INSTRUCTION MANUAL

Models 2518/2519
Ethernet Smart A/D™

Revised July 25, 2003

Sensoray Co., Inc.
7313 SW Tech Center Dr., Tigard, Oregon 97223
voice: 503.684.8005, fax: 503.684.8164, e-mail: sales@sensoray.com
www.sensoray.com

Table of Contents

Basics

Introduction

Network Interface

Configuration

Sensor Connections

Programming

Calibration

Smart A/D™ Theory of Operation

Smart A/D™ Timing

Specifications

Chapter 1: Basics

1.1 Limited Warranty

Sensoray Company, Incorporated (Sensoray) warrants the Model 2518/2519 hardware to be free from defects in material and workmanship and perform to applicable published Sensoray specifications for two years from the date of shipment to purchaser. Sensoray will, at its option, repair or replace equipment that proves to be defective during the warranty period. This warranty includes parts and labor.

The warranty provided herein does not cover equipment subjected to abuse, misuse, accident, alteration, neglect, or unauthorized repair or installation. Sensoray shall have the right of final determination as to the existence and cause of defect.

As for items repaired or replaced under warranty, the warranty shall continue in effect for the remainder of the original warranty period, or for ninety days following date of shipment by Sensoray of the repaired or replaced part, whichever period is longer.

A Return Material Authorization (RMA) number must be obtained from the factory and clearly marked on the outside of the package before any equipment will be accepted for warranty work. Sensoray will pay the shipping costs of returning to the owner parts that are covered by warranty.

Sensoray believes that the information in this manual is accurate. The document has been carefully reviewed for technical accuracy. In the event that technical or typographical errors exist, Sensoray reserves the right to make changes to subsequent editions of this document without prior notice to holders of this edition. The reader should consult Sensoray if errors are suspected. In no event shall Sensoray be liable for any damages arising out of or related to this document or the information contained in it.

EXCEPT AS SPECIFIED HEREIN, SENSORAY MAKES NO WARRANTIES, EXPRESS OR IMPLIED, AND SPECIFICALLY DISCLAIMS ANY WARRANTY OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. CUSTOMER'S RIGHT TO RECOVER DAMAGES CAUSED BY FAULT OR NEGLIGENCE ON THE PART OF SENSORAY SHALL BE LIMITED TO THE AMOUNT THERETOFORE PAID BY THE CUSTOMER. SENSORAY WILL NOT BE LIABLE FOR DAMAGES RESULTING FROM LOSS OF DATA, PROFITS, USE OF PRODUCTS, OR INCIDENTAL OR CONSEQUENTIAL DAMAGES, EVEN IF ADVISED OF THE POSSIBILITY THEREOF.

1.2 Special Handling Instructions

Models 2518 and 2519 hardware assemblies contain CMOS circuitry that is sensitive to Electrostatic Discharge (ESD).

Special care should be taken in handling, transporting, and installing the hardware assembly to prevent ESD damage. In particular:

- ❑ Do not remove the hardware assembly from its protective anti-static bag until you are ready to configure it for installation.
- ❑ Handle the hardware assembly only at grounded, ESD protected stations.
- ❑ Remove power from the target system before installing or removing the hardware assembly.

Chapter 2: Introduction

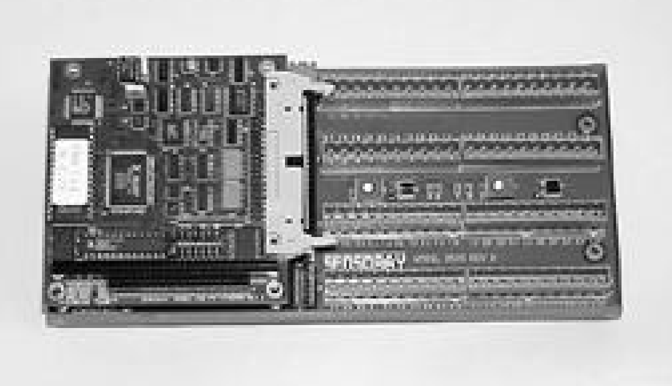
2.1 Functional Description

Sensoray Smart A/D™ models 2518 and 2519 directly interface eight (model 2518) or sixteen (model 2519) process sensors to a 10BASE-T Ethernet network.

These models are members of Sensoray's Ethernet I/O product family. Models 2518 and 2519 function as I/O Servers in a client-server network model. As many as four clients may share access to a single Ethernet Smart A/D™ server.

Models 2518 and 2519 are modular assemblies that consist of one or two model 518 Smart A/D boards mounted onto a model 2509 Ethernet base unit.

Figure 1: Model 2518 8-channel Ethernet Smart A/D™



2.1.1 Ethernet Base Unit

The model 2509 Ethernet base unit is a DIN-mountable circuit board that combines five functional elements: DC-DC converter, intelligent network interface, PC/104 interface, field wiring terminations and optional DIN rail mounting system.

The on-board DC-DC converter accepts any input voltage from 12 to 24V AC/DC and produces all internal supply voltages required by the unit.

A high performance, dedicated network processor is coupled with an optimized TCP/IP stack. This powerful combination simplifies software development and accelerates application performance on virtually any Ethernet-capable client.

Implemented TCP/IP protocols include IP as a foundation for other network layers, UDP for command and acquisition, ICMP for diagnostics, and ARP for network address resolution.

Model 2509 provides a standard PC/104 bus connector and an electrical subset of the PC/104 bus. This PC/104 interface, along with the on-board DC-DC converter, is optimized for supporting a stack of one or two model 518 Smart A/D™ boards.

Integral screw terminals are provided for connecting the Smart A/D™ sensor channels to external sensor field wiring. These robust, pluggable screw terminal blocks simplify installation and enhance maintainability.

The circuit board can be rigidly supported by an optional DIN mounting frame that enables virtually instant snap-on installation onto standard DIN mounting rails.

2.1.2 Smart A/D™ Daughter Boards

Sensoray's model 518 Smart A/D™ is a standard PC/104 compatible board that interfaces eight process sensors to any PC/104 bus.

Each of eight sensor channels may be independently configured, via software, to accept thermocouples, RTD's, strain gages, thermistors, resistors, 4-to-20 mA current loops, or DC voltage inputs.

Pulsed excitation is provided for passive sensors, such as thermistors, resistors, RTD's, and strain gages. Pulsed excitation minimizes sensor self-heating effects and helps to reduce system power consumption.

An embedded microcomputer continuously scans the sensor channels. As each channel is scanned, its sensor is excited, digitized, linearized, converted to engineering units appropriate for the sensor type, and stored in on-board memory. The newest sensor data from all channels are always available for instant access.

2.1.3 Software

Comprehensive support software is supplied with models 2518 and 2519. The SDK includes:

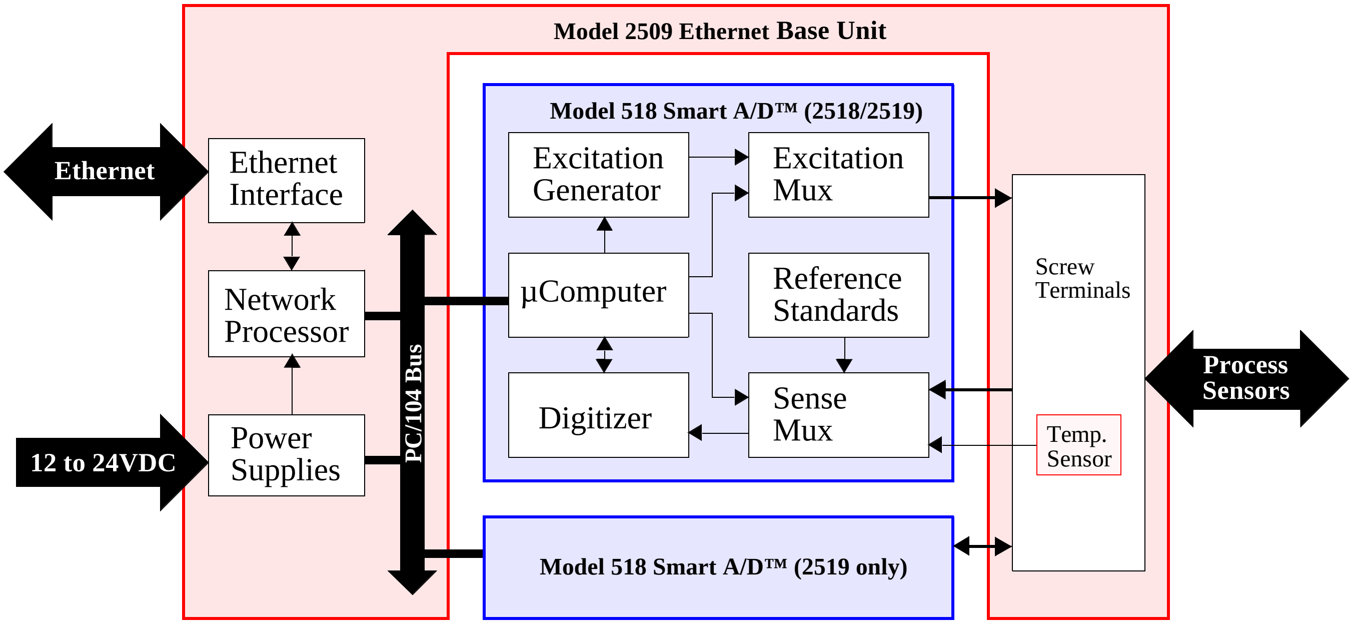
- ❑ A fully documented DLL which, when used with any standard Windows socket, provides straightforward access to all Smart A/D™ functions.
- ❑ Numerous Visual Basic programming examples to help jump-start application development.

2.2 Product Features

- ❑ Remote operation on any 10BASE-T Ethernet network.
- ❑ Up to 4096 sensor channels per network, using up to 256 distributed Ethernet Smart A/D™ units.
- ❑ High-performance dedicated network processor with optimized TCP/IP stack delivers up to 1400 transactions per second.
- ❑ On-board DC-DC converter requires only a single 12V to 24V AC/DC input power source.
- ❑ Robust, pluggable screw terminal blocks support direct sensor connections.
- ❑ Network watchdog timer monitors Ethernet activity.
- ❑ Embedded processors off-load computationally intensive data acquisition functions from the client.
- ❑ Each sensor channel may be dynamically configured to accept any supported sensor type.
- ❑ Fully differential sense inputs help to eliminate ground loops.
- ❑ All sense inputs are protected from high differential and common mode voltages.
- ❑ Pulsed excitation provided for thermistors, resistors, RTDs and strain gages. Pulsed excitation minimizes sensor self-heating and helps to reduce system power consumption.
- ❑ Sensor excitation signals are electrically separate from the sense inputs so that high accuracy three-wire or four-wire circuits may be implemented.
- ❑ Automatic reference junction compensation provided for thermocouples. Integral compensation sensors occupy dedicated measurement channels so that all process sensor channels remain available for application use.
- ❑ Programmable alarm thresholds for each channel. A single Alarm flag enables the rapid detection of limit violations without burdening clients with sensor data evaluation.
- ❑ High quality internal reference standards for excellent long-term stability and accuracy.

2.3 Block Diagram

Figure 2: Model 2518/2519 Block Diagram



2.4 Watchdog Timer

Two watchdog timers are implemented on the Ethernet base board. One timer is implemented in hardware and the other in software. Either of the timers will invoke a reset of the entire Smart A/D™ unit if a time-out occurs.

- The *hardware* watchdog timer will time out if it is not kept refreshed by the local network processor. This timer guards against disruptions due to local power transients and circuit failures.
- The *software* watchdog timer will time out if no personal network packets have been received for a period of ten seconds. This timer protects against system faults such as network and client failures.

2.5 Reset State

A hardware reset of the Ethernet base unit is invoked in response to any of the following events:

- Power-up.
- Watchdog time-out.

Smart A/D™ daughter boards experience a hardware reset whenever the Ethernet base unit performs a hardware reset. In addition, invoking a soft reset (see Section 6.3.1) on a specific Smart A/D™ daughter board will cause that board to undergo a hardware reset.

2.5.1 Reset State

Upon the occurrence of a reset, all Smart A/D™ daughter boards are in the default state as described below:

- ❑ Standard speed mode (22ms/channel) is selected.
- ❑ 60Hz rejection mode is active.
- ❑ All channels are enabled for scanning.
- ❑ Sensor data values are reset to zero. Sensor data is not available immediately after a board reset. The maximum elapsed time that must pass before new sensor data becomes available is equal to 35 channel slot times. Details of channel slot timing can be found in Section 9.2
- ❑ Channels are configured for the 5-volt input range, with 500µV resolution.
- ❑ Software filter constants are set to zero (disabled).
- ❑ Alarm threshold limit values are indeterminate.
- ❑ All channel alarms are disabled.
- ❑ All channels are programmed to fail high upon open-sensor detection.

2.6 Visual Indicators

2.6.1 Fault Indicator

A red light-emitting diode (LED), located on each Smart A/D™ daughter board, is used as a status indicator. This LED is energized as a result of any of the following conditions:

- A board reset is in progress.
- A board fault has been detected.

When a Smart A/D™ is functioning normally, its LED is energized for approximately one-half second in response to either a hardware or a soft reset. In the case of hardware resets, this time period may be longer than 0.5 seconds due to stretching of the reset signal on the local PC/104 bus.

If the LED is persistently energized, there may be a hardware fault on the A/D board. In this case, be sure to check all sensor connections, board orientation and seating, etc. before concluding that the board has a fault.

2.6.2 Heartbeat Indicator

The model 2509 base board includes a LED, labeled “RUN,” that flashes on and off at approximately two Hertz. When flashing at this rate, the RUN LED indicates that the network processor is functioning normally.

2.6.3 Network Status Indicators

Four LEDs on the model 2509 Ethernet base board are used to indicate the status of the Ethernet interface:

- **XMT** — Active for approximately 0.25 seconds when the 2509 transmits a packet.
- **RCV** — Active for approximately 0.25 seconds when the 2509 receives a packet addressed to its Ethernet address.
- **LNK** — Active whenever Link pulses are being detected on the wire.
- **SEL** — Active for approximately 0.25 seconds when the board’s network processor accesses the Ethernet interface circuitry.

Chapter 3: Network Interface

3.1 The Smart A/D™ Paradigm

By virtue of the I/O services they provide, Smart A/D™ products are used exclusively in measurement and control systems. Such systems nearly always rely on deterministic responses from I/O devices in order to acquire data and control processes in real time.

Neither Ethernet nor the TCP/IP protocols were designed to inherently provide real time response. Consequently, the real time performance of an Ethernet measurement and control network depends entirely on the responsiveness of network hosts and on the volume and nature of the network traffic.

Smart A/D™ products are designed to provide exceptionally fast network response times in order to guarantee deterministic, real-time behavior over suitable Ethernet local area networks (LANs). The Smart A/D™ paradigm is this: *Deliver real-time I/O services over Ethernet LANs.*

3.2 I/O Transaction Time

In Smart A/D™ parlance, a *transaction* is the process of receiving a service request from a client, executing the service function, and if required, transmitting a response back to the client. The measure of how well an I/O server will perform in a real time system is determined by the number of transactions per second it is capable of.

Smart A/D™ servers are designed from the ground up to provide exceptionally short transaction times. By using small, efficient packets, and by combining a dedicated network processor with a tightly-coupled, optimized protocol stack, these servers are capable of processing 1400 transactions per second at a sustained rate.

3.2.1 Influence of Network Traffic

Real time I/O behavior can be degraded by excessive network traffic, particularly if large packets are present on the wire. This might be the case, for example, in a general-purpose LAN with many hosts.

This is not to say that I/O servers can't operate on a general-purpose LAN; it is simply an observation that if deterministic performance is to be attained, an I/O server must reside on a LAN with traffic characteristics that are compatible with the desired transaction rates.

In systems requiring high I/O transaction rates or fast response times, the requisite LAN is one that is dedicated to I/O functions. Such a network would enable several Smart A/D™ servers to provide I/O services at high transaction rates without risking interruption from FTP transfers, etc.

3.3 IP Addresses

All Sensoray Ethernet products employ standard Internet Protocol (IP) version 4 addresses. An IP address is a logical address that is assigned to a particular network host. Every host on a TCP/IP network is assigned an IP address so that it can communicate with other hosts on the network.

In its physical form, an IP address is a 32-bit binary value. At the user-interface level, however, IP addresses are almost always represented in a more readable form known as *dotted decimal* notation.

For example, "23.145.172.11" is an IP address written in dotted decimal form. The "dots," which are decimal points, serve to visually delimit the four bytes that constitute the 32-bit binary value. The four delimited bytes are expressed as decimal values.

3.3.1 IP Address Allocation

Sensoray Ethernet products utilize fixed, jumper-selectable IP addresses. This address allocation method has several important benefits over other methods:

- ❑ Smart A/D™ products can be easily integrated into nearly any LAN.
- ❑ Special network support components, such as DHCP servers, are not required. More importantly, all of the side effects of DHCP that negatively impact deterministic behavior are eliminated.
- ❑ Configuration is fast and simple; clients that require I/O services are inherently aware of the affiliated Smart A/D™ servers on a network.
- ❑ System downtime is minimal if a Smart A/D™ product should ever require replacement. Any unit can be quickly replaced by substituting another unit with the same IP address.

3.4 Supported Protocols

TCP/IP is not a single protocol, but rather is a suite of protocols. Not all of the TCP/IP protocols need to be implemented to take advantage of the standards defined by the suite. In fact, significant performance improvements can be realized by implementing only those protocols that are needed for a particular application.

In light of this, Smart A/D™ servers implement only the protocols that are fundamental to providing high-performance I/O services and diagnostics on a dedicated LAN.

3.4.1 Address Resolution Protocol (ARP)

ARP is used to establish the mapping between IP addresses and their corresponding physical Ethernet (MAC) addresses. ARP exists at the boundary between internet protocols and physical network; it is not technically part of the internet protocols. ARP transactions are transparent to the application process.

When a client first connects to a Smart A/D™ server it doesn't know the server's MAC address; it only knows the server's IP address. In order to direct packets to a specific server, the client must first determine the destination MAC address. The following sequence of events occurs when the client first sends data to a Smart A/D™ server:

1. The client broadcasts an *ARP request* to the server's IP address. This is a message to the server, asking it to respond with its MAC address.
2. The server responds by sending an *ARP response* back to the requesting client. This message contains the server's MAC address. The server installs the clients IP/MAC address pair into an *address resolution cache* that it maintains in local memory.
3. The client extracts the server's MAC address from the ARP response and installs the server's IP/MAC address pair to its own address resolution cache.
4. Now that the server's MAC address is known, the client sends the data to the server.

The next time the client wishes to send data to the server, it uses the cached address instead of executing another time- and bandwidth-consuming ARP transaction.

3.4.1.1 ARP Implementation

With regard to address resolution caches, Smart A/D™ servers deviate in several ways from general-purpose hosts. These deviations exist for the purpose of providing the highest possible performance. Specifically:

- General-purpose hosts maintain their address resolution caches by expiring cache entries that have not been accessed for some time interval, typically one minute. *Smart A/D™ servers expire cache entries only upon reset.*
- General-purpose hosts often employ large address resolution caches to support the potentially large number of hosts that might exist on a network. *The address resolution cache on Smart A/D™ servers is limited to four entries.*

3.4.2 Internet Protocol (IP)

IP is the foundation for all of the other TCP/IP protocols employed on Smart A/D™ products. All ICMP and UDP messages are encapsulated in IP datagrams.

3.4.3 Internet Control Message Protocol (ICMP)

Technically, ICMP is an error reporting mechanism. As implemented on Smart A/D™ products, however, ICMP serves as a diagnostic tool for testing reachability.

Smart A/D™ products support two ICMP message types: Echo Request and Echo Reply. In response to receipt of an Echo Request message, a Smart A/D™ server will send an Echo Reply message. Smart A/D™ servers never originate Echo Request messages.

Most general-purpose hosts have a "Ping" utility that can be used to initiate ICMP echo transactions. This is an important diagnostic tool for troubleshooting networks that incorporate Smart A/D™ products, as the Smart A/D™ servers can be "ping'ed" independent of any application software.

3.4.4 User Datagram Protocol (UDP)

UDP is the transport layer mechanism by which commands are sent to and sensor data is received from Smart A/D™ boards. This protocol, combined with application layer handshaking, provides a reliable, high-performance message delivery service when used on dedicated I/O networks.

3.5 Network Architecture

All networks that employ Smart A/D™ servers conform to the classic client-server network model. In the client-server network model, one or more clients access the services of one or more servers.

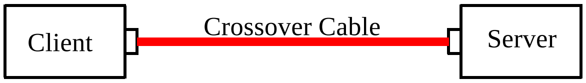
At least one hub is required in any 10BASE-T Ethernet network that connects three or more hosts. Hubs provide a means for connecting hosts to a network.

3.5.1 Trivial Networks

In its simplest form, a client-server Ethernet network consists of two hosts: a single client using the services of a single server. This is the only configuration that does not require the use of a hub.

A simple network, consisting of one client and one server is shown in Figure 3. The server can be either a model 2518 or 2519, depending on whether 8 or 16 sensor channels are needed.

Figure 3: The simplest possible Smart A/D™ network



3.5.2 Non-Trivial Networks

A non-trivial network is one in which three or more hosts are physically connected to the network. There are many situations where a non-trivial network is desirable. Some of the more common possibilities are explored below.

3.5.2.1 Higher Channel Counts

One reason for implementing a non-trivial network is to provide more sensor channels than a single Smart A/D™ server can supply. For example, Figure 4 shows a network that provides 64 sensor channels by means of four model 2519 Smart A/D™ servers.

Figure 4: A network with 64 clustered sensor channels

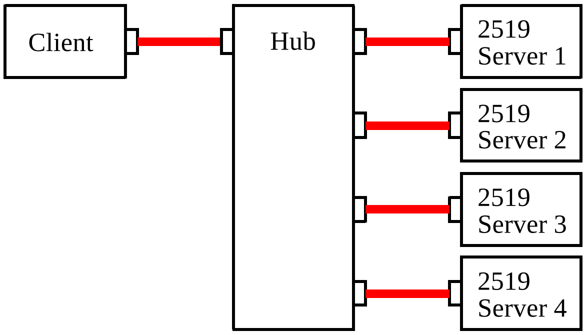
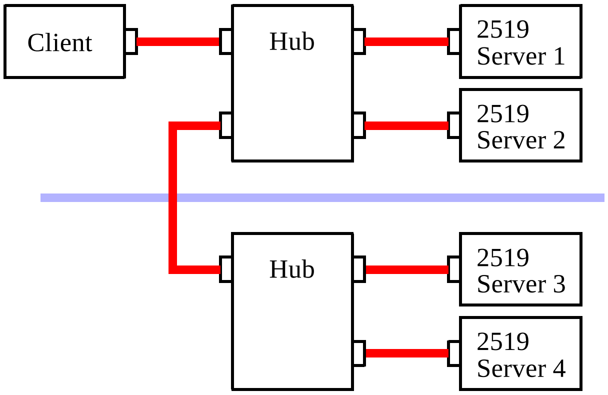


Figure 4 illustrates how one might configure a 64 channel network in the case where all of the sensors are clustered together in close physical proximity.

3.5.2.2 Distributed I/O

Many applications require sensors to be physically distributed throughout a system. Figure 5 illustrates how a network could be configured to accommodate a situation in which two clusters of 32 sensor channels are located a significant distance apart from one another.

Figure 5: A network with two clusters of 32 sensor channels



There are several important advantages to organizing systems into clusters as shown in Figure 5:

- ❑ Wiring complexity is minimized. All sensor field wiring is local to the Ethernet unit.
- ❑ Wire and assembly costs are significantly lowered.
- ❑ Measurement performance is enhanced due to reduced ambient noise pick-up.

3.5.2.3 Distributed Processing

Sometimes it is desirable to distribute processing among multiple clients. In such systems, each client may be allocated a subset of the available sensor channels. In essence, multiple logical networks are operating over a single physical network.

Figure 6: A distributed processing network with 32 sensor channels

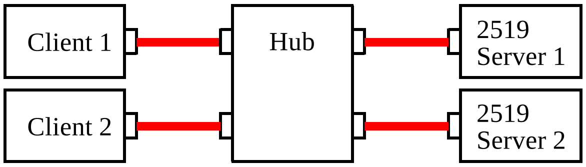


Figure 6 shows an example of a distributed processing system with two clients. Client 1 might be assigned five sensor channels on Server 2, while Client 2 might be assigned the remaining channels on Server 2 and all channels on Client 1. Note that any particular sensor channel may be accessed by more than one client.

3.6 Other Information Sources

It is beyond the scope of this document to discuss detailed network technical issues. If you require more information, an excellent source is *Internetworking with TCP/IP*, by Douglas Comer.

Chapter 4: Configuration

4.1 Overview

Various hardware options are configured by installing or removing shorting shunts at various locations on the Ethernet Smart A/D™ unit. All options must be configured before the unit is installed into the target system.

Some hardware options are configured by setting shunts on the model 2509 Ethernet base board, while other options are configured by setting shunts on the model 518 Smart A/D™ daughter boards.

A supply of shunts, of sufficient number to program all options, is included with the unit.

4.2 Network IP Address Selection

The model 2509 IP address is programmed to a static address value by installing hardware shunts onto the board.

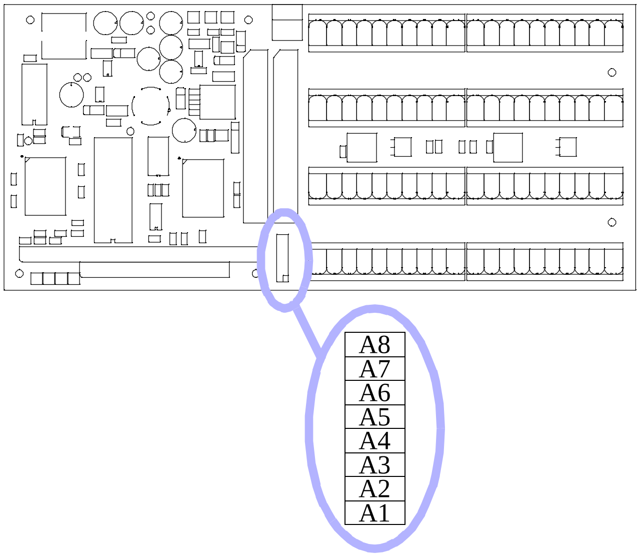
4.2.1 Shunt Locations

The procedure for setting the address is the same for all board revisions, but the board layout differs somewhat for different board revisions.

4.2.1.1 Revision A through D Boards

Eight shunt locations are provided for setting the IP address. These shunt locations, designated A1 to A8, behave as a group to establish the IP address of the model 2518/2519 unit.

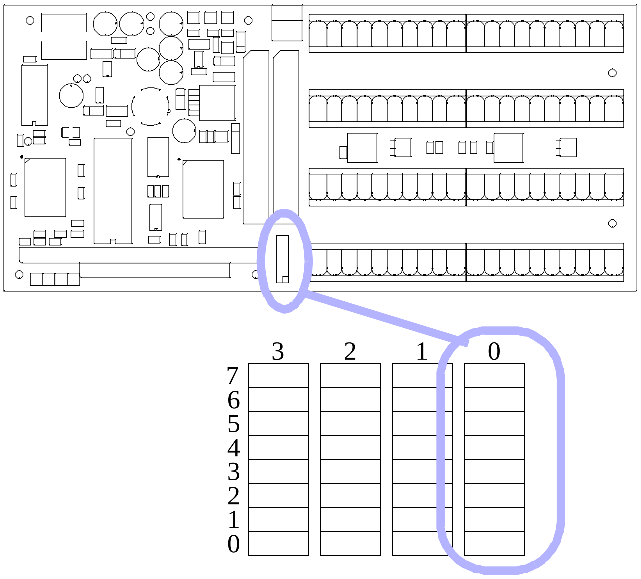
Figure 7: Model 2509 programming shunts, rev. A to rev. D



4.2.1.2 Revision E and Higher Boards

32 shunt locations are present on revision E boards. These locations are organized into four columns (labeled “0” to “3”) and eight rows (labeled “0” to “7”).

Figure 8: Model 2509 programming shunts, rev. E and higher



Only column “0” is used to set the board’s IP address. Shunts should never be installed in columns 1, 2 or 3.

4.2.2 Programming the IP Address

Of the four bytes that comprise the IP address, three are factory programmed to fixed values by means of firmware. The fourth byte of the address may be field-programmed by installing shunts.

Models 2518 and 2519 always respond to IP address `10.0.0.X`, where byte `X` is determined by option shunts column 0, rows 0 to 7 (revision E board) or A1 to A8 (all other revisions). Any address from `10.0.0.0` to `10.0.0.255` may be assigned, although the values 0 and 255 are not recommended as they may be treated as special cases by the network.

Table 1 illustrates how to install shunts for all of the possible values of the field-programmable IP address byte.

Table 1: IP Address programming shunts

Value	High Nibble			
	A8	A7	A6	A5
	7	6	5	4
0	●	●	●	●
1	●	●	●	❍
2	●	●	❍	●
3	●	●	❍	❍
4	●	❍	●	●
5	●	❍	●	❍
6	●	❍	❍	●
7	●	❍	❍	❍
8	❍	●	●	●
9	❍	●	●	❍
A	❍	●	❍	●
B	❍	●	❍	❍
C	❍	❍	●	●
D	❍	❍	●	❍
E	❍	❍	❍	●
F	❍	❍	❍	❍

Value	Low Nibble			
	A4	A3	A2	A1
	3	2	1	0
0	●	●	●	●
1	●	●	●	❍
2	●	●	❍	●
3	●	●	❍	❍
4	●	❍	●	●
5	●	❍	●	❍
6	●	❍	❍	●
7	●	❍	❍	❍
8	❍	●	●	●
9	❍	●	●	❍
A	❍	●	❍	●
B	❍	●	❍	❍
C	❍	❍	●	●
D	❍	❍	●	❍
E	❍	❍	❍	●
F	❍	❍	❍	❍

● indicates shunt is installed
❍ indicates shunt is removed.

Example:

The desired IP address is `10.0.0.42`. The programmable byte value, `42`, which is expressed in decimal, must first be converted to hexadecimal; decimal `42` = hex `2A`. Now, using Table 1, it can be seen that shunts must be installed as follows:

Table 2: Example - Select IP Address 10.0.0.42

Value	High Nibble			
	A8	A7	A6	A5
	7	6	5	4
2	●	●	❍	●

Value	Low Nibble			
	A4	A3	A2	A1
	3	2	1	0
A	❍	●	❍	●

On revision A-D boards, the installed shunts are A8, A7, A5, A3 and A1. On revision E boards, shunts must be installed on column 0, rows 7, 6, 4, 2 and 0

4.2.3 Default IP Address

Models 2518 and 2519 are factory programmed to set the unit's IP address to `10.0.0.127`.

4.2.4 Choosing an IP Address

On most networks, Ethernet IP addresses can be assigned arbitrarily so long as the assigned address does not conflict with another host on the same LAN.

In rare cases, addresses in the range `10.0.0.X` will not work. In such cases, Sensoray can provide custom programming of the units to provide alternative fixed IP address ranges. Contact *sales@sensoray.com* for details.

4.3 Model 518 Smart A/D™ Boards

Model 518 Smart A/D™ daughter boards have two groups of programming shunts. One group is used to set the board base I/O address, and the other group is used to enable channel signal conditioning circuits.

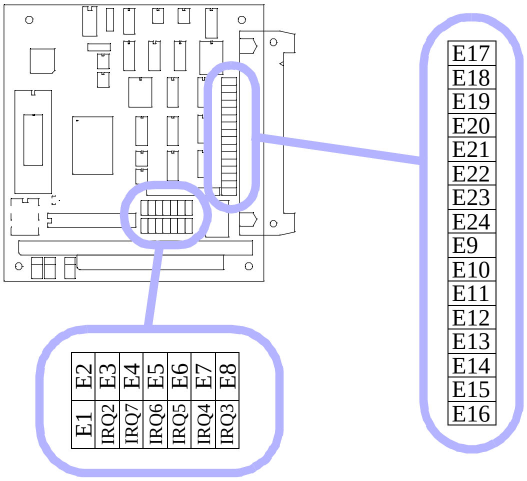
Model 2518 has one Smart A/D™ daughter board which interfaces to Smart A/D™ sensor channels 0 to 7. Model 2519 has two Smart A/D™ boards. The lower Smart A/D™ board interfaces to channels 0 to 7, while the upper board interfaces to channels 8 to 15.

Option shunts are easily accessed on the uppermost Smart A/D™ board. If option shunts must be modified on the lower Smart A/D™ board of a model 2519, you will need to remove the upper Smart A/D™ board to gain access to shunts on the lower board. This is accomplished by removing the four screws from the corners of the upper board, then carefully unplugging the upper board from the lower board. After the shunts have been set on the lower board, plug the upper board back onto the lower board and reinstall the four screws.

4.3.1 Shunt Locations

The following diagram shows the locations of hardware programming shunts on the model 518 Smart A/D™ board.

Figure 9: Model 518 programming shunts



4.3.2 Base Address Selection

IRQ and I/O addresses are factory programmed and should never need to be changed. When installed on a model 2509 Ethernet base board, model 518 daughter boards should be configured as follows:

- **Address (E1 to E8).** The lower model 518 board, which is the only model 518 board residing on model 2518, is set to base address 2B0 hex. In the case of model 2519 only, the upper model 518 board is set to base address 2B4 hex.
- **Interrupts (IRQ2 to IRQ7).** The lower model 518 board is set to generate interrupts on IRQ2, and on model 2519 only, the upper model 518 board is set to generate interrupts on IRQ3.

See the model 518 instruction manual for details on how to configure model 518 interrupt and address options.

4.3.3 Signal Conditioning Circuits

Each of the eight sensor channels on a model 518 Smart A/D™ board is provided with a signal conditioning circuit (SCC) which may be inserted into the sense signal path. This conditioning circuit performs three functions:

❑ **Common-mode tie-down.** This function helps to prevent the common mode voltage (CMV) from straying beyond the input CMV range of the Smart A/D™ measurement section. This function is required when a sensor channel is driven from an isolated source, such as a battery, isolated power supply, or thermocouple.

❑ **Open-sensor detect.** If either of the two sense signals should become disconnected from the source, this function forces a differential voltage of 700mV, minimum, to appear across the sense inputs. This function is useful when a sensor is used in a control loop.

❑ **RF shunt.** This function, which shunts RF noise to ground, is essential when connecting to isolated sources, such as thermocouples.

4.3.4 Side Effects

The input impedance of a channel’s Sense inputs is reduced when that channel’s SCC is enabled. See “General Specifications” on page 26 for details.

Ordinarily, this reduction of input impedance will not be an issue for thermocouples. In the case of other types of voltage sources, however, measurement accuracy may be degraded if such sources have high source impedance.

4.3.5 Recommended Settings

The decision of whether or not to enable a channel SCC is influenced primarily by the type of sensor to be interfaced, and, in the case of active sources, whether or not the source is ground-referenced.

Listed below are recommendations for enabling vs. disabling the SCC:

Table 3: Recommended signal conditioner settings

Sensor Type	SCC
Ungrounded Thermocouple	Enabled
Isolated DC Voltage Source	
Grounded Thermocouple	Disabled
Ground-referenced DC Voltage Source	
4-to-20 mA loop	
Resistor	
RTD	
Strain/Pressure Gage	
Thermistor	

4.3.6 Shunt Mapping

Shunts E9 through E24 are used to control the enabling of SCCs for all sensor channels. Each channel is affiliated with two shunts. To enable a channel’s SCC, you must install both of the affiliated shunts. Similarly, to disable the SCC, you must remove both shunts.

The following table shows the mapping between channels and affiliated shunt pairs for SCC enables.

Table 4: Signal Conditioner shunt mapping

		Channel							
518 Board	**Lower**	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
	Upper	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**
Shunt Numbers		E9	E10	E11	E12	E13	E14	E15	E16
		E17	E18	E19	E20	E21	E22	E23	E24

Example:

To enable the SCC for sensor channel 2, install shunts E11 and E19 on the lower Smart A/D™ board.

Example:

To disable the SCC for sensor channel 14, remove shunts E15 and E23 on the upper Smart A/D™ board.

4.3.7 Default SCC Configuration

Models 2518 and 2519 are factory programmed so that the SCCs for all sensor channels are disabled. Shunts E9 through E24 are all removed as shipped from Sensoray.

Chapter 5: Sensor Connections

5.1 Overview

Sensors connect to the Smart A/D™ by means of pluggable screw terminal blocks on the model 2509 base board. In addition to the terminal blocks, the base board also contains integral temperature sensors for thermocouple reference-junction compensation.

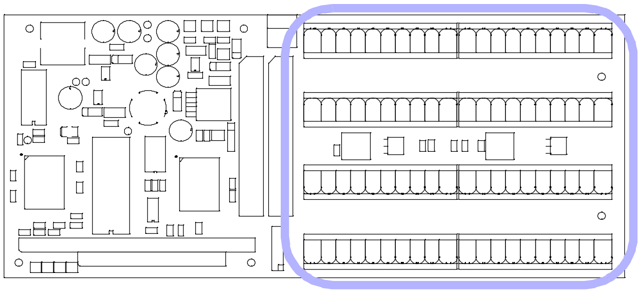
5.1.1 Sensor Hot Insertion

Sensors may be electrically connected to and disconnected from the terminal blocks, and terminal blocks may be plugged onto and unplugged from the base board without removing power from the base board.

5.2 Terminal Block Pinouts

All of the pluggable terminal blocks (TBs) that are used for sensor terminations are located on the base board as shown in Figure 10.

Figure 10: Sensor terminal blocks



Positions for up to eight sensor TBs are provided on the base board. These TBs are referenced as TB1 through TB8.

All eight of the TB positions are used on model 2519. In the case of model 2518, only TB1 through TB4 are used, even though all eight of the TB positions may be populated.

Terminal block pin numbers are silkscreened onto the board using "X.Y" notation, where X is the TB number and Y is the pin number. For example, the silkscreened legend "3.7" designates pin number 7 of TB3.

Each TB connects two sensors to the unit. Table 5 shows the TB numbers and pin assignments for models 2518 and 2519. The bottom table row lists the TB numbers. For any particular TB number in the bottom row, examine the intersecting column to determine which sensor channels connect to that TB.

Table 5: Sensor terminal block pin assignments

Pin No.	Func.	Channel							
1	V+	0	2	4	6	8	10	12	14
2	I+								
3	V-								
4	I-								
5	S								
6	V+	1	3	5	7	9	11	13	15
7	I+								
8	V-								
9	I-								
10	S								
TB:		**5**	**6**	**7**	**8**	**1**	**2**	**3**	**4**

Example:

Let's examine the case of TB7. First, look across the bottom row of Table 5 to find TB7. Now, looking up the intersecting column, it can be seen that TB7 is used to connect sensors to channels 4 and 5. Channel 4 occupies pins 1 to 5, which are designated "7.1" to "7.5." Channel 5 occupies pins 6 to 10, which are designated "7.6" to "7.10."

5.3 Sensor Channels

Each sensor channel provides five signals for interfacing to a sensor: V+, V-, I+, I- and S. A sensor can have as many as five connections to a channel, or as few as two.

5.3.1 Sense Terminals

Two of the channel signals, designated V+ and V-, are universally used for all sensor types. These two signals are, respectively, the positive and negative differential voltage sense inputs.

Since the sense inputs accept differential signals, it is not required for either of the two input signals to be at ground potential. The Smart A/D™ digitizer measures only the difference voltage across the two sense inputs.

Due to common-mode voltage (CMV) input constraints, however, you must ensure that neither sense input, with respect to system ground, exceeds the maximum input CMV permitted by the Smart A/D™.

Small excursions beyond the CMV limit may degrade measurement accuracy on the offending channel, while large excursions may cause measurement errors on other sensor channels, or worse yet, might damage the board.

See “General Specifications” on page 26 for input CMV limit values.

5.3.2 Excitation Terminals

Passive sensors require connection to the I+ and I- signals. The I+ and I- signals supply positive and negative excitation, respectively, to passive sensors.

Different classes of excitation are applied to sensors; the class of excitation that is applied depends on the type of sensor to be measured.

Constant current excitation is used for high-accuracy, low resistance measurements. A current limited, constant voltage excitation is applied when measuring higher resistance values. Finally, a 10 Volt, high current excitation is applied to strain and pressure gages.

5.3.3 Shield Terminal

The fifth lead—named *S* for shield—connects to the cable shield, if present. The S signal is internally connected to the backplane ground. To avoid ground loops, the cable shield should never be connected to anything at the sensor end of the cable, nor should it be connected to another ground point.

A shield conductor is never absolutely required, but is sometimes an unavoidable noise-abatement measure.

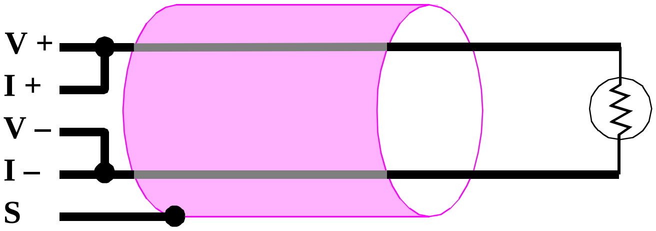
5.4 RTDs, Resistors and Thermistors

RTDs, resistors and thermistors can be connected to the TB in any of three possible configurations: two-wire, three-wire, and four-wire. In the following discussion, *sensor* is used interchangeably with *RTD*, *resistor* and *thermistor*.

5.4.1 Two-wire Circuit

The simplest configuration is the two-wire circuit. As the name implies, this configuration requires only two wires. The V+ and I+ terminals are shorted together at the TB, as are the V- and I- terminals.

Figure 11: Two-wire connection.

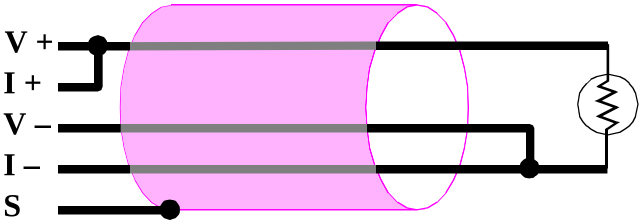


This circuit conserves wire, but it causes measurement error due to the voltage drop between the TB and sensor. There are two potential solutions for this problem: use short wires, or use more than two wires.

5.4.2 Three-wire Circuit

By using three wires, it is possible to reduce cable loss errors by 50 percent. Instead of shorting V- and I- together at the TB, discrete conductors are run from each of these terminals to the sensor. The wires are then shorted together at the sensor.

Figure 12: Three-wire connection

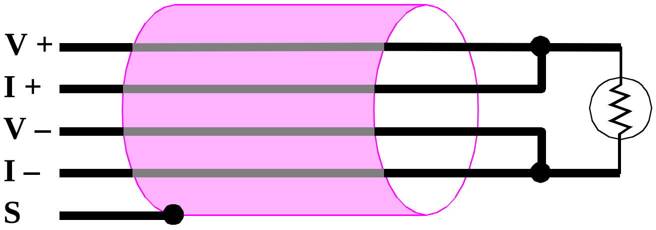


In this situation, the high impedance V- terminal detects the voltage at the sensor before any voltage drop can occur.

5.4.3 Four-wire Circuit

Unlike the three-wire circuit, which exhibits half the cable losses of an equivalent two-wire circuit, a four-wire circuit completely eliminates cable losses.

Figure 13: Four-wire connection



Like the three-wire circuit, separate conductors are run from the V- and I- terminals to the sensor. In addition, separate conductors are run from V+ and I+ to the sensor, where they are tied together. This circuit eliminates cable loss effects from both the V+ and V- lines.

5.4.4 Recommended Practice

If sensor field wiring is exposed to electrical noise (i.e., the cable run is long or is close to noisy conductors) you should consider using shielded cable. The cable shield must be connected only to the S terminal on the TB and left unconnected at the sensor end of the cable.

The four-wire circuit should be used when accuracy is critical. This is especially important when making precision low-resistance value measurements, such as when using RTDs.

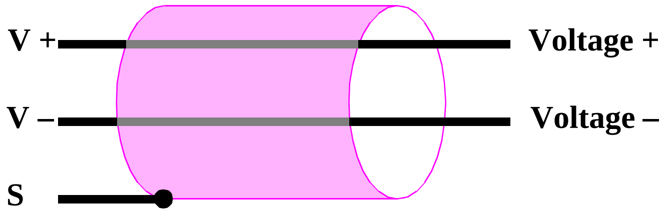
Thermistors have much higher resistance values than RTDs over most of their operating range. Consequently, the two-wire circuit may be used if your sensor will be operating exclusively at high resistance values. However, if your thermistor will be operating at lower resistance values, you should consider implementing a three-wire or four-wire circuit to reduce or eliminate cable-loss errors.

5.5 DC Voltage Sources

DC voltage sources are connected directly to the V+ and V- terminals. DC voltage sources should never be connected to the I+ or I- terminals.

Since all voltage input ranges are bipolar, DC voltages may be connected in either signal polarity. Although the diagram below shows Voltage+ connected to V+ and Voltage- connected to V-, there is no reason that Voltage+ must be positive with respect to Voltage-.

Figure 14: DC voltage connection



5.5.1 Recommended Practice

In order to assure accurate DC voltage measurements, high common-mode voltages (CMV) must not be permitted to appear on V+ or V-.

If your signal source is isolated (i.e., sourced from a battery or isolated power supply), you can connect V+ or V- to either S or the backplane power supply return. This connection can be implemented as a direct short or can be made through, for example, a 10KΩ resistor. There should be no significant DC current flowing through this connection; its purpose is to limit the common-mode voltage at the sense inputs.

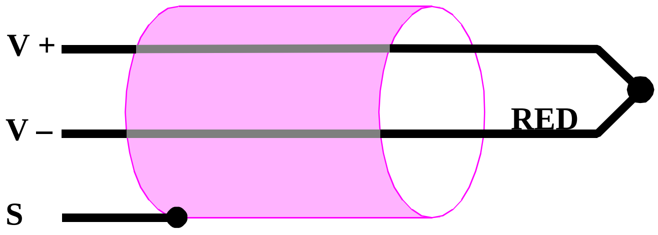
In the case of low source impedance voltage sources, another way to limit CMV is to install the channel signal conditioning circuit as described in Section Chapter 4:.

5.6 Thermocouples

Thermocouple (TC) signals are connected directly to the V+ and V- terminals. TCs should never be connected to the I+ or I- terminals.

TC wires are color coded to indicate polarity. The red thermocouple wire is always negative, by convention. The positive TC wire should always be connected to the V+ terminal, and the negative wire should always be connected to the V- terminal.

Figure 15: Thermocouple connection



5.6.1 Recommended Practice

Two TC reference-junction temperature sensors are located on the model 2509 base board. Because these reference sensors, as well as the sensor terminal blocks, have non-zero thermal mass, it is critical that the base board operate in a stable thermal environment so as to minimize thermal gradients across the termination system.

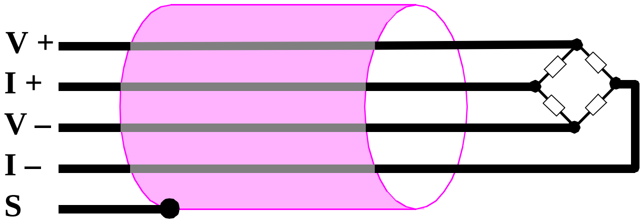
Significant measurement errors can occur if you expose the base board to thermal transients, such as air flows from cooling fans or ambient breezes. For best results, the unit should be located within a protective enclosure after sensor terminations have been made.

If your TC is ungrounded, always enable the channel signal conditioning circuit. See "Configuration" on page 8 for details.

5.7 Strain and Pressure Gages

In a typical strain/pressure gage sensor, four wires are used to connect the gage to the sensor channel. Two of the wires supply gage excitation, while the other two wires transport the gage output signal to the Smart A/D™ measurement circuitry.

Figure 16: Strain/pressure gage connection



5.7.1 Recommended Practice

Due to the high measurement gain used, strain and pressure gages should use shielded cable. The shield should be connected only to the channel S terminal.

If you are using a six-wire gage, you should connect the gage's excitation source and excitation sense leads together at the I+ and I- terminals. Note that remote sensing of the excitation signals is not supported.

The gage impedance seen by the Smart A/D™ excitation source must be at least 120Ω to prevent automatic current limiting. If the input impedance is less than 120Ω, a resistor must be inserted in series with the gage excitation terminals to prevent excessive excitation circuit loading. Note that this will alter the mV/V rating of the sensor.

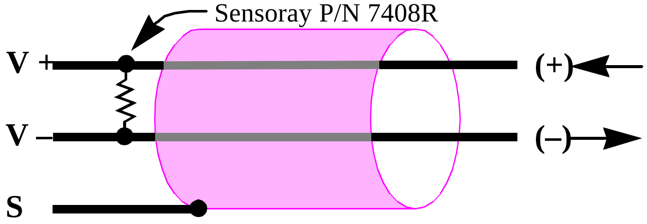
Gage output voltage must fall between -500mV and +500mV under all load conditions, including any offset caused by bridge imbalance. If the gage output voltage, at 10V excitation, might exceed these input limits, it may be necessary to configure the sensor channel for voltage input and supply external excitation to the gage.

5.8 4-to-20 mA Current Loops

The Smart A/D™ supports 4-to-20 mA current loop inputs on any channel, subject to two constraints:

1. A 250Ω, 0.01% resistor must be installed as shown in the following diagram. This resistor, which converts the loop current to a voltage, is available as an option for the Smart A/D™; order Sensoray part number 7408R.
2. Only the *grounded* end of the current loop may be connected to a sensor channel. This ensures that the common-mode voltage will not exceed Smart A/D™ input limits.

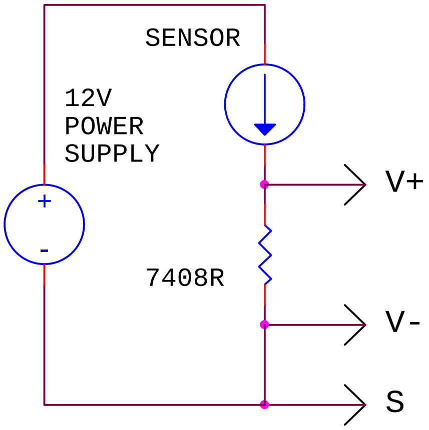
Figure 17: 4-20 mA current loop connection



5.8.1 Current Loop Excitation

It is important to note the Smart A/D™ does *not* provide excitation for current loops. Current loops must be energized from an external power source, such as the isolated 12V power supply in the following schematic:

Figure 18: Typical current loop schematic



If the external power source is not isolated (i.e., its negative output terminal is connected to the backplane power supply return), you should not connect to the S terminal as shown in the above diagram, as a potential ground loop would be created. In such a case, leave the S terminal disconnected and connect to the V+ and V- terminals only.

Chapter 6: Programming

6.1 Overview

The Smart A/D™ is programmed by means of a small, powerful built-in command set. Commands are issued to the Smart A/D™ server by a client, which can be any host on the same Ethernet network. Some commands cause the unit to respond by returning data to the client.

This chapter details the embedded Smart A/D™ command set and the programming interface which encapsulates the command access methods.

6.2 Standardized Interface

Because of the complexity of both network programming and the native Smart A/D™ command set, a standardized application programming interface (API) has been developed. This API enables access to Smart A/D™ functions through an interface that is consistent for all supported operating system and hardware platforms.

6.2.1 Distribution Diskette

The API consists of several software components that are distributed on floppy diskette with the Smart A/D™. The files on the distribution diskette include:

- `SmartAD.dll`—Windows Dynamic Link Library for Sensoray Smart A/D™ boards.
- Application programming examples and demonstration programs.

6.2.2 API Functions

The remainder of this chapter focuses on the functional characteristics of each API function by discussing what happens on the Smart A/D™ when an API function is executed, and the meaning of any values that are passed to or received from a function.

In some cases, arguments that are passed to or from an API function are specified as `void`. These arguments employ nomenclature borrowed from the C/C++ languages to indicate that no parameter value is passed.

Similarly, an argument that is prefixed with an asterisk ("*") is indicative of a pointer.

You should refer to the documentation on the distribution diskette for detailed, up-to-date information about the current API release. This is recommended because:

- API arguments and return values shown in this document are only representative of the actual parameter values and ordering used.
- API functions often include additional arguments that are not discussed here.
- The current version of the API may not include all functions or sensor types listed in this document.
- The API may contain additional platform-specific functions, not mentioned here, that are essential to the creation of working application programs.

6.2.3 Channel Numbering Convention

Many of the API functions require a sensor channel number as a parameter. By convention, channel numbers on Smart A/D™ boards always begin at zero.

Channel numbers range from 0 to 7 on the model 2518, and from 0 to 15 on the model 2519.

6.3 Board Configuration Functions

6.3.1 ResetBoard()

> Arguments: `void`
> Returns: `void`

This function, which invokes a "soft reset," provides a means for forcing the Smart A/D™ to its reset state without altering the states of any other boards in the system. A soft reset has the same effect on the Smart A/D™ as a hard system reset.

After executing this function, the Smart A/D™ is in its default state. See Section 2.5 for detailed information about the default state of the Smart A/D™.

6.3.2 GetFaultFlags()

> Arguments: `void`
> Returns: `FaultFlags`

The Smart A/D™ employs a hardware status register to provide rapid access to the states of various hardware fault conditions. In addition, the API maintains a log of fault events that have occurred in the API software functions. The status of all of these fault conditions, both hardware and software, are available through the GetFaultFlags function.

For details on this function, refer to the documentation supplied on the distribution diskette.

6.3.3 GetFWVersion()

Arguments:	`void`
Returns:	`VersionTimes100`

Returns the Smart A/D™ firmware version number, times 100. The version number is printed on the EPROM device that is plugged into the Smart A/D™.

6.3.4 SetHiSpeedMode()

Arguments:	`void`
Returns:	`void`

Decreases the channel slot time (Section 9.2) to enable higher throughput.

Assuming the board is operating in the default 60Hz rejection mode, all channel slot times are reduced from 22 milliseconds (default) to 9 milliseconds, and the board is said to be operating in the "high-speed mode."

Decreasing the slot time causes measurement accuracy to be reduced by approximately 75 percent and may cause increased measurement noise. Also, line frequency rejection is no longer possible as the digitization period is shorter than one line cycle.

A board reset will restore slot times to their default values. This is the only way to resume the default update rate after invoking the SetHiSpeedMode function.

When the high-speed mode is active, any attempt to invoke this command again will be ignored.

6.3.5 SetReject50Hz()

Arguments:	`void`
Returns:	`void`

Increases the digitizer's integration period from 16.7 milliseconds (default) to 20.0 milliseconds, enabling the Smart A/D™ to better reject 50Hz differential noise.

This function should be used in systems targeted for 50Hz power environments.

A board reset will restore the board to the default 60Hz rejection mode. This is the only way to return to the 60Hz rejection mode after invoking the SetReject50Hz function.

When the 50Hz rejection mode is active, any attempt to invoke this command again will be ignored.

6.4 Channel Configuration Functions

6.4.1 SetSensorType()

Arguments:	`ChannelNumber` `SensorDefinitionCode`
Returns:	`void`

Declares the type of sensor connected to a channel. By declaring the sensor type, you are implicitly selecting the excitation mode, measurement gain setting and linearization curve.

Upon successful execution of this function, the onboard microcomputer will perform all subsequent scans of the specified channel as appropriate for the declared sensor type.

This function has several side-effects that should be noted:

- Sensor data is not available immediately after declaring a channel's sensor type. After executing this function, sensor data is reset to zero. The maximum elapsed time that must pass before new sensor data becomes available is equal to ten (model 2518) or nineteen (model 2519) channel slot times. Details of channel slot timing can be found in Section 9.2.
- The channel filter constant (Section 6.4.2) is reset to zero, effectively disabling the software filter function.
- Limit alarms are disabled and limit values are indeterminate, as the new sensor type may have a new type of engineering units.

Typically, the client will call this function once for each sensor channel. Each execution of this function declares the sensor type for a single channel, therefore eight (model 2518) or sixteen (model 2519) invocations must occur to declare all channel sensor types.

Each sensor type supported by the Smart A/D™ is assigned a unique value, called the Sensor Definition Code. This value, along with the target sensor channel number, is passed to the Smart A/D™ as a function argument.

Upon reset of the Smart A/D™, all channel sensor types default to the ±5-volt input range, with 500µV resolution, and all sensor data values are reset to zero. Channel sensor types remain at the default setting until changed by this function. If the default sensor type is suitable for your application, you need not execute this function at all.

Any attempt to declare an unsupported sensor type will result in selection of the default sensor type.

Declaring a sensor type as “Disabled” inhibits scanning of the corresponding channel, resulting in increased throughput for all remaining active channels.

See “Sensor Specifications” on page 27 for a complete list of supported sensor types. Refer to the API documentation on the distribution diskette for a list of sensor definition codes.

6.4.2 SetFilter()

```
Arguments: ChannelNumber
           FilterConst
Returns:   void
```

Establishes a digital single-pole low-pass filter for the specified channel. Filter weighting is specified by the `FilterConst` parameter, which may have any value from 0 to 255. The relationship between `FilterConst` and filter percentage is:

$$FilterPercentage = \frac{FilterConst}{256}$$

For example, a `FilterConst` value of 0x40 (64 decimal) results in a 25% filter. This means that the sensor data value made available to the client is the sum of 75% of the newest unfiltered value plus 25% of the previous filtered value.

All filter constants default to zero upon board reset. This value disables the filters, thereby maximizing the frequency response of all channels.

A non-zero filter constant will roll off the channel frequency response. Since the filter is implemented in the digital domain, noise frequencies at or above nyquist must be pre-filtered before arriving at the Smart A/D™.

Some applications require known filter time constants. In such systems, you may use the following expression to calculate the time constant (in seconds) as a function of `FilterConst`:

$$\tau \approx \frac{(-1.06)(NumActiveChans)(ChanSlotTime)}{\ln(FilterConst/256)}$$

In the above expression `NumActiveChans` is the number of channels in the scan loop (all channels that have not been disabled), and `ChanSlotTime` is the channel slot time, as described in Section 9.2.

It may not be practical to compute exact filter constants in cases where the application doesn’t require a known time constant. In such cases, you will probably find that it is more effective to find the best filter weighting by experimenting with different filter constants.

6.4.3 SetFailMode()

```
Arguments: ChannelNumber
           FailHigh
Returns:   void
```

Specifies whether, in the event an open sensor is detected, the sensor data value should default to a large positive value (fail high) or a large negative value (fail low).

The `ChannelNumber` argument designates the sensor channel to be affected, while the boolean `FailHigh` specifies whether the channel should fail high or low.

This function is useful for triggering alarms when open sensors are detected, and for forcing the desired system response to sensor fault conditions in closed-loop control applications.

After a board reset, all channels are, by default, programmed to fail high.

6.5 Sensor Acquisition Functions

6.5.1 GetAmbientTemp()

```
Arguments: void
Returns:   Temperature
```

Returns the reference temperature, in degrees Centigrade, from an optional Sensoray termination board.

This is useful for monitoring thermocouple reference junctions and can also serve as a debugging aid during Smart A/D™ installation and application development.

This function returns an indeterminate value if no Sensoray termination board is connected to the Smart A/D™.

6.5.2 GetSensorData()

Arguments:	`ChannelNumber`
Returns:	`SensorData`

Returns the most recently acquired sensor data from one sensor channel. The returned value, `SensorData`, is expressed in engineering units that are appropriate for the channel's sensor type.

For example, channels configured for any type of temperature sensor typically return values in Degrees Centigrade, while channels configured for DC Voltage measurement return values in Volts.

Channels that have been programmed as sensor type "Disabled" return indeterminate values.

Refer to "Sensor Specifications" on page 27 for information regarding the resolution, accuracy and engineering units that apply to specific sensor types.

6.5.3 GetAllSensors()

Arguments:	`*SensorData`
Returns:	`void`

Fills an array with sensor data from all channels on a Smart A/D™. The `SensorData` argument is a pointer to an application array which is to receive the sensor data.

Data values are returned from all sensor channels, even those that have been declared as type "Disabled." Disabled channels return indeterminate values.

As in the GetSensorData function, sensor data are scaled to engineering units suited to the sensor type. Refer to "Sensor Specifications" on page 27 for information regarding the resolution, accuracy and engineering units that apply to specific sensor types.

6.6 Alarm Functions

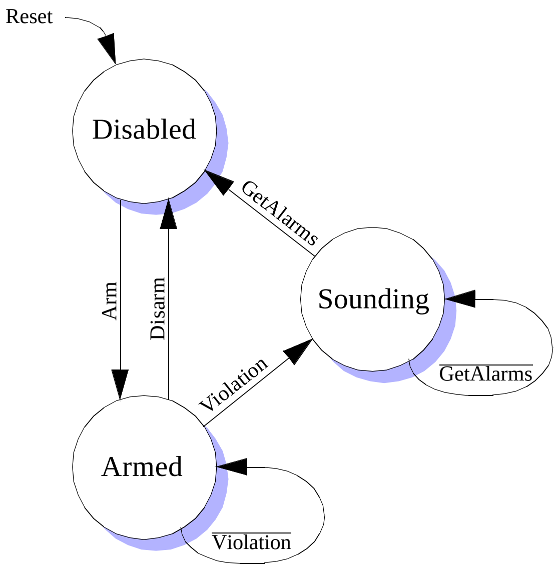
Each sensor channel has the ability to detect limit violations. A limit violation occurs when a channel's sensor data value exceeds a programmed threshold. When such violations are detected, a channel is said to be "sounding" an alarm.

Alarm limit thresholds are independently programmed for each channel by means of the SetAlarmLimits function. After setting these thresholds, alarm sounding may be enabled under software control.

Although each channel has two alarm limit values, each channel technically has only one alarm. When either limit is violated (exceeded), the channel alarm is asserted.

6.6.1 Alarm States

The following state diagram illustrates the operation of alarms for an individual sensor channel:



Initially, all channel alarms are disabled and each channel is in the Disabled state. The SetAlarmEnable function is used to switch between the Disabled and Armed states.

If a limit violation is detected while the alarm is Armed, the channel switches to the Sounding state. This causes the Smart A/D™ Alarm status flag to be asserted. The Alarm status flag may be accessed indirectly through the GetAlarms function.

A channel remains in the Sounding state until a GetAlarms function is executed. When the GetAlarms function executes, the channel switches to the disabled state, where it remains until the client again arms the channel alarm.

6.6.2 SetAlarmLimits()

Arguments:	`ChannelNumber`
	`HighLimit`
	`LowLimit`

Returns:	`void`

Declares the upper (`HighLimit`) and lower (`LowLimit`) alarm threshold values for a channel. An alarm will sound if the target channel's sensor data value becomes positive with respect to `HighLimit`, or negative with respect to LowLimit, while the channel alarm is armed.

The HighLimit and LowLimit arguments are expressed in the same engineering units used by the sensor that is connected to the target channel. For this reason, the sensor type must be declared (see Section 6.4.1) before invoking this function.

Suppose, for example, the target channel has been previously configured for a type K thermocouple. By specifying LowLimit and HighLimit values of 110.0 and 126.0, respectively, and then arming the channel alarms (Section 6.6.3), an alarm will sound if the channel's thermocouple temperature wanders outside the range +110 to +126°C.

The HighLimit value should always be positive with respect to the LowLimit value, otherwise an alarm will sound immediately when alarms are armed.

Alarm limit values are indeterminate after a board reset. Channel alarm limit values should be set by this function before arming the channel alarms.

6.6.3 SetAlarmEnable()

> Arguments: `ChannelNumber`
> `LimitEnable`
> Returns: `void`

Enables or disables limit violation detection for the specified channel. The ChannelNumber parameter designates the target channel, and the boolean LimitEnable argument specifies whether alarm sounding is to be permitted (i.e., the channel is to be armed) on the target channel.

Channel alarm limit values should be set (Section 6.6.2) before invoking this function.

By default, alarms are disabled on all channels after a board reset. In addition, individual channel alarms are disabled in response to any of the following events:

- The client explicitly disables an alarm by calling this function with LimitEnable set to false.
- An alarm sounds on the channel. This prevents the alarm from sounding again after the client has acknowledged the alarm with the GetAlarms function.
- A new sensor type is declared for a channel. This not only disables the channel alarm, but also renders the channel's limit values indeterminate.

6.6.4 GetAlarms()

> Arguments: `*AlarmStatus`
> Returns: `AlarmSounding`

Fills an array with the alarm status of all channels on a Smart A/D™ and restores any sounding channels to the Disabled alarm state.

The AlarmStatus argument is a pointer to an application array which is to receive the alarm status.

The returned boolean value, AlarmSounding, indicates whether any alarms were sounding when this function was called. It is obviously not necessary to evaluate the contents of the AlarmStatus array if AlarmSounding returns false.

6.7 Strain/Pressure Gage Functions

The strain/pressure gage functions discussed in this section are applicable only to sensor channels that have been configured for gages. API gage functions that are directed to non-gage channels (channels which have not been declared to be the strain/pressure gage sensor type) will be ignored.

After a board reset, you must configure any channels that will be connected to strain or pressure gages. Follow this procedure to configure a sensor channel for operation with a gage:

- Call the SetSensorType function to declare that the sensor channel will be connected to a gage.
- Calibrate the gage. The first time this is done, you must use the SetGageZero and SetGageSpan functions. If this is not the first gage calibration for the target channel, you may optionally use the SetGageCal function, or you may continue to use the SetGageZero and SetGageSpan functions.

6.7.1 SetGageZero()

> Arguments: `ChannelNumber`
> Returns: `void`

Declares a zero-load condition for a strain/pressure gage channel.

This function tells the Smart A/D™ that there is presently no load applied to the target channel's gage. The Smart A/D™ registers this condition and uses it as a normalization reference for all subsequent measurements on the target channel.

Typically, this function is invoked just before a call to SetGageSpan. Together, the SetGageZero and SetGageSpan functions serve to completely calibrate a gage channel.

6.7.2 SetGageSpan()

```
Arguments:  ChannelNumber
            LoadValue
Returns:    void
```

Declares a known load condition for a strain/pressure gage channel, and the numerical load value expected from the Smart A/D™ when measuring the current load.

This function is used in conjunction with the SetGageZero function to prepare a sensor channel for gage measurements. Invoke this function after declaring the sensor type, but before acquiring sensor data from a channel.

The SetGageSpan function requires that a precisely known load, called the "calibration load," be applied to the target sensor. In some cases the calibration load can be simulated by using an internal, precision calibration resistor—present in many gages—to unbalance the sensor bridge. In the case of gages that do not provide a calibration resistor, a mechanical load must be applied to the sensor.

Ideally, the calibration load will be close to the actual load values to be measured. If a wide range of load values are to be measured, it is best to employ a calibration load that is equal to or larger than the largest application load to be measured.

Before invoking this command, you must apply either a simulated or true mechanical zero load condition to the sensor and invoke the SetGageZero command, as described in Section 6.7.1. This must be done before (preferably immediately before) executing the SetGageSpan function.

6.7.3 SetGageTare()

```
Arguments:  ChannelNumber
Returns:    void
```

Tares the strain/pressure gage connected to the specified channel. This function is used to compensate application load offsets.

For example, an empty container is often used to hold a material that is to be weighed. Since the container itself is not to be measured, the SetGageTare function can be invoked with a load consisting of only the empty container. After executing this function, the Smart A/D™ will subtract off the container load and return only the load value of the container contents.

Before invoking this command, the gage channel should be calibrated. See Section 6.7.1, Section 6.7.2 and Section 6.7.5 for details.

6.7.4 GetGageCal()

```
Arguments:  ChannelNumber
            *GageCal
Returns:    void
```

Returns the gage calibration parameters from the sensor channel specified by `ChannelNumber`. The `*GageCal` argument is a pointer to an application buffer which is to receive the calibration parameters.

This function makes it possible to restore a channel's gage calibration without using the SetGageZero or SetGageSpan functions. The returned parameters may be saved by the application and used to later restore a gage calibration by means of the SetGageCal function.

The returned parameters are useful only if the gage has already been calibrated. This prior calibration may have been accomplished in either of two ways:

- Using SetGageZero and SetGageSpan functions. This method must be used if the sensor channel has never been calibrated for the gage to which it is connected.
- Using the SetGageCal function.

6.7.5 SetGageCal()

```
Arguments:  ChannelNumber
            *GageCal
Returns:    void
```

This command restores a channel's strain/pressure gage calibration settings by using calibration parameters that were obtained from the GetGageCal function.

The `*GageCal` argument is a pointer to an application buffer which contains the gage calibration parameters.

Chapter 7: Calibration

7.1 Overview

Smart A/D™ boards are factory calibrated to meet or exceed published specifications. Because the boards employ high-stability standards, calibration should not be necessary unless the board has been subjected to environmental extremes or has been in service for an extended time period.

WARNING: RECALIBRATION AFFECTS THE ACCURACY OF ALL SENSOR MEASUREMENTS.

After calibrating a Smart A/D™, you can't "undo" the new calibration. The only way to correct a calibration error is to perform a proper calibration.

7.1.1 User-supplied References

To perform a board calibration, you must supply the following external reference standards:

- 5.0000V signal.
- 500.00mV signal
- 380.00Ω resistor.

The actual values of your standards may deviate from these target values by up to ±1%, but the values must be stable, quiet and precisely known.

7.1.2 Calibrate()

```
Arguments: ChannelNumber
           StdNum
           RefValue
Returns:   Ignore
```

This API function calibrates one of the Smart A/D™ internal reference standards and stores the calibration value in non-volatile memory on the Smart A/D™.

Although this function returns a value, the value is only of use for factory test purposes and should therefore be ignored by your application.

The `RefValue` argument is used to specify the exact value of the user-supplied reference. As mentioned earlier, the absolute value of the reference is not critical, but knowledge of the exact reference value is required.

`ChannelNumber` specifies the sensor channel number that is connected to the user-supplied reference. You may use any sensor channel of your choosing.

The `StdNum` argument specifies which of the external references is supplied to the target channel and the measurement range to be used for the calibration.

Table 6: StdNum Values for the Calibrate Function

StdNum	Reference	Measurement Range
0	5.0000 V	±5V, 200uV resolution
1	500.00 mV	±500mV, 20uV resolution
2	380.00Ω	400Ω, 0.02Ω resolution

7.2 Calibration Procedure

7.2.1 Order of Calibration

Three invocations of the Calibrate function are necessary to completely calibrate the Smart A/D™. These invocations must occur in the order shown in Table 6, from top to bottom.

Specifically, the 5V standard is calibrated first, followed by the 500mV standard and then the resistance standard.

7.2.2 Calibration Process

Each measurement range is calibrated by following a five-step procedure:

1. Connect the appropriate external reference standard, as specified in Table 6, to the target channel. In the case of the reference resistor, you must use a four-wire connection to the resistance standard.

1. Configure the target channel's sensor type as specified under Measurement Range in Table 6. This is done by calling the SetSensorType function with the appropriate sensor definition code.

2. Allow sufficient time for sensor data to stabilize on the target channel. A simple way to do this is to poll the target channel's sensor data, using the GetSensorData function, until non-zero data is returned.

3. Invoke the Calibrate command, using the StdNum value from Table 6, then wait for the return value.

4. Reset the Smart A/D™. This is normally accomplished by invoking the ResetBoard function.

Chapter 8: Smart A/D™ Theory of Operation

8.1 Firmware

The Smart A/D™'s internal microcomputer executes a firmware-resident control program. At the heart of this program is a high-performance, event-driven kernel.

The following paragraphs describe the functions of some of the tasks that are managed by the kernel.

8.1.1 Digitizer

This task performs the fundamental data acquisition function central to the Smart A/D™'s purpose.

The Digitizer begins a conversion by selecting the next channel to be digitized. The sense and excitation multiplexers are switched to the desired channel, and, if the channel is connected to a passive sensor, the pulsed excitation source is configured and activated. The scanner then sleeps while the sensor signal stabilizes.

When the sensor signal has stabilized, the Digitizer initiates an A/D conversion. Other tasks are allowed to run while the conversion is taking place.

Upon completion of the conversion, the digitized value is forwarded to the Cruncher task.

8.1.2 Cruncher

The Cruncher task is responsible for polishing raw A/D data into finished form. Typically, the Cruncher task runs while the scanner is digitizing the next channel in the scan loop.

This task begins when it receives a digitized value from the Digitizer task. The data is normalized to the internal standards, then is linearized and converted to engineering units appropriate for the declared sensor type. The resulting value is passed through a single-pole low-pass filter.

Next, the filtered data is checked for alarm limit violations. If a limit has been exceeded, the channel alarm is disabled and the Alarm status flag is set.

Finally, the processed sensor data is posted to internal dual-ported RAM for rapid client access.

8.1.3 Command Processor

The Command Processor receives and executes commands from the model 2509 Ethernet base board, which in turn forwards commands from the network client. Top priority is given to this task in order to minimize communication latency.

This task is run when a command is received from the base board.

8.2 Analog Circuits

The Smart A/D™ analog circuitry is functionally partitioned into Measurement and Excitation sections.

8.2.1 Measurement Section

The Measurement section selects and conditions a sensor input signal for digitizing.

The Measurement section begins sensor signal processing by switching a pair of channel sense signals through a differential analog multiplexer. The selected channel is then conditioned by a programmable gain amplifier. Finally, the amplified signal is applied to the input of an integrating A/D converter.

Channel selection, amplifier gain, and A/D functions are all under control of the onboard microcomputer. None of these circuits are directly accessible by the client.

8.2.2 Excitation Section

The Excitation section sources a pulsed DC current or voltage that is used to excite passive sensors.

The Excitation section supplies one of three pulsed signals to passive sensors, depending on the declared sensor type. The length of the excitation pulse is equal to the channel slot time (see Section 9.2 for an explanation of slot time).

In the case of strain and pressure gages, 10VDC excitation is applied. This voltage is current limited so that shorting the excitation signals together or to ground will not damage the excitation source.

In the case of RTD's and the 400Ω and 3KΩ resistance ranges, a constant current of approximately 1.2mADC is forced through the sensor.

5VDC is applied, in series with a reference resistor, to sensors measured on the 600KΩ resistance range.

Chapter 9: Smart A/D™ Timing

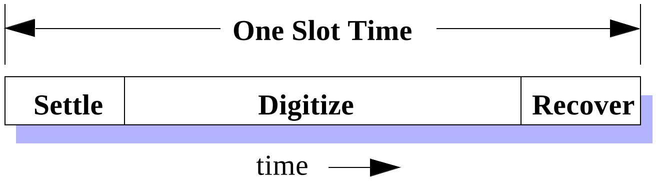
9.1 Overview

Three Smart A/D™ timing parameters are important from the application developer's viewpoint: Channel Update Rate, Data Age and Communication Latency. This section discusses these parameters along with some important related issues.

9.2 Slot Time

Channel Slot Time is the length of time required for the Smart A/D™ to completely process one sensor channel.

Each channel slot time encompasses three functional phases as shown in the following diagram:



In the first phase, a sensor channel is excited and its sense signals are routed through to the Smart A/D™ measurement section. The embedded microcomputer configures the signal path as appropriate for the sensor type, then the analog front end is given time to stabilize.

The digitizer acquires the sensor data value in the second phase. The Update Rate, which is described in the next section, can be increased by reducing the duration of the time slot's *Digitize* phase. See Section 6.3.4 for details.

A *Recover* phase occurs at the end of the time slot. The function of this phase is to reset the analog front end in preparation for the next conversion.

Computationally intensive processes, such as linearization, alarm processing and software filtering, don't affect the slot time because they execute concurrently during the subsequent channel slot time.

9.3 Update Rate

Update Rate is defined as the number of times each second that a sensor channel acquires new sensor data.

9.3.1 Primary Influences

The two primary influences on update rate are the number of active channels and the channel slot time.

Due to timing uncertainties, the update rate is expressed as a range of values bounded by minimum and maximum times. The minimum and maximum update rate for any active channel is given by these functions:

$$UpdateRate_{min} = \frac{1}{(NumActiveChans + 1) \times SlotTime}$$

$$UpdateRate_{max} = \frac{1}{NumActiveChans \times SlotTime}$$

The time difference between the minimum and maximum update rate is due to the automatic, interleaved measurement of internal Smart A/D™ reference standards. Other than their timing impact, these measurements—which are scheduled and executed autonomously by the embedded processor—are transparent to the application program. At most, one internal standard will be measured per every sixteen sensor channel measurements.

Clearly, the update rate increases as the number of active channels decrease. Channels can be removed from the scan loop by disabling them with the SetSensorType command with the Disabled sensor definition code.

For example, a Smart A/D™ that has eight active channels and is running at the default channel slot time (22 milliseconds) would have a worst-case channel update rate ranging from 5.1 to 5.7 samples per second:

$$UpdateRate_{min} = \frac{1}{(8+1) \times 0.022} \approx 5.1Hz$$

$$UpdateRate_{max} = \frac{1}{8 \times 0.022} \approx 5.7Hz$$

Note that these timing calculations apply to each Smart A/D™ board. In the case of model 2519, which employs two model 518 Smart A/D™ boards, the update rates must be independently computed for each Smart A/D™ board.

For example, with all sixteen channels enabled on a model 2519, each model 518 Smart A/D™ will have eight channels enabled, and will therefore have a worst-case channel update rate ranging from 5.1 to 5.7 samples per second as in the above example.

9.3.2 Secondary Influences

Another influence on the update rate is the frequency of communication between client and Smart A/D™.

The Smart A/D™ firmware is designed to minimize the impact of communications on the update rate. Even so, communication activity will sometimes interrupt scanning, which in turn will stretch the current channel's slot time in a non-deterministic way.

Because of the unpredictable effect on update rate, excessively frequent communication traffic between client and Smart A/D™ should be avoided.

9.4 Data Age

Data Age is the measure of time that has elapsed since a channel's data was acquired by the Smart A/D™. The age of any channel's data is directly related to the update rate.

The minimum possible data age is, of course, zero. This would be the case if the client retrieves sensor data immediately after new data is posted by the Smart A/D™.

The maximum possible data age would apply if the client retrieves sensor data immediately before new data is posted by the Smart A/D™. In this case, the data age is the reciprocal of the minimum update rate.

In practice, data age almost always lies between the minimum and maximum possible values.

9.4.1 Example

As an example, take the case shown at the end of Section 9.3.1. The minimum update rate is approximately 4.1 Hertz, so the maximum data age would be:

$$DataAge_{max} = \frac{1}{4.1} \approx 244ms$$

In this example, the age of sensor data from any given channel may range from zero to 244 milliseconds.

9.5 Communication Latency

Communication Latency is the time elapsed from a client request for sensor data to the acquisition of that data. This latency can be viewed as having two components: client overhead and Smart A/D™ response time.

Client overhead is the time spent executing those parts of the Smart A/D™ API functions that are in the computational domain of the client (i.e., not pending on a Smart A/D™ response). Since this overhead—which is purely a function of client clock rate and architecture—varies from one system to another, it must be independently determined for each system.

Smart A/D™ response time is the time spent in an API function while waiting for a Smart A/D™ response. For obvious reasons, short communication latency is a high-priority design objective in most real time applications.

Communication functions are event-driven in the Smart A/D™ local environment and are assigned highest priority in order to minimize the communication latency. As a result, Smart A/D™ response time is predictable with a good degree of accuracy:

Table 7: Command response times

Command	Response Time (microseconds)
GetSensorData	100
GetAllSensors	130

Chapter 10: Specifications

10.1 General Specifications

Table 8: General Specifications

Category	Parameter	Condition	Specification
Power supply	Input range	Absolute max. limits	12 to 24V AC/DC
	Power consumption		8W, typical.
Temperature	Operating range	Guaranteed accuracy	-25°C to +85°C
Sense inputs	CMRR	|CMV| < 5V, f < 1KHz	80dB, minimum.
	CMV	Guaranteed accuracy	±5.0V, maximum.
	Input impedance	SCC disabled	1000 MΩ, minimum.
		SCC enabled	1 MΩ, typical.
	Overload protection	Continuous	±20V CMV, maximum.
		2 seconds, max.	±70V CMV, maximum.
A/D converter	Type		Integrating, 16-bit resolution.
	Measurement period, per channel	60Hz reject mode (default mode)	16.7 milliseconds, typical (default mode). 4.0 milliseconds, typical (high-speed mode).
		50Hz reject mode	20 milliseconds, typical (default mode). 4.8 milliseconds, typical (high-speed mode).
	Total time slot, per channel	60Hz reject mode (default mode)	22 milliseconds, maximum (default mode). 9 milliseconds, maximum (high-speed mode).
		50Hz reject mode	25.3 milliseconds, maximum (default mode). 11 milliseconds, maximum (high-speed mode).
Excitation source	Pulse width		Applied for duration of channel time slot.
	Type	Strain/pressure gage	10VDC, 30mA maximum.
		400Ω/3KΩ ranges	1.2mADC.
		600KΩ range	5VDC, 4KΩ source impedance.
Network	Physical layer		Ethernet, 10BASE-T.
	TCP/IP protocols		ARP, ICMP, IP, UDP.
	Address type		Internet Protocol, version 4.
	Address requirements		Any one IP address in the range `10.0.0.0` to `10.0.0.255`.
	Ethernet packet size	Command	200 bytes, maximum.
		Response	120 bytes, maximum.
	Transaction time		710 µs, maximum.

10.2 Sensor Specifications

Table 9: Sensor Specifications

Sensor Type		Range	Resolution [1]	Accuracy [1]
Thermocouple	B	0 to +1820°C	0.1°C	3.3°C
	C	0 to +1820°C	0.1°C	2.1°C
	E	-270 to +990°C	0.1°C	0.8°C
	J	-210 to +760°C	0.1°C	0.6°C
	K	-270 to +1360°C	0.1°C	1.0°C
	N	-270 to +1347°C	0.1°C	0.9°C
	T	-270 to +400°C	0.1°C	0.6°C
	S	0 to +1760°C	0.1°C	3.0°C
	R	0 to +1760°C	0.1°C	2.8°C
Thermistor	Omega 44006 or 44031	-55°C to +145°C	0.01°C	0.05°C
RTD	Cu 10Ω, 0.0367Ω/°C	0 to +119°C	0.1°C	0.6°C
	Pt 100Ω, 0.385Ω/°C	-200 to +800°C	0.05°C	0.2°C
		-200 to +409°C	0.0125°C	0.2°C
	Pt 100Ω, 0.392Ω/°C	-200 to +800°C	0.05°C	0.2°C
		-200 to +409°C	0.0125°C	0.2°C
	Ni 120Ω, 0.380Ω/°F [5]	-100 to +482°F	0.1°F	0.3°F
	Ni 200Ω, 1.10Ω/°C	-58.9 to +151.6°C	0.02°C	0.07°C
	Ni 1KΩ, 5.60Ω/°C	-50.0 to +129°C	0.025°C	0.08°C
Gage [6]		-500 to +500 mV	5µV [7]	30µV
DC Voltage		-5 to +5V [3]	500µV	600µV
		-5 to +5V	200µV	600µV
		-500 to +500mV	20µV	40µV
		-100 to +100mV	5µV	30µV
Current Loop [2]		4 to 20mA	0.01%	0.08%
Resistance		0 to 400Ω	0.02Ω	0.04Ω
		0 to 3KΩ	0.125Ω	0.25Ω
		0 to 600KΩ	31Ω	130Ω
Disabled [4]		Not Applicable — sensor removed from scan loop		

Notes

1. Measurement resolution and accuracy is specified for the default (versus high-speed) measurement mode. Derate resolution and accuracy by 75% when using the high-speed measurement mode.
2. Channels configured for 4-20mA current loops return data values representing a percentage of full-scale current (4mA= 0%, 20mA=100%). For example, a 12mA current would produce the data value 50.0, indicating 50% of full-scale.
3. This is the default sensor type after a board reset. It is provided for back-compatibility with earlier Sensoray Smart A/D™ products. Use of this type is not recommended for new designs, as higher resolution is available on the 5V, 200uV resolution range.
4. Declaring a channel's sensor type as Disabled removes the channel from the scan loop. Data values returned from disabled channels are indeterminate and should be ignored.
5. This is the only temperature sensor type that returns data in degrees Fahrenheit. All other temperature sensors return data in degrees Centigrade.
6. Gage specifications assume 4-wire bridge circuit, 350Ω minimum impedance, 10V excitation supplied by the sensor channel.
7. Use the following function to express gage measurement resolution in load units:

$$Resolution = \frac{RatedLoad}{(2 \times 10^6) \times RatedVoltageOut}$$

For example, take the case of a pressure gage that is rated at 3mV/V at 100 PSI full scale. The Smart A/D™ would be able to resolve a load to:

$$Resolution = \frac{100PSI}{(2 \times 10^6) \times 0.003V} = 0.017PSI$$