	Return: >01 Error: ?01
%aaSETMQTTPWxx…x	Set MQTT password aa: always 01 xx…x: password, if set null module will disable the username and password function (0~99 character)	Return: >01 Error: ?01
%aaGETMQTTUN	Get MQTT user name aa: always 01	Return: !UserName Error: ?01
%aaGETMQTTPW	Get MQTT password aa: always 01	Return: !Password Error: ?01

7.6.4 How to Start MQTT with ADAM-6000 Modules

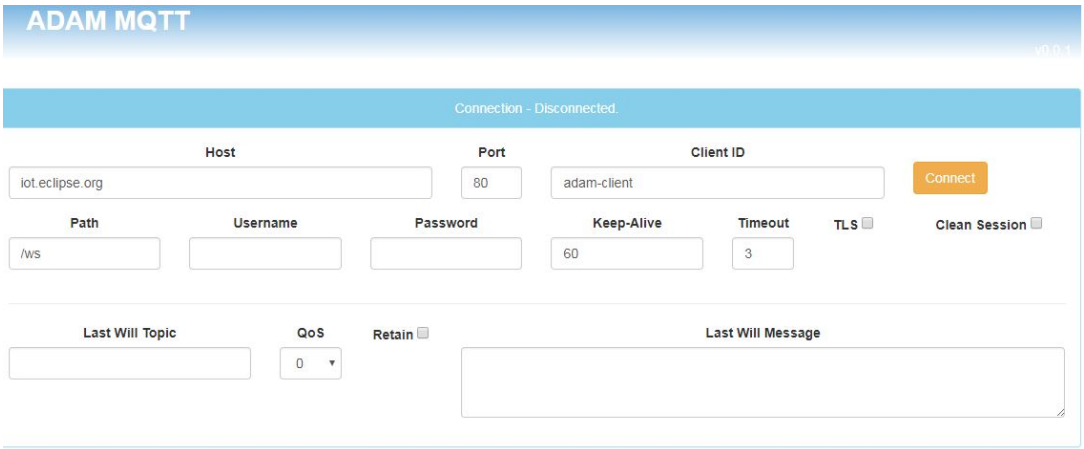
Adam/Apax .NET utility (V2.05.11 or later) provides pages to simulate MQTT client in order to test the MQTT function of ADAM modules. You can thus experience the benefits of ADAM modules with MQTT in four steps.

1. Select **MQTT** from the **Tools** menu; this will forward you to the **ADAM MQTT** page



2. Set up the connection

The **Connection** configuration page allows you to set up the client information. The default host is the public broker source "iot.eclipse.org" at Port 80. You can also set up the host URL or IP address. Click **Connect** once you have completed the configuration.



Note! *1. Path, Username, Password, TLS, and Clean Session functions are not released.*

2. The web page only supports the connection to the broker over WebSocket.

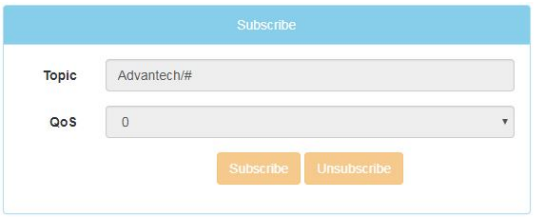
3. Set up the subscribe/publish function

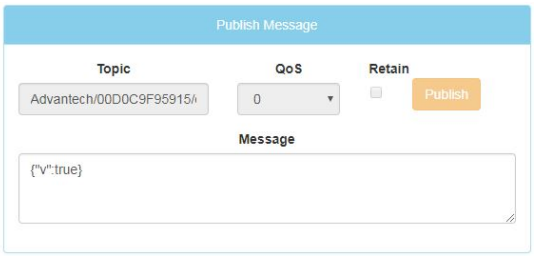
Subscribe

You will need to set up the topic and choose the QoS level and then click Subscribe. The message of the topic will be shown in the history field.

Publish

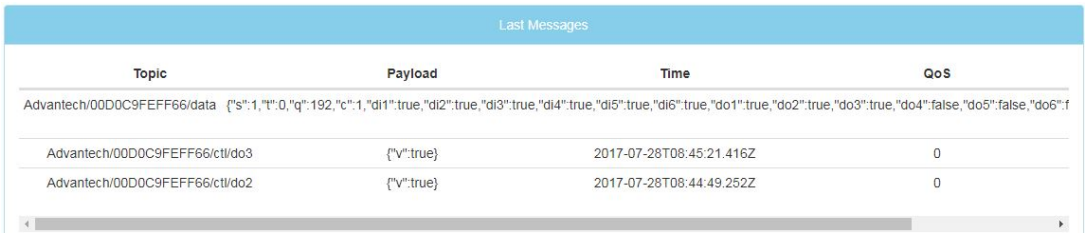
You will need to configure the publish topic settings, QoS, and message and then click Publish. The MQTT message will be published to the broker. If the retain function is enabled, your ADAM-6000 module will receive the last message when it subscribes to the topic.



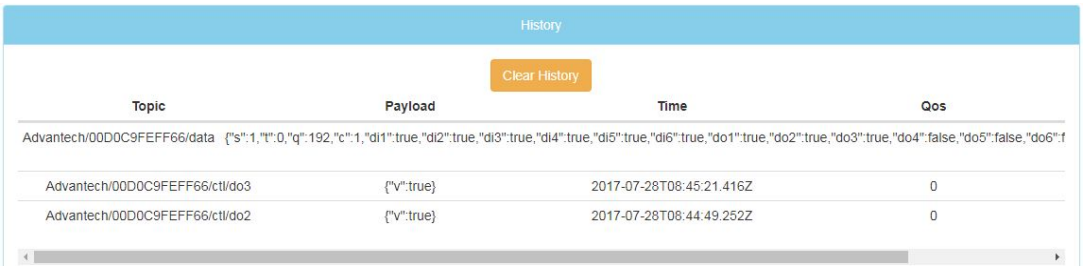


4. Review the MQTT message

You can read the last MQTT message and the historical messages in last message column and history column.



The above images shows the last message published by an ADAM module.



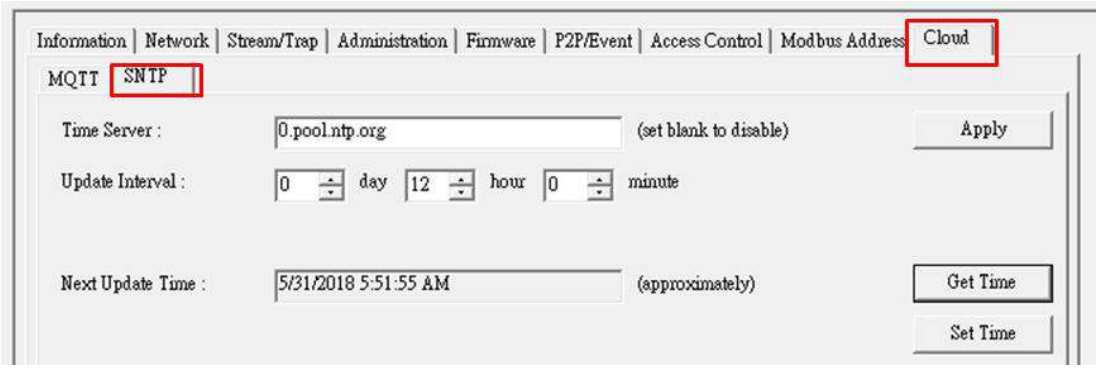
The above image shows the history message of an ADAM module.

7.6.5 Real-Time Clock

The ADAM-6017 (D version or later) supports real-time clock (RTC) in UTC format. Leveraging the RTC, you can determine the timestamp for data or an event. The RTC can be calibrated via SNTP. The SNTP settings can be configured using Adam/Apax .NET Utility or ASCII commands.

7.6.6 SNTP Configuration Using Adam/Apax .NET Utility

From Version 2.05.11 B16 of Adam/Apax .NET Utility and onward, the SNTP settings can be configured from the **Cloud** tab.



Time Server

Set up the SNTP server to synchronize with the RTC of the target module. If the Time Server entry is left blank, the function will be disabled. Click **Apply** when the configuration has been completed.

Update Interval

The RTC of the target module will synchronize with the time server at the interval time based on the updated interval configuration. Click **Apply** when the configuration has been completed

Next Update Time

This field shows the time at which the RTC will synchronize with the SNTP server. When you click **Get Time**, the utility will get the UTC of the ADAM module and convert it into local time based on the time zone of your computer. When you click **Set Time**, the RTC of the module will synchronize with the UTC of your computer.

7.6.7 SNTP Configuration Using ASCII Commands

Description	Command	Remark
Set SNTP server	%01SETSNTPADxxxxxxxx XXXXXXXX: the SNTP server domain or IP	Return: >01 Error: ?01
Get SNTP server	%01GETSNTPAD	Return: the SNTP server domain or IP exam-ple:!0.pool.ntp.org\r the SNTP server at 0.pool.ntp.org
Set SNTP update interval	%01SETSNTPPTxxxxxxxx xxxxxxxx the update interval	Return: >01 Error: ?01
Get SNTP update interval	%01GETSNTPPT	Return: SNTP update interval in second example:!0000A8C0 the update interval is 12 hr (43200 s)
Set RTC time in UTC format	#01TMYYYY-MM-DDThh:mm:ssZ YYYY:year MM:month DD:day hh:hour mm:minute ss:second	Return: >01 Error: ?01
Get RTC time in UTC for-mat	$01TM	Return: the RTC time in UTC format Example: >01TM2018-05-02T05:54:03Z

Chapter 8

Graphic Condition Logic (GCL)

8.1 Overview

In a traditional DA&C system, the system is managed by a single controller. Remote I/O modules, such as ADAM-6000 modules, are only used to acquire data from sensors or to generate signals to control other devices/equipment. In such setups, a computer or controller, such as a PLC, is responsible for acquiring data from the input modules, manipulating the data, and then executing logic operations and processes according to the input data, after which output data are generated and transmitted to the output modules based on the logic decision.

The computer/controller and remote I/O modules form a complete control system in the same network. The complexity of logic operations and processes depends on the application, and the operations are implemented by the program written on the computer/controller. There many software applications that can be used to write programs. Examples include Microsoft Visual Studio (C language) for when a computer is used and RSLogix (Ladder language) for when a PLC controller is used.

In many applications, the logic operation and process are relatively simple and it would seem that it is unnecessary to use a computer or controller, which are seem too powerful for such a simple application. Now, ADAM-6000 modules with Firmware Version 4.x (or later) feature logic operation and process ability with GCL. This makes ADAM-6000 modules smart I/O modules that can act as a standalone control system.

You can define the logic operation and process rules in Adam/Apax .NET Utility and then upload the rules to the ADAM-6000 modules. Then ADAM-6000 modules will execute the logic rules to process different action according to the input conditions. With GCL enabled, the computer/controller used in traditional systems can be removed since the ADAM-6000 modules can handle their role.

The configuration environment for GCL in Adam/Apax .NET Utility is completely graphical, making it very easy and intuitive to complete the logic rule configuration. After completing the logic rule configuration and download, you can view the real-time execution situation and input values in Adam/Apax .NET Utility.

8.2 GCL Configuration Environment

You can configure all GCL-related settings by clicking the **GCL Configuration** item list in the Module Tree Display Area in Adam/Apax .NET Utility.

Note that only P2P or GCL can be enabled at one time; thus, if you have enabled P2P, when you click a GCL configuration item and launch the GCL configuration environment, you will find that it is disabled. Similarly, once you enable the GCL feature, the P2P function will be disabled. Figure 8.1 shows how the Status Display Area will appear after a GCL configuration item has been selected. Note the key areas in the figure: the GCL Menu, the Logic Rule Set Area, and the Individual Logic Rule Configuration Area.

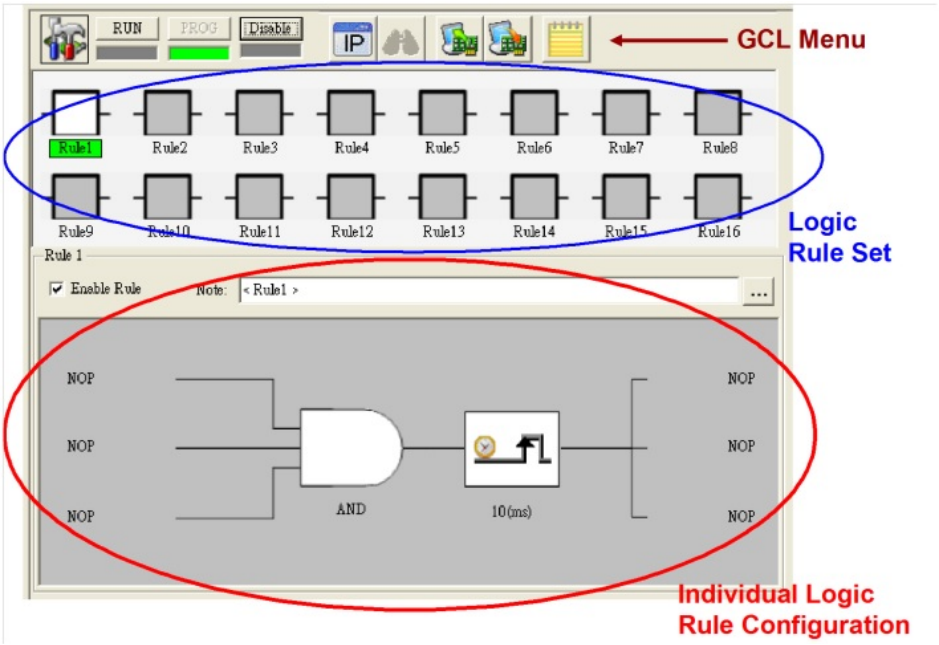


Figure 8.1 GCL Configuration Environment

GCL Menu Area

The icons in the GCL Menu Area are explained in the following table:

Icon	Function	Description
	Current Status	This icon shows the current GCL status. The status is either disable, programming, or running mode (from top to button). Note: You cannot enable peer-to-peer/datastream function and GCL function at the same time. So if you want to enable GCL, the peer-to-peer and datastream function will be disabled automatically.
	Run GCL	When running mode is selected, this button will be lit.
	Program GCL	When programming mode is selected, this button will be lit.
	Disable GCL	When disable mode is selected, this button will be lit.
	IP Table Configuration	Click this button to configure IP table, which can used to define the output destination. This is available only in programming mode.
	Monitoring	Click this to enable online monitoring. This is available only in running mode.
	Upload Project	Click this to upload a GCL configuration from the module to the computer. This is available only in programming mode.
	Download Project	Download the current GCL configuration to the module. This is available only in programming mode.
	Project Content	Click this show the current GCL configuration. You can also save the current configuration to file or load a previous configuration from a saved file.

Logic Rule Set Area

Each ADAM-6000 module can hold 16 logic rules, which are depicted as the 16 logic rule icons in the Logic Rule Set Area. To configure a rule, click on the corresponding logic rule icon; this will cause the text background to be highlighted green and the rule will then appear in the Logic Rule Configuration Area.

Logic Rule Configuration Area

Once you have selected a rule from the Logic Rule Set Area, select the **Enable Rule** check box to enable that rule; this will cause the icon for the selected logic to turn white. You can write a description for each logic rule by clicking the button next to the Note text box. Each rule comprises four stages: Input Condition, Logic, Execution and Output, as shown in the Individual Logic Rule Configuration Area in Figure 8.2. When you click on one of the stages shown in the figure, the corresponding configuration window will appear.

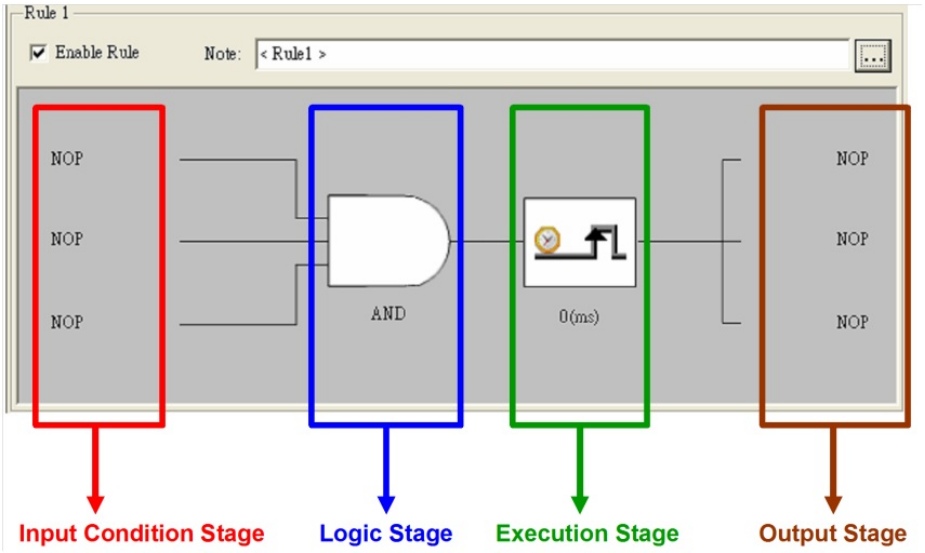


Figure 8.2 Four Stages for One Logic Rule

	Option	Description	Condition	Section
Input Condition Stage	NoOperation	No operation	N/A	8.3.1
	AI	Local analog input channel value	>=, <=, =	
	DI	Local digital input channel value	True, False	
	DI_Counter	Local counter input channel value	>=, <=, =	
	DI_Frequency	Local frequency input channel value	>=, <=, =	
	Timer	Local internal Timer value	>=	
	AuxFlag	Local internal Flag value	True, False	
	DO	Local digital output channel value	True, False	
	Counter	Local internal counter value	>=, <=, =	

	Option	Description	Section
Logic Stage	AND	AND operation	8.3.2
	OR	OR operation	
	NAND	NAND operation	
	NOR	NOR operation	

	Option	Description	Section
Execution Stage	Execution_Period	Define the execution time for this logic rule	8.3.3
	SendTo NextRule	Combine output of this logic rule to next logic rule to form a logic cascade	

	Option	Description	Section
Output Stage	NoOperation	No operation	8.3.4
	AO	Local or remote analog output channel value	
	DO	Local or remote digital output channel value	
	DI_Counter	Local or remote counter input channel setting	
	DO_Pulse	Local or remote pulse output channel setting	
	Timer	Local internal Timer setting	
	AuxFlag	Local or remote internal Flag value	
	RemoteMessage	Remote message	
	Counter	Local internal counter setting	

8.3 Configuring the Four Stages of a Logic Rule

8.3.1 Input Condition Stage

The input condition stage is a logic condition decision based on the input data. The resulting decision will be logic true or false, and this will be sent to the logic stage for logic operation. Take analog input mode as an example; the condition can be defined so that when the analog input value is greater than a specific value (the limit), the input stage will transfer "True" to the logic stage (otherwise, "False").

When you click the input condition stage icon, a window similar to Figure 8.3 will appear. You can select the input mode from the **Mode** box. The input mode options are as follows:

- NoOperation (default)
- AI (local analog input channel)
- DI (local digital input channel)
- DI_Counter (local counter input channel)
- DI_Frequency (local frequency input channel)
- Timer (internal timer)
- AuxFlag (internal flag)
- DO (local digital output channel)
- Counter (internal counter)

After you have selected the input mode and completed all related settings, click OK. That input condition stage icon will then change to represent the selected condition. Each mode is detailed in the following sections.

Input Mode: NoOperation

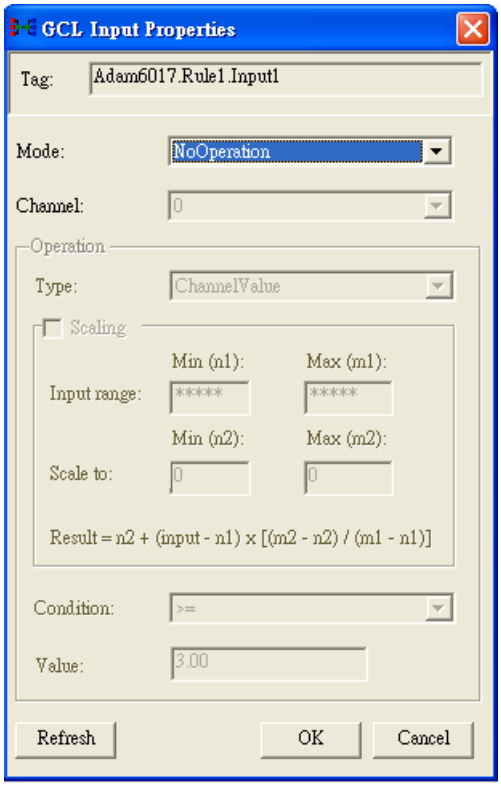


Figure 8.3 Input Condition Stage Configuration

The default mode is NoOperation. Under this mode, there is no input condition.

Input Mode: AI (Local Analog Input Channel)

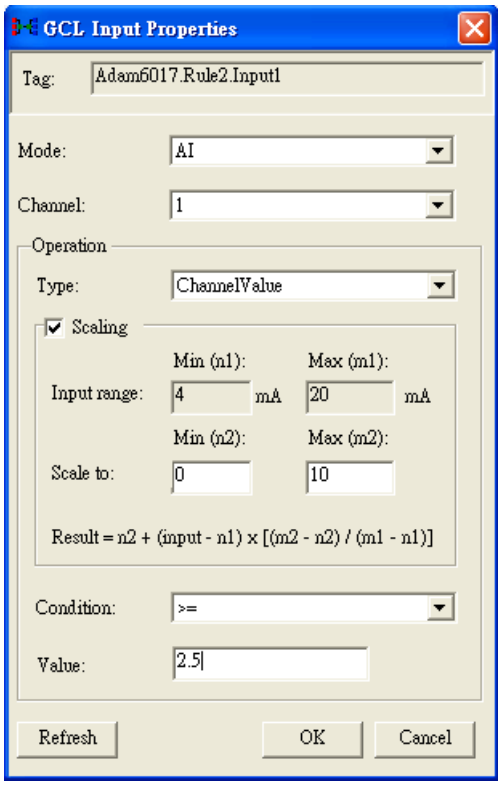


Figure 8.4 Scaling Function of Analog Input Mode

Follow these steps to configure the analog input condition:

1. Select **AI** from the **Mode** box and then select the channel you wish to configure from the **Channel** box.
2. Define the input condition operation in the **Operation** panel. To do this, select the analog input type from the **Type** box; this will be either **ChannelValue** (to use the current value of the selected analog input channel as the input for the condition) or **Deviation** (to use the deviation as the input for the condition; this is obtained by dividing the difference between the present and previous sample values by the total range).
3. Select the condition for the input channel from the **Condition** box and enter the threshold in the **Value** box. The examples in the following table illustrate how to define the analog input condition.

Channel	Type	Condition	Value	Description
0	ChannelValue	>=	5	If the value of Analog Input Channel 0 is more than or equal to 5, the condition result is "logic true" (otherwise, "logic false").
2	ChannelValue	=	3.2	If the value of Analog Input Channel 2 is equal to 3.2, the condition result is "logic true" (otherwise, "logic false").
3	ChannelValue	<=	1.7	If the value of Analog Input Channel 3 is less than or equal to 1.7, the condition result is "logic true" (otherwise, "logic false").
5	Deviation	N/A	20	If the deviation of Analog Input Channel 5 is greater than 20%, the condition result is "logic true" (otherwise, "logic false").

The analog input will read the voltage (or current) from the specified channel. Usually, the voltage (or current) value represents a real-world physical unit value (i.e., an engineer unit value) and there is typically a linear relationship between the voltage (or current) value and the engineer unit value. This linear relationship is depicted in the following figure:

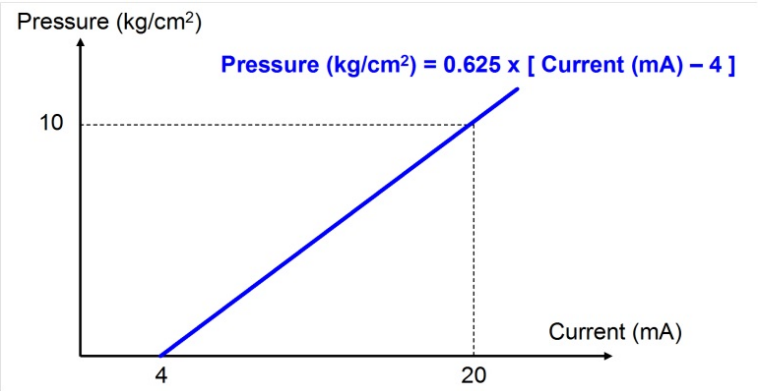


Figure 8.5 Engineer Unit and Current Value

ADAM-6000 analog input modules features a scaling function for converting voltage (or current) values into engineer unit values. For example, assume that the condition is if pressure value ≥ 2.5 kg/cm^2. Without the scaling function, you would need to convert the pressure value (2.5 kg/cm^2) to a current value (8 mA). Then, you would

have to enter the current value into the **Value** box in the **Operation** panel in order to define the condition.

Instead, you can use the scaling function, which is enabled by selecting the **Scaling** check box in the **Operation** panel. This allows you to enter the minimum and maximum values for the engineer unit in the **Min** (n2) and **Max** (m2) boxes of the **Scale to** item, and this establishes the relationship between the voltage (or current) value and the engineer unit value. In the example in Figure 8.4, the values 0 and 10 have been entered as the minimum and maximum pressure values. Since the module can automatically translate the pressure value to a current value, you need only enter the pressure value (2.5 kg/cm2) into the **Value** box in order to define the condition (note that only the value should be entered; the unit of measurement does not need to be included). This function allows you to configure the condition more intuitively without having to calculate the corresponding current value.

Input Mode: DI (Local Digital Input Channel)

Select **DI** as the input mode and then choose the channel you wish to configure from the **Channel** box. The input value of the selected channel will be used as the condition input. If the value is logic high, then the condition result is logic true; if the value is logic low, then the condition result is logic false.

Input Mode: Counter (Local Counter Input Channel)

Select **DI_Counter** as the input mode and then choose the channel you wish to configure from the **Channel** box. The count value of the selected channel will be used as the condition input. Similar to AI mode, you will need to select the appropriate condition for the selected input channel from the **Condition** box and enter the corresponding value in the **Value** box. The condition will compare the counter value from the input channel with the value in the **Value** box. If the condition is satisfied, the condition result is logic true (otherwise, logic false).

Input Mode: DI_Frequency (Local Frequency Input Channel)

Select **DI_Frequency** as the input mode and then choose the channel you wish to configure from the **Channel** box. The frequency value of the frequency input channel will be used as the condition input. Similar to counter mode, you will need to select the condition for the input channel from the **Condition** box and then ensure the corresponding value in the **Value** box. The condition will compare the frequency value from the input channel with the value you have entered in the **Value** box. If the condition is satisfied, the condition result is logic true (otherwise, logic false).

Input Mode: Timer (Internal Timer)

Each ADAM-6000 module has 16 local timers. After a timer starts, its value obviously represents how much time has passed. Under this input mode, you can use the timer value as the condition input. To do this, select Timer as the input mode and choose the appropriate timer from the **Index** box (range 0~15). Then, define the condition from the **Condition** box and enter the corresponding value in the **Value** box (increment: 0.01 s).

When the condition is met, the condition result is logic high. For example, if you choose >= from the **Condition** box and enter "500" in the **Value** box, this means that the condition result will remain logic low until the timer value is greater 5 s. After this point, the condition result will be logic high.

Input Mode: AuxFlag (Internal Flag)

Each ADAM-6000 module has 16 internal flags. The data type of an internal flag is digital, meaning that its value is either logic true or logic false. To use an internal flag value as a condition input, select **AuxFlag** as the input mode and then choose the appropriate internal flag from the **Index** box (range 0~15). Then, define the condition from the **Condition** box.

If you choose True in the **Condition** box, this means that when the internal flag value is logic true, the condition result will also be logic true. If you choose **False** in the **Condition** box, when the internal flag value is logic false, the condition result will be logic true.

You can use internal flag to implement a logic cascade or logic feedback. See Section 8.4 for more details about how to achieve this.

Note! *You can use other program applications to read/write internal flags via an ASCII command or Modbus/TCP address. See Section 7.4.2 and Appendix B.2 for further details.*

Input mode: DO (Local Digital Output Channel)

Select **DO** as the input mode and then choose the channel you wish to configure from the **Channel** box. The value of the selected digital output channel will be used as the condition input. If you select True from the **Condition** box, this means that when the value of the selected digital output channel is logic true, the condition result is also logic true. If you choose **False** from the **Condition** box, when the value of the selected digital output channel is logic false, the condition result will be logic true.

Input Mode: Counter (Internal Counter)

Each ADAM-6000 module has 8 internal counters, the values of which can be used as condition inputs. To do this, select **Counter** as the input mode and then choose the counter index from the **Channel** box (range 0~7). The count value of the selected internal counter will be used as the condition input. Similar to frequency input mode, you will need to select the condition for the counter from the **Condition** box and then enter the corresponding value in the **Value** box. The condition will compare the internal counter value with the value in the **Value** box. If condition is satisfied, the condition result is logic true (otherwise, logic false).

8.3.2 Logic Stage

When you click the logic stage icon, a window similar to Figure 8.6 will appear.

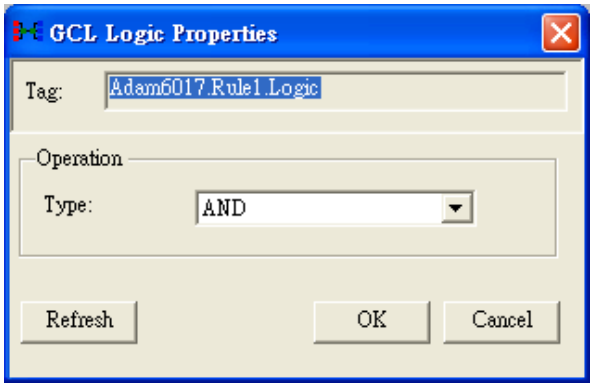


Figure 8.6 Logic Stage Configuration

For each logic rule, there will be up to three input conditions that pass logic true or false values to the logic stage. You can choose four logic operations from the **Type** box: AND, OR, NAND, NOR. The logic operation will process the input logic values and then generate a logic result that will be passed to the execution stage. After you have selected the appropriate logic operation, click **OK**. The logic stage icon will change to represent the current logic operation.

In the following text, truth tables are employed to present how the four logic operations work. in these examples, two input conditions are used. The letter "T" means logic true, and the letter "F" means logic false.

Logic Gate: AND

Input Condition 1	Input Condition 2	Logic Value Passed to the Execution Stage
F	F	F
F	T	F
T	F	F
T	T	T

Logic Gate: OR

Input Condition 1	Input Condition 2	Logic Value Passed to the Execution Stage
F	F	F
F	T	T
T	F	T
T	T	T

Logic Gate: NAND (not AND)

Input Condition 1	Input Condition 2	Logic Value Passed to the Execution Stage
F	F	T
F	T	T
T	F	T
T	T	F

Logic Gate: NOR (not OR)

Input Condition 1	Input Condition 2	Logic Value Passed to the Execution Stage
F	F	T
F	T	F
T	F	F
T	T	F

8.3.3 Execution Stage

When you click the execution stage icon, a dialog window similar to Figure 8.7 will appear. There are two possible execution setting you can choose from the **Type** box in the **Operation** panel; either **Execution_Period** (there is a delay before the logic result is passed to the output stage) or **SendToNextRule** (the input value is passed immediately to the next rule). Once you have selected the execution setting, click

OK. The execution stage icon will change to represent current setting. Each type is detailed in the following text.

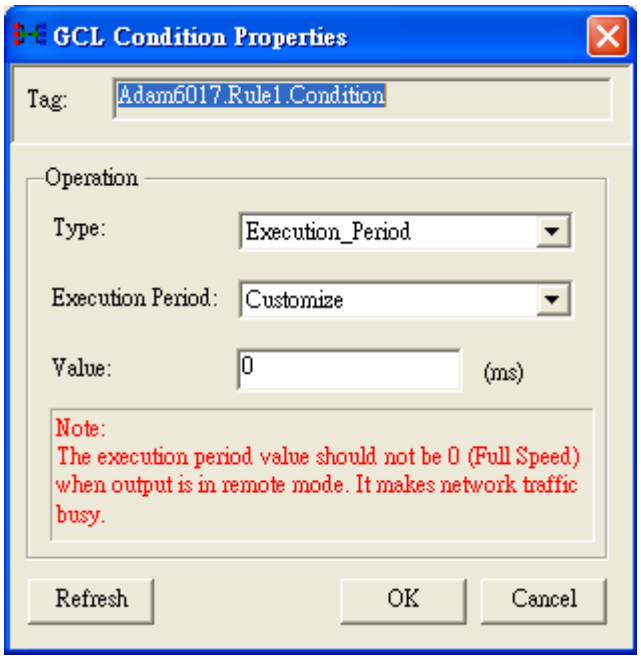


Figure 8.7 Execution Stage Configuration

Execution Type: Execution_Period

As mentioned, the logic stage will transfer the logic result (i.e., logic true or logic false) to the execution stage. The execution stage will then pass this value to the output stage after a specific period of time has expired. Follow these steps to configure the length of this period:

1. Select **Execution_Period** from the **Type** box.
2. Choose the appropriate period from the **Execution Period** box. You can select some a predefined period from 1 to 60000 ms. You can also select **Customize** to define the period by entering a value into the passed text (unit: ms).
3. Click **OK** to complete the configuration.

Note!

*1. If you choose **Full speed** from the **Execution Period** box, the execution speed will be as fast as possible. There might be network communication traffic problems when the data output is to another module (it may result in more packets being transferred than the receiving module can handle).*

2. When you want to use Adam/Apax .NET Utility to configure one ADAM-6000 module which is already running its GCL rules, remember to stop the GCL logic rules first.

Execution Type: SendToNextRule

This setting allows you to combine different logic rules into a single rule, which can help with building a more complex logic architecture. There are two methods for combining different logic rules: using the send to next rule function, and using an internal flag.

When you use the send to next rule function, the output of one logic rule is set to be the input of the subsequent logic rule; this means that it can only combine two logic rules which are consecutive and on the same module. If you want to combine differ-

ent logic rules that are not consecutive or on different modules, then you will need to use an internal flag for the logic rule cascade (this is introduced in Section 8.4).

When you select **SendToNextRule** in the **Type** box, one of the output icons will become the next rule. See Figure 8.8 for an example.

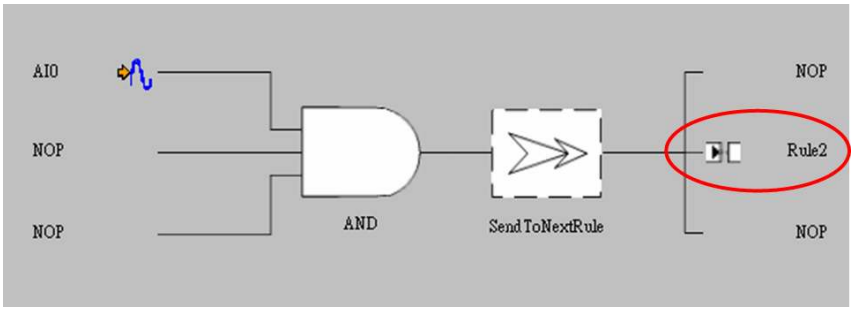


Figure 8.8 Send to Next Rule Function

When you click the next logic rule icon, you will notice that one of the input conditions is the previous logic rule (in Figure 8.9, "Rule1" now appears in the input stage). Therefore, the logic result from the previous logic rule will be one of logic input values of the current logic rule (in this example, "Rule2"). This combines the two neighboring logic rules, which is referred to as a logic cascade.

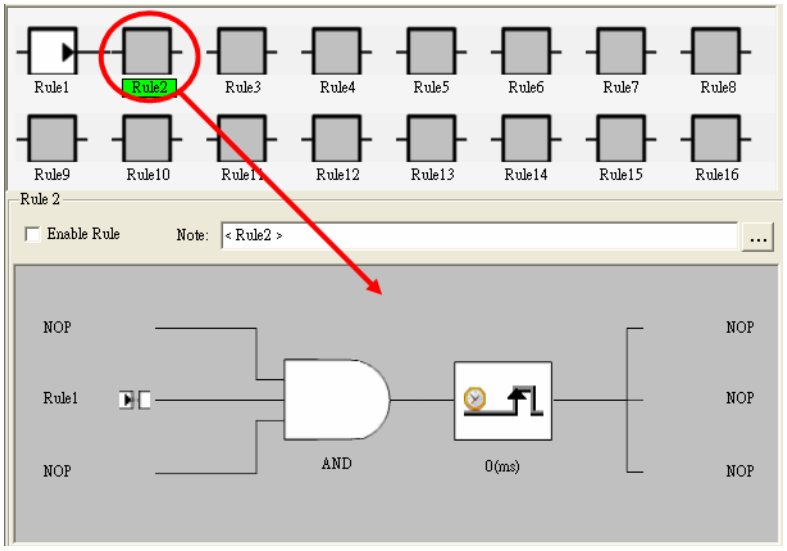


Figure 8.9 The Next Logic Rule

8.3.4 Output Stage

When you click the output stage icon, a window similar to Figure 8.10 will appear. There are three outputs per logic rule. The logic result from the execution stage will be passed to these three outputs. The action taken by the three outputs will depend on the logic result.

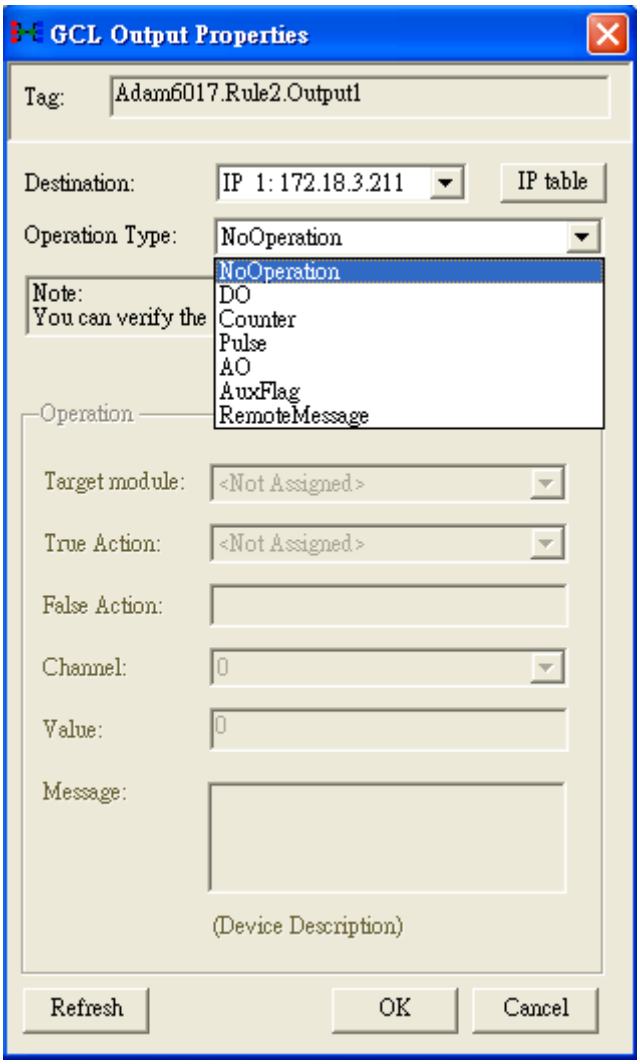


Figure 8.10 Output Stage Configuration

To configure the output stage, you will first need to select the address of the target device for the output from the **Destination** box. This defines where the output signal where be sent to. You can choose **Local** (meaning the output is on the same module) or another remote module by its IP address, which will be listed in the **Destination** box (note that the IP addresses are defined in the IP table, which you can configure clicking on **IP table** in this window or the **IP Table** icon in the GCL Menu Area). The name of the output module can be selected from the **Target module** box.

Note! *When your output destination is not local, remember to use an Ethernet switch to connect the ADAM-6000 module to the target device (do not use an Ethernet hub) in order to prevent data packet collisions.*

After deciding on the target device, you will need to choose the output action from the **Operation Type** box. The options are listed as follows:

- NoOperation (default)
- AO (analog output)
- DO (digital output)
- DI_counter (counter channel setting)
- DO_Pulse (pulse output)
- Timer (local timer)
- AuxFlag (local or remote internal flag)
- RemoteMessage (remote message output)
- Counter (local internal counter setting)

After you have chosen the output action, you will need to click **Verify** to confirm whether the target device exists and that it supports GCL (this does not apply to the NoOperation setting).

In the **True Action** box, you will be able to set the action taken when the logic result passed from execution stage is logic true. The **False Action** box defines the action taken when the logic result passed from the execution stage is logic false, and this will be automatically set according to the defined true action.

Once you have completed the configuration, the output stage icon will change to represent the current condition. The steps for configuring each output action are given in the following text.

Operation Type: NoOperation

This is the default setting. When this is selected, there is no output action.

Operation Type: AO (Analog Output)

Follow these steps to configure the analog output:

1. Select **AO** from the **Operation Type** box
2. Choose the target module from the **Target module** box (skip this step if **Destination** has been set to **Local**)
3. Select the appropriate output range from the **TargetRange** box
4. From the **Channel** box, set which channel will generate the output signal on the target device
5. Define the value that will generated by entering it in the **Value** box (the unit of the value will depend on the range in the **TargetRange** box)
6. Click **OK** to complete the configuration

Note! *You can view the action description by the **True Action/False Action** boxes. For a logic true result, the selected analog output channel will generate the new value that you have defined. For a logic false result, the output value of the selected analog output channel will remain unchanged.*

Operation Type: DO (Digital Output)

Follow these steps to configure the digital output:

1. Select **DO** from the **Operation Type** box
2. Choose the target module from the **Target module** box (skip this step if **Destination** has been set to **Local**)
3. From the **True Action** box, define whether to generate a true or false digital output signal for a true action (the false action will automatically be opposite to the true action)
4. From the **Channel** box, define which channel will generate an output signal on the target device
5. Click **OK** to complete the configuration

Operation Type: DI_Counter (Counter Channel Setting)

Follow these steps to configure the counter channel setting:

1. Select **DI_Counter** from the **Operation Type** box
2. Choose the target module from the **Target module** box (skip this step if **Destination** has been set to **Local**)
3. From the **True Action** box, define what action will be taken for a true action (i.e., when the logic result passed from the execution stage is logic true) by selecting **Start** (start the counter), **Stop** (stop the counter), or **Reset** (reset the counter)
4. From the **Channel** box, define which counter channel will take the defined action
5. Click **OK** to complete the configuration

Operation Type: DO_Pulse (Pulse Output)

Follow these steps to configure the pulse output:

1. Select **DO_Pulse** from the **Operation Type** box
2. Choose the target module from the **Target module** box
3. From the **True Action** box, define what true action will be taken (i.e., when the logic result passed from the execution stage is logic true) by selecting **Continue** (continuously generate a pulse train), **Stop** (stop generating pulses), or **Num of pulse** (generate a finite number of pulses). Note that the false action will always be **Keep current status**, meaning that there will be no action change for the selected digital output channel.
4. From the **Channel** box, define which digital output channel will take the defined action (start or stop pulse generation)
5. If **Num of pulse** has been selected, enter the number of pulses in the **Value** box
6. Click **OK** to complete the configuration

Operation Type: Timer (Local Timer)

Follow these steps to configure the timer:

1. Select **Timer** in the **Operation Type** box.
2. Choose which timer you want to configure from the Index box in the **Operation** panel (each ADAM-6000 module has 16 local timers; range 0~15)
3. Define the timer action from the **Type** box in the **Operation** panel by selecting **ON-Delay** (the timer will start when the logic result is logic true; by contrast, it will stop counting and reset its value to zero when the logic result is logic false) or **OFF-Delay** (this is the opposite of ON-Delay)
4. Click **OK** to complete the configuration

Operation Type: AuxFlag (Local or Remote Internal Flag)

Follow these instructions to assign the logic result from the execution stage to a local or remote internal flag:

1. Select **Auxflag** from the **Operation Type** box
2. From the **Index** box, choose the internal flag you wish to configure
3. From the **True Action** box, define the value you want to assign to the internal flag for the true action (the false action will be opposite to the true action)
4. Click **OK** to complete the configuration

Operation Type: RemoteMessage (Remote Message Output)

We can send the device description as message to the target device.

1. Select **RemoteMessage** from the **Operation Type** box
2. Give the message an index by entering a value in the **Value** box (when several logic rules send a message, it is important to specify which logic rule sends the message to the target device)
3. Enter the message you wish to be sent in the **Message** box
4. Click **OK** to complete the configuration

You do not need to set the **True Action** box for this operation type. When the logic result is logic true, the message will be sent to the target device. When the logic result is logic false, the message will not be sent.

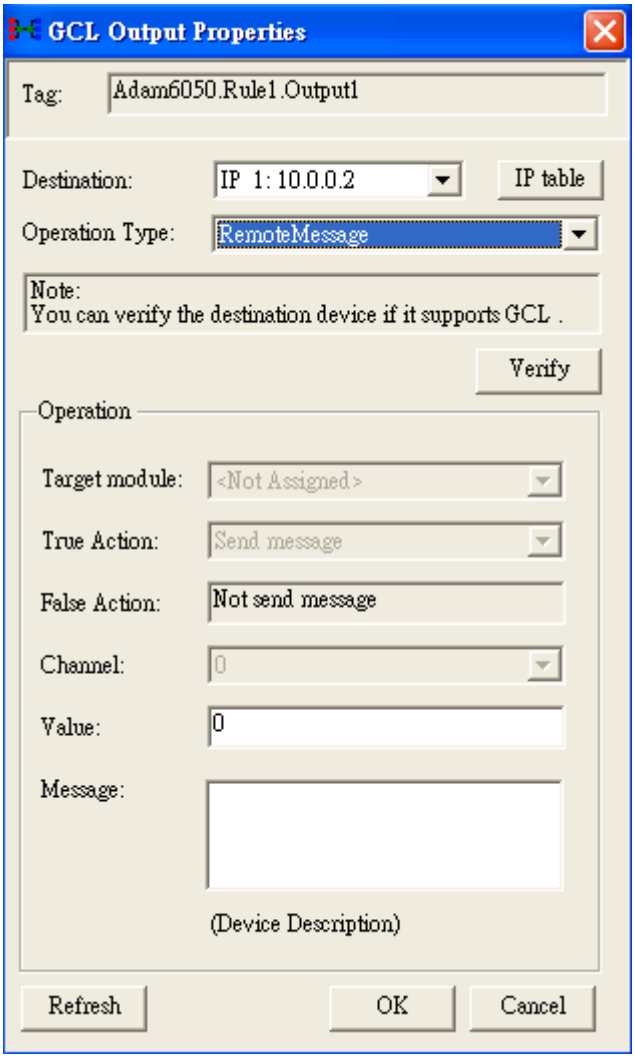


Figure 8.11 Remote Message Output

Note! *The transmitted message will comprise the **Message** box content (Device Description), the number of the logic rule sending the message, the message index, the module IP address, the module name, and all I/O statuses.*

Operation Type: Counter (Local Internal Counter Setting)

Follow these steps to configure the internal counter setting:

1. Select **Counter** from the **Operation Type** box
2. From the **True Action** box, define what action will be taken for the true action by selecting **Positive edge trigger (F→T)** (increment the internal counter by 1) or **Reset** (reset the internal counter).
3. The false action is displayed in the **False Action** box will automatically be opposite to the true action. Refer to the table below to see the relationship between true action and false action.
4. From the **Channel** box, define which counter channel will take the defined action
5. Click **OK** to complete the configuration

Note! *Note: When you choose **Positive edge trigger (F→T)** as the action, the counter will only add one count for the first time that the logic result from the execution stage is logic high. After the first time, the counter value will not change even if the logic result from the execution stage is still logic high. This is why it is called a positive edge trigger.*

The following table shows the true action and false action for different output actions:

Output Action	True action	False action
No Operation	Do nothing	Do nothing
AO	Change the analog output value	Keep current status
DO	Output true value	Output false value
	Output false value	Output true value
DI_Counter	Start counter	Stop counter
	Stop counter	Start counter
	Reset counter	Do nothing
DO_Pulse	Generate a continuous pulse train	Keep current status
	Generate a finite number of pulses	
	Stop generation pulses	
Timer	Start counting time	Stop timer and reset value to zero
	Stop timer and reset value to zero	Start timer
Internal Flag	Assign true value to a flag	Assign false value to a flag
	Assign false value to a flag	Assign true value to a flag
Remote Message	Send a message to the target device	Do nothing
Counter	Increment the counter by 1	Do nothing
	Reset counter	

8.4 Internal Flag for Logic Cascade and Feedback

8.4.1 Logic Cascade

Using an internal flag as an interface, you can combine different logic rules together to form a single logic rule for more complex logic architectures. Logic rules can be combined on the same module or even on different modules. Please refer to the examples in this section to understand how internal flags work.

Local Logic Cascade

Here, we take a simple example to describe the logic cascade. We use two analog input channels (Channels 0 and 1) of an ADAM-6017 to measure a signal from sensors. As long as either of the two input channels reads a voltage signal of 3~5 V, Digital Channel 0 will generate a logic high value. Otherwise, the channel will generate a logic low value. The logic architecture is depicted in Figure 8.12.

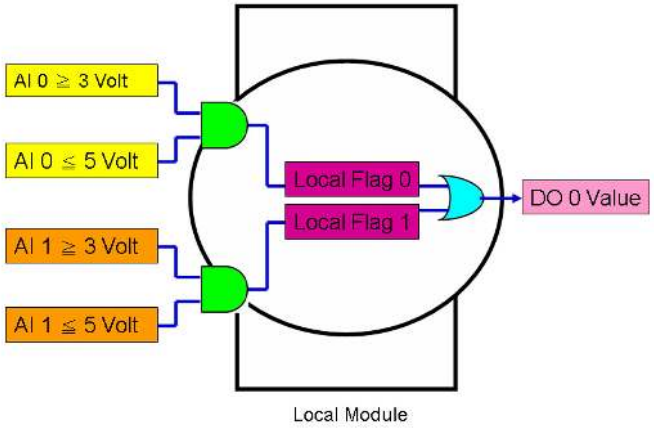


Figure 8.12 Local Logic Cascade Architecture

To implement this logic architecture, it is necessary to use three logic rules and two internal flags. Refer to Figures 8.13~8.15 for how to configure the three logic rules.

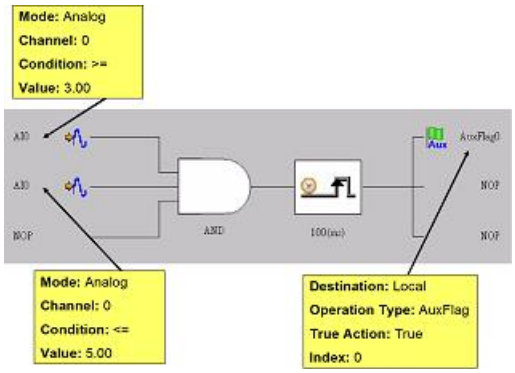


Figure 8.13 Configuration of Logic Rule 1

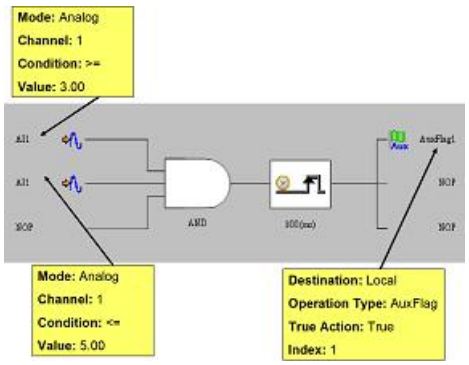


Figure 8.14 Configuration of Logic Rule 2

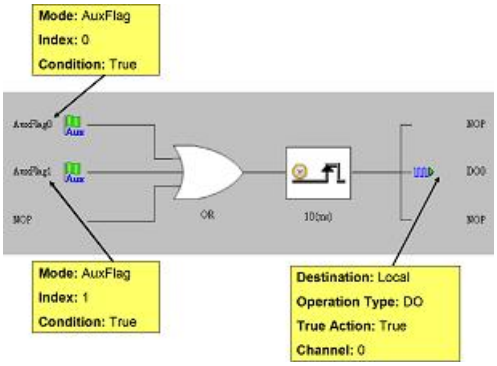


Figure 8.15 Configuration of Logic Rule 3

We use the Logic Rule 1 to check whether the value for Analog Input Channel 0 in the ADAM-6017 is within 3~5 V. Logic Rule 2 is used to check whether the value of Analog Input Channel 1 is within 3~5 V. The comparison result of Logic Rules 1 and 2 is assigned to Internal Flags 0 and 1. Logic Rule 3 reads the value of these two internal flags and uses the OR logic operation to define the output of Digital Output Channel 0. As shown in Figure 8.12, this logic architecture was built using internal flags.

Distributed Logic Cascade

Logic cascade functions are not limited to a single module. Since you can define the internal flag on another module, the logic cascade structure can be across different modules. Using the previous application as example, Figure 8.16 shows Logic Rules 1~3 running on Modules A~ C. The logic structure now spans three ADAM-6000 modules, and this is referred to as a distributed logic cascade. The configurations of the three logic rules are given in Figures 8.17~8.19.

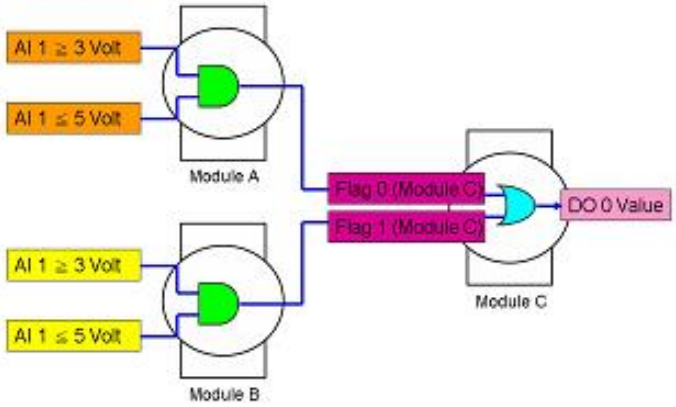


Figure 8.16 Distributed Logic Cascade

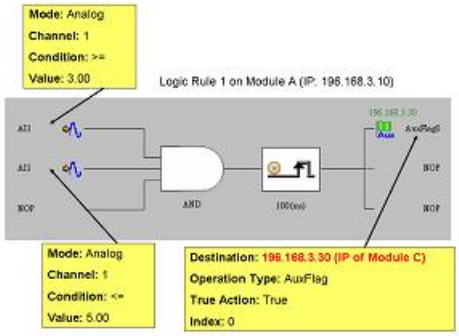


Figure 8.17 Configuration of Logic Rule 1

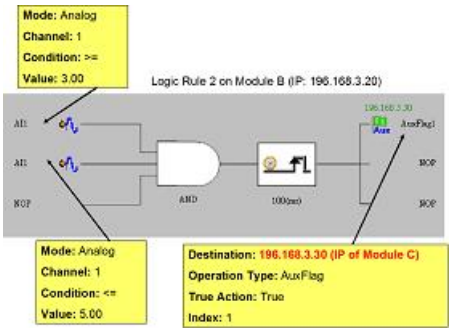


Figure 8.18 Configuration of Logic Rule 2

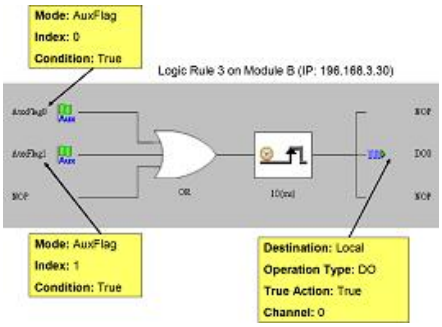


Figure 8.19 Configuration of Logic Rule 3

When a local or distributed logic cascade architecture is employed, there is no limitation for the input numbers of logic rules. This enables you to build any logic architecture to meet your application requirements.

8.4.2 Feedback

When you choose the same internal flag for the input condition and output of a single logic rule, the logic rule has logic feedback ability. In the example in Figure 8.20, one input condition and one output are dedicated to the same internal flag (AuxFlag 0). Thus, the output value in the current execution will become the input of the next execution. This gives this logic rule feedback ability.

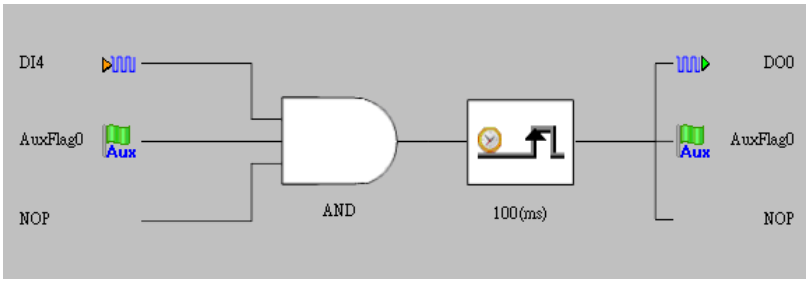


Figure 8.20 Building Logic Feedback

8.5 Logic Download and Online Monitoring

After you have completed all the configurations for GCL logic rules, click the **Download Project** icon in the GCL Menu Area in order to download the entire configuration to the target device. Then you can click the **Run GCL** icon to execute the project on the target module. You will see the current status switch to the **Running Mode** icon.

ADAM-6000 modules feature a special online monitoring function. In running mode, click the **Monitoring** icon in the GCL Menu Area to enable this function. When you do this, you will see the execution status in the Individual Logic Rule Configuration Area. Here, yellow dots indicate which stage the process flow is at. The current input value will also be shown beside the Input Condition Stage Area. Refer to Figure 8.21 for an example of the online monitoring function at work. In this example, the input conditions for DI 1 and DI 3 have been satisfied, and so the yellow dots appear beside the two input condition icons and you can the current input values are above the three input stage icons.

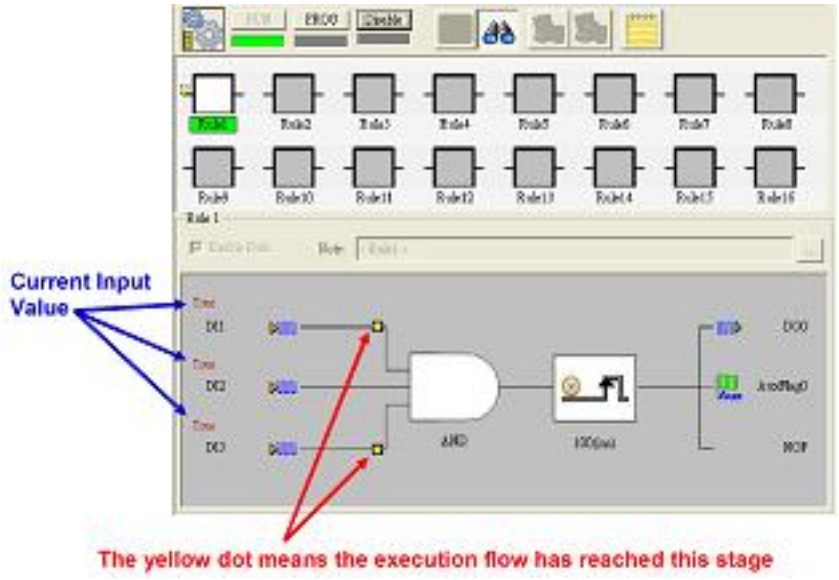


Figure 8.21 Online Monitoring Function

Note! *When you use internal flags (AuxFlag) as the GCL logic rule inputs, you can dynamically change the flag values in the online monitoring window of Adam/Apax .NET Utility. Simply double-click the input icons representing the internal flags and you will see that the flag values change from true to false (or vice versa).*

GCL Rule Execution Sequence

Each ADAM-6000 module has 16 logic rules. Figure 8.22 depicts the execution flow for one cycle. In this figure, there are three groups for one cycle: 1) input condition + logic, 2) execution, and 3) output. All the enabled rules at the input condition + logic stage will be executed first, followed by all the enabled rules at the execution stage, and finally all the enabled rules at the output stage (note that all rules at all stages will be executed in sequence).

For some advanced applications, you can combine different rules by adopting a logic cascade architecture (see Section 8.4.1). For example, the output of Rule 1 can be connected to the input of Rule 2 by assigning the same internal flag to both rules. Based on the aforementioned execution flow, the input condition + logic, execution, and output stages of Rule 1 will be executed sequentially. Therefore, the output of Rule 1 will be updated at the output stage in the first cycle, and the input of Rule 2 will detect the change in the output of Rule 1 in the next cycle.

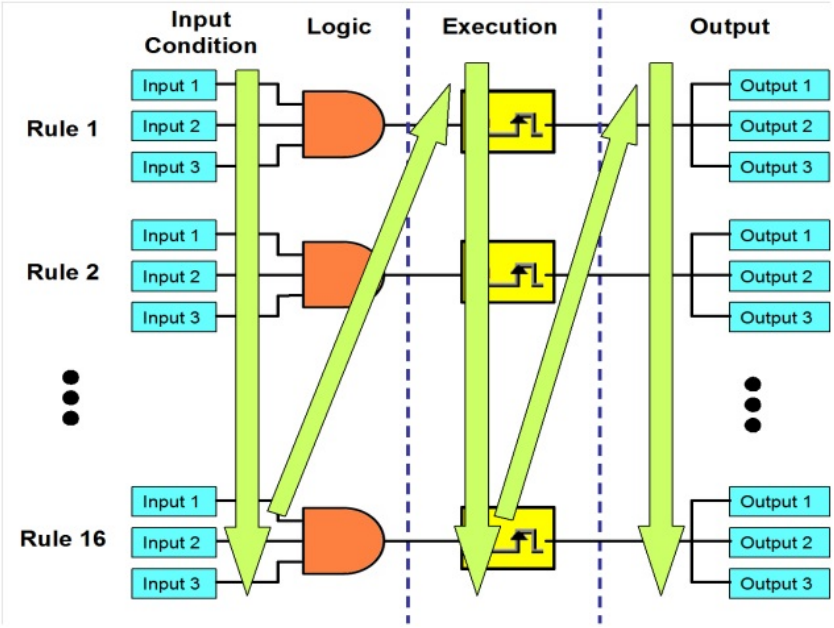


Figure 8.22 GCL Execution Sequence

GCL Execution and Data Transfer Performance

Local Output (Local Cascade)

Condition: Running one logic rule on one ADAM-6050 module

Processing time: <1 ms (this includes the hardware input delay time, execution time for one logic rule, and the hardware output delay time)

If multiple logic rules are used, the processing time can be estimated using the following equation:

Number of logic rules n ≤ 16

Approximate processing time (one cycle) = 600 + n x 370 (?s)

Remote Output (Distributed Cascade)

Condition: Running one logic rule on one ADAM-6050, with the output to another ADAM-6050 module via an Ethernet switch

Processing + communication time: <3 ms

8.6 Typical Applications with GCL

To shorten the GCL configuration time, Advantech provides several example project files for some typical applications. These application example files are included with the .NET Class Library and are available on your HDD after installation. The default location is as follows:

C:\Program Files\Advantech\AdamApax.NET Utility\Source\Example\ADAM-6000 GCL Example Project.

Simply load these files by clicking the **Project Content** icon on the GCL Menu Area. These example projects can be easily modified according to your application requirements. You can then download the modified project to your module and execute it. Each example project is detailed in the following text:

Empty Project

When you want to clear all configurations for GCL, the simple approach is to load this example project. Then you do not need to clear all the configurations manually.

On/Off Control (Two buttons to Control On and Off Separately)

In some automation applications, two digital inputs (e.g., DI 0 and DI 1) are used to control one digital output status (e.g., DO 0). The DO status will become logic high when DI 0 is logic high, and the DO status will return to logic low when DI 1 is logic high. For example, motor operation is controlled by two buttons: when the first button is pressed, the motor starts; when the second button is pressed, the motor stops. PLCs are typically used for this type of industrial automation application, and the ladder diagram will appear like that shown in Figure 8.23.

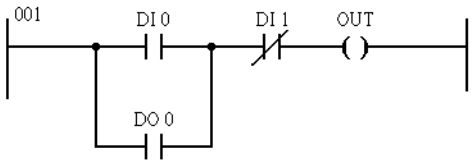


Figure 8.23 Ladder Diagram for On/Off Control

Now, GCL logic can be utilized to achieve the same control operation. For this, two logic rules are used. The complete logic architecture is shown in Figure 8.24.

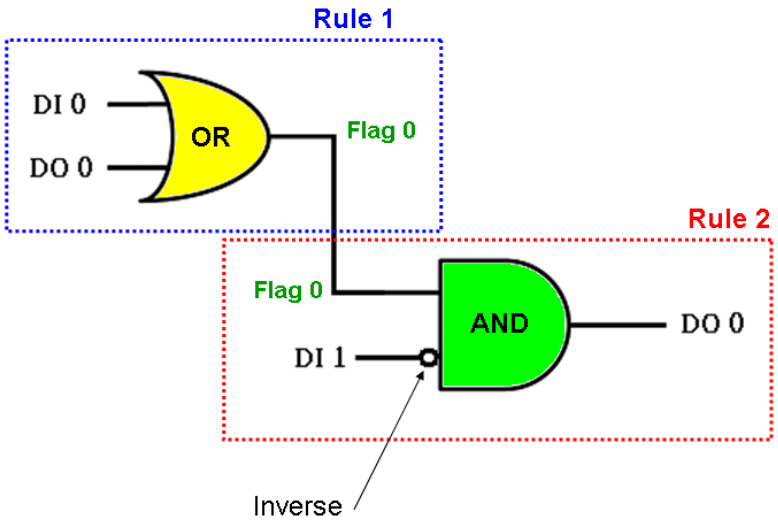


Figure 8.24 GCL Logic for On/Off Control

After you have loaded the example project file, you will find that it uses Rule 1 and Rule 2. One output of Rule 1 and one input of Rule 2 are assigned to the same internal flag: Flag 0. This can combine two or more logic rules together in a logic cascade (see Section 8.4.1). In this example, the condition for the DI 1 is false, and so the digital input value is inverted before entering the AND operator of Rule 2. The GCL logic architecture is similar to the PLC ladder diagram in Figure 8.23.

Sequential Control (Turn On in Sequence and Remain On)

In this type of application, several digital outputs will be activated in sequence and latch their values. In this example project, DO 0~DO 5 will sequentially be controlled to change their status. The time chart for this application is shown in Figure 8.25.

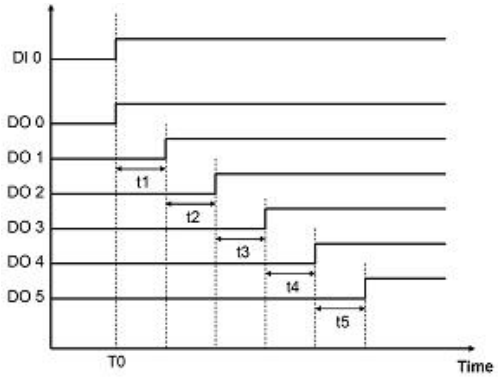


Figure 8.25 Time Chart for Sequence Control

(Digital Outputs are Turned On in Sequence and Remain On)

In the example project, DI 0 is used as a trigger to start the sequential control action. Therefore, when DI 0 becomes logic high (at T0), DO 0 will also become logic high immediately at that point. Then, DO1~DO5 will sequentially be activated to logic high after a specific time interval. You can decide the time interval for t1~t5 (they can be unique values). In this example project, t1~t5 are all set to 5 s.

Six logic rules and one internal timer can be employed for this GCL application. In the first logic rule, DI 0 is used to trigger Timer 0 and DO 0. Since the timer has been triggered, a counter will start and DO 1~DO 5 will be activated after a specific amount of time has elapsed. The GCL architecture is shown in Figure 8.26.

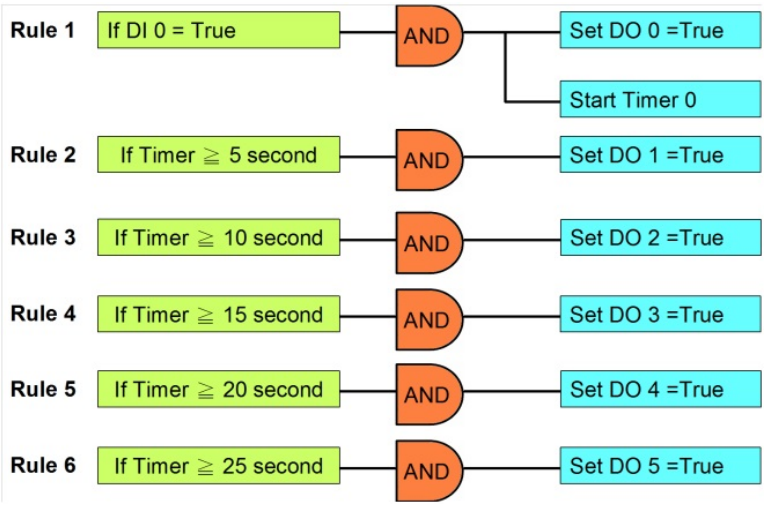


Figure 8.26 GCL Logic for Sequence Control (Turns On in Sequence and Remains On)

Multiple Digital Inputs to Control One Digital Output (12 digital Inputs to 1 Digital Output)

In many applications, only when multiple digital inputs are logic high (i.e., all related conditions are satisfied) will the digital output status become logic high. In this example project, only when DI 0~DI 11 are all logic high at the same time does DO 0 become logic high. The time chart of this application is shown in Figure 8.27. The green band area indicates the moment that all 12 digital inputs are logic high, at

which point DO 0 also becomes logic high. At all other time points, there is at least one digital input channel whose value is not logic high, and so DO 0 is logic low.

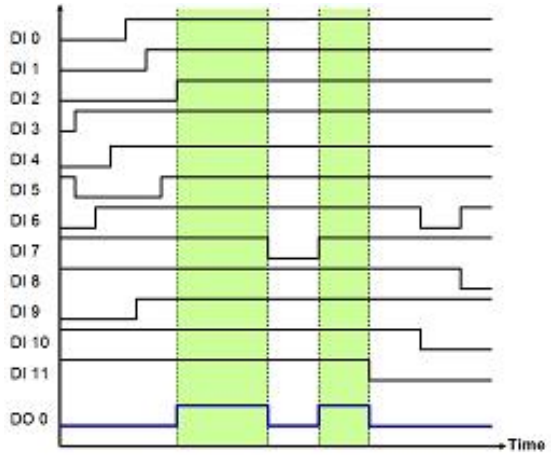


Figure 8.27 Time Chart for 12 Digital Inputs to 1 Digital Output

You can simply implement one AND logic operator to achieve this control system. However, since one logic rule only has three inputs, we need to implement a logic cascade to have 12 inputs. There are two ways to achieve a logic cascade:

1. Select **SendtoNextRule** at the execution stage of one logic rule. This will connect this logic rule to the subsequent logic rule (see Section 8.3.3)
2. Assign the output of one logic rule and input of another logic rule to the same internal flag, thus combining the two logic rules (see Section 8.4.1)

With the first method, the two logic rules must be consecutive. For example, Rules 1 and 2 can be combined together, but Rule 1 and 3 cannot. This limitation does not apply if you use the second method. Using the second method, you can even combine two rules on different modules. This example project adapts the first method for the logic cascade. The GCL logic architecture is shown in Figure 8.28.

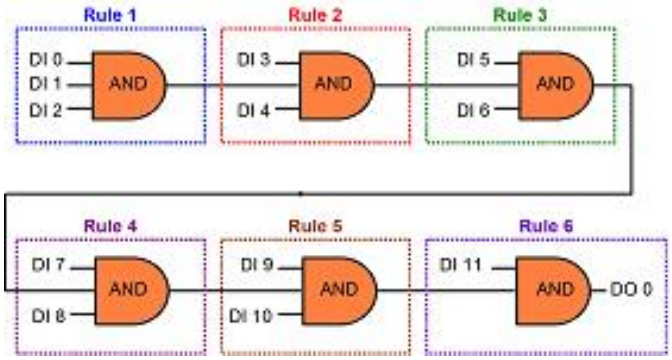


Figure 8.28 GCL Logic for 12 Digital Inputs to 1 Digital Output

Flicker

Flicker is commonly used in automation control applications. A typical example is making an alarm flash under the control of a digital output. This application requires a continuous pulse train to be generated by a digital output channel; all that needs to be determined is the period of the pulse train. The time chart for flicker applications is shown in Figure 8.29.

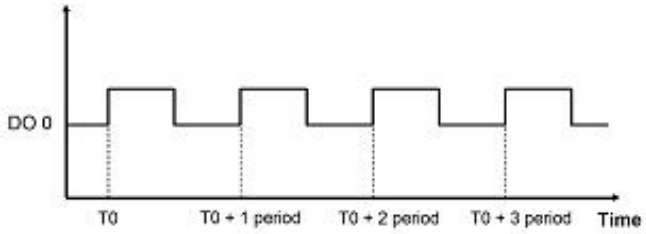


Figure 8.29 Time Chart for Flicker Applications

We need to use one internal flag (Flag 0) and two logic rules for this flicker application. In Logic Rule 1, the value of Flag 0 is periodically inverted (by choosing NAND at the logic stage). In this example, the period is 0.5 s, and this is defined by selecting Execution_Period at the execution stage (see Section 8.3.3). The status of DO 0 is controlled by Flag 0 in Logic Rule 2, and so DO 0 will change every 0.5 s. The GCL logic rule architecture is depicted in Figure 8.30.

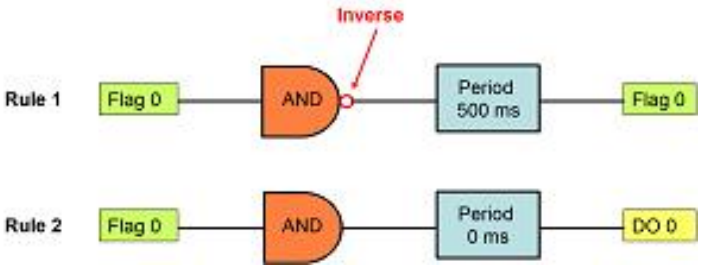


Figure 8.30 GCL Logic for Flicker

Rising Edge

For rising edge applications, the digital output status will be logic high when the digital input value changes from logic low to logic high (i.e., when a rising edge occurs). However, the digital output value will not continuously remain at logic high; instead, after a specific time interval (in the example, it is 1 s), the digital output value will return to logic low. The time chart for this example is shown in Figure 8.31.

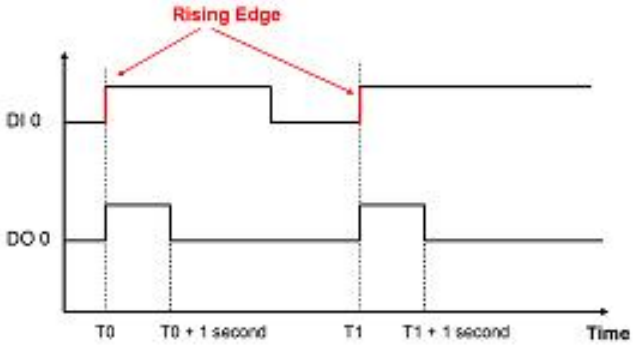


Figure 8.31 Time Chart for Rising Edge

This figure shows that DO 0 will only be triggered when DI 0 exhibits a rising edge. In this example project, the digital output status will remain logic high for 1 s, after which it will return to logic low. When a PLC is used for this type of application, the ladder diagram will look similar to that shown in Figure 8.32.

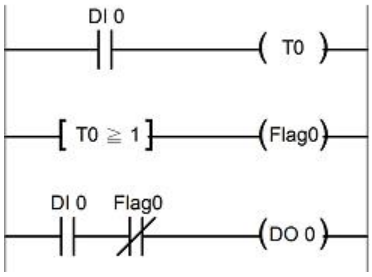


Figure 8.32 Ladder Diagram for Rising Edge

To use GCL for a rising edge application, you will need to use three logic rules, one internal timer (Timer 0), and one internal flag (Flag 0). Refer to Figure 8.33 for the GCL logic architecture. With Logic Rule 3, the value of DO 0 is controlled by DI 0 and Flag 0. Flag 0 is initially set to false.

When a rising edge occurs (i.e., the digital input value changes from logic low to logic high), the digital output will be activated (i.e., Logic Rule 3 is satisfied), and Timer 0 starts to count (Logic Rule 1 is satisfied). When Timer 0 reaches the specific time interval (1 s in this example), Flag 0 becomes logic true as a result of Logic Rule 2, making the value of DO 0 become logic low (i.e., Logic Rule 3 is not satisfied). The GCL architecture is similar to the ladder diagram in Figure 8.32.

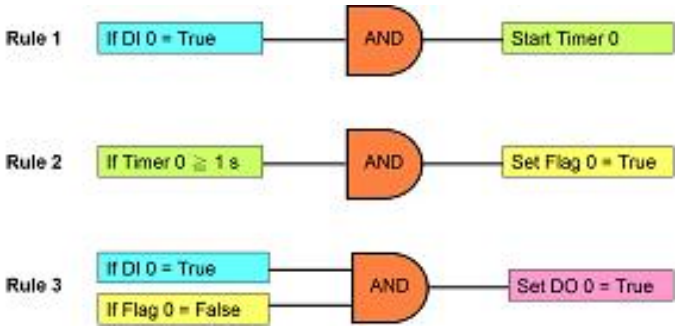


Figure 8.33 GCL Logic for Rising Edge

Falling Edge

For falling edge applications, the digital output value will be set to logic high when the digital input value changes from logic high to logic low (i.e., when a falling edge occurs). However, the digital output value will not continuously remain logic high. Instead, after a specific period (in the example, it is 1 s), the digital output value will return to logic low. Refer to Figure 8.34 for the time chart.

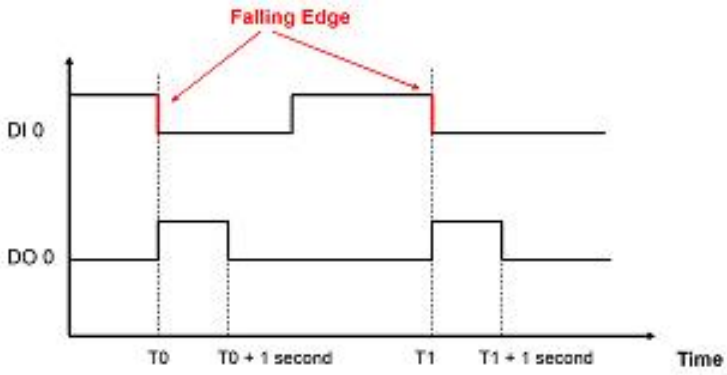


Figure 8.34 Time Chart for Falling Edge

The figure shows that DO 0 will only be triggered when DI 0 exhibits a falling edge. In this example project, the digital output status will remain logic high for 1 s before returning to logic low. When a PLC is used for this type of application, the ladder diagram will look similar to that shown in Figure 8.35.

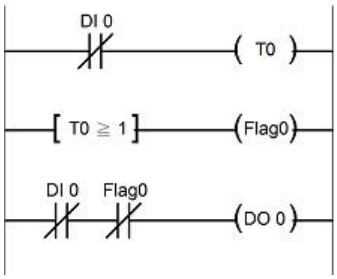


Figure 8.35 Ladder Diagram for Falling Edge

To use GCL for falling edge applications, you will need three logic rules, one internal timer (Timer 0), and one internal flag (Flag 0). Refer to Figure 8.36 for the GCL logic architecture. With Logic Rule 3, the value of DO 0 is controlled by DI 0 and Flag 0. Flag 0 value is initially set to logic false.

When a falling edge occurs (i.e., the digital input value changes from logic high to logic low), the digital output will be activated (i.e., Logic Rule 3 is satisfied), and Timer 0 starts to count (Logic Rule 1 is satisfied). When Timer 0 reaches the specific time (1 s in this example), Flag 0 will become logic true as a result of Logic Rule 2, making the value of DO 0 logic low (Logic Rule 3 is not satisfied). The GCL architecture is similar to the ladder diagram in Figure 8.35.

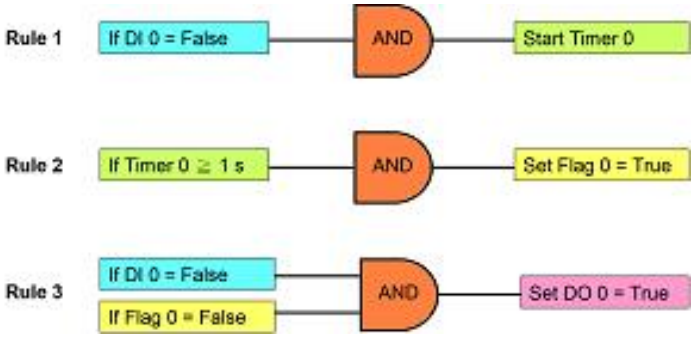


Figure 8.36 GCL Logic for Falling Edge

Sequential Control (Continuously Turn On and Off in Sequence)

This type of automation application is similar to the previous sequential control application (i.e., the third application introduced in this section), except that this application forms a continuous loop. The time chart for this application is shown in Figure 8.37. One time base is needed to control the digital output sequence. In this example, the period of the time base to turn off one digital output and turn on the subsequent digital output is 1 s.

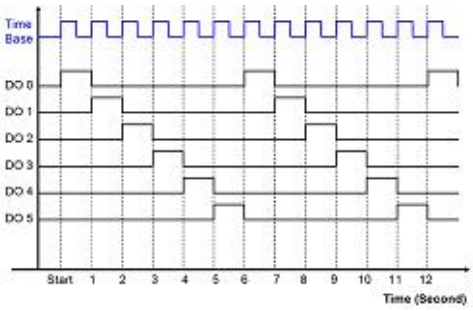


Figure 8.37 Time Chart for Sequence Control (Continuously Turn On and Off in Sequence)

This type of application requires nine logic rules, one internal counter (Counter 0), and one internal flag (Flag 0). In this example project, Logic Rules 1 and 8 are employed to create the time base. By Logic Rule 8, the value of Flag 0 will change every 0.5 s. In Logic Rule 1, once the value of Flag 0 is logic high, Counter 0 will increment by 1. Thus, every 1 s, Counter 0 will increase by 1, making Counter 0 the time base.

Logic Rules 9~14 are used to control DO 0~5. Which logic rule will be executed is based on the value of Counter 0. Since this value will continuously increment by 1 every 1 s, Logic Rules 9~14 will be executed in sequence every 1 s. Therefore, DO 0~DO 5 will be activated sequentially in 1-s intervals. When Logic Rule 15 is executed, Counter 0 will reset and its value will return to zero. This makes the logic rule execution to form a continuous loop. Refer to Figure 8.38 for the GCL architecture.

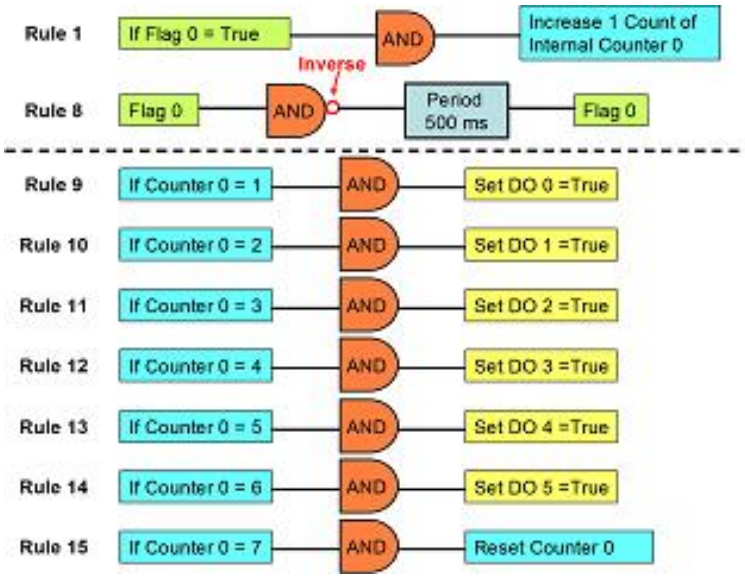


Figure 8.38 GCL Logic for Sequence Control (Continuously Turn On and Off in Sequence)

Digital Input Event Trigger (Only Occurs Once)

GCL can be employed to perform an event trigger. For this type of application, a digital input channel is used to trigger an action. The input condition of the GCL logic rule will be if the digital input value is logic true, and the output of the rule will be the desired action (in this case, transmitting a remote message). When the digital input value becomes logic true, the input condition is satisfied. The GCL logic rule will then send a message continuously until the digital input value returns to logic false. However, for this specific application, the message will only be sent once when the condition is satisfied rather than being sent continuously.

This type of application can be achieved by using one counter input channel. The GCL logic is depicted in Figure 8.39. Simply select a local counter input channel (DI_Counter) at the input condition stage for one logic rule. There are two outputs used for the same logic rule; one is to reset the counter input channel and the other is the desired action. Then, when the counter input channel detects a digital input signal, the condition is satisfied and the desired action will be executed. Concurrently, the GCL rule will reset the counter input channel, thus causing the desired action to be executed only once.

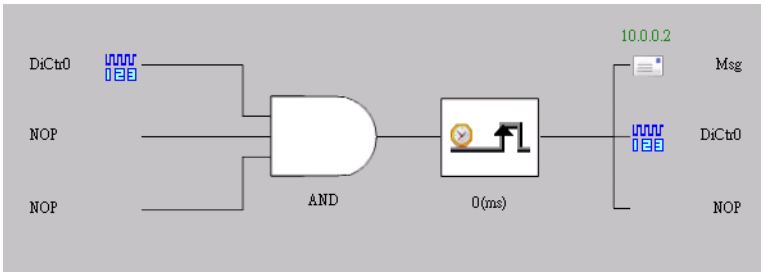


Figure 8.39 GCL Logic for Event Trigger (Only Occurs Once)

The true image of the configuration of the GCL logic rule in Adam/Apax .NET Utility is shown in Figure 8.40. In this example, the desired action is to send a remote message.

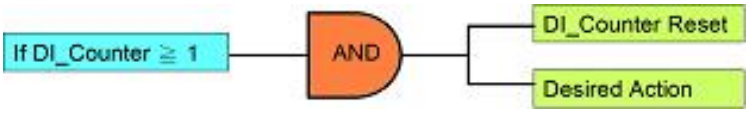


Figure 8.40 Event Trigger Configuration (Only Occurs Once)

Appendix A

Design Worksheets

An organized system configuration will lead to efficient performance and reduce engineer effort. This appendix provides the necessary worksheets to help you configure your DA&C system in an orderly manner.

Follow these steps to build up your system relational document:

Answer these questions for your control strategy

- What will be monitored and controlled?
  (List the equipment)
- What will be monitored and controlled separately?
  (Divide the function area)
- What will be monitored and controlled by ADAM-6000 I/O?
  (List the target equipment in different function areas)

Identify the I/O types and complete the following table to establish an I/O reference sheet

I/O Reference Sheet

Function Area	Equipment Input or Output	I/O Module Type	I/O Module Product No.	Voltage Range	Current Range	Special Requirements

In the following table, map the I/O reference sheet to specific ADAM-6000 I/O modules by listing the following information in each column:

- Column A: TCP/IP addresses for individual function areas
- Column B: product numbers for the I/O modules you need
- Column C: maximum number of I/O points available per module
- Column D: total number of I/O points you need
- Column E: total number of these modules you will need
- Column F: number of spare modules needed for future expansion
- Column G: total number (required + spare) modules you need for these systems

Summary Required Modules

ADAM-6000 IP Address	I/O Module Product No.	Max. No. I/O Points per Module	Total I/O Points Required	Total No. I/O Modules Required	Spare I/O Modules	Total I/O Modules

List the Modbus addresses in the following I/O table.

ADAM-6000 IP Addresses

Table for Programming

I/O Type	Channel Number I/O Address Tag Name	Equipment & Description

These worksheets will be useful for hardware wiring and software integration. You should make copies to establish your own system configuration documentation.

Appendix B

Data Formats and I/O Ranges

B.1 ADAM-6000 Command Data Formats

ADAM-6000 modules can communicate with a host computer in the command-response form. When data are not being transmitted, the modules will be in listen mode. Each module will be assigned a unique address. When issuing a command to a system, the host computer will use these addresses to communicate with specific modules and then wait for a response. If none is detected, a timeout will occur, the sequence will be aborted, and control will be returned to the host.

Command Structure

It is important to understand the encapsulation of a Modbus request or response carried on the Modbus/TCP network. A complete command consists of a "command head" (i.e., Modbus application protocol header) and "command body" (i.e., protocol data unit). The command head is prefixed by six bytes and follows the Modbus data packet format; the command body defines the target device and requested action. The following example will help you to understand this structure.

Example:

The request for reading the first two values of ADAM-6017 (Addresses 40001~40002) is structured as follows:

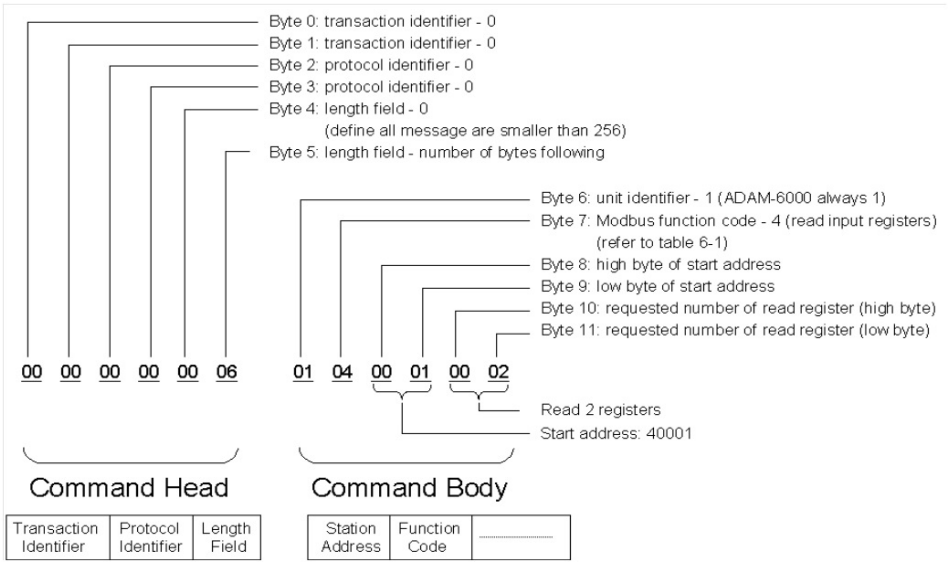


Figure B.1 Request Comment Structure

For this request, the response would be as follows:

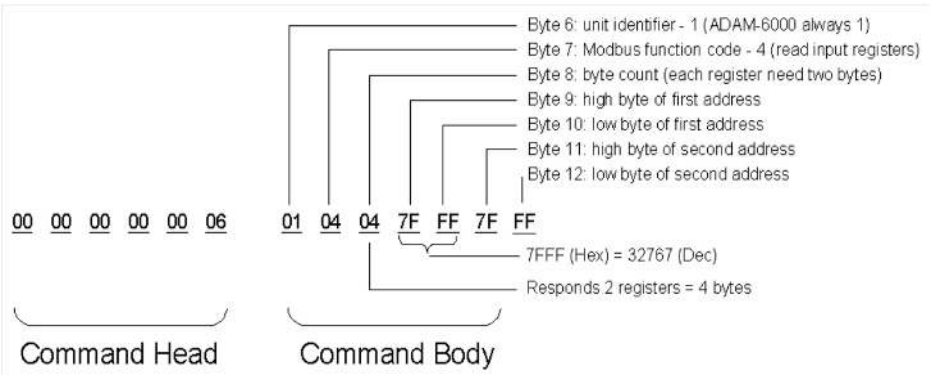


Figure B.2 Response Comment Structure

Modbus Function Codes

Response Comment Structure

Code (Hex)	Name	Usage
01	Read coil status	Read discrete output bit
02	Read input status	Read discrete input bit
03	Read holding registers	Read 16-bit register; used to read integer or floating point process data
04	Read input registers	
05	Force single coil	Write data to force coil ON/OFF
06	Preset single register	Write data in 16-bit integer format
08	Loopback diagnosis	Diagnostic testing of the communication port
0F	Force multiple coils	Write multiple data to force coil ON/OFF
10	Preset multiple registers	Write multiple data in 16-bit integer format

Function Code 01

Reads the discrete output ON/OFF status of an ADAM-6000 module in a binary format.

Request message format:

Command Body					
Station Address	Function Code	Start Address High Byte	Start Address Low Byte	Requested Number of Coil High Byte	Requested Number of Coil Low Byte

Example: Read Coils 1~8 (Addresses 00017~00024) from an ADAM-6000 module.

01 01 00 17 00 08

Response message format:

Command Body					
Station Address	Function Code	Byte Count	Data	Data	…

Example: Coils 2~7 are on, all others are off.

01 01 01 42

In the response, the status of Coils 1~8 is shown as the byte value 42 (hex), which is equivalent to 0100 0010 in binary format.

Function Code 02

Reads the discrete input ON/OFF status of an ADAM-6000 module in binary format.

Request message format:

Command Body					
Station Address	Function Code	Start Address High Byte	Start Address Low Byte	Requested Number of Input High Byte	Requested Number of Input Low Byte

Example: Read coils 1~8 (Addresses 00001~00008) from an ADAM-6000 module.

01 02 00 01 00 08

Response message format:

Command Body					
Station Address	Function Code	Byte Count	Data	Data	…

Example: Inputs 2 and 3 are on, all others are off.

01 02 01 60

In the response, the status of Inputs 1~8 is shown as the byte value 60 (hex), which is equivalent to 0110 0000 in binary format.

Function Codes 03 and 04

Reads the binary content of input registers.

Request message format:

Command Body					
Station Address	Function Code	Start Address High Byte	Start Address Low Byte	Requested Number of Register High Byte	Requested Number of Register Low Byte

Example: Read Analog Inputs 1 and 2 at Addresses 40001~40002 as a floating point value from an ADAM-6017 module

01 04 00 01 00 02

Response message format:

Command Body					
Station Address	Function Code	Byte Count	Data	Data	…

Example: The raw data of Analog inputs 1 and 2 are 17096 and 0, respectively. When the input range is set to 0~10 V, the voltages can be calculated as follows:

Analog Input 1 = (17097/65535) x 10 V = 2.608 V

Analog Input 2 = (0/65535) x 10 V = 0 V

01 04 04 42 C8 00 00

Function Code 05

Forces a single coil to either ON or OFF. The requested ON/OFF state is specified by a constant in the query data field. A value of FF 00 (hex) requests it to be ON; a value of 00 00 (hex) requests it to be OFF; a value of FF FF (hex) requests the forced value to be released.

Request message format:

Command Body					
Station Address	Function Code	Coil Address High Byte	Coil Address Low Byte	Force Data High Byte	Force Data Low Byte

Example: Force Coil 3 (Address 00003) to ON in an ADAM-6000 module.

01 05 00 03 FF 00

Response message format:

The normal response is an echo of the query, returned after the coil state has been forced.

Command Body					
Station Address	Function Code	Coil Address High Byte	Coil Address Low Byte	Force Data High Byte	Force Data Low Byte

Function Code 06

Presets an integer value into a single register.

Request message format:

Command Body

Command Body					
Station Address	Function Code	Coil Address High Byte	Coil Address Low Byte	Force Data High Byte	Force Data Low Byte

Example: Preset Register 40002 to 00 04 (hex) in an ADAM-6000 module.

01 06 00 02 00 04

Response message format:

The normal response is an echo of the query, returned after the coil state has been preset.

Command Body					
Station Address	Function Code	Coil Address High Byte	Coil Address Low Byte	Force Data High Byte	Force Data Low Byte

Function Code 08

Echoes a received query message. Messages can be any length up to half the length of the data buffer minus 8 bytes.

Request message format:

Command Body		
Station Address	Function Code	Any data, length limited to approximately half the length of the data buffer

Example: 01 08 00 02 00 04

Response message format:

Command Body		
Station Address	Function Code	Data bytes received

Example: 01 08 00 02 00 04

Function Code 15 ("0F" in hex)

Forces each coil in a sequence of coils to either an ON or OFF state.

Request message format:

Command Body								
Station Address	Function Code	Start Address High Byte	Start Address Low Byte	Requested Number of Coil High Byte	Requested Number of Coil Low Byte	Byte Count	Force Data High Byte	Force Data Low Byte

Example: Request to force a series of 10 coils starting at Address 00017 ("11" in hex) in an ADAM-6000 module.

01 0F 00 11 00 0A 02 CD 01

The query data contents are two bytes: CD 01 (hex), equivalent to 1100 1101 0000 0001 in binary format. The binary bits are mapped to the addresses in the following manner.

Bit:	1	1	0	0	1	1	0	1	0	0	0	0	0	0	0	1
Address (000XX):	24	23	22	21	20	19	18	17	-	-	-	-	-	-	26	25

Response message format:

A normal response returns the station address, function code, start address, and requested number of the forced coil.

Command Body					
Station Address	Function Code	Start Address High Byte	Start Address Low Byte	Requested Number of Coil High Byte	Requested Number of Coil Low Byte

Example: 01 0F 00 11 00 0A

Function Code 16 ("10" in hex)

Applies a preset value in a sequence of holding registers.

Request message format:

Command Body							
Station Address	Function Code	Start Address High Byte	Start Address Low Byte	Requested Number of Register High Byte	Requested Number of Register Low Byte	Byte Count	Data

Example: Preset Constant 1 (Address 40009) to 100.0 in an ADAM-6000 module.

01 10 00 09 00 02 04 42 C8 00 00

Response message format:

A normal response returns the station address, function code, start address, and requested number of preset registers.

Command Body					
Station Address	Function Code	Start Address High Byte	Start Address Low Byte	Requested Number of Register High Byte	Requested Number of Register Low Byte

Example: 01 10 00 09 00 02

B.2 ADAM-6000 I/O Modbus Mapping Tables

ADAM-6015

Address (0X):

Address (0X)	Channel	Description	Attribute
00101	0	Reset historical max. value	R/W
00102	1		R/W
00103	2		R/W
00104	3		R/W
00105	4		R/W
00106	5		R/W
00107	6		R/W
00109	Average 0~6		R/W

Address (0X)	Channel	Description	Attribute
00111	0	Reset historical min. value	R/W
00112	1		R/W
00113	2		R/W
00114	3		R/W
00115	4		R/W
00116	5		R/W
00117	6		R/W
00119	Average 0~6		R/W

Address (0X)	Channel	Description	Attribute
00121	0	Burnout flag[1]	Read
00122	1		Read
00123	2		Read
00124	3		Read
00125	4		Read
00126	5		Read
00127	6		Read

Address (0X)	Channel	Description	Attribute
00131	0	High alarm flag[2]	Read
00132	1		Read
00133	2		Read
00134	3		Read
00135	4		Read
00136	5		Read
00137	6		Read
00139	Average 0~6		Read

Address (0X)	Channel	Description	Attribute
00141	0	Low alarm flag[3]	Read
00142	1		Read
00143	2		Read
00144	3		Read
00145	4		Read
00146	5		Read
00147	6		Read

Address (0X)	Channel	Description	Attribute
00149	Average 0~6	Low alarm flag[3]	Read

Address (0X)	Channel	Description	Attribute
00301	0	Clear GCL internal counter value	Write
00302	1		Write
00303	2		Write
00304	3		Write
00305	4		Write
00306	5		Write
00307	6		Write
00308	7		Write

Address (4X):

Address (4X)	Channel	Description	Attribute
40001	0	AI value	Read
40002	1		Read
40003	2		Read
40004	3		Read
40005	4		Read
40006	5		Read
40007	6		Read
40009	Average 0~6		Read

Address (4X)	Channel	Description	Attribute
40011	0	Historical max. AI value	Read
40012	1		Read
40013	2		Read
40014	3		Read
40015	4		Read
40016	5		Read
40017	6		Read
40019	Average 0~6		Read

Address (4X)	Channel	Description	Attribute
40021	0	Historical min. AI value	Read
40022	1		Read
40023	2		Read
40024	3		Read
40025	4		Read
40026	5		Read
40027	6		Read
40029	Average 0~6		Read

Address (4X)	Channel	Description	Attribute
40031~40032	0	AI floating point value (IEEE754)	Read
40033~40034	1		Read
40035~40036	2		Read
40037~40038	3		Read
40039~40040	4		Read
40041~40042	5		Read
40043~40044	6		Read
40047~40048	Average 0~6		Read

Address (4X)	Channel	Description	Attribute
40051~40052	0	Historical max. AI floating point value (IEEE754)	Read
40053~40054	1		Read
40055~40056	2		Read
40057~40058	3		Read
40059~40060	4		Read
40061~40062	5		Read
40063~40064	6		Read
40067~40068	Average 0~6		Read

Address (4X)	Channel	Description	Attribute
40071~40072	0	Historical min. AI floating point value (IEEE754)	Read
40073~40074	1		Read
40075~40076	2		Read
40077~40078	3		Read
40079~40080	4		Read
40081~40082	5		Read
40083~40084	6		Read
40087~40088	Average 0~6		Read

Address (4X)	Channel	Description	Attribute
40101~40102	0	AI status[4]	Read
40103~40104	1		Read
40105~40106	2		Read
40107~40108	3		Read
40109~40110	4		Read
40111~40112	5		Read
40113~40114	6		Read

Address (4X)	Channel	Description	Attribute
40201	0	Type code[5]	R/W
40202	1		R/W
40203	2		R/W
40204	3		R/W
40205	4		R/W
40206	5		R/W
40207	6		R/W
40209	Average		R/W

Address (4X)	Channel	Description	Attribute
40211		Module Name 1	Read
40212		Module Name 2	Read

Address (4X)	Channel	Description	Attribute
40305	0~15	GCL internal flag value	R/W

Address (4X)	Channel	Description	Attribute
40311~40312	0	GCL internal counter value	Read
40313~40314	1		Read
40315~40316	2		Read
40317~40318	3		Read
40319~40320	4		Read
40321~40322	5		Read
40323~40324	6		Read
40325~40326	7		Read

Note: The blue Modbus address is only supported by the ADAM-6015-DE

Remarks:

1. When a channel cannot detect an RTD signal, the bit value will be 1.
2. You can configure the high alarm value using Adam/Apax .NET Utility. When the analog input value is higher than the high alarm value, the bit value will be 1.
3. You can configure the low alarm value using Adam/Apax .NET Utility. When the analog input value is lower than the low alarm value, the bit value will be 1.

4. Analog input status

Bit	Description
0	Normal
3	Open circuit/burnout

5. Type code

Type Code (Hex)	Input Range	
0x03A4	PT100(385)	-50~150ºC
0x03A5	PT100(385)	0~100ºC
0x03A6	PT100(385)	0~200ºC
0x03A7	PT100(385)	0~400ºC
0x03A2	PT100(385)	-200~200ºC
0x03C4	PT100(392)	-50~150ºC
0x03C5	PT100(392)	0~100ºC
0x03C6	PT100(392)	0~200ºC
0x03C7	PT100(392)	0~400ºC
0x03C2	PT100(392)	-200~200ºC
0x03E2	PT1000	-40~160ºC
0x0300	Balco500	-30~120ºC
0x0320	NI604(518)	-80~100ºC
0x0321	NI604(518)	0~100ºC

6. The model name of an ADAM module can be retried by reading the hexadecimal value stored at Modbus address 40211.

ADAM-6017

Address (0X):

Address (0X)	Channel	Description	Attribute
00017	0	DO value	R/W
00018	1		R/W

Address (0X)	Channel	Description	Attribute
00101	0	Reset historical max. AI value	R/W
00102	1		R/W
00103	2		R/W
00104	3		R/W
00105	4		R/W
00106	5		R/W
00107	6		R/W
00108	7		R/W
00109	Average 0~7		R/W

Address (0X)	Channel	Description	Attribute
00111	0	Reset historical min. AI value	R/W
00112	1		R/W
00113	2		R/W
00114	3		R/W
00115	4		R/W
00116	5		R/W
00117	6		R/W
00118	7		R/W
00119	Average 0~7		R/W

Address (0X)	Channel	Description	Attribute
00121	0	Open-circuit flag[1] (burn-out)	Read
00122	1		Read
00123	2		Read
00124	3		Read
00125	4		Read
00126	5		Read
00127	6		Read
00128	7		Read

Address (0X)	Channel	Description	Attribute
00131	0	High alarm flag[2]	Read
00132	1		Read
00133	2		Read
00134	3		Read
00135	4		Read
00136	5		Read
00137	6		Read
00138	7		Read
00139	Average 0~7		Read

Address (0X)	Channel	Description	Attribute
00141	0	Low alarm flag[3]	Read
00142	1		Read
00143	2		Read
00144	3		Read
00145	4		Read
00146	5		Read
00147	6		Read
00148	7		Read
00149	Average 0~7		Read

Address (0X)	Channel	Description	Attribute
00301	0	Clear GCL internal counter value	Write
00302	1		Write
00303	2		Write
00304	3		Write
00305	4		Write
00306	5		Write
00307	6		Write
00308	7		Write

Address (4X)

Address (4X)	Channel	Description	Attribute
40001	0	AI value	Read
40002	1		Read
40003	2		Read
40004	3		Read
40005	4		Read
40006	5		Read
40007	6		Read
40008	7		Read
40009	Average 0~7		Read

Address (4X)	Channel	Description	Attribute
40011	0	Historical max. AI value	Read
40012	1		Read
40013	2		Read
40014	3		Read
40015	4		Read
40016	5		Read
40017	6		Read
40018	7		Read
40019	Average 0~7		Read

Address (4X)	Channel	Description	Attribute
40021	0	Historical min. AI value	Read
40022	1		Read
40023	2		Read
40024	3		Read
40025	4		Read
40026	5		Read
40027	6		Read
40028	7		Read
40029	Average 0~7		Read

Address (4X)	Channel	Description	Attribute
40031~40032	0	AI floating point value (IEEE754)	Read
40033~40034	1		Read
40035~40036	2		Read
40037~40038	3		Read
40039~40040	4		Read
40041~40042	5		Read
40043~40044	6		Read
40045~40046	7		Read
40047~40048	Average 0~7		Read

Address (4X)	Channel	Description	Attribute
40051~40052	0	Historical max. AI floating point value (IEEE754)	Read
40053~40054	1		Read
40055~40056	2		Read
40057~40058	3		Read
40059~40060	4		Read
40061~40062	5		Read
40063~40064	6		Read
40065~40066	7		Read
40067~40068	Average 0~7		Read

Address (4X)	Channel	Description	Attribute
40071~40072	0	Historical min. AI floating point value (IEEE754)	Read
40073~40074	1		Read
40075~40076	2		Read
40077~40078	3		Read
40079~40080	4		Read
40081~40082	5		Read
40083~40084	6		Read
40085~40086	7		Read
40087~40088	Average 0~7		Read

Address (4X)	Channel	Description	Attribute
40101	0	AI status[4]	Read
40103	1		Read
40105	2		Read
40107	3		Read
40109	4		Read
40111	5		Read
40113	6		Read
40115	7		Read

Address (4X)	Channel	Description	Attribute
40201	0	Type code[5]	R/W
40202	1		R/W
40203	2		R/W
40204	3		R/W
40205	4		R/W
40206	5		R/W
40207	6		R/W
40208	7		R/W
40209	Average 0~7		Read

Address (4X)	Channel	Description	Attribute
40211		Module Name 1	Read
40212		Module Name 2	Read
40221	All	AI channel enable	R/W
40305	0~15	GCL internal flag value	R/W

Address (4X)	Channel	Description	Attribute
40311~40312	0	GCL internal counter value	Read
40313~40314	1		Read
40315~40316	2		Read
40317~40318	3		Read
40319~40320	4		Read
40321~40322	5		Read
40323~40324	6		Read
40325~40326	7		Read

Note! *The blue Modbus addresses are only supported by the ADAM-6000 CE or D version.*

Remarks

1. When the channel cannot detect the current input signal in the 4~20 mA input range, the bit value will be 1.
2. You can configure the high alarm value using Adam/Apax .NET Utility. When the analog input value is higher than the high alarm, the bit value will be 1.
3. You can configure the low alarm value using Adam/Apax .NET Utility. When the analog input value is lower than the low alarm, the bit value will be 1.
4. Analog input status

Bit	Description
0	Fail to provide the AI value
1	Over range
2	Under range
3	Open circuit/burnout

4	Reserved
5	Reserved
6	Reserved
7	ADC initialization error
8	Reserved
9	Zero/span calibration error
10	Reserved
11	Reserved
12	Reserved
13	Reserved
14	Reserved
15	Reserved

5. Type Code

Type Code (Hex)	Input Range
0x0103	±150 mV
0x0104	±500 mV
0x0105	0~150 mV
0x0106	0~500 mV
0x0140	±1 V
0x0142	±5 V
0x0143	±10 V
0x0145	0~1 V
0x0147	0~5 V
0x0148	0~10 V
0x0181	±20 mA
0x0180	4~20 mA
0x0182	0~20 mA

1. The model name of an ADAM module can be retried by reading the hexadecimal value stored at Modbus address 40211.

ADAM-6018, ADAM-6018+(only support the analog input)

Address (0X)

Address (0X)	Channel	Description	Attribute
00017	0	DO value (only for ADAM-6018)	R/W
00018	1		R/W
00019	2		R/W
00020	3		R/W
00021	4		R/W
00022	5		R/W
00023	6		R/W
00024	7		R/W

Address (0X)	Channel	Description	Attribute
00101	0	Reset historical max. value	R/W
00102	1		R/W
00103	2		R/W
00104	3		R/W
00105	4		R/W
00106	5		R/W
00107	6		R/W
00108	7		R/W
00109	Average 0~7		R/W

Address (0X)	Channel	Description	Attribute
00111	0	Reset historical min. value	Read
00112	1		Read
00113	2		Read
00114	3		Read
00115	4		Read
00116	5		Read
00117	6		Read
00118	7		Read
00119	Average 0~7		Read

Address (0X)	Channel	Description	Attribute
00121	0	Burnout flag[1]	Read
00122	1		Read
00123	2		Read
00124	3		Read
00125	4		Read
00126	5		Read
00127	6		Read
00128	7		Read

Address (0X)	Channel	Description	Attribute
00131	0	High alarm flag[2]	Read
00132	1		Read
00133	2		Read
00134	3		Read
00135	4		Read
00136	5		Read
00137	6		Read
00138	7		Read
00139	Average 0~7		Read

Address (0X)	Channel	Description	Attribute
00141	0	Low alarm flag[3]	Read
00142	1		Read
00143	2		Read
00144	3		Read
00145	4		Read
00146	5		Read
00147	6		Read
00148	7		Read
00149	Average 0~7		Read

Address (4X)

Address (4X)	Channel	Description	Attribute
40001	0	AI value	Read
40002	1		Read
40003	2		Read
40004	3		Read
40005	4		Read
40006	5		Read
40007	6		Read
40008	7		Read
40009	Average 0~7		Read

Address (4X)	Channel	Description	Attribute
40011	0	Historical max. AI value	Read
40012	1		Read
40013	2		Read
40014	3		Read
40015	4		Read
40016	5		Read
40017	6		Read
40018	7		Read
40019	Average 0~7		Read

Address (4X)	Channel	Description	Attribute
40021	0	Historical min. AI value	Read
40022	1		Read
40023	2		Read
40024	3		Read
40025	4		Read
40026	5		Read
40027	6		Read
40028	7		Read
40029	Average 0~7		Read

Address (4X)	Channel	Description	Attribute
40305	0~15	GCL internal flag value	R/W

Remarks

1. When the specific channel cannot detect a thermocouple signal, the bit value will be 1.
2. User can configure the high alarm value using Adam/Apax .NET Utility. When the analog input value is higher than the high alarm value, the bit value will be 1.
3. You can configure the low alarm value using Adam/Apax .NET Utility. When the analog input value is lower than the low alarm value, the bit value will be 1.

ADAM-6024

Address(0X)

Address(0X)	Channel	Description	Attribute
00001	0	DI value	Read
00002	1		Read
00017	0	DO value	R/W
00018	1		R/W
00101~00106	0~5	Reset historical max AI value	R/W
00111~00116	0~5	Reset historical min AI value	R/W

Address(4X)

Address(4X)	Channel	Description	Attribute
40001	0	AI value	Read
40002	1		Read
40003	2		Read
40004	3		Read
40005	4		Read
40006	5		Read
40011	0	AO value	R/W
40012	1		R/W

40021	0	AI status [1]	Read
40022	1		Read
40023	2		Read
40024	3		Read
40025	4		Read
40026	5		Read
40111~40116	0~5	Historical max AI value	Read
40121~40126	0~5	Historical min AI value	Read
40201~40206	0~5	AI type code[2]	R/W
40209~40210	0~1	AO type code[3]	R/W
40211	-	Module name	Read
40221	0~5	AI channel enable	R/W
40401~40402	0~1	AO startup value	R/W

Remark

[1]. AI status

- Bit value = 0: normal
- Bit value = 1: over high
- Bit value = 2: over low
- Bit value = 0: invalid calibration

[2]. AI type

- 7 in hex → 4~20mA
- 8 in hex → +/- 10V
- D in hex → 0~20mA

[3]. AO type

- 0 in hex → 0~20mA
- 1 in hex → 4~20mA
- 2 in hex → 0~10V

ADAM-6050

Address (0X)

Address (0X)	Channel	Description	Attribute
00001	0	DI value	Read
00002	1		Read
00003	2		Read
00004	3		Read
00005	4		Read
00006	5		Read
00007	6		Read
00008	7		Read
00009	8		Read
00010	9		Read
00011	10		Read
00012	11		Read

Address (0X)	Channel	Description	Attribute
00017	0	DO value	R/W
00018	1		R/W
00019	2		R/W
00020	3		R/W
00021	4		R/W
00022	5		R/W

Address (0X)	Channel	Description	Attribute
00033	0	Counter start (1)/stop (0)	R/W
00034		Clear counter (1)	Write
00035		Clear overflow[3]	R/W
00036		DI latch status[4]	R/W
00037	1	Counter start (1)/stop (0)	R/W
00038		Clear counter (1)	Write
00039		Clear overflow[3]	R/W
00040		DI latch status[4]	R/W
00041	2	Counter start (1)/stop (0)	R/W
00042		Clear counter (1)	Write
00043		Clear overflow[3]	R/W
00044		DI latch status[4]	R/W
00045	3	Counter start (1)/stop (0)	R/W
00046		Clear counter (1)	Write
00047		Clear overflow[3]	R/W
00048		DI latch status[4]	R/W
00049	4	Counter start (1)/stop (0)	R/W
00050		Clear counter (1)	Write
00051		Clear overflow[3]	R/W
00052		DI latch status[4]	R/W
00053	5	Counter start (1)/stop (0)	R/W
00054		Clear counter (1)	Write
00055		Clear overflow[3]	R/W
00056		DI latch status[4]	R/W
00057	6	Counter start (1)/stop (0)	R/W
00058		Clear counter (1)	Write
00059		Clear overflow[3]	R/W
00060		DI latch status[4]	R/W
00061	7	Counter start (1)/stop (0)	R/W
00062		Clear counter (1)	Write
00063		Clear overflow[3]	R/W
00064		DI latch status[4]	R/W
00065	8	Counter start (1)/stop (0)	R/W
00066		Clear counter (1)	Write

00067		Clear overflow[3]	R/W
00068		DI Latch status[4]	R/W
00069	9	Counter start (1)/stop (0)	R/W
00070		Clear counter (1)	Write
00071		Clear overflow[3]	R/W
00072		DI latch status[4]	R/W
00073	10	Counter start (1)/stop (0)	R/W
00074		Clear counter (1)	Write
00075		Clear overflow[3]	R/W
00076		DI latch status[4]	R/W
00077	11	Counter start (1)/stop (0)	R/W
00078		Clear counter (1)	Write
00079		Clear overflow[3]	R/W
00080		DI latch status[4]	R/W

Address (0X)	Channel	Description	Attribute
00301	0	Clear GCL internal counter	Write
00302	1		Write
00303	2		Write
00302	3		Write
00305	4		Write
00306	5		Write
00307	6		Write
00308	7		Write

Address (4X)

Address (4X)	Channel	Description	Attribute
40301	All	DI value	Read
40303	All	DO value	R/W
40305	0~15	GCL internal flag value	R/W
40307	All	DO diagnostic status (for D version or later)	Read
40308	All	DO diagnostic mask Read (for D version or later)	Read

Address (4X)	Channel	Description	Attribute
40001~40002	0	Counter/Frequency value[1]	Read
40003~40004	1		Read
40005~40006	2		Read
40007~40008	3		Read
40009~40010	4		Read
40011~40012	5		Read
40013~40014	6		Read
40015~40016	7		Read
40017~40018	8		Read
40019~40020	9		Read
40021~40022	10		Read
40023~40024	11		Read

Address (4X)	Channel	Description	Attribute
40025~40026	0	Pulse output low-level width[2]	Read
40027~40028	1		Read
40029~40030	2		Read
40031~40032	3		Read
40033~40034	4		Read
40035~40036	5		Read

Address (4X)	Channel	Description	Attribute
40037~40038	0	Pulse output high-level width[2]	Read
40039~40040	1		Read
40041~40042	2		Read
40043~40044	3		Read
40045~40046	4		Read
40047~40048	5		Read

Address (4X)	Channel	Description	Attribute
40049~40050	0	Set absolute pulse[5]	R/W
40051~40052	1		R/W
40053~40054	2		R/W
40055~40056	3		R/W
40057~40058	4		R/W
40059~40060	5		R/W

Address (4X)	Channel	Description	Attribute
40061~40062	0	Set incremental pulse[6]	R/W
40063~40064	1		R/W
40065~40066	2		R/W
40067~40068	3		R/W
40069~40070	4		R/W
40071~40072	5		R/W

Address (4X)	Channel	Description	Attribute
40211	All	Module name 1	Read
40212	All	Module name 2	Read

Address (4X)	Channel	Description	Attribute
40311~40312	0	GCL internal counter value	Read
40313~40314	1		Read
40315~40316	2		Read
40317~40318	3		Read
40319~40320	4		Read
40321~40322	5		Read
40323~40324	6		Read
40325~40326	7		Read

Note! *The blue Modbus addresses are only supported by the CE version (or later).*

Remarks

1. How to retrieve the counter/frequency value:
   Counter (decimal) = (value of 40002) x 65536 + (value of 40001)
   Frequency (decimal) = (value of 40001) / 10 Hz
2. Time increment: 0.1 ms
3. If the count number is an overflow, the bit value will be 1. Once this bit has been read, the value will return to 0.
4. When the digital input channel is configured as "high-to-low latch" or "low-to-high latch," the bit value will be 1 if the latch condition is met. Subsequently, the bit value will remain at 1 until you write 0 to this bit (i.e., when you clear the latch status).
5. Decide how many pulses will be generated. When you write 0 to this bit, continuous pulses will be generated.
6. During pulse generation, you can use this bit to generate additional pulses. For example, assume that "Absolute pulse" is set as 100; during its generation, you can set "Incremental pulse" to 10 in order for an extra 10 pulses to be generated. The 100 pulses will continue to be generated.
7. The model name of an ADAM module can be retried by reading the hexadecimal value stored at Modbus address 40211.

ADAM-6051

Address (0X)

Address (0X)	Channel	Description	Attribute
00001	0	DI value	Read
00002	1		Read
00003	2		Read
00004	3		Read
00005	4		Read
00006	5		Read
00007	6		Read
00008	7		Read
00009	8		Read
00010	9		Read
00011	10		Read
00012	11		Read
00013	12		Read
00014	13		Read

Address (0X)	Channel	Description	Attribute
00033	0	Counter start (1)/stop (0)	R/W
00034		Clear counter (1)	Write
00035		Clear overflow[3]	R/W
00036		DI latch status[4]	R/W
00037	1	Counter start (1)/stop (0)	R/W
00038		Clear counter (1)	Write
00039		Clear overflow[3]	R/W
00040		DI latch status[4]	R/W
00041	2	Counter start (1)/stop (0)	R/W
00042		Clear counter (1)	Write
00043		Clear overflow[3]	R/W
00044		DI latch status[4]	R/W
00045	3	Counter start (1)/stop (0)	R/W
00046		Clear counter (1)	Write
00047		Clear overflow[3]	R/W
00048		DI latch status[4]	R/W
00049	4	Counter start (1)/stop (0)	R/W
00050		Clear counter (1)	Write
00051		Clear overflow[3]	R/W
00052		DI latch status[4]	R/W
00053	5	Counter start (1)/stop (0)	R/W
00054		Clear counter (1)	Write
00055		Clear overflow[3]	R/W
00056		DI latch status[4]	R/W
00057	6	Counter start (1)/stop (0)	R/W

00058		Clear counter (1)	Write
00059		Clear overflow[3]	R/W
00060		DI latch status[4]	R/W
00061	7	Counter start (1)/stop (0)	R/W
00062		Clear counter (1)	Write
00063		Clear overflow[3]	R/W
00064		DI latch status[4]	R/W
00065	8	Counter start (1)/stop (0)	R/W
00066		Clear counter (1)	Write
00067		Clear overflow[3]	R/W
00068		DI latch status[4]	R/W
00069	9	Counter start (1)/stop (0)	R/W
00070		Clear counter (1)	Write
00071		Clear overflow[3]	R/W
00072		DI latch status[4]	R/W
00073	10	Counter start (1)/stop (0)	R/W
00074		Clear counter (1)	Write
00075		Clear overflow[3]	R/W
00076		DI latch status[4]	R/W
00077	11	Counter start (1)/stop (0)	R/W
00078		Clear counter (1)	Write
00079		Clear overflow[3]	R/W
00080		DI latch status[4]	R/W
00081	12 7	Counter start (1)/stop (0)	R/W
00082		Clear counter (1)	Write
00083		Clear overflow[3]	R/W
00084		DI latch status[4]	R/W
00085	137	Counter start (1)/stop (0)	R/W
00086		Clear counter (1)	Write
00087		Clear overflow[3]	R/W
00088		DI latch status[4]	R/W

Address (0X)	Channel	Description	Attribute
00301	0	Clear GCL internal counter value	Write
00302	1		Write
00303	2		Write
00304	3		Write
00305	4		Write
00306	5		Write
00307	6		Write
00308	7		Write

Address (4X)

Address (4X)	Channel	Description	Attribute
40001~40002	0	Counter/frequency value[1]	Read
40003~40004	1		Read
40005~40006	2		Read
40007~40008	3		Read
40009~40010	4		Read
40011~40012	5		Read
40013~40014	6		Read
40015~40016	7		Read
40017~40018	8		Read
40019~40020	9		Read
40021~40022	10		Read
40023~40024	11		Read
40025~40026	12		Read
40027~40028	13		Read

Address (4X)	Channel	Description	Attribute
40029~40030	0	Pulse output	
Low Level Width 2	Read		
40031~40032	1		Read

Address (4X)	Channel	Description	Attribute
40033~40034	0	Pulse output high-level width[2]	Read
40035~40036	1		Read

Address (4X)	Channel	Description	Attribute
40037~40038	0	Set absolute pulse[5]	Read
40039~40040	1		Read

Address (4X)	Channel	Description	Attribute
40041~40042	0	Set incremental pulse[6]	Read
40043~40044	1		Read

Address (4X)	Channel	Description	Attribute
40033~40034	0	Pulse output high-level width	R/W
40035~40036	1		R/W

Address (4X)	Channel	Description	Attribute
40301	All	DI value	Read
40303	All	DO value	R/W
40305	0~15	GCL internal flag value	R/W
40307	All	DO diagnostic status (for D version)	Read

Address (4X)	Channel	Description	Attribute
40211		Module name 1	Read
40212		Module name 2	Read

Address (4X)	Channel	Description	Attribute
40311~40312	0	GCL internal counter value	Read
40313~40314	1		Read
40315~40316	2		Read
40317~40318	3		Read
40319~40320	4		Read
40321~40322	5		Read
40323~40324	6		Read
40325~40326	7		Read

Note! *The blue Modbus addresses are only supported by the CE version or later.*

Remarks

1. How to retrieve the counter/frequency value:
   Counter (decimal) = (value of 40002) x 65536 + (value of 40001)
   Frequency (decimal) = (value of 40001) / 10 Hz
2. Time increment: 0.1 ms
3. If the count number is an overflow, the bit value will be 1. Once this bit has been read, the value will return to 0.
4. When the digital input channel is configured as "high-to-low latch" or "low-to-high latch," the bit value will be 1 if the latch condition is met. Subsequently, the bit value will remain at 1 until you write 0 to this bit (i.e., when you clear the latch status).
5. Decide how many pulses will be generated. When you write 0 to this bit, continuous pulses will be generated.
6. During pulse generation, you can use this bit to generate additional pulses. For example, assume that "Absolute pulse" is set as 100; during its generation, you can set "Incremental pulse" to 10 in order for an extra 10 pulses to be generated. The 100 pulses will continue to be generated.
7. The model name of an ADAM module can be retried by reading the hexadecimal value stored at Modbus address 40211.

ADAM-6052

Address (0X)

Address (0X)	Channel	Description	Attribute
00001	0	DI value	Read
00002	1		Read
00003	2		Read
00004	3		Read
00005	4		Read
00006	5		Read
00007	6		Read
00008	7		Read

Address (0X)	Channel	Description	Attribute
00017	0	DO value	R/W
00018	1		R/W
00019	2		R/W
00020	3		R/W
00021	4		R/W
00022	5		R/W
00023	6		R/W
00024	7		R/W

Address (0X)	Channel	Description	Attribute
00033	0	Counter start (1)/stop (0)	R/W
00034		Clear counter (1)	Write
00035		Clear overflow[3]	R/W
00036		DI latch status[4]	R/W
00037	1	Counter start (1)/stop (0)	R/W
00038		Clear counter (1)	Write
00039		Clear overflow[3]	R/W
00040		DI latch status[4]	R/W
00041	2	Counter start (1)/stop (0)	R/W
00042		Clear counter (1)	Write
00043		Clear overflow[3]	R/W
00044		DI latch status[4]	R/W
00045	3	Counter start (1)/stop (0)	R/W
00046		Clear counter (1)	Write
00047		Clear overflow[3]	R/W
00048		DI latch status[4]	R/W
00049	4	Counter start (1)/stop (0)	R/W
00050		Clear counter (1)	Write
00051		Clear overflow[3]	R/W
00052		DI latch status[4]	R/W
00053	5	Counter start (1)/stop (0)	R/W
00054		Clear counter (1)	Write
00055		Clear overflow[3]	R/W
00056		DI latch status[4]	R/W
00057	6	Counter start (1)/stop (0)	R/W
00058		Clear counter (1)	Write
00059		Clear overflow[3]	R/W
00060		DI latch status[4]	R/W
00061	7	Counter start (1)/stop (0)	R/W
00062		Clear counter (1)	Write
00063		Clear overflow[3]	R/W
00064		DI latch status[4]	R/W

Address (0X)	Channel	Description	Attribute
00301	0	Clear GCL internal counter value	Write
00302	1		Write
00303	2		Write
00304	3		Write
00305	4		Write
00306	5		Write
00307	6		Write
00308	7		Write

Address (4X)

Address (4X)	Channel	Description	Attribute
40001~40002	0	Counter/frequency value[1]	Read
40003~40004	1		Read
40005~40006	2		Read
40007~40008	3		Read
40009~40010	4		Read
40011~40012	5		Read
40013~40014	6		Read
40015~40016	7		Read
40017~40018	0	Pulse output low-level width[2]	R/W
4001~40020	1		R/W
40021~40022	2		R/W
40023~40024	3		R/W
40025~40026	4		R/W
40027~40028	5		R/W
40029~40030	6		R/W
40031~40032	7		R/W
40033~40034	0	Pulse output high-level width[2]	R/W
40035~40036	1		R/W
40037~40038	2		R/W
40039~40040	3		R/W
40041~40042	4		R/W
40043~40044	5		R/W
40045~40046	6		R/W
40047~40048	7		R/W

Address (4X)	Channel	Description	Attribute
40049~40050	0	Set absolute pulse[5]	R/W
40051~40052	1		R/W
40053~40054	2		R/W
40055~40056	3		R/W
40057~40058	4		R/W
40059~40060	5		R/W
40061~40062	6		R/W
40063~40064	7		R/W
40065~40066	0	Set incremental pulse[6]	R/W
40067~40068	1		R/W
40069~40070	2		R/W
40071~40072	3		R/W
40073~40074	4		R/W
40075~40076	5		R/W
40077~40078	6		R/W
40079~40080	7		R/W

Address (4X)	Channel	Description	Attribute
40301	All	DI value	Read
40303	All	DO value	R/W
40305	0~15	GCL internal flag value	R/W
40307	All	DO diagnostic status (for D version)	Read

Address (4X)	Channel	Description	Attribute
40211		Module name 1	Read
40212		Module name 2	Read

Address (4X)	Channel	Description	Attribute
40311~40312	0	GCL internal counter value	Read
40313~40314	1		Read
40315~40316	2		Read
40317~40318	3		Read
40319~40320	4		Read
40321~40322	5		Read
40323~40324	6		Read
40325~40326	7		Read

Note! *The blue Modbus addresses are only supported by the CE version or later.*

Remarks

1. How to retrieve the counter/frequency value:
   Counter (decimal) = (value of 40002) x 65536 + (value of 40001)
   Frequency (decimal) = (value of 40001) / 10 Hz
2. Time increment: 0.1 ms
3. If the count number is an overflow, the bit value will be 1. Once this bit has been read, the value will return to 0.
4. When the digital input channel is configured as "high-to-low latch" or "low-to-high latch," the bit value will be 1 if the latch condition is met. Subsequently, the bit value will remain at 1 until you write 0 to this bit (i.e., when you clear the latch status).
5. Decide how many pulses will be generated. When you write 0 to this bit, continuous pulses will be generated.
6. During pulse generation, you can use this bit to generate additional pulses. For example, assume that "Absolute pulse" is set as 100; during its generation, you can set "Incremental pulse" to 10 in order for an extra 10 pulses to be generated. The 100 pulses will continue to be generated.
7. The model name of an ADAM module can be retried by reading the hexadecimal value stored at Modbus address 40211.

ADAM-6060/6066

Address (0X)

Address (0X)	Channel	Description	Attribute
00001	0	DI value	Read
00002	1		Read
00003	2		Read
00004	3		Read
00005	4		Read
00006	5		Read

Address (0X)	Channel	Description	Attribute
00017	0	DO value	R/W
00018	1		R/W
00019	2		R/W
00020	3		R/W
00021	4		R/W
00022	5		R/W

Address (0X)	Channel	Description	Attribute
00033	0	Counter start (1)/stop (0)	R/W
00034		Clear counter (1)	Write
00035		Clear overflow[3]	R/W
00036		DI latch status[4]	R/W
00037	1	Counter start (1)/stop (0)	R/W
00038		Clear counter (1)	Write
00039		Clear overflow[3]	R/W
00040		DI latch status[4]	R/W
00041	2	Counter start (1)/stop (0)	R/W
00042		Clear counter (1)	Write
00043		Clear overflow[3]	R/W
00044		DI latch status[4]	R/W
00045	3	Counter start (1)/stop (0)	R/W
00046		Clear counter (1)	Write
00047		Clear overflow[3]	R/W
00048		DI latch status[4]	R/W
00049	4	Counter start (1)/stop (0)	R/W
00050		Clear counter (1)	Write
00051		Clear overflow[3]	R/W
00052		DI latch status[4]	R/W
00053	5	Counter start (1)/stop (0)	R/W
00054		Clear counter (1)	Write
00055		Clear overflow[3]	R/W
00056		DI latch status[4]	R/W

Address (0X)	Channel	Description	Attribute
00301	0	Clear GCL internal counter value	Write
00302	1		Write
00303	2		Write
00304	3		Write
00305	4		Write
00306	5		Write
00307	6		Write
00308	7		Write

Address (4X)

Address (4X)	Channel	Description	Attribute
40001~40002	0	Counter/frequency value[1]	Read
40003~40004	1		Read
40005~40006	2		Read
40007~40008	3		Read
40009~40010	4		Read
40011~40012	5		Read

Address (4X)	Channel	Description	Attribute
40013~40014	0	Pulse output low-level width[2]	R/W
40015~40016	1		R/W
40017~40018	2		R/W
40019~40020	3		R/W
40021~40022	4		R/W
40023~40024	5		R/W

Address (4X)	Channel	Description	Attribute
40025~40026	0	Pulse output high-level width[2]	R/W
40027~40028	1		R/W
40029~40030	2		R/W
40031~40032	3		R/W
40033~40034	4		R/W
40035~40036	5		R/W

Address (4X)	Channel	Description	Attribute
40037~40038	0	Set absolute pulse[5]	R/W
40039~40040	1		R/W
40041~40042	2		R/W
40043~40044	3		R/W
40045~40046	4		R/W
40047~40048	5		R/W

Address (4X)	Channel	Description	Attribute
40049~40050	0	Set incremental pulse[6]	R/W
40051~40052	1		R/W
40053~40054	2		R/W
40055~40056	3		R/W
40057~40058	4		R/W
40059~40060	5		R/W

Address (4X)	Channel	Description	Attribute
40301	All	DI value	Read
40303	All	DO value	R/W
40305	0~15	GCL internal flag value	R/W

Address (4X)	Channel	Description	Attribute
40211		Module name 1	Read
40212		Module name 2	Read

Address (4X)	Channel	Description	Attribute
40311~40312	0	GCL internal counter value	Read
40313~40314	1		Read
40315~40316	2		Read
40317~40318	3		Read
40319~40320	4		Read
40321~40322	5		Read
40323~40324	6		Read
40325~40326	7		Read

Note! *The blue Modbus addresses are only supported by the CE version or later.*

Remarks

1. How to retrieve the counter/frequency value:
   Counter (decimal) = (value of 40002) x 65536 + (value of 40001)
   Frequency (decimal) = (value of 40001) / 10 Hz
2. Time increment: 0.1 ms
3. If the count number is an overflow, the bit value will be 1. Once this bit has been read, the value will return to 0.
4. When the digital input channel is configured as "high-to-low latch" or "low-to-high latch," the bit value will be 1 if the latch condition is met. Subsequently, the bit value will remain at 1 until you write 0 to this bit (i.e., when you clear the latch status).

5. Decide how many pulses will be generated. When you write 0 to this bit, continuous pulses will be generated.
6. During pulse generation, you can use this bit to generate additional pulses. For example, assume that "Absolute pulse" is set as 100; during its generation, you can set "Incremental pulse" to 10 in order for an extra 10 pulses to be generated. The 100 pulses will continue to be generated.
7. The model name of an ADAM module can be retried by reading the hexadecimal value stored at Modbus address 40211.

Appendix C

Grounding Reference

C.1 Field Grounding and Shielding Application

C.1.1 Overview

Unfortunately, it is seldom possible to complete a system integration task in one pass because of unforeseeable problems encountered in the field. A communication network or system could be unstable, or there may be noise from equipment damage or a nearby thunderstorm. However, the most common issue is improper wiring, and such problems are generally related to grounding and shielding. Although wiring and cabling incurs considerably less cost than hardware, the reliability of a network actually depends mostly on grounding and shielding. Thus, it is a worthwhile to invest in both time and quality wiring/cabling to ensure system reliability. This appendix describes some fundamental concepts about field grounding and shielding.

Grounding	C.2
The Earth for reference	C.2.1
The Frame Ground and Grounding Bar	C.2.2
Normal Mode and Common Mode	C.2.3
Wire impedance	C.2.4
Single-Point Grounding	C.2.5
Shielding	C.3
Cable Shield	C.3.1
System Shielding	C.3.2
Noise Reduction Techniques	C.4
Checklist	C.5

C.2 Grounding

C.2.1 The Earth as a Reference

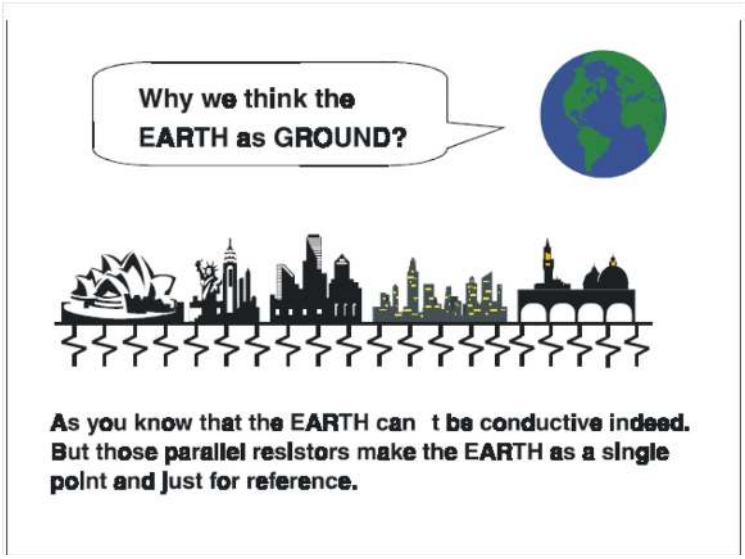


Figure C.1 Thinking of the Earth as a Ground

Obviously, the earth is not conductive. Consider that all buildings lie on (or in) the earth; thus, steel, concrete, and associated cables (such as lightning arresters) and

power system are connected to earth. It is helpful to think of them as resistors. The countless number of parallel resistors makes the earth a single reference point.

C.2.2 Frame Grounds and Grounding Bars

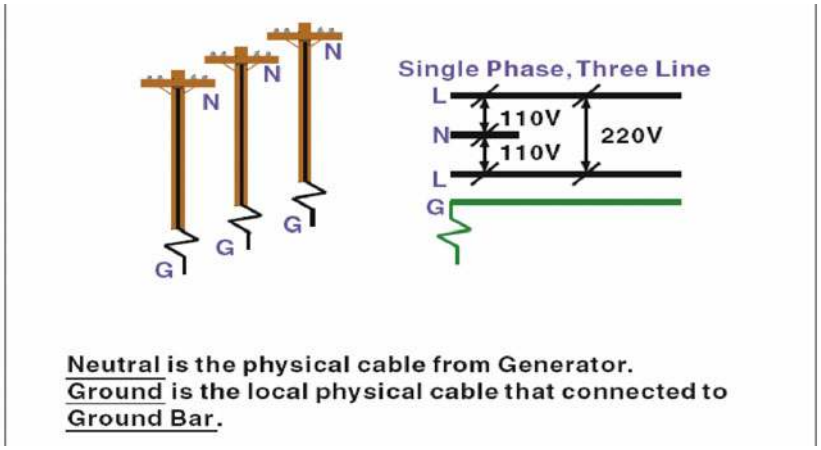


Figure C.2 Grounding Bar

Grounding is one of the most important issues for our systems. Just like the frame ground of a computer, a data signal acquired by a channel offers a reference point of the electronic circuit inside the computer. If we want to communicate with this computer, both the signal ground and frame ground should be connected to each other in order to make a reference point for each other's electronic circuit. For computer networks, power systems, telecommunication networks, and so on, it is generally necessary to install an individual grounding bar for each system. These grounding bars not only provide an individual reference point but they also make the earth the ground.

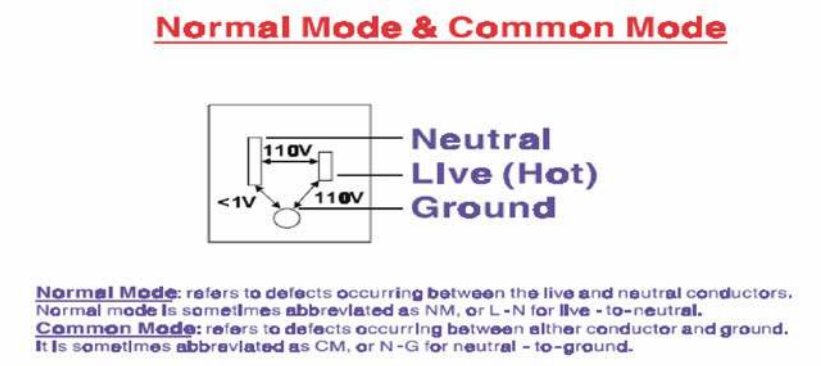


Figure C.3 Figure C.3: Normal and Common Mode

C.2.3 Normal Mode and Common Mode

Attempting to measure the voltage between a live circuit and a concrete floor or between neutral and a concrete floor will give nonsensical values. "Hot" and "neutral" are just relational signals: you will get 110 or 220 V_{AC} by measuring these signals. Normal mode and common mode show that the frame ground is the most important reference signal for all systems and equipment.

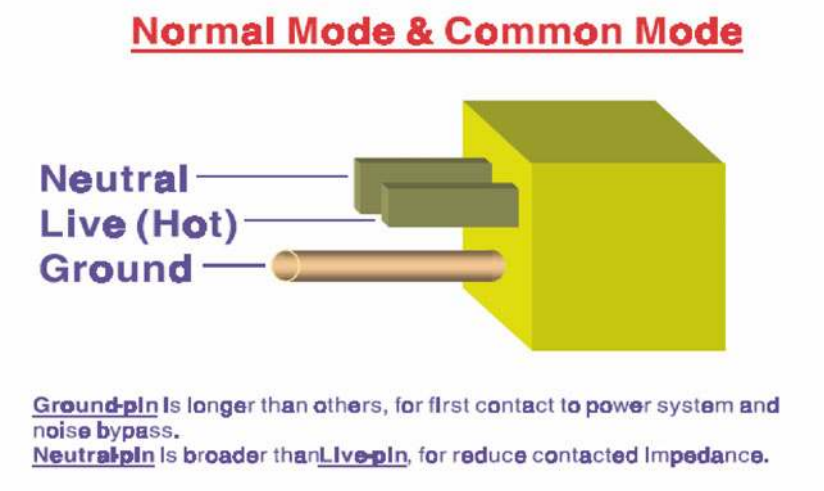


Figure C.4 Normal and Common Mode

- The ground pin is longer than the other pins so that it is the first in contact with a power system and to act as a noise bypass.
- The neutral pin is broader than live pin in order to reduce contact impedance.

C.2.4 Wire impedance

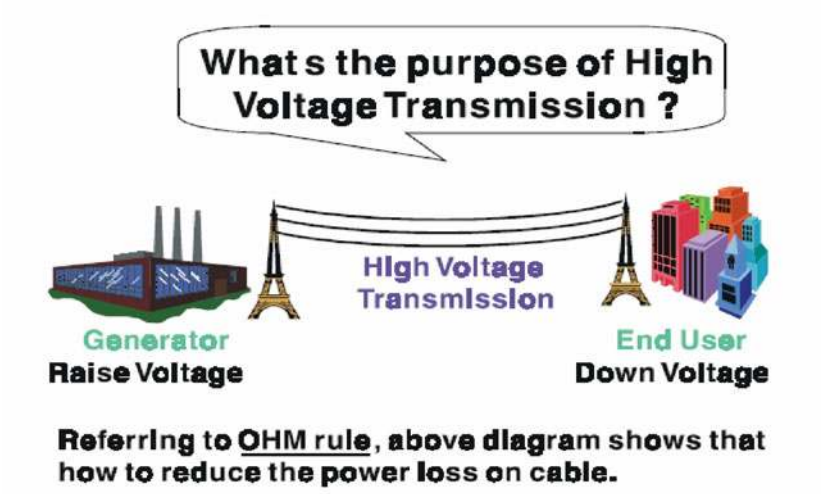


Figure C.5 High Voltage Transmission

High-voltage power lines are necessary because power plants generate high-voltage currents, and then local power stations step down the voltage for local distribution. According to the power formula, P = I * V, the current decreases when the voltage is increased. Given that the impedance of a cable is determined by the material it is made of, Ohm's law (V = I * R) tells us that using a high-impendence material means that the decrease in current will reduce power loss during transmission. Thus, using high-voltage power lines reduces the cost of moving electrical power from one place to another.

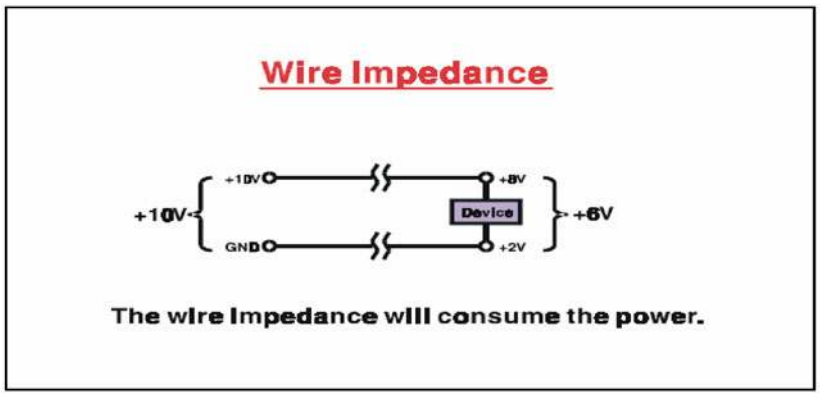


Figure C.6 Wire Impedance

C.2.5 Single-Point Grounding

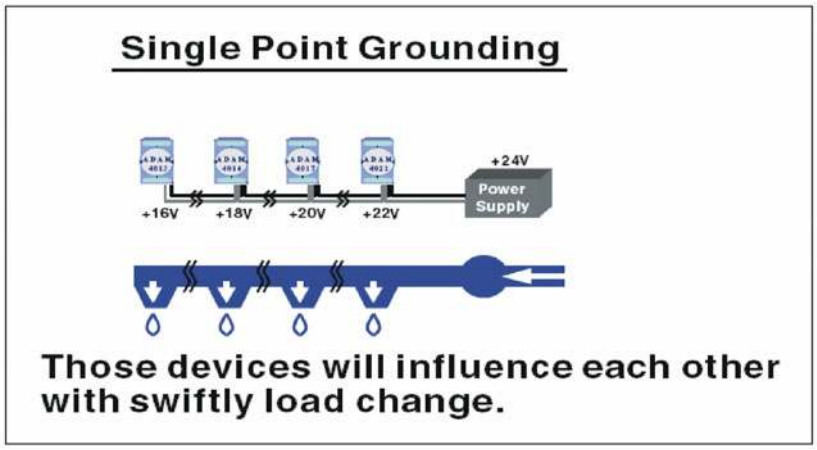


Figure C.7 Single-Point Grounding

Single-point grounding can be understood by considering a simple household water system; when one person is taking a hot shower and someone else turns on a hot water faucet, the water in the shower will become cold. The lower section of Figure C.7 depicts this concept and shows how devices will be influenced when there is a sudden change in load. In this example, assume that all four outlets are open; when Outlets 3 and 4 are closed, the flow to the other two outlets will increase. In other words, the flow rate will not be constant.

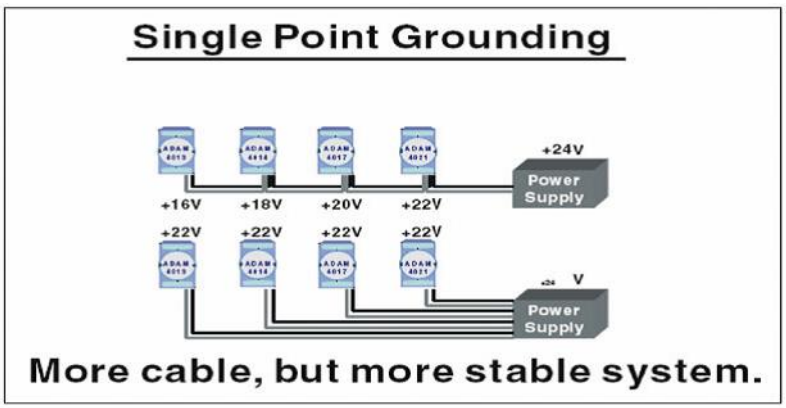


Figure C.8 Single point grounding

Figure C.8 shows how a single-point grounding system will be a more stable system than if only one cable is used. If you use thin cable for powering these devices, the end device will actually receive less power than the other devices. The thin cable will consume the energy.

C.3 Shielding

C.3.1 Cable Shield

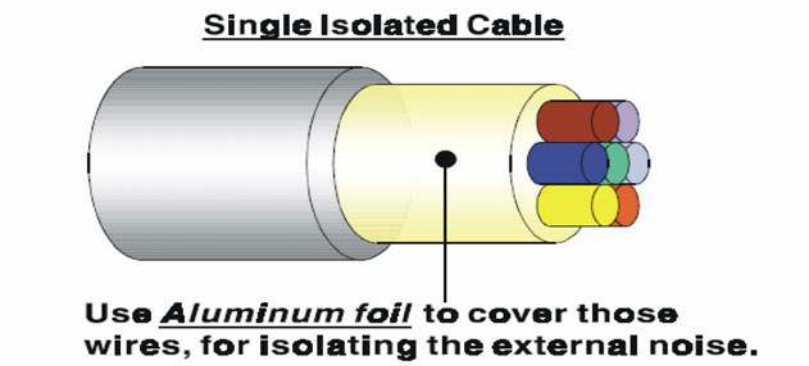


Figure C.9 Single Isolation Cable

Single Isolation Cable

Figure C.9 shows the structure of isolation cable. The image shows the isolation layer, which is spiraled aluminum foil that covers the wires. This spiraled structure provides a shielding layer that protects the cables from external noise.

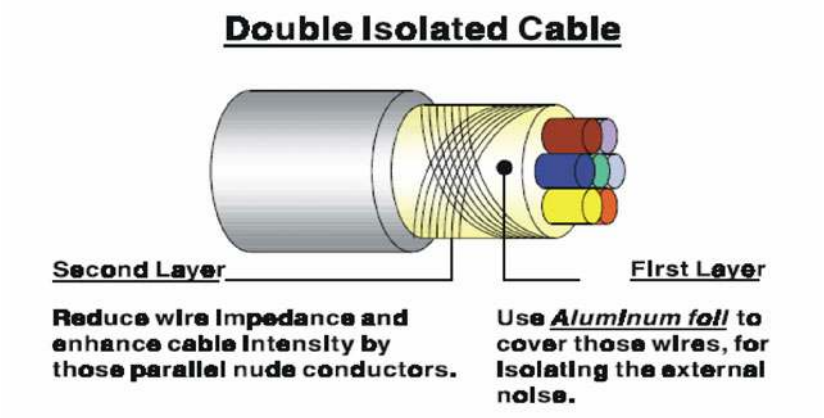


Figure C.10 Double Isolation Cable

Double Isolation Cable

Figure C.10 shows the structure of double isolation cable. Here, the first isolation layer of spiraled aluminum foil covers the conductors and the second isolation layer, which comprises several bare conductors, spirals and crosses over the first shield layer. This spiraled structure forms an isolation layer that, compared to single isolation cable, is more efficient for reducing external noise.

To maximize noise reduction, the following tips should be considered:

- Do not use a cable shield as a signal ground. The shield is designed to carry noise, so any environmental noise will couple and interfere with your system.
- Higher density shielding is better, especially for communication networks.

- It is best to use double isolation cable for communication networks as well as analog I/O applications.
- Both sides of shields should be connected to their frame while inside the device (for EMI considerations)
- Do not strip an excessive amount of the plastic cover when soldering.

C.3.2 System Shielding

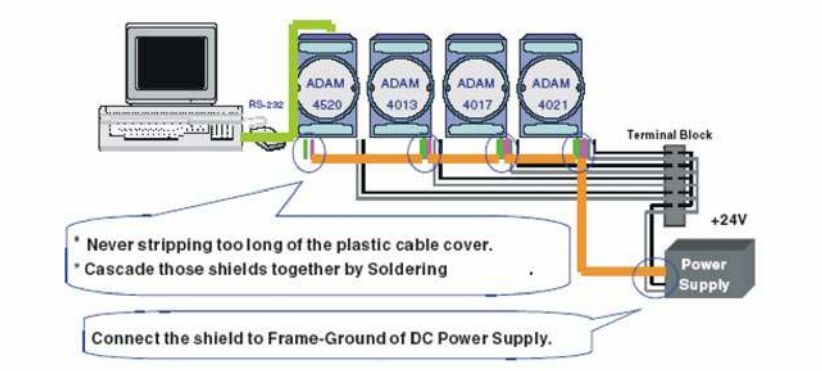


Figure C.11 System Shielding

- Never strip too much of the plastic cable cover. This can destroy the characteristics of the STP cable, and bare wire shield easily conducts noise.
- Cascade shields together by soldering (refer to the following figure for further details).
- Connect the shield to the frame ground of a DC power supply to force the conducted noise to flow to the frame ground of the DC power supply (the frame ground of the DC power supply should be connected to the system ground).

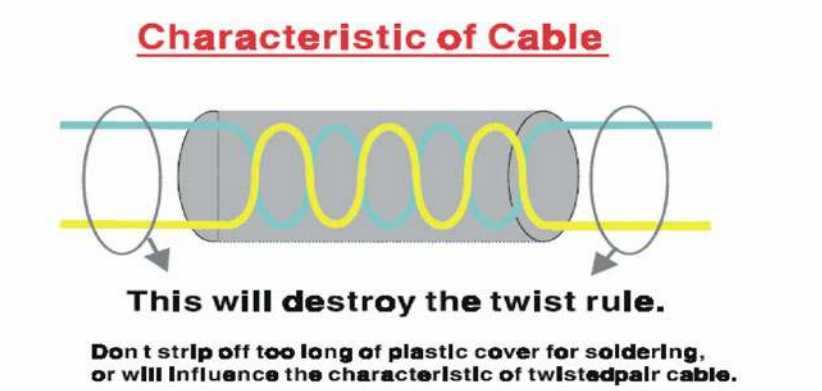


Figure C.12 The Characteristics of the Cable

Remember that you should not strip off too much insulation for soldering. This could reduce the effectiveness of the STP cable and open a path that introduces unwanted noise.

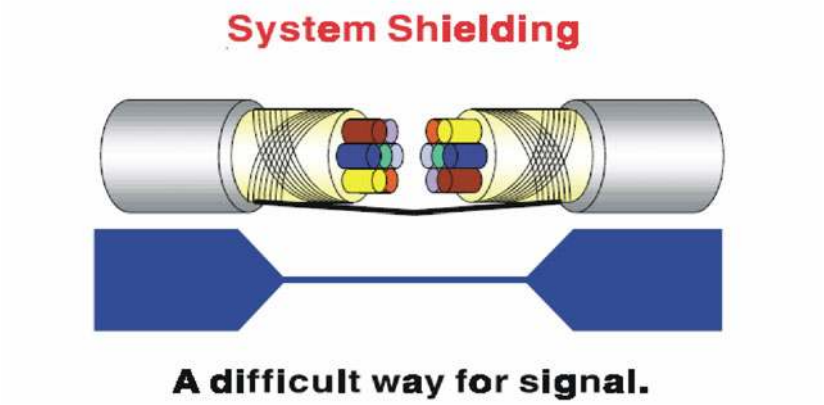


Figure C.13 System Shielding (1)

Shield Connection (1)

Be careful not to rush when setting you a system, as this can easily lead to cable breakage. As is the case with all electronic circuits, a signal will always take the path of least resistance. Thus, if the connection between two cables is poor, this will create a poor path for the signal. This may cause the noise to find another path with less resistance and thereby introduce interference.

Figure C.14 System Shielding (2)

Shield Connection (2)

Figure C.14 shows you that the fill soldering just makes an easier way for the signal.

C.4 Noise Reduction Techniques

Consider the following techniques when seeking to implement noise reduction:

- Isolate noise sources in shielded enclosures
- Place sensitive equipment in a shielded enclosure and away from computer equipment
- Use separate grounds between noise sources and signals
- Keep ground/signal leads as short as possible
- Use twisted and shielded signal leads
- Ground shields on one end ONLY when the reference grounds are not the same
- Check for stability in communication lines
- Add additional grounding bars if necessary
- The diameter of a power cable must be >2.0 mm2

- Independent grounding is needed for analog I/Os and communication networks while using a jumper box
- Use noise reduction filters if necessary (TVS, etc.)
- Refer to FIPS 94 Standard, which recommends that the computer system should be placed closer to its power source to eliminate load-induced common mode noise

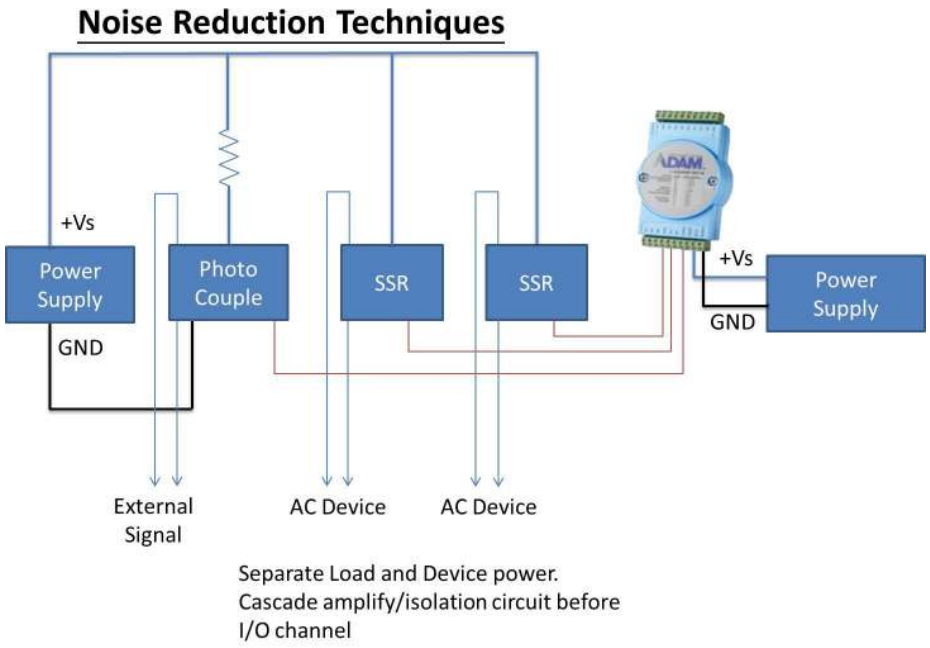


Figure C.15 Noise Reduction Techniques

C.5 Checklist

- Have you followed the single-point grounding rule?
- Normal mode and common mode voltage?
- Have the DC and AC ground been separated?
- Reject the noise factor?
- Is the shield connected correctly?
- Is the wire size correct?
- Are the soldered connections good?
- Are the terminal screw are tight?

Appendix D

REST for ADAM-6000

D.1 REST Introduction

REpresentational State Transfer (REST) is a software architecture for web applications and services including image indication, resource requests/responses, and message delivery. It can be developed to be compatible with common protocols and standards such as HTTP, URI, XML, and HTML. With the advantage of scalability, simplicity, and performance, REST has been adopted in web services by Amazon and Yahoo.

The web service of ADAM-6000 series modules is based on HTML5. Should you need to integrate this into other web services, the following information/command list will provide a useful reference for implementation.

D.2 REST Resources for ADAM

D.2.1 Analoginput

GET/analoginput/(all|{id})/value

Request	ContentType: application/x-www-form-urlencoded {id}: the analog input channel ID (starting from 0)
Examples	Use the following URI to get the value of AI-0: http://10.0.0.1/analoginput/0/value Use the following URI to get the all analog input values: http://10.0.0.1/analoginput/all/value
Response	ContentType: text/xml If the result is OK, the content will appear as follows: <?xml version="1.0" ?> <ADAM-6017 status="OK"> <AI> <ID>0</ID> <VALUE>7FFF</VALUE> </AI> </ADAM-6017> If the request fails, the content will appear as follows: <?xml version="1.0" ?> <ADAM-6017 status="{error}"> </ADAM-6017> {error}: The error message
Remarks	If the {id} is out of range, the response will return HTTP status code 501 (not implemented). The content of <VALUE> is in hex format (range 0000~FFFF), which maps to the min./max. of the range (analog input value is 16 bits).

GET /analoginput/(all|{id})/range

Request	ContentType: application/x-www-form-urlencoded {id}: the analog input channel ID (starting from 0)
Examples	Use the following URI to get the input range of AI-0: http://10.0.0.1/analoginput/0/range Use the following URI to get all analog input ranges: http://10.0.0.1/analoginput/all/range

Response	ContentType: text/xml If the result is OK, the content will appear as follows: <?xml version="1.0" ?> <ADAM-6017 status="OK"> <AI> <ID>0</ID> <RANGE>7</RANGE> <NAME>4~20 mA</NAME> <MAX>20</MAX> <MIN>4</MIN> <UNIT>mA</UNIT> </AI> </ADAM-6017> If the request fails, the content will appear as follows: <?xml version="1.0" ?> <ADAM-6017 status="{error}"> </ADAM-6017> {error}: The error message
Remarks	If the {id} is out of range, the response will return HTTP status code 501 (not implemented).

D.2.2 Analogoutput

GET /analogoutput/(all|{id})/value

Request	ContentType: application/x-www-form-urlencoded {id}: the analog output channel ID (starting from 0)
Examples	Use the following URI to get the value of AI-0: http://10.0.0.1/analogoutput/0/value Use the following URI to get the all analog output values: http://10.0.0.1/analogoutput/all/value
Response	ContentType: text/xml If the result is OK, the content will appear as follows: <?xml version="1.0" ?> <ADAM-6224 status="OK"> <AO> <ID>0</ID> <VALUE>0FFF</VALUE> </AO> </ADAM-6224> If the request fails, the content will appear as follows: <?xml version="1.0" ?> <ADAM-6224 status="{error}"> </ADAM-6224> {error}: The error message
Remarks	If the {id} is out of range, the response will return HTTP status code 501 (Not implemented) The content of <VALUE> is in hex format (range 0000~0FFF), which maps to the min./max. value of the range (analog output value is 12 bits).

POST /analogoutput/all/value

Request	ContentType: application/x-www-form-urlencoded

Examples	Use the following URI to set the analog output value(s) http://10.0.0.1/analogoutput/all/value The data accompanying the request will be given as {name}={value} pair(s) {name}: The channel name (e.g., AO-0) {value}: The value to be set to the indicated channel For example, if the request is going to set Channels 0, 1, and 2 to a value of 3, then the name-value pairs will appear as follows: AO0=00FF&AO1=0000&AO2=0FFF
Response	ContentType: text/xml The content will appear as follows: <?xml version="1.0" ?> <ADAM-6224 status="{status}"> </ADAM-6224> {status}: The result (if successful, the result will be 'OK'; otherwise, the result will be the error message).
Remarks	The {value} of the post data is in HEX format and from 0000 to 0FFF, which maps to the min./max. value of the range (analog output value is 12 bits).

GET /analogoutput/(all|{id})/range

Request	ContentType: application/x-www-form-urlencoded {id}: the AO channel ID (starting from 0)
Examples	Use the following URI to get the AO-0 range: http://10.0.0.1/analogoutput/0/range Use the following URI to get all analog output ranges: http://10.0.0.1/analogoutput/all/range
Response	ContentType: text/xml If the result is OK, the content will appear as follows: <?xml version="1.0" ?> <ADAM-6224 status="OK"> <AO> <ID>0</ID> <RANGE>7</RANGE> <NAME>4~20 mA</NAME> <MAX>20</MAX> <MIN>4</MIN> <UNIT>mA</UNIT> </AO> </ADAM-6224> If the request fails, the content will appear as follows: <?xml version="1.0" ?> <ADAM-6224 status="{error}"> </ADAM-6224> {error}: The error message
Remarks	If the {id} is out of range, the response will return HTTP status code 501 (not implemented)

D.2.3 Digitalinput

GET /digitalinput/(all|{id})/value

Request	ContentType: application/x-www-form-urlencoded {id}: the DI channel ID (starting from 0)
Examples	Use the following URI to get the DI-0 value: http://10.0.0.1/digitalinput/0/value Use the following URI to get all digital input values: http://10.0.0.1/digitalinput/all/value

Response	ContentType: text/xml If the result is OK, the content will appear as follows: <?xml version="1.0" ?> <ADAM-6050 status="OK"> <DI> <ID>0</ID> <VALUE>0</VALUE> </DI> </ADAM-6050> If the request fails, the content will appear as follows: <?xml version="1.0" ?> <ADAM-6050 status="{error}"> </ADAM-6050> {error}: The error message
Remarks	If the {id} is out of range, the response will return HTTP status code 501 (not implemented)

D.2.4 Digitaloutput

GET /digitaloutput/(all|{id})/value

Request	ContentType: application/x-www-form-urlencoded {id}: the digital output channel ID (starting from 0)
Examples	Use the following URI to get the DO-0 value: http://10.0.0.1/digitaloutput/0/value Use the following URI to get all digital output values: http://10.0.0.1/digitaloutput/all/value
Response	ContentType: text/xml If the result is OK, the content will appear as follows: <?xml version="1.0" ?> <ADAM-6050 status="OK"> <DO> <ID>0</ID> <VALUE>1</VALUE> </DO> </ADAM-6050> If the request fails, the content will appear as follows: <?xml version="1.0" ?> <ADAM-6050 status="{error}"> </ADAM-6050> {error}: The error message
Remarks	If the {id} is out of range, the response will return HTTP status code 501 (not implemented)

POST /digitaloutput/all/value

Request	ContentType: application/x-www-form-urlencoded
Examples	Use the following URI to set the digital output value(s): http://10.0.0.1/digitaloutput/all/value The data accompanying the request will be given as {name}={value} pair(s) {name}: The channel name (e.g., DO-0) {value}: The value to be set to the indicated channel For example, if the request is going to set Channel 0, 1, and 2 to a value of 1, then the name-value pairs would be as follows: DO0=1&DO1=1&DO2=1

Response	ContentType: text/xml The content will appear as follows: <?xml version="1.0" ?> <ADAM-6050 status="{status}"> </ADAM-6050> {status}: The result. If successful, the result will be "OK"; otherwise, the result will be the error message.
Remarks	

D.2.5 Counter

GET /counter/(all|{id})/value

Request	ContentType: application/x-www-form-urlencoded {id}: the counter channel ID (starting from 0)
Examples	Use the following URI to get the Counter-0 value: http://10.0.0.1/counter/0/value Use the following URI to get the all Counter values: http://10.0.0.1/counter/all/value
Response	ContentType: text/xml If the result is OK, the content will appear as follows: <?xml version="1.0" ?> <ADAM-6051 status="OK"> <CNT> <ID>0</ID> <VALUE>102938</VALUE> </CNT> </ADAM-6051> If the request fails, the content will appear as follows: <?xml version="1.0" ?> <ADAM-6051 status="{error}"> </ADAM-6051> {error}: The error message
Remarks	If the {id} is out of range, the response will return HTTP status code 501 (not implemented)

ADVANTECH

Enabling an Intelligent Planet

www.advantech.com

Please verify specifications before quoting. This guide is intended for reference purposes only.

All product specifications are subject to change without notice.

No part of this publication may be reproduced in any form or by any means, such as electronically, by photocopying, recording, or otherwise, without prior written permission from the publisher.

All brand and product names are trademarks or registered trademarks of their respective companies.

© Advantech Co., Ltd. 2022