Operation and Service Manual

# Analog PID Controller

## SIM960

SRS Stanford Research Systems

Revision 1.2 ● June 8, 2005

## Certification

Stanford Research Systems certifies that this product met its published specifications at the time of shipment.

## Warranty

This Stanford Research Systems product is warranted against defects in materials and workmanship for a period of one (1) year from the date of shipment.

## Service

For warranty service or repair, this product must be returned to a Stanford Research Systems authorized service facility. Contact Stanford Research Systems or an authorized representative before returning this product for repair.

Stanford Research Systems, Inc.
1290–C Reamwood Avenue
Sunnyvale, CA 94089 USA
Phone: (408) 744-9040 • Fax: (408) 744-9049
**www.thinkSRS.com • e-mail: info@thinkSRS.com**

Printed in U.S.A.

# Contents

# General Information

## Safety and Precautions for Use

Because of the variety of uses for the SIM960, those responsible for the application and use of this control equipment must satisfy themselves that all necessary steps have been taken to assure that each application and use meets all performance and safety requirements, including any applicable laws, regulations, codes and standards.

The SIM960 is not designed, intended, or sold for use in hazardous environments requiring fail-safe operation, including without limitation, operation of nuclear facilities, aircraft or spacecraft control systems, and life support or weapons systems. The user must assure that any failure or misapplication of the SIM960 cannot lead to a consequential failure of any interconnected equipment that could lead to loss of life or limb, or property damage.

The illustrations, charts, and discussions shown in this manual are intended solely for purposes of example. Since there are many variables and requirements associated with any particular control application, Stanford Research Systems does not assume responsibility or liability for actual use based upon the examples shown in this publication.

### Service

Do not install substitute parts or perform any unauthorized modifications to this instrument.

The SIM960 is a double-wide module designed to be used inside the SIM900 Mainframe. Do not turn on the power to the Mainframe or apply voltage inputs to the module until the module is completely inserted into the mainframe and locked in place. Do not exceed ±18 V at any input or output connector.

## Symbols you may Find on SRS Products

| Symbol | Description |
|---|---|
| ∼ | Alternating current |
| ⚡ | Caution - risk of electric shock |
| ⊥ | Frame or chassis terminal |
| ⚠ | Caution - refer to accompanying documents |
| ⏚ | Earth (ground) terminal |
| ⊣⊢ | Battery |
| ∿ | Fuse |
| \| | On (supply) |
| ○ | Off (supply) |

## Notation

The following notation will be used throughout this manual:

- Front-panel buttons are set as [Button];
  [Adjust ▲▼] is shorthand for "[Adjust ▲] & [Adjust ▼]".
- Front-panel indicators are set as *Overload*.
- Remote command names are set as *IDN?.
- Literal text other than command names is set as `OFF`.

## Specifications

### Performance Characteristics

| | | Min | Typ | Max | Units |
|---|---|---|---|---|---|
| Amplifier Settings | Control type | Analog, *PID*+Offset | | | |
| | Input Range | −10 | | +10 | V common mode |
| | | −1 | | +1 | V differential |
| | Proportional gain | $10^{-1}$ | | $10^{3}$ | V/V |
| | Integral gain | $10^{-2}$ | | $5 \times 10^{5}$ | 1/s |
| | eff. time const. | $2 \times 10^{-6}$ | | $10^{2}$ | s |
| | Derivative gain | $10^{-7}$ | | 1 | s |
| | Offset | −10 | | +10 | V |
| | resolution | | 1 | | mV |
| Amplifier Performance | Bandwidth | 100 | | | kHz |
| | Propagation delay | | 1 | | $\mu$s |
| | Noise ($f > 20$ Hz) | | 8 | | nV/$\sqrt{\text{Hz}}$, RTI |
| | Output Range | −10 | | +10 | V |
| Configuration | Parameter control | Digital | | | |
| | Parameter accuracy | | | 1 | % |
| | Stability | | | 200 | ppm/°C |
| | Display Resolution | | 4 | | digits |
| Inputs | Measure | BNC, 1 MΩ, ±10 V range | | | |
| | Ext. Setpoint | BNC, 1 MΩ, ±10 V range | | | |
| Setpoint Generator | Setting | −10 | | +10 | V |
| | resolution | | 1 | | mV |
| | Ramp Rate | $10^{-3}$ | | $10^{4}$ | V/s |
| | Noise ($f > 100$ Hz) | | | 20 | nV/$\sqrt{\text{Hz}}$, RTI |
| Operating | Temperature [14] | 0 | | 40 | °C |
| | Power | ±15, +5 | | | V DC |
| | Supply current | 150 (±15 V), 80 (+5 V) | | | mA |

### General Characteristics

| | |
|---|---|
| Number of inputs | 2 |
| Interface | Serial (RS-232) through SIM interface |
| Connectors | BNC (3 front, 2 rear); DB–15 (male) SIM interface |
| Weight | 2.1 lbs |
| Dimensions | 3.0″ W × 3.6″ H × 7.0″ D |

# 1 Getting Started

This chapter gives you the necessary information to get started quickly with your SIM960 Analog PID Controller.

## In This Chapter

## 1.1 General

The SIM960 is designed to maintain stability in systems requiring low noise and wide bandwidth. The controller design consists of a front end differential input amplifier, followed by an integrator and a differentiator, arranged in what is known as the "ideal" *PID* topology. The input amplifier (the "error amplifier") differences the the two single ended inputs, ***Setpoint*** and ***Measure***, and multiplies the resulting error signal ($\varepsilon$) by the proportional gain. The amplified error is then passed to three parallel control paths:

1. The proportional path, no change is made to the signal.
2. The integral path with gain ***I***.
3. The derivative path gain ***D***.

These three signals can be independently selected to combine at a summing amplifier, which is then buffered to the output. A constant offset can also be added, which can be useful in applications that do not use the ***I*** term. Mathematically, the behavior is

$$\varepsilon \equiv \textbf{\textit{Setpoint}} - \textbf{\textit{Measure}} \tag{1.1}$$

$$\textbf{\textit{Output}} = \boldsymbol{P} \times \left\{ \varepsilon + \boldsymbol{I} \int \varepsilon \, \mathrm{d}t + \boldsymbol{D} \frac{\mathrm{d}\varepsilon}{\mathrm{d}t} \right\} + \textbf{\textit{Offset}} \tag{1.2}$$

where the three terms within the braces, and ***Offset***, can be independently enabled or zeroed.

For internal stability, the actual differentiator is "rolled off" to limit the derivative gain to +40 dB.

The output circuitry includes a soft limiter that turns on when the output exceeds user specified upper and lower limits and clamps the output to the limit level. The output bar display on the right side of the front panel has red LEDs at each end to indicate when the output is being limited.

## 1.2 Front Panel Operation

This section discusses the essentials of operating the SIM960 locally, from the front panel. See Chapter 3 for remote operation.

- Press [Select] to choose which configuration parameter to view in the numerical display. The indicator to the left of each descriptor shows which parameter is displayed. When *Shift* is highlighted, pressing [Select] steps the parameter selection backwards.

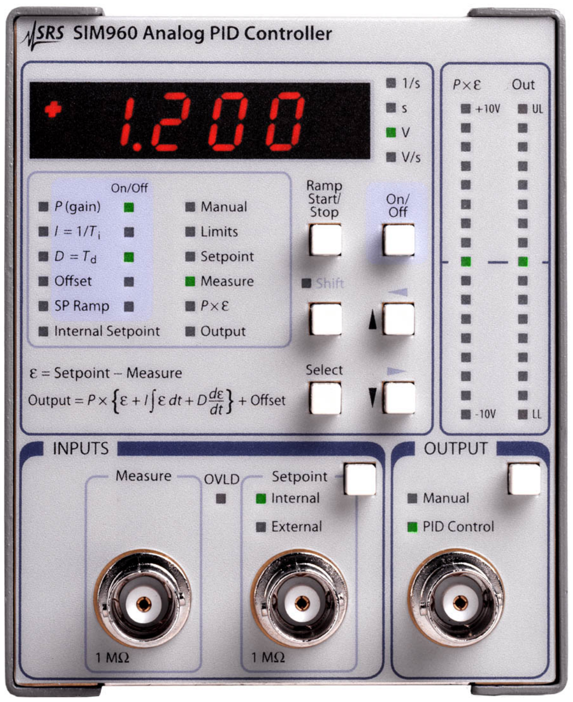

Figure 1.1: The SIM960 front panel.

- The ***P***, ***I***, ***D***, ***Offset***, and ***SP Ramp*** parameters may be enabled/disabled with [On/Off]. Each of these parameters has an additional indicator to the right of the descriptor to indicate the on/off status.

- The ***Limits*** parameter has two sublevels: upper and lower limit.

- The ***Setpoint***, ***Measure***, $P \times \varepsilon$, and ***Output*** values are display-only. All the other values can be changed using the [▲▼] buttons; the digit selected for adjustment is indicated by its flashing brightness. Change the digit selection while *Shift* is highlighted ([◄►]).

- There are two formats for the numeric display: Exponential, and fixed decimal. The format used for a particular parameter depends on its range. Parameters ***P***, ***I***, ***D*** and ***SP Ramp*** (rate) vary by several orders of magnitude and are therefore displayed in exponential format, while all other parameters range from −10 V to +10 V and are displayed in fixed decimal format.

- For exponential format, the mantissa may be changed using the up/down arrow buttons. The active digit may be selected using the left/right buttons (= shift, followed by up/down button).

The right-most digit (after E) is the power of ten exponent. For example, the display 1.2E 3 = 1200.

*Polarity* $\Longrightarrow$

- The ***P*** parameter has a selectable "±" indicator before the mantissa.This allows the polarity of the controller to be toggled by the user. All other exponentially displayed parameters are unipolar, so no sign is displayed for these parameters.
- In fixed decimal format a value between −10 and +10 may be selected using [▲▼] (and [Shift]).
- The two outputs, $\boldsymbol{P \times \varepsilon}$, and ***Output***, are accompanied by bar displays on the right side of the front panel. $\boldsymbol{P \times \varepsilon}$ simply ranges from −10 V to +10 V. However, since the controller output ranges between the user-programmed upper and lower limits, the output bar display is normalized to that range. For example, if the limits were set to +5 V and −1 V, the full range of the bar display would be 6 V, and 0 V would no longer correspond to the center of the bar display, but would be 1/6th of the way up from the bottom. The default limits are ±10 V.
- Use [Setpoint] in the INPUTS section of the front panel to choose between an external setpoint input, and the internally generated setpoint. An external setpoint can be supplied at the Setpoint BNC input. When the internal setpoint is selected the BNC connector is disconnected from the SIM960 circuitry.
- The Output BNC connector can be toggled between PID Control mode and Manual mode using [Output] (in the OUTPUT section of the front panel). In manual mode, the SIM960 output is set to the value indicated by the manual parameter.

### 1.2.1 Inputs

The common mode range of the "Measure" and "Setpoint" inputs extends from −10 V to +10 V. If either input is outside this range, the overload LED indicator lights.

The differential input range is ±1 V. Whenever the difference between Setpoint and Measure exceeds this range, the overload LED indicator turns on. When connected with overall negative feedback and reasonably well tuned, the SIM960 keeps the difference between the setpoint and measure inputs as small as possible, so the differential input range is unlikely to be exceeded. Before the SIM960 has been tuned for a given system, however, this may not be true. It is helpful to keep in mind that exceeding the ±1 V differential input range will saturate the error amplifier, even if the output signal would otherwise be within the upper and lower ***Limits*** setting. In such situations, the controller will be effectively limited at some intermediate value.

### 1.2.2 Ramping

The ramping feature of the SIM960 PID Controller allows the user to linearly slew the internally generated setpoint level from its current value to a new value. The slew rate may be changed using the ***SP Ramp*** parameter on the front panel.

The indicator to the right of ***SP Ramp*** shows whether ramping is enabled or disabled. Use [On/Off] (with ***SP Ramp*** selected) to enable/disable ramping. When disabled, changes to the ***Internal Setpoint*** parameter take effect immediately. When ramping is enabled, however, changes to ***Internal Setpoint*** do not immediately take effect. Instead, *Internal* (in the Setpoint block of the INPUTS section of the front panel) begins to blink, showing that a new setpoint has been entered and a ramp event is now pending.

To begin the ramp, press [Ramp Start/Stop]. Now, the *Internal* blink rate doubles, indicating that the setpoint is ramping. To pause the ramp, press [Ramp Start/Stop]When the ramp is paused, the *Internal* blink rate becomes uneven. To continue the ramp, press [Ramp Start/Stop] again. When the setpoint reaches the new programmed value, the ramp automatically terminates, and *Internal* stops blinking.

Note, ***SP Ramp*** has no sign in the numerical display. This is because the polarity of the ramp rate is unambiguously determined by whether the newly entered setpoint is greater or less than the current setpoint. The range of available ramp rates is from 1 mV/s to 10,000 V/s. For ramp rates less than or equal to 1 V/s, the rate is dynamically trimmed based on real-time measurements from the onboard A-to-D converter.

### 1.2.3 Connections

Connect the sensor output of the system to be controlled to the "Measure" input of the SIM960. If an external setpoint is to be supplied, connect this to the "Setpoint" input, and use the button in the INPUTS section of the front panel to select "External" input. Before connecting the SIM960 output to the system to control, it may be necessary to set the user programmable output upper and lower ***Limits*** to guard against damaging the system. Care should be taken to insure that the programmed output range is consistent with the system input range. Once the limits have been programmed, connect the SIM960 output to the system input.

### 1.2.4 Bar displays

Two LED bar displays have been included on the right side of the SIM960 front panel to provide visual information about the ***P* × *ε*** and ***Output*** signals. This reduces the need to frequently return to those fields on the numerical display while trying to adjust other tuning parameters. Some time should be taken to understand what information these bar displays provide.

Each bar has two lighted LEDs; one for the maximum peak of the signal, and one for the minimum peak. The peaks are determined with respect to time variation of the signal, and they decay back to the DC level with a decay time of ∼100 ms.

To understand how a signal is represented in the bar display, consider an input sine wave of frequency 1 Hz. Since frequency is low compared to the inverse of the decay time, the maximum and minimum peak values are indistinguishable, and the signal appears as a single LED that tracks the sine wave. As the frequency increases, the maximum peak does not decay quickly enough to track the negative excursions the signal, and the minimum peak also fails to track positive excursions. So there appear to be two lighted LEDs slightly separated, roughly tracking the sine wave. As the frequency is further increased to well above the decay time inverse, the two lighted LEDs no longer decay at all from their peak levels, so there appear to be two lighted LEDs marking the maximum and minimum peaks of the sine wave.

Thus, a slowly varying signal appears as a single lighted LED in the display, tracking the signal changes with time. But a quickly varying signal, however, appears as two lighted LEDs marking the maximum and minimum excursions of the signal in time.

The range of the ***P* × *ε*** bar display is ±10 V. The ***Output*** bar display has a range determined by the user programmed upper and lower limits. For example, if the limits were set to +5 V and −1 V, the full range of the bar display would be 6 V, and 0 V would no longer correspond to the center of the bar display, but would be 1/6th of the way up from the bottom. Also, the ***Output*** bar display has a red LED on each end to indicate whether the controller output is saturated at its limit.

### 1.2.5 Restoring the default configuration

The default configuration of the SIM960 can be restored in either of two ways: From the front panel, or via the remote interface.

To restore from the front panel, first turn off the power to the SIM960 by switching its SIM900 Mainframe to "Standby," then switch the power on while holding down [Ramp Start/Stop]. Keep the button depressed for about one second after power comes on.

The default configuration can also be restored via the remote interface using the *RST command.

## 1.3 Rear Panel Monitoring

Two analog monitor signals are available at the rear panel of the SIM960 (see Figure 1.2).

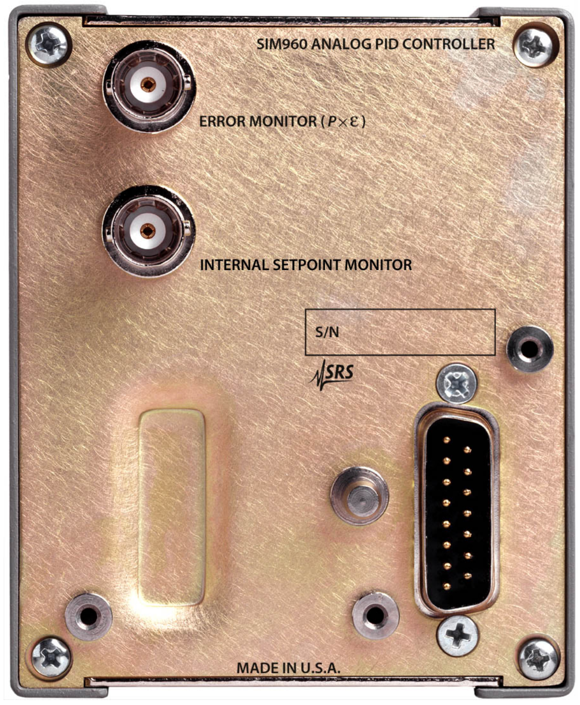
SIM960 ANALOG PID CONTROLLER
ERROR MONITOR ($P\times\varepsilon$)
INTERNAL SETPOINT MONITOR
S/N
SRS
MADE IN U.S.A.

Figure 1.2: The SIM960 rear panel.

### 1.3.1 Error Monitor

The upper BNC connector carries a buffered copy of the $\boldsymbol{P} \times \boldsymbol{\varepsilon}$ output of the error amplifier. This output is always available, even when the $\boldsymbol{P}$ term is disabled from the control law. It is also active when the main SIM960 output is set to Manual mode.

### 1.3.2 Input Setpoint Monitor

The lower BNC is a copy of the internally generated setpoint voltage. This output is also always available, even when the Setpoint mode is set to External.

## 1.4 SIM Interface

The primary connection to the SIM960 Analog PID Controller is the rear-panel DB–15 SIM interface connector. Typically, the SIM960 is mated to a SIM900 Mainframe via this connection, either through one of the internal mainframe slots, or the remote cable interface.

It is also possible to operate the SIM960 directly, without using the SIM900 Mainframe. This section provides details on the interface.

### 1.4.1 SIM interface connector

The DB–15 SIM interface connector carries all the power and communications lines to the instrument. The connector signals are specified in Table 1.1

| Pin | Signal | Direction<br>Src ⇒ Dest | Description |
|---|---|---|---|
| 1 | SIGNAL_GND | MF ⇒ SIM | Ground reference for signal |
| 2 | −STATUS | SIM ⇒ MF | Status/service request (GND=asserted, +5V=idle) |
| 3 | RTS | MF ⇒ SIM | HW Handshake (+5 V=talk; GND=stop) |
| 4 | CTS | SIM ⇒ MF | HW Handshake (+5 V=talk; GND=stop) |
| 5 | −REF_10MHZ | MF ⇒ SIM | 10 MHz reference (optional connection) |
| 6 | −5V | MF ⇒ SIM | Power supply (No connection in SIM960) |
| 7 | −15V | MF ⇒ SIM | Power supply (analog circuitry) |
| 8 | PS_RTN | MF ⇒ SIM | Power supply return |
| 9 | CHASSIS_GND | | Chassis ground |
| 10 | TXD | MF ⇒ SIM | Async data (start bit="0"=+5 V; "1"=GND) |
| 11 | RXD | SIM ⇒ MF | Async data (start bit="0"=+5 V; "1"=GND) |
| 12 | +REF_10MHz | MF ⇒ SIM | 10 MHz reference (optional connection) |
| 13 | +5V | MF ⇒ SIM | Power supply (digital circuitry) |
| 14 | +15V | MF ⇒ SIM | Power supply (analog circuitry) |
| 15 | +24V | MF ⇒ SIM | Power supply (No connection in SIM960) |

Table 1.1: SIM Interface Connector Pin Assignments, DB-15

### 1.4.2 Direct interfacing

The SIM960 is intended for operation in the SIM900 Mainframe, but users may wish to directly interface the module to their own systems without the use of additional hardware.

The mating connector needed is a standard DB–15 receptacle, such as Amp part # 747909-2 (or equivalent). Clean, well-regulated supply voltages of +5,±15 VDC must be provided, following the pin-out specified in Table 1.1. Ground must be provided on Pins 1 and 8, with chassis ground on Pin 9. The −STATUS signal may be monitored on Pin 2 for a low-going TTL-compatible output indicating a status message.

*The SIM960 has no internal protection against reverse polarity, missing supply, or overvoltage on the power supply pins.*

#### 1.4.2.1 Direct interface cabling

If the user intends to directly wire the SIM960 independent of the SIM900 Mainframe, communication is usually possible by directly connecting the appropriate interface lines from the SIM960 DB–15 plug to the RS-232 serial port of a personal computer. [1] Connect RXD from the SIM960 directly to RD on the PC, TXD directly to TD, and similarly RTS→RTS and CTS→CTS. In other words, a null-modem style cable is *not* needed.

To interface directly to the DB–9 male (DTE) RS-232 port typically found on contemporary personal computers, a cable must be made with a female DB–15 socket to mate with the SIM960, and a female DB–9 socket to mate with the PC's serial port. Separate leads from the DB–15 need to go to the power supply, making what is sometimes know as a "hydra" cable. The pin-connections are given in Table 1.2.

| DB–15/F to SIM960 | | | Name |
|---|---|---|---|
| | | DB–9/F | |
| 3 | ⟷ | 7 | RTS |
| 4 | ⟷ | 8 | CTS |
| 10 | ⟷ | 3 | TxD |
| 11 | ⟷ | 2 | RxD |
| | | 5 | Computer Ground |
| | | to P/S | |
| 7 | ⟷ | −15 VDC | |
| 14 | ⟷ | +15 VDC | |
| 13 | ⟷ | +5 VDC | |
| 8,9 | ⟷ | Ground (P/S return current) | |
| 1 | ⟷ | Signal Ground (separate wire to Ground) | |

Table 1.2: SIM960 Direct Interface Cable Pin Assignments

[1] Although the serial interface lines on the DB-15 do not satisfy the minimum voltage levels of the RS-232 standard, they are typically compatible with desktop personal computers

### 1.4.2.2 Serial settings

The initial serial port settings at power-on are: 9600 Baud, 8–bits, no parity, 1 stop bit, and RTS/CTS flow control. These may be changed with the BAUD, FLOW, or PARI commands.

The maximum *standard* baud rate that the SIM960 supports is 38400. The minimum baud rate is 110. Above 38400, the SIM960 can be set to the following (non-RS–232-standard) baud rates: 62500, 78125, 104167, 156250. Note that these rates are typically not accessible on a standard PC RS–232 port, but can be used between the SIM960 and the SIM900 Mainframe.

# 2 Advanced Topics

This chapter discusses a simple "closed-loop" tuning procedure, along with some of the advanced features of the SIM960 Analog PID Controller.

## In This Chapter

## 2.1 PID Tuning Basics

PID control provides a simple way to minimize the effect of disturbances to a system. The system consists of a closed feedback loop between two elements, the SIM960 **controller** and the user **process**. The controller has two inputs, ***Setpoint*** and ***Measure***, and an ***Output***. The process consists of a power source that can be directly changed by the controller, in conjunction with a sensor to monitor the process behavior. The sensor signal, after any necessary conditioning, is the process output. This should be connected to the ***Measure*** input of the SIM960, and the SIM960 ***Output*** should be connected to the process input, forming a feedback loop.

The difference between the ***Setpoint*** and ***Measure*** inputs is the error signal, $\varepsilon \equiv$ ***Setpoint*** − ***Measure*** (Eqn 1.1). In the SIM960 the error signal is amplified by the proportional gain. The controller uses the amplified error, $\boldsymbol{P} \times \varepsilon$, to generate three control signals:

1. Proportional, the ***P*** amplified error with no changes.
2. Integral, the time integral of the amplified error signal multiplied by a gain coefficient ***I***.
3. Derivative, the time derivative of the amplified error signal multiplied by a gain coefficient ***D***.

These signals, as well as an ***Offset***, are combined at a summing junction to produce the controller output (see Eqn 1.2). Figure 2.1 shows a schematic representation of the SIM960 controller topology. Note the proportional gain coefficient is common to all three terms, so the net integral and derivative gains are ***PI*** and ***PD***, respectively, whether or not proportional control is enabled.

The controller monitors the process output and makes small adjustments to the process in order to minimize deviations of ***Measure*** from ***Setpoint*** due to external disturbances. To accomplish this, the controller must be properly tuned, meaning that the gains for each of the three control signals—proportional, integral, and derivative—must be chosen appropriately to match the behavior of the process. A well-tuned controller should be able to maintain a stable process output.

The control loop feedback should be negative. However, because the polarity of the process response to the controller output is an arbitrary function of the design of the system, it is vital that the controller polarity be chosen properly. Based on the topology of the SIM960 design, feedback polarity can be changed simply by changing the polarity of the proportional gain parameter ***P***. The user must first determine which polarity will provide negative feedback. If the

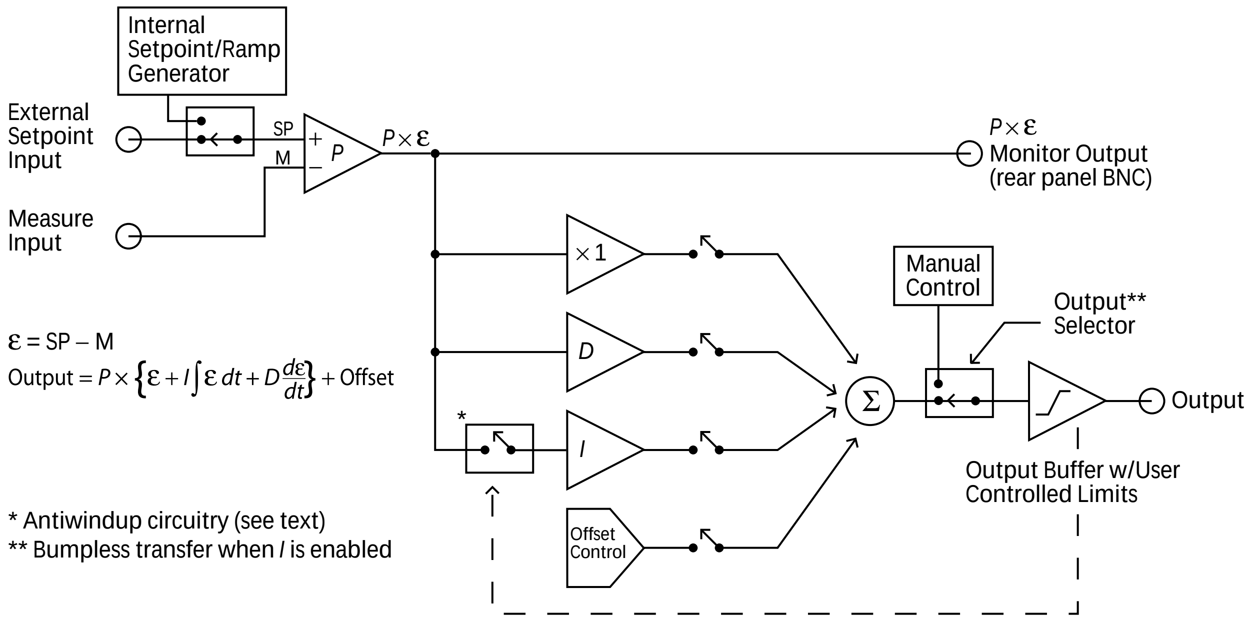
Internal Setpoint/Ramp Generator
External Setpoint Input
Measure Input
SP
M
P
P × ε
P × ε Monitor Output (rear panel BNC)
×1
D
I
Offset Control
Manual Control
Output** Selector
Σ
Output
Output Buffer w/User Controlled Limits
ε = SP – M
Output = P × {ε + I ∫ ε dt + D dε/dt} + Offset
* Antiwindup circuitry (see text)
** Bumpless transfer when I is enabled

Figure 2.1: The SIM960 block diagram.

process is noninverting, i.e. a small positive change at its input results in a positive change at its output, then using positive ***P*** polarity will ensure negative feedback in the loop. To see this, follow the effect of a small positive change at the process output. Since the process output is connected to the ***Measure*** input of the SIM960, a small positive change would cause a negative change to $\varepsilon$. The resulting change at the controller ***Output*** would also be negative, as would be that of the process output. Thus, the initial small positive change at the process output is "corrected" by a negative change after going around the feedback loop. As a general rule, if the process is noninverting, then the ***P***-polarity should be positive. If the process is inverting, negative ***P***-polarity should be used.

Care should be taken in designing the process. The sensor should be situated so that it is responsive to changes to the part of the system under control. Placing the sensor too remotely can result in a time delay which limits the quality of control. Also, the sensor should primarily measure the system's *response* to external changes, rather than measure the changes directly. The latter can sometimes be used to help the controller anticipate transients, but at the risk of sacrificing accuracy in reaching the target setpoint.

Tuning a PID controller amounts to determining what the relative contributions should be from each of the three types of control. The simplest approach is to start with proportional control and add integral and derivative one at a time. A simple ***P***-controller generates a control variable that is proportional to the error signal.[1]

Increasing the ***P*** gain should cause the process output to respond by moving closer to the setpoint. Generally, enough amplification should be used so that the process output is brought reasonably close to the setpoint. Too much gain, however, will cause the system to oscillate. Start with a small ***P*** gain, and increase by factors of two until the system begins to oscillate. Then back off in small amounts until stability is recovered.

While it is possible to maintain stability with a simple ***P*** controller, in general this will lead to a finite, non-zero $\boldsymbol{\varepsilon}$. Increasing ***P*** will tend to reduce the resulting $\boldsymbol{\varepsilon}$, but too much proportional gain will eventually lead to oscillations.

One way to eliminate this nonzero error problem is to include an offset at the controller output. The SIM960 ***Offset*** parameter can be turned on and adjusted to hold the process power at a level that maintains a smaller error. However, this is only a coarse improvement, since the necessary power level may change with time.

Integral control provides an "automatic" way to dynamically adjust the effective offset to zero the error; in older controllers, integral action was called "automatic reset" for this reason. Integral control simply integrates the error signal with respect to time. Thus the controller output changes until the error has been reduced to zero, near which point the controller output slows and stops changing. If the error drifts over time, the integrator responds by adjusting the controller output to cancel the error. So it is much like having a dynamic output offset constantly responding to system changes. As with proportional gain, too much integral gain can cause oscillation. Again, start with a small ***I*** gain and increase by factors of two until oscillation begins, then back off until stability is recovered.

Though integral control is effective at reducing the error, it is not as effective as proportional control at responding quickly to changes. This is because the integrator needs time to build up a response. To further enhance the response of the process to rapid changes, derivative control is often employed. Derivative control is proportional to the rate of change of the error, so it is relatively unresponsive to slow changes, but rapid changes to the system produce a significant response. Derivative control reduces oscillations that can result from step changes to a system.

[1] During the tuning process, it is important to keep in mind that the differential input range of the SIM960 is ±1.0 V. It is good practice to occasionally glance at the *OVLD* indicator to ensure the input amplifier is not saturated.

## 2.2 Anti-Windup and Conditional Integration

For better integral performance, the SIM960 features anti-windup circuitry in the form of conditional integration. The purpose of anti-windup is to improve the controller's ability to recover from output saturation. When the output saturates, the error is likely to be large, since the process is unable to provide power fast enough to recover the process output. However, the integrator contribution may not account for the full amount of the controller output in this case. Subsequently, the integrator continues to integrate the error until the integrator output saturates. This "winding up" aspect of integral control becomes a problem when the process recovers and the error level passes through zero, because the error must move significantly beyond zero for the integrator to "unwind" from saturation. In general, once the controller output is clamped at a limit, nothing is accomplished by driving it harder into that limit by more integration. In fact, it only makes it harder to recover from saturation, since the result is usually large swings back and forth from limit to limit.

There are a variety of anti-windup strategies to mitigate this effect. A simple way to implement anti-windup is to switch off the integrator whenever the output saturates. This is not the same as resetting the integrator (zeroing its output by discharging the feedback capacitance) because the output simply stops moving, but does not go to zero. It is equivalent to momentarily zeroing the integrator *input*, so that there is no signal to integrate while the output is saturated.

An improvement to this scheme comes from recognizing that not all saturation conditions cause unwanted integrator wind-up. For example, suppose the controller/process history were such as to produce the following conditions:

- Error signal negative
- Integrator output finite, not saturated
- Controller output saturated at the positive limit

Then, the integrator output would be moving in the negative direction, since its input, the error, is negative. This would not cause the controller output to be pushed harder into saturation; in fact it may eventually pull it out of saturation. So stopping the integrator would hinder the controller's effort to recover the process variable. The SIM960 uses a technique called "conditional integration:" *Conditional integration only stops the integrator when the polarity of the error is such as to drive the integrator toward the saturated limit.*

## 2.3 Bumpless Transfer

When switching the output mode between ***Manual*** and PID Control, transients on the output signal can disturb the system under control. Minimizing these switching transients is known as "bumpless transfer." The SIM960 supports bumpless transfer under certain conditions, as described below.

### 2.3.1 Manual-to-PID

When switching from ***Manual*** output to PID Control output, bumpless transfer is only possible if the integral term is enabled.[2] When ***I*** is turned on and the SIM960 is in ***Manual*** output mode, the input to the integrator is rerouted to integrate the difference between ***Manual*** and the (deselected) PID Control output. This effectively allows the PID Control to "track" the ***Manual*** value, presetting the integrator as necessary. Then, when the output is switched back to PID Control, the controller output is already the same as the ***Manual*** output level. Were this not the case, the integrator output would likely saturate while in manual mode, and upon switching to PID Control mode, the controller output would suddenly jump. Bumpless transfer insures that the transition from ***Manual*** to PID Control mode is smooth.

### 2.3.2 PID-to-Manual

An additional feature of the SIM960 is the ability to preset the manual level to the current PID control output level, so that switching from PID mode to manual mode will also be smooth. With the module in PID mode, select the ***Manual*** field. Press [On/Off] and hold it down for at least one second. After one second the manual display reading will shift to the current PID output level. The output mode will remain in PID control mode until it is manually switched on the front panel or through the remote interface. But the new manual output level will be equal to the PID control output.

[2] This can be understood mathematically, since only the integral term has an "unspecified" initial offset value that can be set to an arbitrary value without violating Eqn 1.2.

# 3 Remote Operation

This chapter describes operating the module over the serial interface.

## In This Chapter

## 3.1 Index of Common Commands

| symbol | definition |
|---|---|
| *i,j* | Integers |
| *f,g* | Floating-point values |
| *z* | Literal token |
| (?) | Required for queries; illegal for set commands |
| *var* | Parameter always required |
| {*var*} | Required parameter for set commands; illegal for queries |
| [*var*] | Optional parameter for both set and query forms |

### Controller Settings

| | | |
|---|---|---|
| DISP(?) {*z*} | 3 – 10 | Select Field |
| SHFT(?) {*z*} | 3 – 10 | Shift Status |
| PCTL(?) *z* | 3 – 10 | Proportional action ON/OFF |
| ICTL(?) *z* | 3 – 10 | Integral action ON/OFF |
| DCTL(?) *z* | 3 – 10 | Derivative action ON/OFF |
| OCTL(?) *z* | 3 – 10 | Offset ON/OFF |
| RAMP(?) *z* | 3 – 11 | Internal setpoint ramping ON/OFF |
| INPT(?) *z* | 3 – 11 | Input (Internal/External Setpoint) |
| AMAN(?) *z* | 3 – 11 | Output (Manual Output/PID Control) |
| GAIN(?) {*f*} | 3 – 11 | Proportional Gain |
| APOL(?) *z* | 3 – 11 | Controller Polarity |
| INTG(?) {*f*} | 3 – 11 | Integral Gain |
| DERV(?) {*f*} | 3 – 11 | Derivative Gain |
| OFST(?) {*f*} | 3 – 11 | Output Offset |
| RATE(?) {*f*} | 3 – 11 | Setpoint ramping Rate |
| RMPS? | 3 – 12 | Setpoint ramping status |
| STRT *z* | 3 – 12 | Pause or continue ramping |
| SETP(?) {*f*} | 3 – 12 | New setpoint |
| MOUT(?) {*f*} | 3 – 12 | Manual Output |
| ULIM(?) {*f*} | 3 – 12 | Upper Output Limit |
| LLIM(?) {*f*} | 3 – 12 | Lower Output Limit |
| SMON? [*i*] | 3 – 12 | Setpoint Input Monitor |
| MMON? [*i*] | 3 – 13 | Measure Input Monitor |
| EMON? [*i*] | 3 – 13 | Amplified Error Monitor |
| OMON? [*i*] | 3 – 13 | Output Monitor |
| RFMT(?) {*z*} | 3 – 13 | Output Streaming Records Format |
| SOUT [*z*] | 3 – 13 | Stop Streaming |
| FPLC(?) {*i*} | 3 – 13 | Frequency of Power Line Cycle |
| DISX(?) {*z*} | 3 – 14 | Front Panel Display Enable |

## 3.2 Alphabetic List of Commands

| | | |
|---|---|---|
| **★** | | |
| *CLS | 3 – 14 | Clear Status |
| *ESE(?) [*i*,] {*j*} | 3 – 15 | Standard Event Status Enable |
| *ESR? [*i*] | 3 – 15 | Standard Event Status |
| *IDN? | 3 – 17 | Identify |
| *OPC(?) | 3 – 17 | Operation Complete |
| *RST | 3 – 16 | Reset |
| *SRE(?) [*i*,] {*j*} | 3 – 14 | Service Request Enable |
| *STB? [*i*] | 3 – 14 | Status Byte |
| *TST? | 3 – 17 | Self Test |
| **A** | | |
| ADSE(?) [*i*], {*j*} | 3 – 16 | A-to-D status enable register |
| ADSR? [*i*] | 3 – 15 | A-to-D status register |
| AMAN(?) *z* | 3 – 11 | Output (Manual Output/PID Control) |
| APOL(?) *z* | 3 – 11 | Controller Polarity |
| **B** | | |
| BAUD(?) {*i*} | 3 – 14 | Baud Rate |
| **C** | | |
| CESE(?) [*i*,]{*j*} | 3 – 15 | Comm Error Status Enable |
| CESR? [*i*] | 3 – 15 | Comm Error Status |
| CONS(?) {*z*} | 3 – 17 | Console Mode |
| **D** | | |
| DCTL(?) *z* | 3 – 10 | Derivative action ON/OFF |
| DERV(?) {*f*} | 3 – 11 | Derivative Gain |
| DISP(?) {*z*} | 3 – 10 | Select Field |
| DISX(?) {*z*} | 3 – 14 | Front Panel Display Enable |
| **E** | | |
| EMON? [*i*] | 3 – 13 | Amplified Error Monitor |
| **F** | | |
| FLOW(?) {*z*} | 3 – 14 | Flow Control |
| FPLC(?) {*i*} | 3 – 13 | Frequency of Power Line Cycle |
| **G** | | |
| GAIN(?) {*f*} | 3 – 11 | Proportional Gain |

| | | |
|---|---|---|
| **I** | | |
| ICTL(?) *z* | 3 – 10 | Integral action ON/OFF |
| INCR? [*i*] | 3 – 15 | Instrument condition register |
| INPT(?) *z* | 3 – 11 | Input (Internal/External Setpoint) |
| INSE(?) [*i*], {*j*} | 3 – 15 | Instrument status enable register |
| INSR? [*i*] | 3 – 15 | Instrument status register |
| INTG(?) {*f*} | 3 – 11 | Integral Gain |
| **L** | | |
| LBTN? | 3 – 19 | Last Button |
| LCME? | 3 – 18 | Command Error |
| LEXE? | 3 – 18 | Execution Error |
| LLIM(?) {*f*} | 3 – 12 | Lower Output Limit |
| **M** | | |
| MMON? [*i*] | 3 – 13 | Measure Input Monitor |
| MOUT(?) {*f*} | 3 – 12 | Manual Output |
| **O** | | |
| OCTL(?) *z* | 3 – 10 | Offset ON/OFF |
| OFST(?) {*f*} | 3 – 11 | Output Offset |
| OMON? [*i*] | 3 – 13 | Output Monitor |
| **P** | | |
| PARI(?) {*z*} | 3 – 14 | Parity |
| PCTL(?) *z* | 3 – 10 | Proportional action ON/OFF |
| PSTA(?) {*z*} | 3 – 16 | Pulse −STATUS Mode |
| **R** | | |
| RAMP(?) *z* | 3 – 11 | Internal setpoint ramping ON/OFF |
| RATE(?) {*f*} | 3 – 11 | Setpoint ramping Rate |
| RFMT(?) {*z*} | 3 – 13 | Output Streaming Records Format |
| RMPS? | 3 – 12 | Setpoint ramping status |
| **S** | | |
| SETP(?) {*f*} | 3 – 12 | New setpoint |
| SHFT(?) {*z*} | 3 – 10 | Shift Status |
| SMON? [*i*] | 3 – 12 | Setpoint Input Monitor |
| SOUT [*z*] | 3 – 13 | Stop Streaming |
| STRT *z* | 3 – 12 | Pause or continue ramping |
| **T** | | |
| TERM(?) {*z*} | 3 – 19 | Response Termination |

## 3.3 Introduction

Remote operation of the SIM960 is through a simple command language documented in this chapter. Both set and query forms of most commands are supported, allowing the user complete control of the amplifier from a remote computer, either through the SIM mainframe or directly via RS-232 (see section 1.4.2.1).

See Table 1.1 for the specification of the DB–15 SIM Interface Connector.

### 3.3.1 Power-on configuration

The settings for the remote interface are 9600 baud with no parity and hardware flow control, and local echo disabled (CONS OFF).

Most of the SIM960 instrument settings are stored in non-volatile memory, and at power-on the instrument returns to the state it was last in when power was removed. Exceptions are noted in the command descriptions.

Reset values of parameters are shown in **boldface**.

### 3.3.2 Buffers

The SIM960 stores incoming bytes from the host interface in a 32-byte Input Buffer. Characters accumulate in the Input Buffer until a command terminator (either ⟨CR⟩ or ⟨LF⟩) is received, at which point the message is parsed and executed. Query responses from the SIM960 are buffered in a 32-byte Output Queue.

If the Input Buffer overflows, then all data in *both* the Input Buffer and the Output Queue are discarded, and an error is recorded in the CESR and ESR status registers.

### 3.3.3 Device Clear

The SIM960 host interface can be asynchronously reset to its power-on configuration by sending an RS-232-style ⟨break⟩ signal. From the SIM900 Mainframe, this is accomplished with the SRST command; if directly interfacing via RS-232, then use a serial break signal. After receiving the Device Clear, the interface is reset to 9600 baud and CONS mode is turned OFF. Note that this *only* resets the communication interface; the basic function of the SIM960 is left unchanged; to reset the meter, see *RST.

The Device Clear signal will also terminate any streaming outputs from the SIM960 due to an SMON?, MMON?, EMON? and/or OMON? query of multiple conversions.

## 3.4 Commands

This section provides syntax and operational descriptions for remote commands.

### 3.4.1 Command syntax

The four letter mnemonic (shown in CAPS) in each command sequence specifies the command. The rest of the sequence consists of parameters.

Commands may take either *set* or *query* form, depending on whether the "?" character follows the mnemonic. *Set only* commands are listed without the "?", *query only* commands show the "?" after the mnemonic, and *optionally query* commands are marked with a "(?)".

Parameters shown in { } and [ ] are not always required. Parameters in { } are required to set a value, and should be omitted for queries. Parameters in [ ] are optional in both set and query commands. Parameters listed without any surrounding characters are always required.

Do *not* send ( ) or { } or [ ] as part of the command.

Multiple parameters are separated by commas. Multiple commands may be sent on one command line by separating them with semicolons (;) so long as the Input Buffer does not overflow. Commands are terminated by either ⟨CR⟩ or ⟨LF⟩ characters. Null commands and whitespace are ignored. Execution of the command does not begin until the command terminator is received.

tokens Token parameters (generically shown as *z* in the command descriptions) can be specified either as a keyword or integer value. Command descriptions list the valid keyword options, with each keyword followed by its corresponding integer value. For example, to set the response termination sequence to ⟨CR⟩+⟨LF⟩, the following two commands are equivalent:

TERM CRLF —or— TERM 3

For queries that return token values, the return format (keyword or integer) is specified with the TOKN command.

The following table summarizes the notation used in the command descriptions:

| symbol | definition |
|---|---|
| *i*,*j* | Integers |
| *f*,*g* | Floating-point values |
| *z* | Literal token |
| | |
| (?) | Required for queries; illegal for set commands |
| *var* | Parameter always required |
| {*var*} | Required parameter for set commands; illegal for queries |
| [*var*] | Optional parameter for both set and query forms |

### 3.4.2 Controller settings commands

| | |
|---|---|
| DISP(?) {*z*} | Select Field<br><br>Set (query) the field level to be displayed {to *z*}. Allowed values of *z* are<br>**PRP 0** Proportional gain<br>IGL 1 Integral gain<br>DER 2 Derivative gain<br>OFS 3 Output offset<br>RTE 4 Setpoint rate<br>STP 5 Setpoint value<br>MNL 6 Manual output value<br>ULM 7 Upper limit of output<br>LLM 8 Lower limit of output<br>SMN 9 ADC measurement of Setpoint input<br>MMN 10 ADC measurement of Measure input<br>EMN 11 ADC measurement of P-Amplified error<br>OMN 12 ADC measurement of PID/Manual Output |
| SHFT(?) {*z*} | Shift Status<br><br>Set (query) the current shift status {to *i*=(**OFF 0**, ON 1)}. |
| PCTL(?) *z* | Proportional action ON/OFF<br><br>Set (query) the proportional control {to *z*=(OFF 0, **ON 1**)}.<br><br>When ON, the PID Control path includes the proportional control term. |
| ICTL(?) *z* | Integral action ON/OFF<br><br>Set (query) the integral control {to *z*=(**OFF 0**, ON 1)}.<br><br>When ON, the PID Control path includes the integral control term. |
| DCTL(?) *z* | Derivative action ON/OFF<br><br>Set (query) the derivative control {to *z*=(**OFF 0**, ON 1)}.<br><br>When ON, the PID Control path includes the derivative control term. |
| OCTL(?) *z* | Offset ON/OFF<br><br>Set (query) the offset control {to *z*=(**OFF 0**, ON 1)}.<br><br>When ON, the PID Control path includes the constant output offset. |

| | |
|---|---|
| RAMP(?) *z* | Internal setpoint ramping ON/OFF<br>Set (query) internal setpoint ramping {to *z*=(**`OFF 0`**, `ON 1`)}.<br>When `ON`, the changes to the internal setpoint are made with constant slew-rate ramping enabled. |
| INPT(?) *z* | Input (Internal/External Setpoint)<br>Set (query) setpoint input state {to *z*=(`INT 0`, **`EXT 1`**)}. |
| AMAN(?) *z* | Output (Manual Output/PID Control)<br>Set (query) controller output state {to *z*=(`MAN 0`, **`PID 1`**)}. |
| GAIN(?) {*f*} | Proportional Gain<br>Set (query) the proportional gain {to *f* }.<br>Values may be entered in exponential format. |
| APOL(?) *z* | Controller Polarity<br>Set (query) the proportional gain polarity {to *z*=(**`POS 1`**, `NEG 0`)}. |
| INTG(?) {*f*} | Integral Gain<br>Set (query) the integral gain {to *f* }.<br>Values may be entered in exponential format. |
| DERV(?) {*f*} | Derivative Gain<br>Set (query) the derivative gain {to *f* }.<br>Values may be entered in exponential format. |
| OFST(?) {*f*} | Output Offset<br>Set (query) the output offset {to *f* }. |
| RATE(?) {*f*} | Setpoint ramping Rate<br>Set (query) the setpoint rate {to *f* }.<br>Values may be entered in exponential format. |

| | |
|---|---|
| RMPS? | Setpoint ramping status<br>Query the ramp status.<br>The response is one of the following token values: **`IDLE 0`**, `RAMP_PENDING 1`, `RAMPING 2`, `PAUSED 3`. |
| STRT *z* | Pause or continue ramping<br>Cause a ramping event in progress to pause (STOP) or continue (START). *z*=(`STOP 0`, `START 1`). |
| SETP(?) {*f*} | New setpoint<br>Set (query) the setpoint value {to *f*}.<br>If ramping is enabled, SETP will initiate a ramp to *f*. Otherwise, the setpoint value changes immediately to the new value. |
| MOUT(?) {*f*} | Manual Output<br>Set (query) the manual output value {to *f*}. |
| ULIM(?) {*f*} | Upper Output Limit<br>Set (query) the upper output limit {to *f*}.<br>The default value after Reset is +10.00 V. |
| LLIM(?) {*f*} | Lower Output Limit<br>Set (query) the lower output limit {to *f*}.<br>The default value after Reset is −10.00 V. |
| SMON? [*i*] | Setpoint Input Monitor<br>Query the ***Setpoint*** input voltage to the error amplifier.<br>If INPT `INT` is set, then SMON? monitors the value of the internally-generated setpoint. If INPT `EXT`, then SMON? monitors the voltage applied at the front-panel Setpoint BNC input.<br>*i* is an optional parameter that causes streaming of ***Setpoint*** data. If *i* is specified, then *i* measurements will be output at a rate of approximately half a second per measurement. If *i* is specified as 0, then measurements will be output indefinitely. The SOUT command can be used to stop streaming. |

| | |
|---|---|
| MMON? [*i*] | Measure Input Monitor<br><br>Query the ***Measure*** input voltage to the error amplifier. This is always the voltage applied at the front-panel Measure BNC input.<br><br>*i* is an optional parameter that causes streaming of ***Measure*** data. If *i* is specified, then *i* measurements will be output at a rate of approximately half a second per measurement. If *i* is specified as 0, then measurements will be output indefinitely. The SOUT command can be used to stop streaming. |
| EMON? [*i*] | Amplified Error Monitor<br><br>Query the $\boldsymbol{P} \times \varepsilon$ voltage.<br><br>*i* is an optional parameter that causes streaming of ($\boldsymbol{P} \times \varepsilon$) data. If *i* is specified, then *i* measurements will be output at a rate of approximately half a second per measurement. If *i* is specified as 0, then measurements will be output indefinitely. The SOUT command can be used to stop streaming. |
| OMON? [*i*] | Output Monitor<br><br>Query the ***Output*** voltage.<br><br>*i* is an optional parameter that causes streaming of ***Output*** data. If *i* is specified, then *i* measurements will be output at a rate of approximately half a second per measurement. If *i* is specified as 0, then measurements will be output indefinitely. The SOUT command can be used to stop streaming. |
| RFMT(?) {*z*} | Output Streaming Records Format<br><br>Set (query) the output streaming format {to *z*=(**`OFF 0`**, `ON 1`)}.<br><br>When `ON`, data are output on a single line with three comma delimiters. Since there are four channels that can be streamed to output, and any combination of the four may be streamed, the comma delimiters allow unambiguous identification of channel data. |
| SOUT [*z*] | Stop Streaming<br><br>Turn off streaming (of channel *z*= (`SMN 0`, `MMN 1`, `EMN 2`, `OMN 3`)).<br><br>If the optional parameter *z* is not specified, then all streaming outputs are turned off. |
| FPLC(?) {*i*} | Frequency of Power Line Cycle<br><br>Set (query) the power line cycle frequency {to *i*=(`50`, **`60`**)} Hz. |

| | |
|---|---|
| DISX(?) {*z*} | Front Panel Display Enable<br>Set (query) the front panel display status {to *z*=(`OFF 0`, **`ON 1`**)}. |

### 3.4.3 Serial communication commands

| | |
|---|---|
| BAUD(?) {*i*} | Baud Rate<br>Set (query) the baud rate {to *i*}.<br>At power-on, the baud rate defaults to 9600. |
| FLOW(?) {*z*} | Flow Control<br>Set (query) flow control {to *z*=(`NONE 0`, **`RTS 1`**, `XON 2`)}.<br>At power-on, the SIM960 defaults to FLOW `RTS` flow control. |
| PARI(?) {*z*} | Parity<br>Set (query) parity {to *z* = (**`NONE 0`**, `ODD 1`, `EVEN 2`, `MARK 3`, `SPACE 4`)}.<br>At power-on, the SIM960 defaults to PARI `NONE`. |

### 3.4.4 Status commands

The Status commands query and configure registers associated with status reporting of the SIM960.

| | |
|---|---|
| *CLS | Clear Status<br>*CLS immediately clears the ESR, CESR, and the SIM960 status registers. |
| *STB? [*i*] | Status Byte<br>Reads the Status Byte register [bit *i*].<br>Execution of the *STB? query (without the optional bit *i*) always causes the −STATUS signal to be deasserted. Note that *STB? *i* will *not* clear −STATUS, even if bit *i* is the only bit presently causing the −STATUS signal. |
| *SRE(?) [*i*,] {*j*} | Service Request Enable<br>Set (query) the Service Request Enable register [bit *i*] {to *j*}. |

| | |
|---|---|
| *ESR? [*i*] | Standard Event Status<br>Reads the Standard Event Status Register [bit *i*].<br>Upon executing *ESR?, the returned bit(s) of the ESR register are cleared. |
| *ESE(?) [*i*,] {*j*} | Standard Event Status Enable<br>Set (query) the Standard Event Status Enable Register [bit *i*] {to *j*}. |
| CESR? [*i*] | Comm Error Status<br>Query Comm Error Status Register [for bit *i*].<br>Upon executing a CESR? query, the returned bit(s) of the CESR register are cleared. |
| CESE(?) [*i*,]{*j*} | Comm Error Status Enable<br>Set (query) Comm Error Status Enable Register [bit *i*] {to *j*}. |
| INCR? [*i*] | Instrument condition register<br>Query the instrument condition register [bit *i*].<br>The values of the bits in the instrument condition register are determined by the current (real-time) condition of the events defined in the instrument status register (see Section 3.5.8).<br>Reading the instrument condition register does not affect the register. |
| INSR? [*i*] | Instrument status register<br>Query the instrument status register [bit *i*]. |
| INSE(?) [*i*], {*j*} | Instrument status enable register<br>Set (query) the instrument status enable register [bit *i*] {to *j*}. |
| ADSR? [*i*] | A-to-D status register<br>Query the analog to digital status register [bit *i*].<br>When new data become available from the A-to-D converter, the A-to-D status register bit corresponding to the channel of the new data is set (see Section 3.5.10). |

ADSE(?) [*i*], {*j*}

A-to-D status enable register

Set (query) the analog to digital converter status enable register [bit *i*] {to *j*}.

PSTA(?) {*z*}

Pulse −STATUS Mode

Set (query) the Pulse −STATUS Mode {to *z*=(**OFF 0**, ON 1)}.

When PSTA ON is set, any new service request will only *pulse* the −STATUS signal low (for a minimum of 1 $\mu$s). The default behavior is to latch −STATUS low until a *STB? query is received.

At power-on, PSTA is set to OFF.

### 3.4.5 Interface commands

The Interface commands provide control over the interface between the SIM960 and the host computer.

*RST

Reset

Reset the SIM960 to its default configuration. The effect of this command is equivalent to the following sequence of commands:

- DISX ON
- DISP PRP
- SHFT OFF
- GAIN 1.0
- APOL POS
- INTG 1.0
- DERV 1.0E-6
- OFST 0.0
- RATE 1.0
- PCTL ON
- ICTL OFF
- DCTL OFF
- OCTL OFF
- RAMP OFF
- SETP 0.0 (must not precede RAMP OFF)

- MOUT `0.0`
- ULIM `+10.0`
- LLIM `-10.0`
- INPT `EXT`
- AMAN `PID`
- TOKN `OFF`
- SOUT

The baud rate of the SIM960 is unaffected by *RST. The entire status model is also unaffected by *RST.

---

CONS(?) {*z*}

Console Mode

Set (query) the Console mode {to *z*=(**OFF 0**, ON 1)}.

CONS causes each character received at the Input Buffer to be copied to the Output Queue.

At power-on, CONS is set to `OFF`.

---

*IDN?

Identify

Read the device identification string.

The identification string is formatted as:
`Stanford_Research_Systems,SIM960,s/n******,ver#.#`
where `SIM960` is the model number, `******` is the 6-digit serial number, and `#.#` is the firmware revision level.

---

*TST?

Self Test

There is no internal self-test in the SIM960, so this query always returns `0`.

---

*OPC(?)

Operation Complete

Operation Complete. Sets the OPC flag in the ESR register.

The query form *OPC? writes a `1` in the Output Queue when complete, but does not affect the ESR register.

---

WAIT *i*

Wait

Wait *i* milliseconds before processing more commands from the host.

LEXE?

Execution Error

Query the last execution error code. Valid codes are:

| Value | Definition |
|---|---|
| 0 | No execution error since last LEXE? |
| 1 | Illegal value |
| 2 | Wrong token |
| 3 | Invalid bit |
| 16 | Invalid parameter |
| 17 | Missing parameter |
| 18 | No change |
| 20 | Ramp in progress |
| 21 | Limits conflict |

LCME?

Command Error

Query the last command error code. Valid codes are:

| Value | Definition |
|---|---|
| 0 | No execution error since last LCME? |
| 1 | Illegal command |
| 2 | Undefined command |
| 3 | Illegal query |
| 4 | Illegal set |
| 5 | Missing parameter(s) |
| 6 | Extra parameter(s) |
| 7 | Null parameter(s) |
| 8 | Parameter buffer overflow |
| 9 | Bad floating-point |
| 10 | Bad integer |
| 11 | Bad integer token |
| 12 | Bad token value |
| 13 | Bad hex block |
| 14 | Unknown token |

LBTN?

Last Button

Query the last button that was pressed. The values returned are:

| Value | Button |
|---|---|
| 0 | no button pressed since last LBTN? |
| 1 | [Setpoint] |
| 2 | [Output] |
| 3 | [Ramp Start/Stop] |
| 4 | [Shift] |
| 5 | [Select] |
| 6 | [On/Off] |
| 7 | [▲]/ [◄] |
| 8 | [▼]/ [►] |

TOKN(?) {*z*}

Token Mode

Set (query) the Token Query mode {to *z*=(**OFF 0**, ON 1)}.

If TOKN ON is set, then queries to the SIM960 that return tokens will return the text keyword; otherwise they return the decimal integer value.

Thus, the only possible responses to the TOKN? query are ON and 0.

At power-on, TOKN is set to OFF.

TERM(?) {*z*}

Response Termination

Set (query) the ⟨term⟩ sequence {to *z*=(NONE 0, CR 1, LF 2, **CRLF 3**, LFCR 4)}.

The ⟨term⟩ sequence is appended to all query responses sent by the module, and is constructed of ASCII character(s) 13 (carriage return) and 10 (line feed). The token mnemonic gives the sequence of characters.

At power-on, the default is TERM CRLF.

## 3.5 Status Model

The SIM960 status registers follow the hierarchical IEEE–488.2 format. A block diagram of the status register array is given in Figure 3.1.

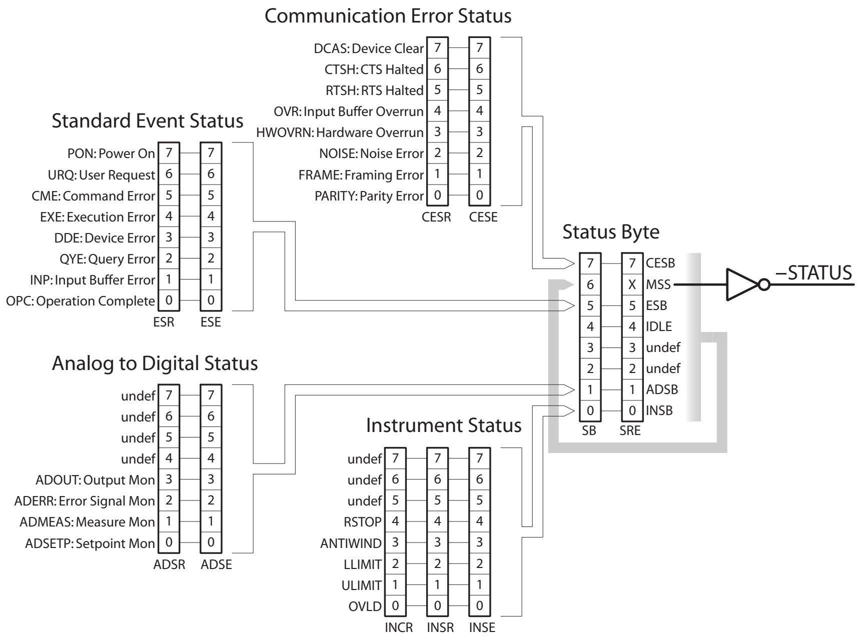

Figure 3.1: Status Register Model for the SIM960 Analog PID Controller.

There are three categories of registers in the SIM960 status model:

Condition Registers : These read-only registers correspond to the real-time condition of some underlying physical property being monitored. Queries return the latest value of the property, and have no other effect. Condition register names end with CR.

Event Registers : These read-only registers record the occurrence of defined events. If the event occurs, the corresponding bit is set to 1. Upon querying an event register, any set bits within it are cleared These are sometimes known as “sticky bits,” since once set, a bit can only be cleared by reading its value. Event register names end with SR.

Enable Registers : These read/write registers define a bitwise mask for their corresponding event register. If any bit position is set in an event register while the same bit position is also set in the enable register, then the corresponding summary bit message is set. Enable register names end with SE.

### 3.5.1 Status Byte (SB)

The Status Byte is the top-level summary of the SIM960 status model. When masked by the Service Request Enable register, a bit set in the Status Byte causes the −STATUS signal to be asserted on the rear-panel SIM interface connector.

| Weight | Bit | Flag |
|---|---|---|
| 1 | 0 | INSB |
| 2 | 1 | ADSB |
| 4 | 2 | undef (0) |
| 8 | 3 | undef (0) |
| 16 | 4 | IDLE |
| 32 | 5 | ESB |
| 64 | 6 | MSS |
| 128 | 7 | CESB |

| | |
|---|---|
| INSB : | Instrument Status Summary Bit. Indicates whether one or more of the enabled flags in the Instrument Status Register has become true. |
| ADSB : | Analog to Digital Status Bit. Indicates whether one or more of the enabled flags in the Analog to Digital Status Register has become true. |
| IDLE : | Indicates that the Input Buffer is empty and the command parser is idle. Can be used to help synchronize SIM960 query responses. |
| ESB : | Event Status Bit. Indicates whether one or more of the enabled events in the Standard Event Status Register is true. |
| MSS : | Master Summary Status. Indicates whether one or more of the enabled status messages in the Status Byte register is true. |
| CESB : | Communication Error Summary Bit. Indicates whether one or more of the enabled flags in the Communication Error Status Register has become true. |

### 3.5.2 Service Request Enable (SRE)

Each bit in the SRE corresponds one-to-one with a bit in the SB register, and acts as a bitwise AND of the SB flags to generate MSS/RQS. Bit 6 of the SRE is undefined—setting it has no effect, and reading it always returns 0. This register is set and queried with the *SRE(?) command.

This register is cleared at power-on.

### 3.5.3 Standard Event Status (ESR)

The Standard Event Status register consists of 8 event flags. These event flags are all "sticky bits" that are set by the corresponding event, and cleared only by reading or with the *CLS command. Reading a single bit (with the *ESR? *i* query) clears only bit *i*.

| Weight | Bit | Flag |
|---|---|---|
| 1 | 0 | OPC |
| 2 | 1 | INP |
| 4 | 2 | QYE |
| 8 | 3 | DDE |
| 16 | 4 | EXE |
| 32 | 5 | CME |
| 64 | 6 | URQ |
| 128 | 7 | PON |

| | |
|---|---|
| OPC : | Operation Complete. Set by the *OPC command. |
| INP : | Input Buffer Error. Indicates data has been discarded from the Input Buffer. |
| QYE : | Query Error. Indicates data in the Output Queue has been lost. |
| DDE : | Device Dependent Error. Undefined for SIM960. |
| EXE : | Execution Error. Indicates an error in a command that was successfully parsed. Out-of-range parameters are an example. The error code can be queried with LEXE?. |
| CME : | Command Error. Indicates a parser-detected error. The error code can be queried with LCME?. |
| URQ : | User Request. Indicates a front-panel button was pressed. |
| PON : | Power On. Indicates that an off-to-on transition has occurred |

### 3.5.4 Standard Event Status Enable (ESE)

The ESE acts as a bitwise AND with the ESR register to produce the single bit ESB message in the Status Byte Register (SB). It can be set and queried with the *ESE(?) command.

This register is cleared at power-on.

### 3.5.5 Communication Error Status (CESR)

The Communication Error Status register consists of 8 event flags; each of which is set by the corresponding event, and cleared only by reading or with the *CLS command. Reading a single bit (with the CESR? *i* query) clears only bit *i*.

| Weight | Bit | Flag |
|---|---|---|
| 1 | 0 | PARITY |
| 2 | 1 | FRAME |
| 4 | 2 | NOISE |
| 8 | 3 | HWOVRN |
| 16 | 4 | OVR |
| 32 | 5 | RTSH |
| 64 | 6 | CTSH |
| 128 | 7 | DCAS |

| | |
|---|---|
| PARITY : | Parity Error. Set by serial parity mismatch on incoming data byte. |
| FRAME : | Framing Error. Set when an incoming serial data byte is missing the STOP bit. |
| NOISE : | Noise Error. Set when an incoming serial data byte does not present a steady logic level during each asynchronous bit-period window. |
| HWOVRN : | Hardware Overrun. Set when an incoming serial data byte is lost due to internal processor latency. Causes the Input Buffer to be flushed, and resets the command parser. |
| OVR : | Input Buffer Overrun. Set when the Input Buffer is overrun by incoming data. Causes the Input Buffer to be flushed, and resets the command parser. |
| RTSH : | Undefined for the SIM960. Command Error. Indicates a parser-detected error. |
| CTSH : | Undefined for the SIM960. |
| DCAS : | Device Clear. Indicates the SIM960 received the Device Clear signal (an RS-232 ⟨break⟩). Clears the Input Buffer and Output Queue, and resets the command parser. |

### 3.5.6 Communication Error Status Enable (CESE)

The CESE acts as a bitwise AND with the CESR register to produce the single bit CESB message in the Status Byte Register (SB). It can be set and queried with the CESE(?) command.

This register is cleared at power-on.

### 3.5.7 Instrument Status (INCR)

The Instrument Condition Register consists of 5 single-bit monitors of condition events within the SIM960. Bits in the INCR reflect the real-time values of their corresponding signals. Reading the entire register, or individual bits within it, does not affect the value of INCR.

| Weight | Bit | Flag |
|---|---|---|
| 1 | 0 | OVLD |
| 2 | 1 | ULIMIT |
| 4 | 2 | LLIMIT |
| 8 | 3 | ANTIWIND |
| 16 | 4 | RSTOP |
| 32 | 5 | undef (0) |
| 64 | 6 | undef (0) |
| 128 | 7 | undef (0) |

| | |
|---|---|
| OVLD : | Amplifier Overload. Set to indicate an overload (either differential *or* common-mode) is presently occurring in the front-end amplifier. |
| ULIMIT : | Upper Limit Reached. Set to indicate the output signal is presently saturated into the programmable upper-limit voltage. |
| LLIMIT : | Lower Limit Reached. Set to indicate the output signal is presently saturated into the programmable lower-limit voltage. |
| ANTIWIND : | Anti-windup Active. Set to indicate the anti-windup circuit is actively inhibiting integration of the error signal. |
| RSTOP : | Ramp Stopped. Set to indicate that no internal setpoint ramp is in progress; cleared to indicate ramping is presently underway. |

### 3.5.8 Instrument Status (INSR)

The Instrument Status Register consists of (latching) event flags that correspond one-to-one with the bits of the INCR (see above). Upon the transition $0 \rightarrow 1$ of any bit within the INCR, the corresponding bit in the INSR becomes set.

Bits in the INSR are unaffected by the $1 \rightarrow 0$ transitions in the INCR, and are cleared only by reading or with the *CLS command. Reading a single bit (with the INSR? *i* query) clears only bit *i*.

### 3.5.9 Analog to Digital Status Enable (INSE)

The INSE acts as a bitwise AND with the INSR register to produce the single bit INSB message in the Status Byte Register (SB). It can be set and queried with the INSE(?) command.

This register is cleared at power-on.

### 3.5.10 Analog to Digital Status (ADSR)

The Analog to Digital Status Register consists of 4 event flags; each of which is set by a corresponding conversion completion for one of the 4 monitored analog signals. Bits in the ADSR are cleared only by reading or with the *CLS command. Reading a single bit (with the ADSR? *i* query) clears only bit *i*.

| Weight | Bit | Flag |
|---|---|---|
| 1 | 0 | ADSETP |
| 2 | 1 | ADMEAS |
| 4 | 2 | ADERR |
| 8 | 3 | ADOUT |
| 16 | 4 | undef (0) |
| 32 | 5 | undef (0) |
| 64 | 6 | undef (0) |
| 128 | 7 | undef (0) |

ADSETP : Setpoint Monitor Conversion Complete. Indicates a new conversion result is available for SMON?.

ADMEAS : Measure Monitor Conversion Complete. Indicates a new conversion result is available for MMON?.

ADERR : Amplified Error Monitor Conversion Complete. Indicates a new conversion result is available for EMON?.

ADOUT : Output Monitor Conversion Complete. Indicates a new conversion result is available for OMON?.

While reading this register (with the ADSR? query) will clear any Trip*n* bit(s) that are set, it will *not* reset the overvoltage protection circuit. To do that, the user must issue the TRIP command. As long as a channel remains tripped off, the Trip*n* bit will continuously be reasserted.

### 3.5.11 Analog to Digital Status Enable (ADSE)

The ADSE acts as a bitwise AND with the ADSR register to produce the single bit ADSB message in the Status Byte Register (SB). It can be set and queried with the ADSE(?) command.

This register is cleared at power-on.