SDI-12
A Serial-Digital Interface Standard
for
Microprocessor-Based Sensors

Version 1.4

August 10, 2016

Prepared By

SDI-12 Support Group
(Technical Committee)
165 East 500 South
River Heights, Utah
435-752-4200
435-752-1691 (FAX)
http://www.sdi-12.org

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

SDI-12
A SERIAL-DIGITAL INTERFACE STANDARD
FOR MICROPROCESSOR-BASED SENSORS
SDI-12 Version 1.4

1.0 INTRODUCTION

This document describes Version 1.4 of the SDI-12 standard. Version 1.4 is an upgrade from Version 1.3, dated January 28, 2016. The purpose of this document is to describe SDI-12 in detail and to provide examples of all SDI-12 commands and responses. (See appendix D for a list of upgrades made since Version 1.0.)

SDI-12 is a standard for interfacing data recorders with microprocessor-based sensors. SDI-12 stands for serial/digital interface at 1200 baud. This document describes the electrical interface, the communications protocol, and the timing requirements for SDI-12 data recorders and SDI-12 sensors.

SDI-12 is intended for applications with the following requirements:

- Battery powered operation with minimal current drain
- Low system cost
- Use of a single data recorder with multiple sensors on one cable (see section 3.0 for details)

2.0 ADVANTAGES OF SDI-12

A serial-digital interface is a logical choice for interfacing microprocessor-based sensors with a data recorder. This has advantages for sensors and data recorders.

- Unique and complex self-calibration algorithms can be done in microprocessor-based sensors.
- Sensors can be interchanged without reprogramming the data recorder with calibration or other information.
- Power is supplied to sensors through the interface.
- Hybrid circuit and surface mount technologies make it practical to include the power supply regulator, a microprocessor, and other needed circuitry in small sensor packages.

- Sensors can use low cost EEPROMs (electrically erasable programmable read only memory) for calibration coefficients and other information instead of internal trimming operations.

- The use of a standard serial interface eliminates significant complexity in the design of data recorders.

- Data recorders can be designed and produced independently of future sensor development.

- SDI-12 data recorders interface with a variety of sensors.

- SDI-12 sensors interface with a variety of data recorders.

- Personnel trained in SDI-12 will have skills to work with a variety of SDI-12 data recorders and SDI-12 sensors.

- SDI-12 sensors with the most recent version of SDI-12 will work with data recorders using earlier versions of SDI-12 and vice versa.

3.0 SDI-12 ELECTRICAL INTERFACE

The SDI-12 electrical interface uses the SDI-12 bus to transmit serial data between SDI-12 data recorders and sensors. The SDI-12 bus is the cable that connects multiple SDI-12 devices. This is a cable with three conductors:

1) a serial data line
2) a ground line
3) a 12-volt line

In the following specifications, all values not indicating specific limits, have an allowable tolerance of ±10% of the value.

Figure 1 shows the SDI-12 bus connecting one data recorder with two sensors. The SDI-12 bus is capable of having at least 10 sensors connected to it, each with 200 feet of cable. With fewer sensors, longer cable lengths are possible.

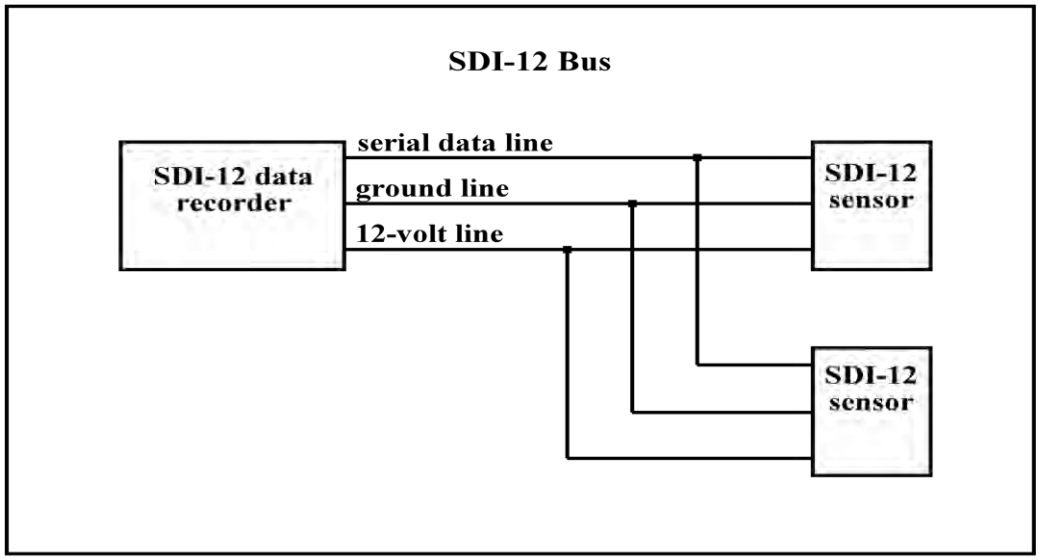


Figure 1. The SDI-12 Bus

3.1 Serial Data Line

The data line is a bidirectional, three-state, data transfer line. Table 1 shows the logic and voltage levels for the transmission of serial data for the SDI-12 standard. The data line uses negative logic.

Condition	Binary state	Voltage range
marking	1	-0.5 to 1.0 volts
spacing	0	3.5 to 5.5 volts
transition	undefined	1.0 to 3.5 volts

Table 1. Logic and voltage levels for serial data

3.1.1 Voltage Transitions

During normal operation, the data line voltage slew rate must not be greater than 1.5 volts per microsecond.

3.1.2 Impedance

When an SDI-12 device has its transmitter on, its direct current (DC) source resistance must be greater than 1000 ohms and less than 2000 ohms. Due to this impedance, the maximum cable length depends on the capacitance of all cables connected to the data line. When any SDI-12 device's transmitter is off, including during a low-power standby mode, the DC resistance to ground must be within 160K to 360K ohms. If an SDI-12 sensor does not use the 12-volt line for power, its data line resistance to ground while powered down must be within 160K to 360K ohms. Figure 2 shows an equivalent circuit.

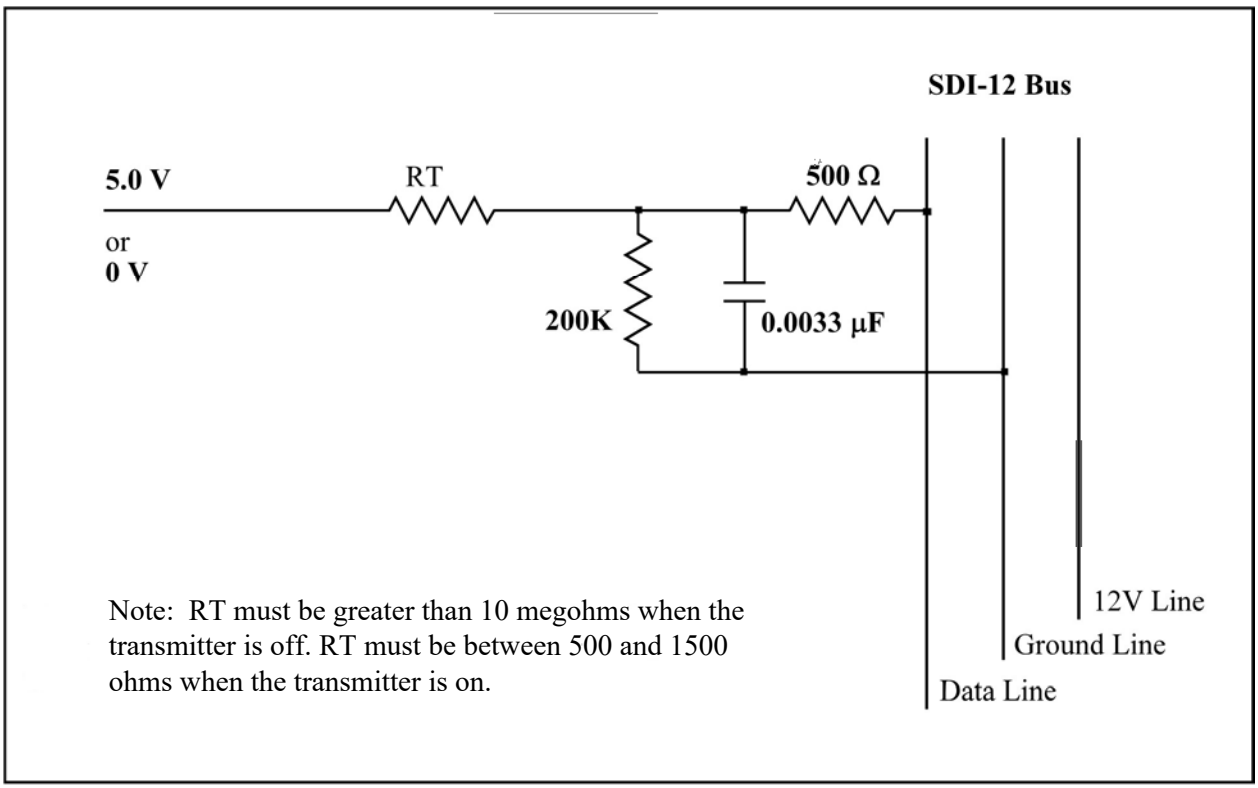


Figure 2. Equivalent circuit

3.2 Ground Line

The ground line must be connected to the circuit ground and the earth ground at the data recorder. The sensor circuit ground also must be connected to the ground line, but not normally to its own earth ground. If it is necessary to connect the sensor circuitry to earth ground, a heavy (12 AWG or larger) ground wire should be connected between the sensor earth ground and the data recorder earth ground for lightning protection.

The ground conductor should be large enough to keep the voltage drop between the data recorder and all sensors less than 0.5 volts during the maximum combined sensor current drain.

3.2.1 Transient Protection

Transient protection is recommended on the SDI-12 bus. See appendix A for a suggested SDI-12 transient protection method.

3.3 12 Volt-Line

The data recorder (or the external power supply) provides between 9.6 volts and 16 volts to the 12-volt line, with respect to ground, as measured under a maximum sensor load of 0.5 amperes. SDI-12 does not require the data recorder to be the source of power to the 12-volt line.

For sensors connected to the 12-volt line that exhibit an inductive load, a series diode is recommended. SDI-12 does not require voltage limiting for transient protection in the sensor. Transient protection is, however, recommended. See appendix A for suggested circuits for transient protection.

3.4 Connectors

A connector type for SDI-12 is not specified.

4.0 SDI-12 COMMUNICATIONS PROTOCOL

SDI-12 data recorders and sensors communicate by an exchange of ASCII characters on the data line. The data recorder sends a break to wake up the sensors on the data line. A break is continuous spacing on the data line for at least 12 milliseconds. The data recorder then sends a command. The sensor, in turn, returns the appropriate response. Each command is for a specific sensor. The first character of each command is a unique sensor address that specifies with which sensor the recorder wants to communicate. Other sensors on the SDI-12 bus ignore the command and return to low-power standby mode. When a data recorder tells a sensor to start its measurement procedure, the recorder does not communicate with any other sensor until the data collection from the first sensor is complete. (During a concurrent measurement command, however, a data recorder can communicate with other sensors while one or more sensors are taking measurements. See section 4.4.7.)

A typical recorder/sensor measurement sequence proceeds as follows:

Step 1. The data recorder wakes all sensors on the SDI-12 bus with a break.

Step 2. The recorder transmits a command to a specific, addressed sensor, instructing it to make a measurement.

Step 3. The addressed sensor responds within 15.0 milliseconds returning the maximum time until the measurement data will be ready and the number of data values it will return.

Step 4. If the measurement is immediately available, the recorder transmits a command to the sensor instructing it to return the measurement(s). If the measurement is not ready, the data recorder waits for the sensor to send a request to the recorder, which indicates that the data are ready. The recorder then transmits a command to get the data.

Step 5. The sensor responds, returning one or more measurements.

4.1 Baud Rate and Byte Frame Format

The baud rate for SDI-12 is 1200. Table 2 shows the byte frame format for SDI-12.

1 start bit
7 data bits, least significant bit transmitted first
1 parity bit, even parity
1 stop bit

Table 2. SDI-12 byte frame format

4.2 Allowable Characters

All characters transmitted on the SDI-12 bus must be printable ASCII characters. Table 3 shows the printable characters.

space, 32 decimal
through
~, 126 decimal

Table 3. Printable characters

There are three exceptions:

1) all responses from an SDI-12 sensor end with a carriage return (0D hex, 13 decimal) and a line feed (0A hex, 10 decimal) character, shown as <CR><LF> in this document;

2) in some cases the second and third character of a CRC code may not be printable ASCII characters;

3) the contents of data packets returned by the High Volume Binary command.

4.3 Device Addresses

The first character of every command must be a sensor address. Likewise, the first character of a response is also the address character. This lets an SDI-12 recorder verify that the

response has come from the correct sensor. (An address is a single character used to indicate which sensor is to respond to the command.) Table 4 shows the address codes.

ASCII address (a single character)	Decimal	Hex	Description
"0" (zero)	48	30	Default address, all sensors are initially set to "0" (zero) by the manufacturer for use in single sensor systems
"1" to "9"	49 to 57	31 to 39	Addresses for additional sensors on the SDI-12 bus

Table 4. Sensor address codes

ASCII '0' through ASCII '9' are the standard addresses which all sensors and data recorders must support. Should there be a need for more than 10 sensors, use an address in the range ASCII 'A' through ASCII 'Z' (decimal 65 through 90) and ASCII 'a' through ASCII 'z' (decimal 97 through 122).

4.4 SDI-12 Commands and Responses

Table 5 lists each basic SDI-12 command, its format, and the format of each response to a command. All SDI-12 Version 1.4 sensors and data recorders must support all commands in this table. None of the basic commands should affect the sensor's calibration. In addition, sensors may support extended commands as described in section 4.4.13. The terms in this table (a,ll,cccccccc,mmmmmm,vvv,xxx,<values>, etc.) are described in sections 4.4.1 to 4.4.12.

Name	Command	Response
Break	Continuous spacing for at least 12 milliseconds	None
Acknowledge Active	a!	a<CR><LF>
Send Identification	aI!	allccccccccmmmmmmvvvxxx...xx<CR><LF>
Change Address	aAb!	b<CR><LF> (support for this command is required only if the sensor supports software changeable addresses)
Address Query	?!	a<CR><LF>
Start Measurement*	aM!	atttn<CR><LF>
Start Measurement and Request CRC*	aMC!	atttn<CR><LF>
Send Data	aD0!	a<values><CR><LF> or a<values><CRC><CR><LF>
	.	a<values><CR><LF> or a<values><CRC><CR><LF>
	.	a<values><CR><LF> or a<values><CRC><CR><LF>
	.	a<values><CR><LF> or a<values><CRC><CR><LF>
	aD9!	a<values><CR><LF> or a<values><CRC><CR><LF>
Additional Measurements*	aM1!	atttn<CR><LF>
	.	atttn<CR><LF>
	.	atttn<CR><LF>
	.	atttn<CR><LF>
	aM9!	atttn<CR><LF>
Additional Measurements and Request CRC*	aMC1! ... aMC9!	atttn<CR><LF>
Start Verification*	aV!	atttn<CR><LF>
Start Concurrent Measurement	aC!	atttnn<CR><LF>
Start Concurrent Measurement and Request CRC	aCC!	atttnn<CR><LF>
Additional Concurrent Measurements	aC1!	atttnn<CR><LF>
	.	atttnn<CR><LF>
	.	atttnn<CR><LF>
	.	atttnn<CR><LF>
	aC9!	atttnn<CR><LF>
Additional Concurrent Measurements and Request CRC	aCC1! ... aCC9!	atttnn<CR><LF>
Continuous Measurements	aR0! ... aR9!	a<values><CR><LF> (formatted like the D commands)
Continuous Measurements and Request CRC	aRC0! ... aRC9!	a<values><CRC><CR><LF> (formatted like the D commands)

*This command may result in a service request. See section 4.4.6.

Table 5. The SDI-12 basic command/response set

The first character of all commands and responses is always a device address. The last character of a command is the "!" character. The “!” can only be in a command as the command terminator. The last two bytes of a response are a carriage return and line feed (<CR><LF>).

The maximum number of characters that can be returned in the <values> part of the response to a D command is either 35 or 75. If the D command is issued to retrieve data in response to a concurrent measurement command, or in response to a high volume ASCII measurement command, the maximum is 75. The maximum is also 75 in response to a continuous measurement command. Otherwise, the maximum is 35.

4.4.1 Acknowledge Active Command (a!)

This command is used to ensure that a sensor is responding to a data recorder or another SDI-12 device. It asks a sensor to acknowledge its presence on the SDI-12 bus. Table 6 shows the acknowledge active command.

Command	Response
a!	a<CR><LF>
a - the sensor address	a - the sensor address
! - terminates the command	<CR><LF> - terminates the response

Table 6. The acknowledge active command (a!)

4.4.1.1 Examples of the Acknowledge Active Command (a!)

0!0<CR><LF>
1!1<CR><LF>

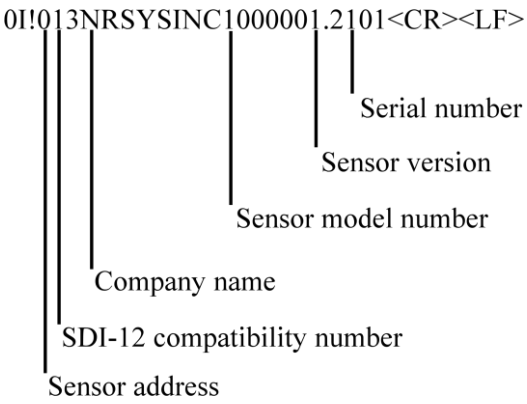
4.4.2 Send Identification Command (aI!)

This command is used to query sensors for their SDI-12 compatibility level, model number, and firmware version number. Table 7 shows the send identification command.

Command	Response
aI!	allccccccccmmmmmmvvvxxx . . . xxx<CR><LF>
a - the sensor address	a - the sensor address
aI!	allccccccccmmmmmmvvvxxx . . . xxx<CR><LF>
a - the sensor address	a - the sensor address
I - the send identification command	ll - the SDI-12 version number, indicating SDI-12 version compatibility; for example, version 1.3 is encoded as 13
	xxx . . . xx - an optional field, up to 13 characters, used for a serial number or other specific sensor information that is not relevant for operation of the data recorder
	<CR><LF> - terminates the response

Table 7. The send identification command (aI!)

4.4.2.1 Example of the Send Identification Command (AI!)



4.4.3 Address Query Command (?!)

When a question mark (?) is used as the address character with the acknowledge active command (a!), the sensor will respond as if it is being addressed on the SDI-12 bus. For example, if a sensor detects ?!, it will respond with a<CR><LF>, no matter what its address may be. This will allow a user to determine the address of a sensor. The user should understand that if more than one sensor is connected to the bus, they will all respond, causing a bus contention. A sensor is required to support the wild card address character (?) with the acknowledge active command (?!). The sensor may also support the wild card address with all other commands; this is not, however, a requirement.

4.4.4 Change Address Command (aAb!)

This command changes the address of a sensor. If the sensor supports software changeable addresses, it must support the change address command. Table 8 shows this command.

After this command has been issued and responded to, the sensor is not required to respond to another command for one second. This gives the sensor time to write the new address to non-volatile memory.

Command	Response
aAb!	b<CR><LF>
a - the sensor address	b - the address of the sensor (will equal the new address or the original address if the sensor is unable to change the address)
A - the change address command b - the address to change to ! - terminates the command	<CR><LF> - terminates the response

Table 8. The change address command (aAb!)

4.4.5 Start Measurement Command (aM!)

This command tells the sensor to take a measurement. The sensor does not, however, return the measurement to the data recorder after this command. It returns the time until one or more measurements will be ready and the number of measurements that it will make. The send data (D0!) command must be issued to get the measurement(s). Table 9 shows the start measurement command.

Command	Response
aM!	atttn<CR><LF>
a - the sensor address	a - the sensor address
M - the start measurement	ttt - the specified time, in seconds, until the sensor will have the measurement(s) ready
! - terminates the command	n - the number of measurement values the sensor will make and return in one or more subsequent D commands; n is a single digit integer with a valid range of 1 to 9

Table 9. The start measurement command (aM!)

If the sensor returns 000 (ttt), the measurement is immediately available for transfer to the data recorder. The data recorder should issue the D0 command to get the data.

If ttt is not equal to zero (000), the data recorder must wait for the specified time to elapse. The ttt time period begins upon completion of the transmission of the line feed character. Then it wakes the sensor with a break and issues the D0 command. If, however, the sensor has the measurement ready before ttt seconds elapse, it will send a service request to the recorder. This tells the recorder to stop marking time and issue the D0 command.

When a data recorder issues an M command, it must complete the command/response sequence with the sensor before it sends any command to any other sensor. For example, suppose that following the issuance of the M command, the sensor responds

0M!00101<CR><LF>

This response shows that one data value will be ready in 10 seconds. The data recorder must wait for either of two events to occur before issuing a D0 command:

1) receipt of a service request (a<CR><LF>) from the sensor
2) the specified time to elapse (10 seconds)

The recorder then issues the D0 command to get the data. After the M command and before the sensor issues the service request, the data recorder will not communicate with any other sensor. **The sensor must not drive the data line until it sends a service request. The recorder must not drive the data line while waiting for the service request.**

A sensor should return a ttt value greater than the time it takes to make a measurement, to allow for timing tolerances and for the service request. (See section 4.4.6.) The data recorder may wait for the entire ttt time. Therefore, the tolerance above the measurement, added to ttt, should be minimal.

After a sensor finishes its measurement procedure, the sensor must retain the measured data in memory until it receives another M or V command, or another command that returns data via the D0 command.

See the send data command, aD0!, aD1! . . . aD9!, for examples of this command. Because the start measurement command is closely related to the send data command, examples for both commands are shown on page 17.

4.4.5.1 Aborting a Measurement

If a sensor detects a break after it receives an M command, but before it issues a service request, it must abort its measurement procedure. The sensor address, followed by <CR><LF>, (or followed by <CRC><CR><LF> if a CRC was requested) should be returned in response to subsequent D commands. This requirement provides a way for a data recorder to abort a

measurement so it can communicate with another sensor, or this sensor, on the SDI-12 bus.

4.4.6 Service Request (a<CR><LF>)

A service request is a response from a sensor. It is not a command. It is sent, after an M command, to tell the data recorder that the sensor has finished its measurement(s) and the data are ready. A service request is issued by the sensor after an M, MC, or V command, when it has finished its measurement. The entire service request must be returned before ttt seconds (see Start Measurement Command, page 11) have elapsed. The time (ttt) is the maximum time that a sensor will take before it has data available.

If a data recorder does not detect a service request, the data recorder must wait for the specified time (ttt), given in response to the M or V command, and then issue the D0 command to get the data. The minimum time before a data recorder can issue the D0 command to get the data, in this case, is ttt seconds.

A sensor is required to issue a service request if it indicates that it will take one second or longer before data are ready, in response to an M or V command. If a sensor says it will take zero seconds before the data are ready, then a sensor must not issue a service request.

4.4.7 Start Concurrent Measurement Command (aC!)

This command, introduced in Version 1.2 of the SDI-12 Specification, tells the sensor to take a concurrent measurement. A concurrent measurement is one which occurs while other SDI-12 sensors on the bus are also taking measurements. The sensor does not, however, return the measurement to the data recorder after this command. It returns the time until all measurements will be ready and the number of measurement that it will make. The send data (D0!) command must be issued to collect the measurements(s). Table 10 shows the start concurrent measurement command.

If the sensor returns 000 (ttt), the measurement is immediately available for transfer to the data recorder. The data recorder should issue the D0 command to get the data.

If ttt is not equal to zero (000), the data recorder must wait for the specified time to elapse before attempting to retrieve the data. The ttt time period begins upon completion of the transmission of the line feed character. During this time the data recorder can collect data from sensors at other addresses. Communicating with other sensors will not abort a concurrent measurement. After the specified time has elapsed, the data recorder wakes the sensor with a break and issues the D0 command. **The sensor will not issue a service request when it has completed the measurement. The sensor must not drive the data line from the time it completes its atttnn<CR><LF> response until it starts responding to the D0 command.**

Command	Response
aC!	atttnn<CR><LF>
a - the sensor address	a - the sensor address
C - the start concurrent measurement command	ttt - the specified time, in seconds, until the sensor will have the measurement(s) ready
! - terminates the command	nn - the number of measurement values the sensor will make and return in response to one or more subsequent D commands

Table 10. The start concurrent measurement command (aC!)

The data recorder should document the number of data values it can store in response to a C command.

After a sensor finishes its measurement procedure, the sensor must retain the measured data in memory until it receives another C, M, or V command, or another command that returns data via the D0 command.

See the send data command, aD0!, aD1! . . . aD9!, section 4.4.8, for the data response requirements. Because the start concurrent measurement command is closely related to the send data command, examples for both commands are in sections 4.4.8.3 and 4.4.8.4.

For a sensor or data logger to claim support of Version 1.2 or higher of the SDI-12 Specification, it must support this command. Claiming zero data values in response to this command (i.e. a response of a00000<CR><LF>) is not a valid support of this command. The command must initiate a true measurement cycle. Sensors or data recorders that conform to a previous version of SDI-12 will most likely not support this command, and will therefore not respond to this command.

4.4.7.1 Aborting a Concurrent Measurement

If a sensor receives a valid command addressed to it while it is in the process of a concurrent measurement, it should abort the measurement procedure. The sensor address followed by <CR><LF> (or <CRC><CR><LF> if a CRC was requested) should be returned in response to subsequent D commands. This requirement provides a way for a data recorder to abort a measurement.

4.4.8 Send Data Command (aD0!, aD1! . . . aD9!)

This command is used to get groups of data from the sensor. D0! is issued after an M, MC, C, CC, V, or HA command. The sensor responds by sending the data. If the expected

number of measurements is not returned in response to a D0! command, the data recorder issues D1!, D2!, etc. until all measurement values are received. (The expected number of measurements is given in the response to an M, C, or V command.) Table 11 shows the send data command.

Command	**Response**
aD0! (aD1! . . . aD9!)	a<values><CR><LF> or a<values><CRC><CR><LF>
a - the sensor address	a - the sensor address
D0 - the send data command, D1 . . . D9 additional send data commands	values (see below)
! - terminates the command	<CR><LF> - terminates the response
	<values> - pd.d p - the polarity sign (+ or -) d - numeric digits before the decimal place . - the decimal point (optional) d - numeric digits after the decimal point the maximum number of digits for a data value is 7, even without a decimal point the minimum number of digits for a data value (excluding the decimal point) is 1 the maximum number of characters in a data value is 9 (the (polarity sign + 7 digits + the decimal point))
	<CRC> - 3 character CRC code, appended if data was requested with the aMC!, aMC1! ... aMC9!, aCC!, or aCC1! ... aCC9! commands (see section 4.4.12)

Table 11. The send data command (aD0!, aD1! . . . aD9!)

If the response to a D command is valid, but no data are returned, the sensor has aborted the measurement. To obtain data the recorder must issue another M, C, or V command.

Notes: in response to certain commands, the data returned after a D command may have a Cyclic Redundancy Check (CRC) appended to it. See section 4.4.12. The high volume ASCII command, HA, extends the range of the send data commands. See section 5.1.

The maximum number of characters that can be returned in the <values> part of the response to a D command is either 35 or 75. If the D command is issued to retrieve data in response to a concurrent command, or a high volume ASCII command, the maximum is 75. Otherwise, the maximum is 35.

4.4.8.1 Continuous Measurements (aR0! ... aR9!)

Sensors that are able to continuously monitor the phenomena to be measured, such as a shaft encoder, do not require a start measurement command (M!, M1! . . . M9!). They can be read directly with the R commands (R0! ... R9!). For example:

if (the sensor is operating in a continuous measurement mode) then
aR0! will get and return the current reading of the sensor

The response to R commands (R0! ... R9!) are formatted like the D commands (D0! ... D9!). The main difference is that the R commands do not need to be preceded with an M command, which tells the sensor to take a measurement. The maximum number of characters that can be returned in the <values> part of the response to an R command is 75.

Each R command is an independent measurement. For example, R5 need not be preceded by R0 ... R4.

If a sensor is unable to take a continuous measurement, then it must return its address followed by a carriage return/line feed (a<CR><LF>) in response to an R command. If a CRC was requested, then the <CR><LF> must be preceded by the CRC. For example: 0AP@<CR><LF>.

4.4.8.2 Example of the aR0! Command

a. One measurement is immediately available after the R0! command:

0R0!0+3.14<CR><LF>

4.4.8.3 Return of Multiple Measurements (Parameters) by a Sensor (D1! . . . D9!)

The commands D1 . . . D9 are used with sensors that return multiple measurements. The purpose of the D commands is for the sensor to return as many measurements as possible in response to each command. The limiting constraint is that the total number of characters that can be returned in the <values> field (see section 4.4.8). If the total number of characters exceeds the maximum length of the <values> field, the sensor fragments the response, sending the first group of measurements in response to D0, the next group in response to D1, and so on. A group can have one or more measurements; data collection always begins with the D0 command. See section 4.4.8.4 for examples.

The sensor must never split individual data values, sending part of a value in response to one D command, and then sending the rest of the characters, for that value, in response to the next D command.

If possible, a sensor should return all measurements in response to the D0 command. This is not, however, a requirement.

4.4.8.4 Examples of the Start Measurement Command (aM!) and the Send Data Commands

a. One measurement is immediately available after the M command:

0M!00001<CR><LF>
0D0!0+3.14<CR><LF>

b. Three measurements will be ready 5 seconds after the M command, and the sensor issues a service request. All 3 measurements are returned in response to the D0 command:

0M!00053<CR><LF>
0<CR><LF>
0D0!0+3.14+2.718+1.414<CR><LF>

c. Nine measurements will be ready 35 seconds after the M command, and the sensor issues a service request. Because the number of characters in all 9 measurements exceeds the limit in the <values> field, a D1 command must be issued to get the second group of measurements:

0M!00359<CR><LF>
0<CR><LF>
0D0!0+1.11+2.22+3.33+4.44+5.55+6.66<CR><LF>
0D1!0+7.77+8.88+9.99<CR><LF>

d. Two measurements will be available in 1 second, and the sensor does not issue a service request. After 1 second, the data recorder sends a break to wake the sensor and issues the D0 command:

0M!00012<CR><LF>
0D0!0+3.14+2.718<CR><LF>

Note: this example shows the proper operation of the data recorder, but the sensor is out of compliance because it did not issue a service request.

e. Three measurements will be ready 5 seconds after the M command, and the sensor issues a service request. Upon receipt of the service request, the data recorder issues D0 to get the data. However, only 1 measurement is returned. The data recorder then issues the D1 command to get the next group of data. In response, the second measurement is returned. Then the recorder issues D2 to get the next, and last, group of data. In the example, each group contains only one measurement.

0M!00053<CR><LF>
0<CR><LF>
0D0!0+3.14<CR><LF>
0D1!0+2.718<CR><LF>
0D2!0+1.414<CR><LF>

Note: This is in compliance with the standard. As many measurements as possible, however, should be returned in response to each D command.

4.4.8.5 Example of the Concurrent Measurement Command (aC!) and the Send Data Command (aD0!)

Two sensors, one returning 12 readings after 45 seconds and the other returning 4 readings after 15 seconds. The measurements are taken concurrently. Fifteen seconds after starting a measurement from sensor 1, the data recorder issues a break followed by the D0 command to sensor address 1. Forty-five seconds (or longer) after starting a measurement from the sensor at address 0, the data recorder sends a break and a D0 command to sensor 0. Note that since a concurrent measurement was requested of sensor 0, it is allowed to return up to 75 characters in its <values> field of the response. An M command only allows 35 characters in its <values> field to ensure compatibility with data recorders prior to version 1.2.

0C!004512<CR><LF>
1C!101504<CR><LF>
1D0!1+1.23+2.34+345+4.4678<CR><LF>
0D0!0+1.234-4.56+12354-0.00045+2.223+145.5+7.7003+4328.8+9+10+11.433+12<CR><LF>

4.4.9 Additional Measurement Commands (aM1! . . . aM9!)

Additional M commands provide a means to request different types of measurements from a sensor or to instruct a sensor to do a calibration or a control function. For example, a sensor could measure pressure and temperature: M tells it to measure pressure and M1 tells it to measure the temperature.

Additional M commands have the same format as the aM! command. **Data collection always begins with the D0 command.** If the sensor does not return the expected number of measurements in response to the D0 command, the recorder should issue aD1, aD2, etc. until the sensor returns all measurements.

To comply with Version 1.2 or higher of SDI-12, sensors must respond to the additional measurement commands (aM1! . . . aM9!) and data recorders must be able to log data from the additional measurement commands. If a sensor has no data defined for an additional measurement command, it should return a0000<CR><LF>, saying that it has zero data values ready. Not responding to the command is not acceptable.

4.4.9.1 Examples of the Additional M Commands (aMn!)

a. A sensor supports the aM1! command:

0M1!00011<CR><LF>
0<CR><LF>
0D0!0+3.14<CR><LF>

b. A sensor takes 9 measurements in response to the M2 command:

0M2!00359<CR><LF>
0<CR><LF>
0D0!0+1.11+2.22+3.33+4.44+5.55+6.66<CR><LF>
0D1!0+7.77+8.88+9.99<CR><LF>

4.4.10 Additional Concurrent Measurement Commands (aC1! . . . aC9!)

Additional C commands provide a means to request different types of measurements from a sensor or to instruct a sensor to do a calibration or a control function. To comply with Version 1.2 or higher of SDI-12, sensors must respond to the additional concurrent measurement commands and data recorders must be able to log data from the additional concurrent measurement commands. If a sensor has no parameters defined for an additional concurrent measurement command, then it should return a00000<CR><LF>, saying that it has zero data values ready. Not responding to the command is not acceptable.

Additional C commands have the same format and constraints as the aC! command. Data collection always begins with the D0 command. If the sensor does not return the expected number of measurements in response to the D0 command, the recorder should issue aD1, aD2, etc., until the sensor returns all measurements.

4.4.11 Start Verification (aV!)

This command tells the sensor to return a verification in response to a subsequent D command. A verification sequence may include ROM signatures, CRC's, RAM test results, or the results of other diagnostics in the sensor. A standard response to the V command is not specified.

The format of this command is the same as the M commands. The format of the response is the same as the D commands.

4.4.11.1 Example of the Start Verification Command (aV!)

0V!00011<CR><LF>
0<CR><LF>
0D0!0+1<CR><LF>

4.4.12 Requesting a Cyclic Redundancy Check (CRC)

To enhance the error detection capability in SDI-12 data collection systems, a variation of the Start Measurement Commands (M!, M1! ... M9!), Start Concurrent Measurement Commands (C!, C1! ... C9!), and Continuous Measurement Commands (aR0! ... aR9!) request that the data be returned with a 16 bit Cyclic Redundancy Check (CRC) appended to it. These commands use the existing command letters with a C appended, namely: aMC!, aMC1! ... aMC9!, aCC!, aCC1! ... aCC9!, and aRC0! ... aRC9!. When these commands are used, the data returned in response to the D commands, or R commands, must have a CRC code appended to it.

The number of measurements returned in response to a CRC command should be the same as the measurement that was made in response to a non-CRC command. In other words, the CRC command causes the same measurements to be taken as the non-CRC command.

To be version 1.3 (or higher) compliant, the sensor must support CRCs.

4.4.12.1 CRC-16 Computation

The computation of the CRC is performed on the data response before parity is added. All operations are assumed to be on 16 bit unsigned integers. The least significant bit is on the right. Numbers preceded by 0x are in hexadecimal. All shifts shift in a zero. The algorithm is:

Initialize the CRC to zero. For each character beginning with the address, up to but not including, the carriage return (<CR>)

```
{
        Set the CRC equal to the exclusive OR of the character and itself
        for count = 1 to 8
        {
                if the least significant bit of the CRC is one
                {
                        right shift the CRC one bit
                        set CRC equal to the exclusive OR of 0xA001 and itself
                }
                else
                {
                        right shift the CRC one bit
                }
        }
}
```

4.4.12.2 Encoding the CRC as ASCII Characters

The 16 bit CRC is encoded as three ASCII characters using the following algorithm:

1st character = 0x40 OR (CRC shifted right 12 bits)
2nd character = 0x40 OR ((CRC shifted right 6 bits) AND 0x3F)
3rd character = 0x40 OR (CRC AND 0x3F)

The three ASCII characters are placed after the data before the <CR><LF>. Parity is applied to all three characters when they are transmitted.

Note: the AND and OR operators are bitwise operators, not logical operators.

4.4.12.3 Examples of the CRC-16 Start Measurement Command (aMC!) and the Send Data Command (aD0!)

a. One measurement is immediately available after the MC command:

0MC!00001<CR><LF>
0D0!0+3.14OqZ<CR><LF>

b. Three measurements will be ready 5 seconds after the MC command, and the sensor issues a service request. All 3 measurements are returned in response to the D0 command:

0MC!00053<CR><LF>
0<CR><LF>
0D0!0+3.14+2.718+1.414Ipz<CR><LF>

c. Nine measurements will be ready 35 seconds after the MC command, and the sensor issues a service request. Because the number of characters in all 9 measurements exceeds the limit for the <values> field, a D1 command must be issued to get the second group of measurements:

0MC!00359<CR><LF>
0<CR><LF>
0D0!0+1.11+2.22+3.33+4.44+5.55+6.66I]q<CR><LF>
0D1!0+7.77+8.88+9.99IvW<CR><LF>

d. Two measurements will be available in one second, and the sensor does not issue a service request. After one second, the data recorder sends a break to wake the sensor and issues the D0 command:

0MC!00012<CR><LF>
0D0!0+3.14+2.718IWO<CR><LF>

Note: this example shows the proper operation of the data recorder, but the sensor is out of compliance because it did not issue a service request.

e. Three measurements will be ready five seconds after the MC command, and the sensor issues a service request. Upon receipt of the service request, the data recorder issues D0 to get the data. Only one measurement, however, is returned. The data recorder then issues the D1 command to get the next group of data. In response, the second measurement is returned. Then the recorder issues D2 to get the next, and last, group of data. In this example, each group contains only one measurement.

0MC!00053<CR><LF>
0<CR><LF>
0D0!0+3.14OqZ<CR><LF>
0D1!0+2.718Gbc<CR><LF>
0D2!0+1.414GtW<CR><LF>

Note: this is in compliance with the standard. As many measurements as possible, however, should be returned in response to each D command.

f. Two sensors, one returning 12 readings after 45 seconds and the other returning 4 readings after 15 seconds. The measurements are taken concurrently. Fifteen seconds after starting a measurement from sensor 1, the data recorder issues a break followed by the D0 command to sensor address one. Forty-five seconds (or longer) after starting a measurement from the sensor at address 0, the data recorder sends a break and a D0 command to sensor 0. Since a concurrent measurement was requested of sensor 0, it is allowed to return up to 75 characters in the <values> field of its response:

0CC!004512<CR><LF>
1CC!101504<CR><LF>

1D0!1+1.23+2.34+345+4.4678KoO<CR><LF>
0D0!0+1.234-4.56+12354-0.00045+2.223+145.5+7.7003+4328.8+9+10+11.433+12Ba]<CR><LF>

4.4.13 Extended Commands

Sensors are required only to respond to the basic SDI-12 command set. Sensors, however, usually require calibration or other setup commands. Extended commands provide the means for such functions. An extended command is a command for a specific make of sensor to tell that sensor to do a specific task. Extended commands are defined and documented by the manufacturer of each sensor.

Extended commands have the following attributes:

- an extended command must be prefixed with an address
- an extended command must be terminated with an exclamation point
- the response must be prefixed with an address
- the response must be terminated with <CR><LF>
- the transparent mode must support basic SDI-12 commands and extended commands

Extended commands should be prefixed with an upper case X, for example, aXNNN!, where X says that an extended command follows and NNN is the extended command. NNN is not limited to three characters. Prefixing extended commands with an upper case X is a recommendation only and is not a requirement. Future versions of SDI-12, however, may require this.

4.4.13.1 Transparent Mode

SDI-12 data recorders must have a mode in which extended commands can be sent to sensors. This is called the transparent mode. The transparent mode has the following characteristics.

- The data recorder buffers a command string received from a computer, terminal, or modem, until the command string is terminated.

- The data recorder wakes the sensor with a break, then it sends the buffered command to the sensor, using the SDI-12 protocol.

- The data recorder receives the response from the sensor and transmits the response to the computer, the terminal, or the modem.

5.0 HIGH VOLUME COMMANDS

The high volume commands, introduced in version 1.4 of the SDI-12 Specification, expand the concurrent measurement commands to allow up to 999 parameters to be returned from a sensor.

5.1 Start High Volume ASCII Measurement

Table 12 shows the high volume ASCII measurement command.

Command Name	Command	Response
High Volume ASCII	aHA!	atttnnn<CR><LF>

Table 12. High volume ASCII measurement

The commands to get high volume data after the aHA! command are: aD0! … aD999!

If after obtaining the data values from aD9! there are still more data values to obtain, continue to send data requests with aD10!...aD99! as needed. If after obtaining the data values from aD99! there are still more data values to obtain, continue to the send data requests with aD100!...aD999! as needed. Leading zeros are not placed after the "D".

The responses to the send data commands follow the same rules as with the Concurrent Measurement send data commands. The maximum number of characters that can be returned in the <values> part of the response is 75 and a three character <CRC> is appended to the data before the <CR><LF>. The CRC must be present.

5.1.1 Example of High Volume ASCII Measurement

In this example there are two sensors, one returning 12 data values after 45 seconds and the other returning 4 data values after 15 seconds. The first measurement uses the high volume ASCII command; the second one uses the concurrent command with a CRC. Fifteen seconds after requesting data from sensor one, the data recorder issues a break followed by the D0 command to sensor address 1. Forty-five seconds (or longer) after starting the measurement from the sensor at address 0, the data recorder sends a break and a D0 command to sensor 0. Since a high volume ASCII measurement was requested of sensor 0, it is allowed to return up to 75 characters in the <values> field of its response, and it does not abort when sensor 1 is addressed:

0HA!0045012<CR><LF>
1CC!101504<CR><LF
1D0!1+1.23+2.34+345+4.4678KoO<CR><LF>
0D0!0+1.234-4.56+12354-0.00045+2.223+145.5+7.7003+4328.8+9+10+11.433+12Ba]<CR><LF>

5.2 Start High Volume Binary Measurement

The high volume binary measurement allows for collection of large volumes of data from a sensor more efficiently than the ASCII transfer methods.

Command Name	Command	Response
High Volume Binary	aHB!	atttnnn<CR><LF>

Table 13. High volume binary measurement

The commands to get high volume binary data after the aHB! command are: aDB0! … aDB999!

If after obtaining the data values from aDB9! there are still more data values to obtain, continue the send data requests with aDB10!...aDB99! as needed. If after obtaining the data values from aDB99! there are still more data values to obtain, continue the send data requests with aDB100!...aDB999! as needed.

Responses to the aDB0! … aDB999! commands are an exception to section 4.1 Table 2 because the byte frame format is 8 data bits, no parity bit. The SDI-12 address is transmitted as the ASCII character, but no parity bit. The remaining fields are encoded as binary numbers. The least significant byte is transferred first for multi-byte binary numbers.

SDI-12 Address	Packet Size	Data Type	Binary Data Payload	CRC
ASCII	16 bit unsigned integer, indicates the size, in bytes, of the binary data payload	8 bit unsigned integer, indicates the data type in the binary data payload	must be <= 1,000 bytes	16 bit CRC value, using the same algorithm as the other measurement commands that request a CRC, but encoded in binary (not converted to 3 byte ASCII)

Table 14. Data packet

If the value of n in aDBn! is invalid due to being higher than necessary to return data values, then the data package must be:

SDI-12 Address	Packet Size	Data Type	Binary Data Payload	CRC
address	0	0	nil	16 bit CRC

Table 15. Empty data packet

All data values in a particular response must be of the same type, but data types can differ between aDBn! commands.

Nil, as shown in Table 15, means that the binary data payload is empty: there are zero data bytes in it. An empty data packet, therefore, has six bytes only:

1 byte: the address;
2 bytes: the packet size (zero);
1 byte: the data type (zero);
2 bytes: the CRC value.

5.2.1 High Volume Binary Data Types

Table 16 shows the high volume binary data types.

Data Type	Range	Size
0	Indicates an invalid request	No data returned
1	-128 to 127	Signed 8-bit integer
2	0 to 255	Unsigned 8-bit integer
3	-32,768 to 32,767	Signed 16-bit integer
4	0 to 65,535	Unsigned 16-bit integer
5	-2,147,483,648 to 2,147,483,647	Signed 32-bit integer
6	0 to 4,294,967,295	Unsigned 32-bit integer
7	-9,223,372,036,854,775,808 to 9,223,372,036,854,775,807	Signed 64-bit integer
8	0 to 18,446,744,073,709,551,615	Unsigned 64-bit integer
9	$\pm 1.18\times10^{-38}$ to $\pm 3.4\times10^{38}$	IEEE 32-bit floating point single precision/binary32
10	$\pm 2.23\times10^{-308}$ to $\pm 1.80\times10^{308}$	IEEE-64 bit floating point double precision/binary64

Table 16. Data types

5.2.2 Example of High Volume Binary Command

In this example a data recorder transmits a high volume binary command and the sensor responds, indicating that four data values will be available after five seconds:

1HB!1005004<CR><LF>

Table 17 shows the data values, returned in binary data packets, that are used in this example. Two of the four data values are returned in a data packet as signed 16-bit integers; the other two data values are returned in a data packet as 32-bit single precision floating point numbers.

Data Type	Size	Data Value	Data Value in Hexadecimal
3	Signed 16-bit integer	-1	0xFFFF
3	Signed 16-bit integer	1	0x0001
9	IEEE-32 bit floating point	3.14	0x4048F5C3
9	IEEE-32 bit floating point	1.0	0x3F800000

Table 17. Data values in the high volume binary command example

Table 18 shows each data packet returned in response to the commands 1DB0!,1DB1!, and 1DB2! as a string of bytes, shown in hexadecimal.

The commands, 1DB0!, 1DB1!, and 1DB2 are transmitted to the sensor as 7 bit ASCII characters with even parity. The SDI-12 address, 0x31 in this example, is returned as an 8 bit ASCII character, without a parity bit.

Table 18 also shows that it was unnecessary to transmit the command 1DB2!, because all four data values were already received. Therefore, the sensor returned an empty data packet.

Command	Data Packet
1DB0!	0x31 0x04 0x00 0x03 0xFF 0xFF 0x01 0x00 0xC2 0xAC
1DB1!	0x31 0x08 0x00 0x09 0xC3 0xF5 0x48 0x40 0x00 0x00 0x80 0x3F 0x3B 0x6E
1DB2!	0x31 0x00 0x00 0x00 0x0E 0xFC

Table 18. Data packet examples

5.3 Concurrency of High Volume Commands

The high volume commands are concurrent commands, as described in section 4.4.7, with section 4.4.7.1 explaining how to abort a concurrent measurement command.

5.4 Compliance with High Volume Commands

To claim compliance with version 1.4, SDI-12 data recorders must support the high volume commands.

No requirement is imposed on SDI-12 sensors to support the high volume commands. The functionality of an SDI-12 sensor dictates the need to support the high volume commands, because there is no advantage in collecting a low number of parameters from a sensor with a high volume command.

6.0 METADATA COMMANDS

The metadata commands, introduced in version 1.4 of the SDI-12 Specification, provide a means to get the response to a command without actually initiating a measurement.

6.1 Identify Measurement Commands

The identify measurement commands are formed by placing the capital letter I into the measurement commands immediately after the address. The response is identical to having issued the command without the capital letter I following the address. The format is atttn<CR><LF>, atttnn<CR><LF>, or atttnnn<CR><LF> depending on the measurement command.

The continuous measurement commands (aR0! … aR9! And aRC0! … aRC9!) provide their data instantly and therefore do not have an identify measurement command.

Command	Response
aIM!	atttn<CR><LF>
aIMC!	atttn<CR><LF>
aIM1!	atttn<CR><LF>
.	atttn<CR><LF>
.	atttn<CR><LF>
.	atttn<CR><LF>
aIM9!	atttn<CR><LF>
aIMC1!	atttn<CR><LF>
.	atttn<CR><LF>
.	atttn<CR><LF>
.	atttn<CR><LF>
aIMC9!	atttn<CR><LF>

Command	Response
aIV!	atttn<CR><LF>
aIC!	atttnn<CR><LF>
aICC!	atttnn<CR><LF>
aIC1!	atttnn<CR><LF>
.	atttnn<CR><LF>
.	atttnn<CR><LF>
.	atttnn<CR><LF>
aIC9!	atttnn<CR><LF>
aICC1!	atttnn<CR><LF>
.	atttnn<CR><LF>
.	atttnn<CR><LF>
.	atttnn<CR><LF>
aICC9!	atttnn<CR><LF>
aIHA!	atttnnn<CR><LF>
aIHB!	atttnnn<CR><LF>

Table 19. The identify measurement commands

6.1.1 Examples of the Identify Measurement Commands

a. One data value will be immediately available after the M command:

8IM!80001<CR><LF>

b. Nine data values will be available 10 seconds after the M command:

8IM!80109<CR><LF>

c. Ninety-nine data values will be available 10 seconds after the C5 command:

8IC5!801099<CR><LF>

6.2 Identify Measurement Parameter Commands

The identify measurement parameter commands provide details about the parameters returned by a particular command. The form of the command is an expansion of the Identify Measurement Commands. An underscore character ("_") plus a three-digit decimal number is placed immediately before the exclamation point ("!"). The decimal number is the data value of interest.

The response is a comma separated value (CSV) string with several fields that provide information about the data value of interest. Two fields are required. Additional fields can be added by the sensor manufacturer. Fields are expected to contain printable ASCII characters other than the comma character (“,”) or the semicolon character (“;”) since they are used to delimit fields.

6.2.1 Field One

The first field contains a concise identification of the parameter, which is the data value of interest. The recommendation is to use a Standard Hydrometeorological Exchange Format (SHEF) code.

SHEF codes are published by the National Oceanic and Atmospheric Administration (NOAA), National Weather Service in the “Standard Hydrometeorological Exchange Format (SHEF) Code Manual.” This document is available on the Internet at:

http://www.nws.noaa.gov/oh/hrl/shef/indexshef.htm

The SHEF codes are listed in “Appendix G, Physical Element Definitions,” in the NOAA document.

If an appropriate SHEF code does not exist for the parameter, or if the sensor manufacturer chooses not to use a SHEF code for the parameter, then field one may contain a concise identification of the parameter as determined by the sensor manufacturer. The recommendation, however, is to use a SHEF code when an appropriate SHEF code does exist.

6.2.2 Field Two

Field two contains the units for the parameter. If the parameter is unit-less, the field must still be present. A single space character is recommended for an empty field to make it easier to read.

6.2.3 Optional Fields

The sensor manufacturer may provide additional information relevant to the parameter by adding additional fields. This may be a more descriptive name than found in field one. For example, it may contain calibration data or dates. If the parameter represents a probe that has a unique serial number, there may be a field that contains that serial number. The only limit on the number of additional fields is that the maximum length of the response, through the terminating semicolon, is 75 characters.

The last field is terminated by a semicolon “;” and then followed by either the <CR><LF> sequence or a three character CRC followed by the <CR><LF> sequence if the measurement command was one that returns a CRC.

If the parameter number, nnn, is invalid for the measurement, then the response shall be: a<CR><LF> or a<CRC><CR><LF> if a CRC was expected.

To poll the Continuous Measurement commands (aR0! … aR9! and aRC0! … aRC9!) to identify their measurement parameters, increment sequentially through the parameters until the response indicates the parameter number, nnn, is invalid for the measurement.

Command	Response
aIM_001!	a,field1,field2;<CR><LF>
.	a,field1,field2;<CR><LF>
.	a,field1,field2;<CR><LF>
.	a,field1,field2;<CR><LF>
aIM_009!	a,field1,field2;<CR><LF>
aIMC_001!	a,field1,field2;<CR><LF>
.	a,field1,field2;<CR><LF>
.	a,field1,field2;<CR><LF>
.	a,field1,field2;<CR><LF>
aIMC_009!	a,field1,field2;<CR><LF>
aIM1_001! … aIM1_009!	a,field1,field2;<CR><LF>
.	a,field1,field2;<CR><LF>
.	a,field1,field2;<CR><LF>
.	a,field1,field2;<CR><LF>
aIM9_001! … aIM9_009!	a,field1,field2;<CR><LF>
aIMC1_001! … aIMC1_009!	a,field1,field2;<CR><LF>
.	a,field1,field2;<CR><LF>
.	a,field1,field2;<CR><LF>
.	a,field1,field2;<CR><LF>
aIMC9_001! … aIMC9_009!	a,field1,field2;<CR><LF>
aIV_001! … aIV_009!	a,field1,field2;<CR><LF>
aIC_001! … aIC_099!	a,field1,field2;<CR><LF>
aICC_001! … aICC_099!	a,field1,field2;<CR><LF>
aIC1_001 … aIC1_009!	a,field1,field2;<CR><LF>
.	a,field1,field2;<CR><LF>
.	a,field1,field2;<CR><LF>
.	a,field1,field2;<CR><LF>
aIC9_001 … aIC9_009!	a,field1,field2;<CR><LF>
aICC1_001! … aICC1_099!	a,field1,field2;<CR><LF>
.	a,field1,field2;<CR><LF>
.	a,field1,field2;<CR><LF>
.	a,field1,field2;<CR><LF>
aICC9_001! … aICC9_099!	a,field1,field2;<CR><LF>
aIR0_001! … aIR0_099!	a,field1,field2;<CR><LF>
.	a,field1,field2;<CR><LF>

Command	Response
.	a,field1,field2;<CR><LF>
.	a,field1,field2;<CR><LF>
aIR9_001! … aIR9_099!	a,field1,field2;<CR><LF>
aIRC0_001! … aIRC0_099!	a,field1,field2;<CR><LF>
.	a,field1,field2;<CR><LF>
.	a,field1,field2;<CR><LF>
.	a,field1,field2;<CR><LF>
aIRC9_001! … aIRC9_099!	a,field1,field2;<CR><LF>
aIHA_001! … aIHA_999!	a,field1,field2;<CR><LF>
aIHB_001! … aIHB_999!	a,field1,field2;<CR><LF>

Table 20. The identify measurement parameter commands

6.2.4 Examples of the Identify Measurement Parameter Commands

a. Generic example:

0M!0001!<CR><LF>
0IM_001!0,field1,field2,field3;<CR><LF>

b. Metadata for an M command, showing that the M command takes a precipitation measurement, using optional field 3 to describe the data value:

0IM!00001<CR><LF>
0IM_001!0,PR,mm,precipitation rate per day;<CR><LF>

c. Metadata CC command, showing that the CC command takes an average air temperature measurement, using additional/optional fields to provide additional metadata, with a CRC appended to the response:

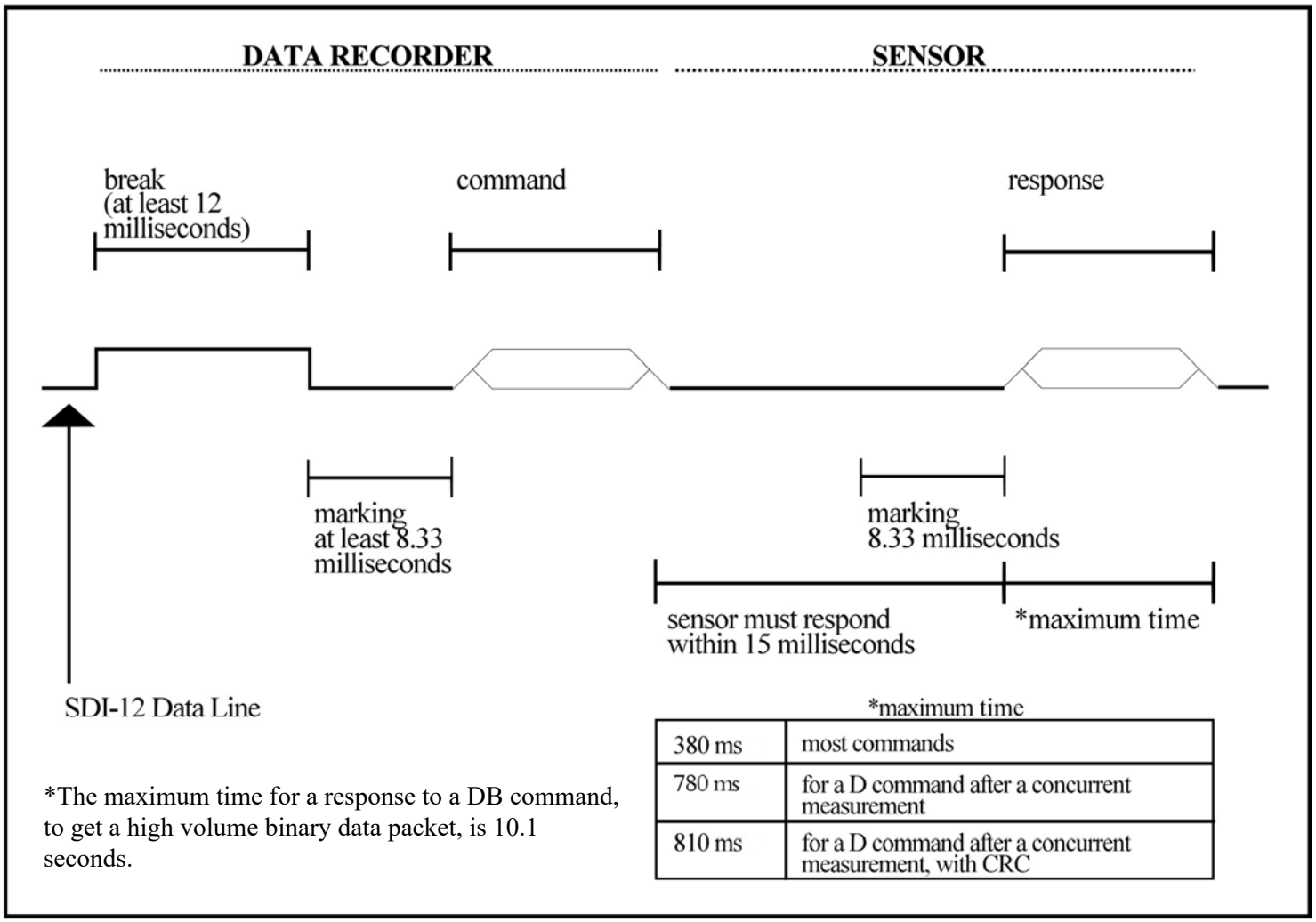
0CC!000502<CR><LF>
0ICC_001!0,CU,degrees C,average air temperature,calibration data,40,1235;@|e<CR><LF>

6.3 Compliance with Metadata Commands

To claim compliance with version 1.4, an SDI-12 sensor must support the metadata commands. Because there is no requirement for an SDI-12 data recorder to manage or store the metadata, the transparent mode (see section 4.4.13.1) on the data recorder is sufficient support for these commands. Explicit data recorder support for these commands is not otherwise required.

7.0 SDI-12 Timing

Figure 3 shows a timing diagram for an SDI-12 command and its response. The tolerance for all SDI-12 timing is ± 0.40 milliseconds. The only exception to this is the time between the stop bit of one character and the start bit of the next character. The maximum time for this is 1.66 milliseconds, with no tolerance.



*maximum time

380 ms	most commands
780 ms	for a D command after a concurrent measurement
810 ms	for a D command after a concurrent measurement, with CRC

*The maximum time for a response to a DB command, to get a high volume binary data packet, is 10.1 seconds.

Figure 3. SDI-12 Timing

- A data recorder transmits a break by setting the data line to spacing for at least 12 milliseconds.

- The sensor will not recognize a break condition for a continuous spacing time of less than 6.5 milliseconds and will always recognize a break when the line is continuously spacing for more than 12 milliseconds.

- Upon receiving a break, a sensor must detect 8.33 milliseconds of marking on the data line before it looks for an address.

- A sensor must wake up from a low-power standby mode and be capable of detecting a start bit from a valid command within 100 milliseconds after detecting a break.

- After a data recorder transmits the last character of a command, it must relinquish control of the data line within 7.5 milliseconds following the end of the stop bit. (Tolerance: +0.40 milliseconds.)

- After receiving the break and the command, the addressed sensor sets the data line to marking for 8.33 milliseconds and then send the response. (Tolerance: -0.40 milliseconds.) The start bit of the first response byte must start within 15 milliseconds after the stop bit of the last byte of the command. (Tolerance: +0.40 milliseconds.)

- After a sensor transmits the last character of a response, it must relinquish control of the data line within 7.5 milliseconds. (Tolerance: +0.40 milliseconds.)

- No more than 1.66 milliseconds of marking are allowed between the end of the stop bit and the start bit (e.g., between characters) on any characters in the command or the response. (No tolerance.) This permits a response to an M command to be sent within a 380 millisecond window.

- Sensors must return to a low-power standby mode after receiving an invalid address or after detecting a marking state on the data line for 100 milliseconds. (Tolerance: +0.40 milliseconds.)

- When a recorder addresses a different sensor, or if the data line has been in the marking state for more than 87 milliseconds, the next command must be preceded by a break.

Note: The low power standby mode, in addition to being a power consumption state, is a protocol state and a break is required to leave that state.

7.1 Rules for the Break

The data recorder sends a break when it is necessary to wake a sensor from low-power standby mode. An SDI-12 sensor is required to return to low-power standby mode after receiving an invalid address or after detecting a marking state on the data line for 100 milliseconds. For this reason, a break must precede a command whenever a new sensor is

addressed and after 87 milliseconds of marking on the data line.

When a data recorder receives a service request, it does not have to send a break if it issues the D0 command within 87 milliseconds after the service request. If, however, more than 87 milliseconds elapse, the D0 command must be preceded with a break.

7.2 Retries

A data recorder must support retries. Sensors have up to 100 milliseconds to wake up after detecting a break and will not respond to any commands until they have awakened. Also, sensors will go to sleep after detecting 100 milliseconds of idle time (marking) on the SDI-12 bus.

When issuing retries, if no response is received from a sensor, the recorder must wait for at least 16.67 milliseconds after the last stop bit of the command, but no longer than 87 milliseconds, and then issue a retry (without a break). (This period of 87 milliseconds includes the 16.67 milliseconds spent waiting for a response from the sensor.) If a correct response is not received after re-transmitting the command at least two more times, **with at least one of those retries more than 100 milliseconds after the end of the break**, the entire sequence (including the break and the retries) should be repeated at least two more times. The flow chart in appendix B of this document illustrates this retry logic. At least one of the retries must be issued after 100 milliseconds after the falling edge of the break to ensure that the sensor has been given the full 100 milliseconds to wake up after the break. A retry is needed if one of the following three conditions exist:

1) no response from the sensor;

2) 8.33 milliseconds of marking on the data line, after receiving the start bit of the response;

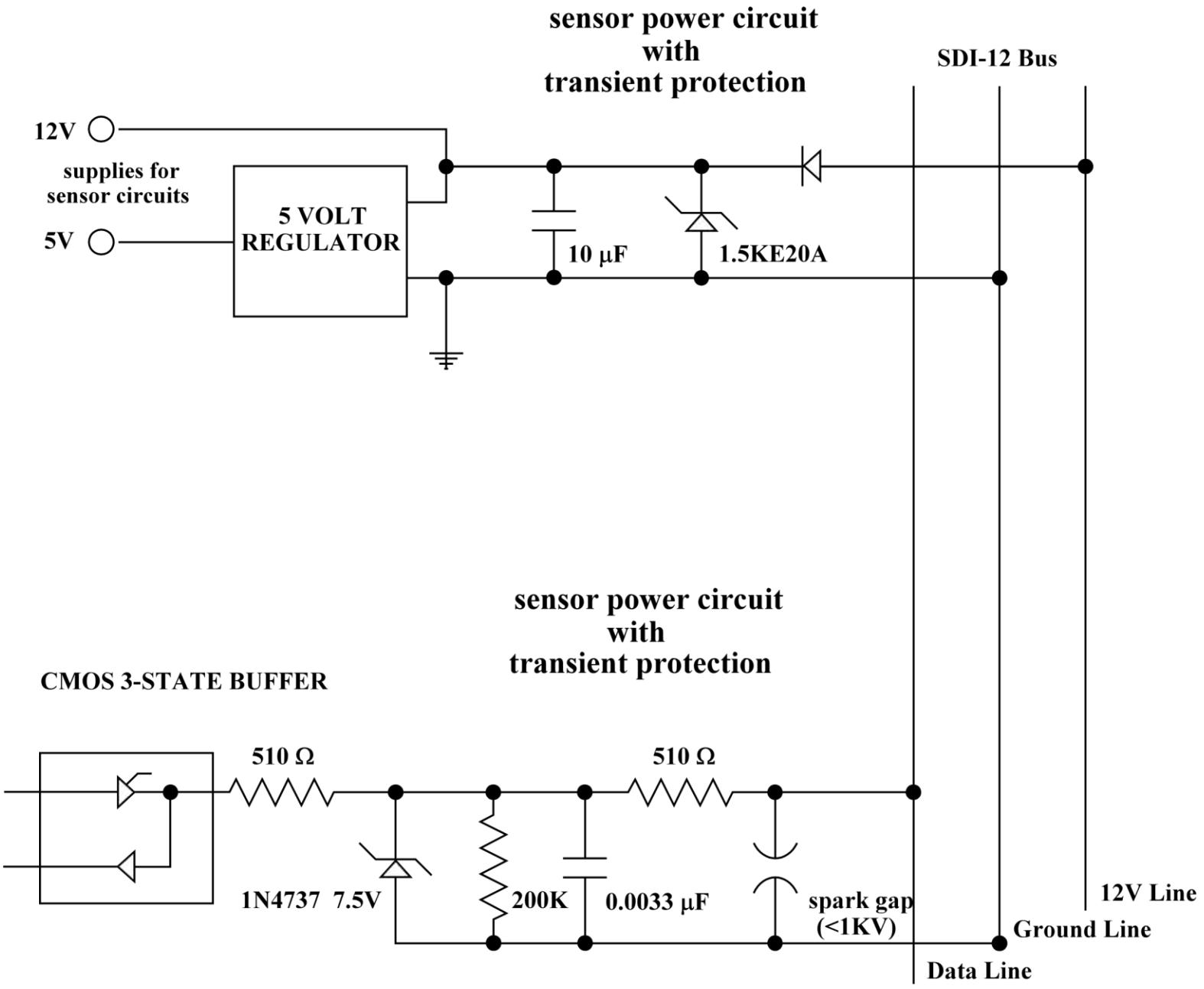
3) an invalid response.

Invalid responses include responses in an incorrect format, parity errors, framing errors, CRC errors, or bus contentions. The data recorder must wait for the response to complete before sending a retry. Because the data recorder is the master of the SDI-12 bus, it is not required to issue a retry. The requirement is for a data recorder to support retries. If one of the above conditions exists, the data recorder will issue retries. Under extraordinary conditions, not as normal operating procedure, the data recorder has the option of not issuing retries.

APPENDICES

Appendix A

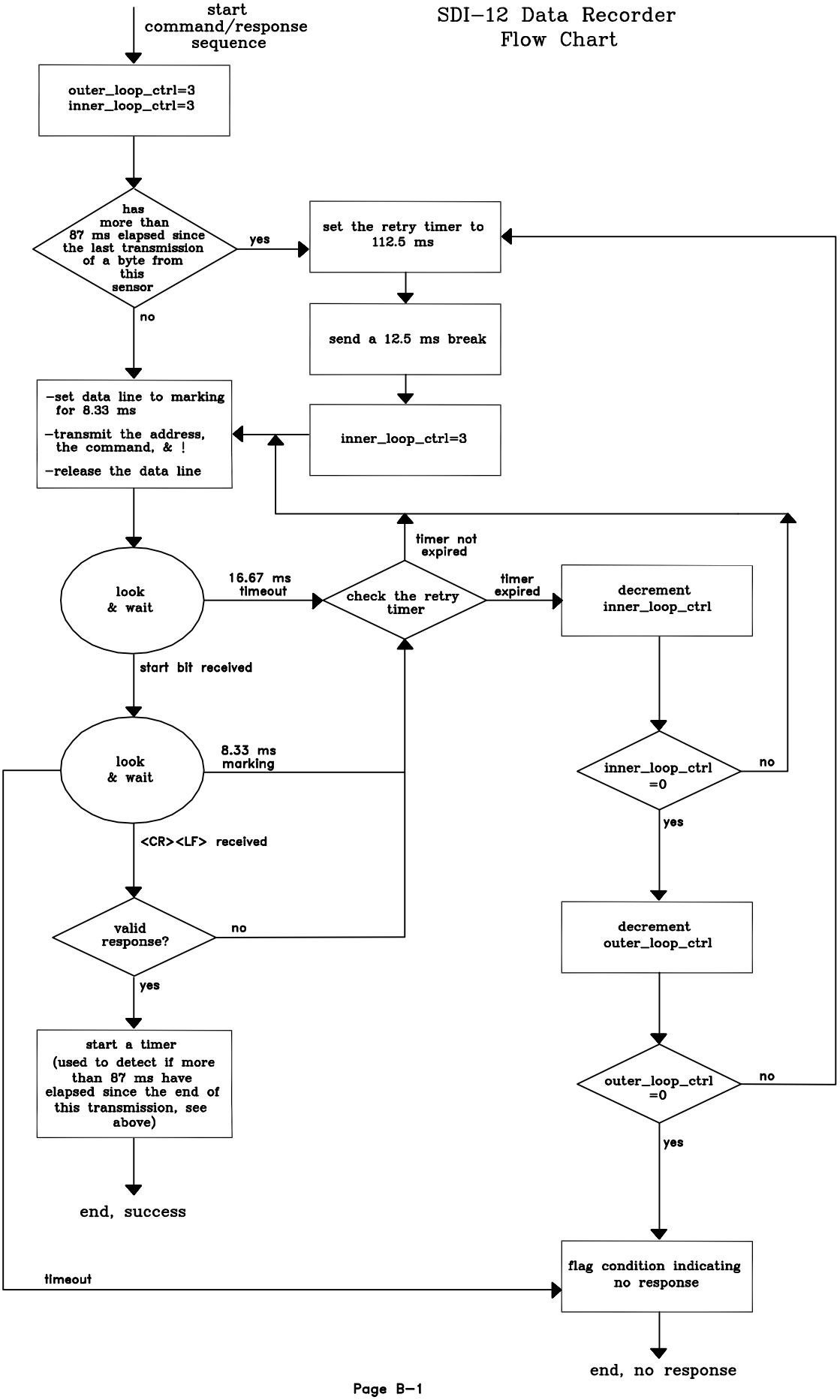
Suggested SDI-12 Circuits


sensor power circuit
with
transient protection
SDI-12 Bus
12V
supplies for
sensor circuits
5V
5 VOLT
REGULATOR
10 µF
1.5KE20A
sensor power circuit
with
transient protection
CMOS 3-STATE BUFFER
510 Ω
510 Ω
1N4737 7.5V
200K
0.0033 µF
spark gap
(<1KV)
12V Line
Ground Line
Data Line

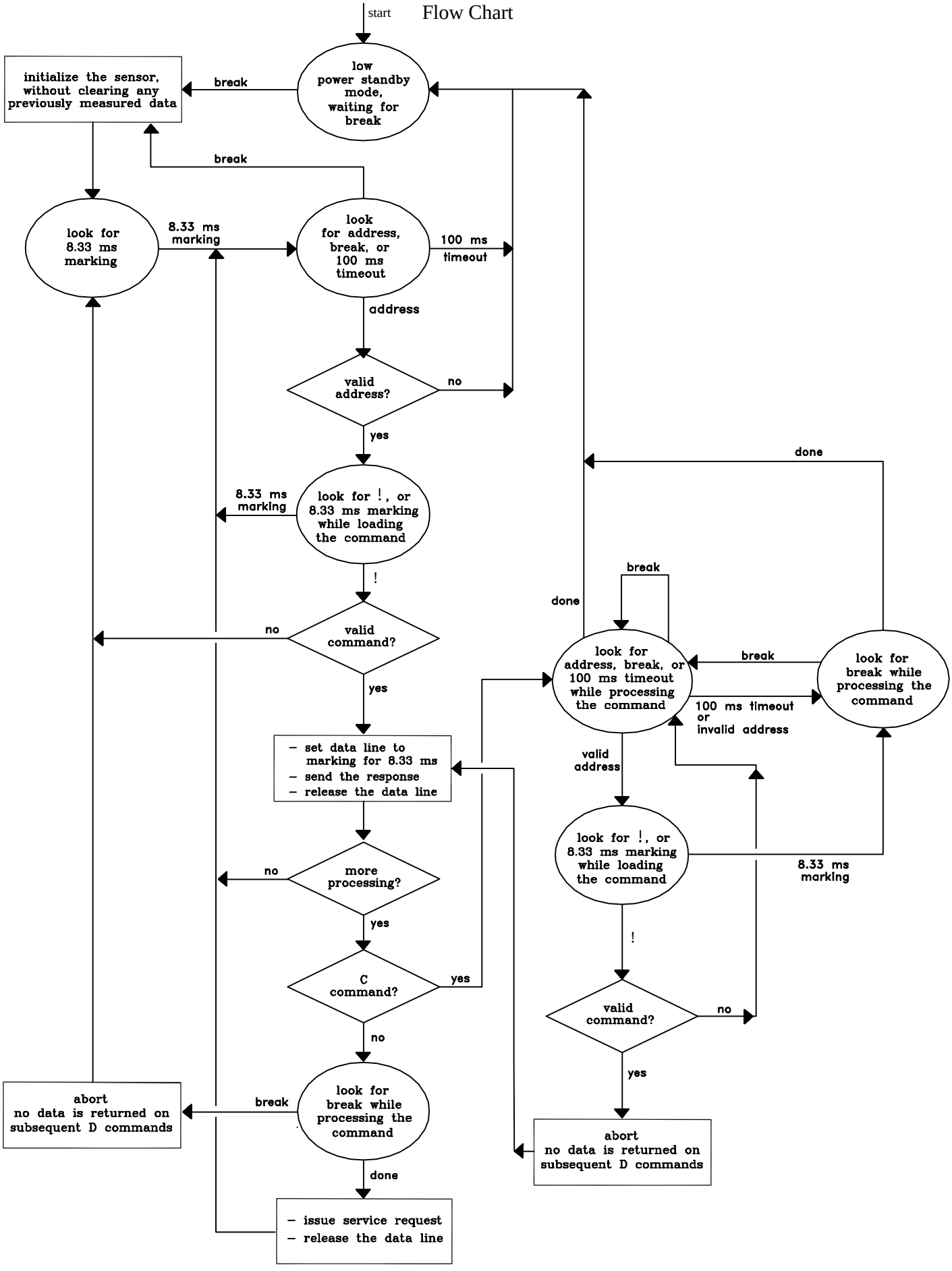
Appendix B

Suggested SDI-12 Flow Control
for
SDI-12 Data Recorders and Sensors

SDI-12 Data Recorder Flow Chart



SDI-12 Sensor Flow Chart



Appendix C

SDI-12 Glossary

Appendix C
SDI-12 GLOSSARY

Address. A single character used to identify each sensor on the SDI-12 bus. The first character of every command and the first character of every response is an address. The normal address characters are "0" to "9."

Basic command set. The following SDI-12 commands: a!, aAb!, ?!, aI!, aM!, aM1! . . . aM9!, aC!, aC1! . . . aC9!, aD0! . . . aD9!, aR0! . . . aR9!, aV!, aMC!, aMC1! . . . aMC9!, aCC!, aCC1! .. . .aCC9!, aRC0! . . . aRC9!

Break. Continuous spacing, by the data recorder, on the data line for at least 12 milliseconds. This is a special condition used to wake sensors from a low power standby mode.

Buffer. Memory in the sensor that holds the sensor's most recent measurement. A sensor is required to keep this information until it receives a new M or V command. When a sensor has a measurement in its buffer, repeated D commands cause it to return the same data each time the D command is issued. If, however, a break occurs while a sensor is making a measurement in response to an M or V command, the sensor must abort its measurement and empty its data buffer. In this case, no data is returned in response to a D command.

Byte frame format. The manner in which a character is encoded for serial transmission. SDI-12 uses 1 start bit, 7 data bits, even parity, and 1 stop bit.

CRC. Cyclical redundancy check, a form of error checking. The sensor appends a number, which is encoded as ASCII characters, that is related mathematically to the characters to be transmitted to the data recorder. Upon receiving the data, the data recorder recomputed the CRC value to verify that the data was received without error.

Data bits. The bits in a character that carry information, as opposed to the start bit, the parity bit, and the stop bit. SDI-12 uses 7 data bits.

Extended command. A command not in the basic SDI-12 command set. Extended commands are defined by the manufacturer of the sensor. Extended commands calibrate sensors, run diagnostic tests in sensors, and perform other tasks specific to a sensor.

Even parity. Setting the parity bit to 0 or 1, as needed, to ensure that the total number of binary ones in the data bits and the parity bit are an even number. SDI-12 uses even, as opposed to odd, parity.

Marking. A binary state of 1 on the SDI-12 data line, with a voltage range of -0.5 to 1.0 volts.

Parity bit. A bit after the data bits in a character, used for error detection. SDI-12 uses even parity.

Printable characters. ASCII characters in the range 20 hex (a space) through 7E hex (~).

SDI-12 command. A string with 2 or more printable characters that will be sent to an SDI-12 sensor. The first character of an SDI-12 command is an address and the last character is an exclamation point (!).

SDI-12 bus. A cable with three conductors: (1) serial data, (2) ground, and (3) 12 volts--used to connect an SDI-12 data recorder with one to ten SDI-12 sensors.

SDI-12 data recorder. A data acquisition device that can be interfaced with one or more SDI-12 sensors. The recorder polls the sensor using the SDI-12 electrical interface and the SDI-12 communications protocol.

SDI-12 response. A string with 3 or more characters, returned to an SDI-12 data recorder by an SDI-12 sensor. The first character of an SDI-12 response is an address and the last two characters are a carriage return/line feed (<CR><LF>). Other than the carriage return/line feed, all characters must be printable ASCII characters.

SDI-12 sensor. A measurement device that can be polled by an SDI-12 data recorder, using the SDI-12 electrical interface and the SDI-12 communications protocol.

Service request. An address followed by carriage return/line feed (a<CR><LF). This is issued by a sensor, after an M or V command, to tell the data recorder that the requested measurements are ready.

Spacing. A binary state of 0 on the SDI-12 data line, with a voltage range of 3.5 to 5.5 volts.

Start bit. A bit indicating the start of a new character in an SDI-12 command or response.

Stop bit. A bit indicating the end of a character in an SDI-12 command or response. SDI-12 specifies one stop bit.

Transparent mode. A mode in an SDI-12 data recorder in which extended commands (or basic commands), received from a computer, a terminal, or a modem can be sent to a sensor. In transparent mode the data recorder buffers characters until terminated, wakes the sensor with a break, and sends the command string to the sensor. An example of transparent mode is a data recorder interfaced to a PC via a serial port. The user enters SDI-12 commands on the keyboard, and the command is transmitted to the data recorder. The recorder reads the command, transmits it to the sensor, and returns the response to the PC. The PC then displays the response.

<values> A string of one or more data values (e.g. measurements) returned from a sensor in response to a D or an R command; each data value is in this format:

pd.d

where:

p - the polarity sign (+ or -) (required)
d - numeric digits before the decimal point
. - the decimal point (optional)
d - numeric digits after the decimal point

the maximum number of digits for a data value is 7, even without a decimal point

the minimum number of digits for a data value (excluding the decimal point) is 1

the maximum number of characters in a data value is 9 (polarity sign + 7 digits + the decimal point)

Appendix D

Revisions

Appendix D
Revisions

Version 1.4 August 10, 2016

Corrected an error: Page 27, Table 17, Rows 1 and 2, Column 2, incorrectly said "Signed-32 bit integer," rather than "Signed-16 bit integer."

Version 1.4 July 14, 2016

Added Sections:

5.0 High Volume Commands
6.0 Metadata Commands

This adds the High Volume ASCII Command, the High Volume Binary Command, extends the use of the D commands to get up to 999 data values, adds binary data packets in response to the High Volume Binary Command, adds the Identify Measurement Commands, and the Identify Measurement Parameter Commands.

Renumbered section 5.0 SDI-Timing to 7.0 SDI-12 Timing.

Updated the table of contents to include the additions.

Other modifications, as needed, to include text about the High Volume Commands and the Metadata Commands, throughout the document.

Numerous clarifications in response to a full review by the SDI-12 Support Group's Technical Committee: some rewording, moved (without changing) some sentences and paragraphs, corrected some typo-graphical errors, and corrected some minor formatting issues.

Version 1.3 January 28, 2016

Clarifications only made to the specification:

1) Page 26, Section 5.2. Added "CRC error" to the first sentence of the last paragraph as one of five invalid response conditions, from a sensor, that require a data recorder retry.

2) Page 26, Section 5.2. Deleted the word three in the sentence, "If one of the above ~~three~~ conditions exists the data recorder will issue retries."

3) Appendix B-2, SDI-12 Sensor Flow Chart. Added "set data line to marking for 8.33 ms" to the box between "valid command?" and "more processing?" This clarifies the supporting text on page 24, Section 5.0, which says, "After receiving the break and the command, the addressed sensor sets the data line to marking for 8.33 milliseconds and then sends the response."

4) Appendices B-1 and B-2, SDI-12 Data Recorder Flow Chart/SDI-12 Sensor Flow Chart. Changed all instances of “msec” to the correct abbreviation of “ms.”

Version 1.3 January 26, 2013

Clarifications only made to the specification:

1) Page 1, Section 1.0. Added “see section 3.0 for details” to the third bullet.

2) Page 1, Section 1.0. Deleted the fourth bullet, “Up to 200 feet of cable between a sensor and data recorder.”

3) Page 11, Table 9. Made a correction. Changed “a valid range of 0 to 9" to “a valid range of 1 to 9" in row 3, column 2.

4) Page 14, Table 10. Deleted this text, “(a data recorder must be able to read and store at least 20 parameters from a sensor, nn = 20)”

5) Page 14, Section 4.4.7. Deleted this text, “The maximum number of data values a sensor can return for nn is 20. A data recorder is also required to store 20 values. Future versions of SDI-12 may increase this to a maximum of up to 99 values.”

6) Page 14, Section 4.4.7. Added this text, “The data recorder should document the number of data values it can store in response to a C command.”

7) Page 14, Section 4.4.7. Changed the last paragraph from:

See the send data command, aD0!, aD1! . . . Ad9! for examples of this command. Because the start concurrent measurement command is closely related to the send data command, examples for both commands are in sections 4.4.8.3 and 4.4.8.4.

To:

“See the send data command, aD0!, aD1! . . . aD9!, section 4.4.8, for the data response requirements. Because the start concurrent measurement command is closely related to the send data command, examples for both commands are in sections 4.4.8.3 and 4.4.8.4.”

8) Page 14, Section 4.4.8. Change the paragraph heading from “aD0! ... aD9!” to “aD0!, ...aD1 ... aD9!.”

Version 1.3 January 3, 2012

Clarifications only made to the specification:

1) Page 5, Section 4.0. Reference to page 14 changed to reference to section 4.4.7.

2) Page 12, Section 4.4.5.1. Deleted the sentence, “It must also empty its buffer so that no data are returned in response to a D command.” This sentence was deleted because the sentence after it specifically states what information is to be returned to the sensor in response to subsequent D commands.

3) Page 14, Section 4.4.7.1. Deleted the sentence, “It must also empty is buffer so that no data are returned in response to a D command.” This sentence was deleted because the sentence after it specifically states what information is to be returned to the sensor in response to subsequent D commands.

4) Page 16, Section 4.4.8.3. Reference to section 4.4 changed to section 4.4.8.

5) Page 16, Section 4.4.8.3. Circular reference to section 4.4.8.3 changed to section 4.4.8.4.

6) Page 23, Section 4.4.13. Corrected typographical error. Deleted the letter s from “extendeds.”

7) Formatted all paragraph headings in bold.

8) Changed Appendix B, Suggested SDI-12 Flow Control for SDI-12 Data Recorded so that the logic for the concurrent command matches the text in the body of the document:

a) top left rectangle: deleted the text “in its buffer”
b) top left oval: changed “making” to “marking.”
c) bottom left rectangle: deleted the phrase “empty buffer so”
d) bottom right rectangle: changed the text so that it matches the text in the bottom left rectangle
e) the “yes” branch of the “C command” decision now goes to the oval “look for address, break, or 100 msec timeout while processing the command,” rather than the oval “look for break while processing the command”
f) the “no” branch of “valid command” in the C command processing section now goes to the oval “look for address, break, or 100 msec timeout while processing the command,” rather than the oval “look for break while processing the command” oval
g) switched the positions of the two columns of logic for the C command logic to minimize the crossing of lines

Version 1.3 January 12, 2009

Clarifications only made to the specification:

1) Page 2, Section 3.0. Appended a phrase and a sentence: “, each with 200 feet of cable. With fewer sensors, longer cable lengths are possible.”

2) Page 4, Section 3.12. Added a sentence: “Due to this impedance, the maximum cable length depends on the capacitance of all cables connected to the SDI-12 data line.”

3) Page 12, Section 4.4.5. Added a sentence: “The data recorder may wait for the entire ttt time. Therefore, the tolerance above the measurement time, added to ttt, should be minimal

Version 1.3 July 18, 2005

Clarifications only made to the specification:

1) Page 7, Section 4.4. Added a sentence: “None of the basic commands should affect the sensor’s calibration.”

2) Page 12, Section 4.4.5.1. Added “or this sensor,” to the sentence at the bottom of the page.

3) Page 16. Corrected a typographical error.

Version 1.3 July 25, 2004

Clarifications only made to the specification:

1) Page 8, Table 5. Changed “same as the D commands” to “formatted like the D commands.”

2) Page 16. Changed “The R commands (R0! ... R9!) work exactly like the D commands (D0! ... D9!).” to “The response to R commands (R0! ... R9!) are formatted like the D commands (D0! ... D9!).”

3) Page 16. Changed “The only difference is that the R commands do not need to be preceded with an M command ...” to “The main difference is that the R commands do not need to be precede with an M command ...”

4) Page 16. Added a new paragraph:

Each R command is an independent measurement. For example, R5 need not be preceded by R0 ... R4.

5) Page 16. Changed “If a CRC was requested then a CRC must be appended to the address.” to “If a CRC was requested then the <CR><LF> must be preceded by the CRC.”

6) Page 20. Changed “When these commands are used, the data returned in response to the D commands must have a CRC code appended to it.” to “When these commands are used, the data returned in response to the D commands, or R commands, must have a CRC code appended to it.”

Version 1.3 September 17, 2002

Clarifications only made to the specification:

1) Page 8. Corrected the footnote at the bottom of Table 5 to reference section 4.4.6 rather than section 4.4.12. Section 4.4.6 is the correct section that footnote should reference.

2) Page 9. Added "The "!" character can only be used in a command as the terminator character."

3) Page 9. Corrected typographical error; changed "of" to "or."

4) Page 11, table 9. Added "n is a single digit integer with a valid range of 0 to 9."

5) Page 12 and Page 14. Added "(or followed by <CRC><CR><LF> if a CRC was requested)."

6) Page 13. Changed: "A service request is issued by the sensor, after an M or V command, when it has finished its measurement." To: "A service request is issued by the sensor, after an M, MC, or V command, when it has finished its measurement."

7) Page 15, table 11, Added "the minimum number of digits for a data value (excluding the decimal point) is 1."

8) Page 16. Added "If a CRC was requested then a CRC must be appended to the address. For example: 0AP@<CR><LF>."

9) Page 20. Added "All shifts shift in a zero."

10) Page 22. Corrected an error in an example that shows a sensor response with a CRC value (changed 1.1234 to 1.234). With the additional '1' in the data, the CRC value shown in the example was incorrect.

11) Page 25. Corrected typographical error; changed "sensor" to "data recorder."

12) Page B-2. Corrected typographical error; changed "emty" to "empty."

13) Page C-3. Added "the minimum number of digits for a data value (excluding the decimal point) is 1."

Version 1.3 April 7, 2000

1) Added a Cyclic Redundancy Check (CRC) to data returned from SDI-12 Sensors, by the use of existing commands with a C appended, namely: aMC!, aMC1! ... aMC9!, aCC1! ... aCC9!, and aRC0! ... aRC9!.

2) Changed the Data Line "OFF" impedance to be less restrictive (160K to 360K ohms, rather than 200K ohms with a plus or minus 10% tolerance.

3) Reworded the paragraph about inductive loading.

4) Clarified the definition of low power standby mode.

Version 1.2 October 21, 1996

Clarifications only made to the specification:

1) Page 2. Added the following statement, which was in Version 1.0, but omitted from Versions 1.1 and 1.2 (4/12/96) by mistake.

"In the following specifications, all values not indicating specific limits, have an allowable tolerance of ±10% of the value."

2) Page 12. Added text to clarify when ttt seconds begins and added text to advise sensor designers to return a ttt value that is greater than the time it takes the sensor to take a measurement.

"The ttt time period begins upon completion of the transmission of the line feed character."

"A sensor should return a ttt value greater than the time it takes to make a measurement, to allow for timing tolerances and for the service request. (See section 4.4.6.)"

3) Page 14. Added text to clarify when ttt seconds begins.

"The ttt time period begins immediately after the sensor transmits the <LF> character."

Version 1.2 April 12, 1996

1) Added the concurrent measurement command (aC!) and the additional concurrent measurement commands (aC1! . . . aC9!) to the basic command set.

2) Clarified the section on data recorder retries to indicate that a data recorder must issue retries unless it has received a valid response or if it wishes to abort the measurement.

3) Clarified the no data response to the additional measurement commands (aM! . . . aM9!)

4) Added the address query command (?!) to the basic command set.

5) Added the change address command (aAb!) to the basic command set.

6) Added the continuous measurement command (aR0! . . . aR9!) to the command set.

Version 1.1 July 7, 1994

Rewrite and clarification of the original specification. No technical changes.

Version 1.0 October 1988

Original release.