Trident DM Wear Debris Monitor User’s Manual

PS-0106-0300 – Inline wear debris monitor with 12mm bore and RS-485 / RS-232 interface / Pulse Output

May 26, 2017
03-PS-0106-0300
Revision: B

REVISION TRACKING

DCO	Version	Author	Description
-	A	BXN	Initial version
0223	B	RCB	Rebrand rewrite

CONTENTS

1 OVERVIEW

Poseidon Systems' Trident DM Metallic Wear Debris Monitor is a real-time, in-line fluid sensing technology for the detection of metallic particles in oil. By tracking metallic wear over time, users can identify component faults in their ealiest stages, enabling reduced repair costs, reduced downtime, and increased safety of operation. This technology has been applied across a number of machinery platforms including marine systems, wind power generation, hydraulics, transmissions and gearboxes. While primarily designed for use in oils, it can be used in any nonconductive media to detect the presence of metallic particles.

The Trident DM will detect, categorize (ferrous vs. nonferrous), and size metals within a machinery lubrication system. The compact, robust device can be mounted within just about any lubrication system on any machine. The monitor will detect and measure particles with an estimated spherical diameter of 50 micron ferrous and 150 micron nonferrous and larger. Several outputs formats are available including particle type/size, approximate mass, and particle counts in user configurable bins.

When connected to a host system the device will give indication of the machinery condition relative to wear metals generated. This gives you the control to improve your asset health management by being able to make informed maintenance decisions based on real-time data. The monitor communicates in real-time using the MODBUS® serial digital communication standard over RS485 or RS232. This industry standard allows for easy integration into your current systems.

Two models of the Trident DM have been designed to help minimize costs associated with mechanical integration. The DM4500 model features female ½" o-ring boss ports (per SAE J1926/1) that can accept a variety of fitting adapters. The DM4600 model is designed to directly integrate with CJC™ Fine Filters (HDU 15/- and 27/-).

2 SCOPE

This product user's manual is written specifically for the Trident DM4500 and DM4600 Wear Debris Monitors, model numbers PS-0106-0300-A and PS-0106-0300-C. Verify the model number on the device label before using information in this manual. An image of the PS-0106-0300-A device is shown in Figure 1 for reference.

Figure 1: Trident DM4500 – Model PS-0106-0300-A

Figure 2 shows the PS-0106-0300-C device designed to integrate directly with CJC™ Fine Filters. This variant integrates between the pump and filter assembly of HDU 15/- and HDU 27/- systems. Please refer to installation manuals for the DM4600 for more information.

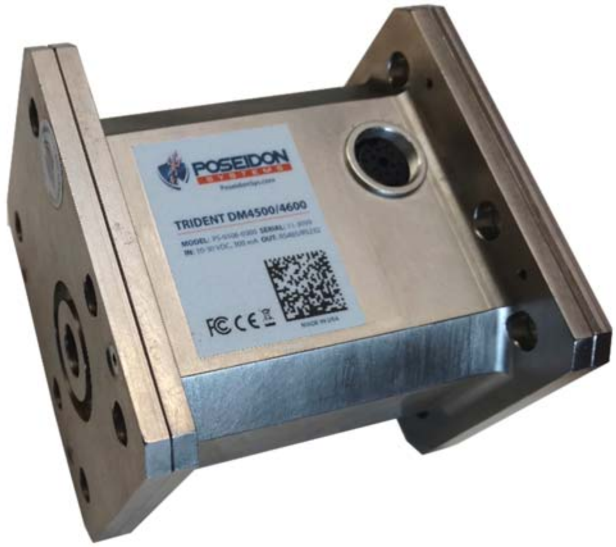

Figure 2: DM4600 Wear Debris Monitor – Model PS-0106-0300-C

3 SPECIFICATIONS

Table 1: Specifications

Characteristic	Specification
Flow Rate	0.25 to 10 GPM (0.95 – 38 LPM)
Working Pressure	150 psi (10.3 bar) max
Fluid Temperature	-40 to 185°F (-40 to 85°C)
Ambient Temperature	-40 to 185°F (-40 to 85°C)
Viscosity	Viscosity independent
Sensor Bore Diameter	0.472 in (12mm)
Oil Connection	DM4500: ½" ORB (per SAE J1926/1) DM4600: Direct integration with CJC™ Fine Filters (HDU 15/- and 27/-)
Ingress Protection	IP65
Power Supply	10-30 Vdc, 300mA
Communications	RS-485 Modbus RTU, RS-232 Modbus RTU, Pulse Ouput
Default Communications Settings	115200 baud, 8-bit, 1 stop, no parity
Weight	1.5 lbs (0.68 kg)

4 INSTALLATION

Proper installation of the Trident DM sensor is critical to insuring reliable operation. The following sections detail the considerations required when selecting an installation location and plumbing the unit into the system.

4.1 LOCATION SELECTION

The following considerations should be used to determine the optimal location for the Trident DM debris monitor:

1.) The sensor must be installed in a line with flow rates between 0.25 and 10 gallons per minute.
2.) The sensor must be mounted downstream of the gearbox/component to be monitored and prior to any filters or screens.
3.) Minimize the number of sharp bends between the component of interest and the debris monitor.
4.) Avoid sharp bends immediately after the debris monitor to avoid buildup of particles at the sensor exit.
5.) Avoid mounting the device to hot surfaces.
6.) The sensor is insensitive to flow direction, but should be oriented such that the I/O cable will not interfere with nearby equipment.

4.2 MOUNTING

The DM4500 sensor should be securely mounted to a rigid structure using either #8 or M8 clearance holes located at the sensor's corners as shown in Figure 3. Screws/bolts should be placed in four mounting locations. Alternatively, magnetic mounting kits are available for low vibration environments.

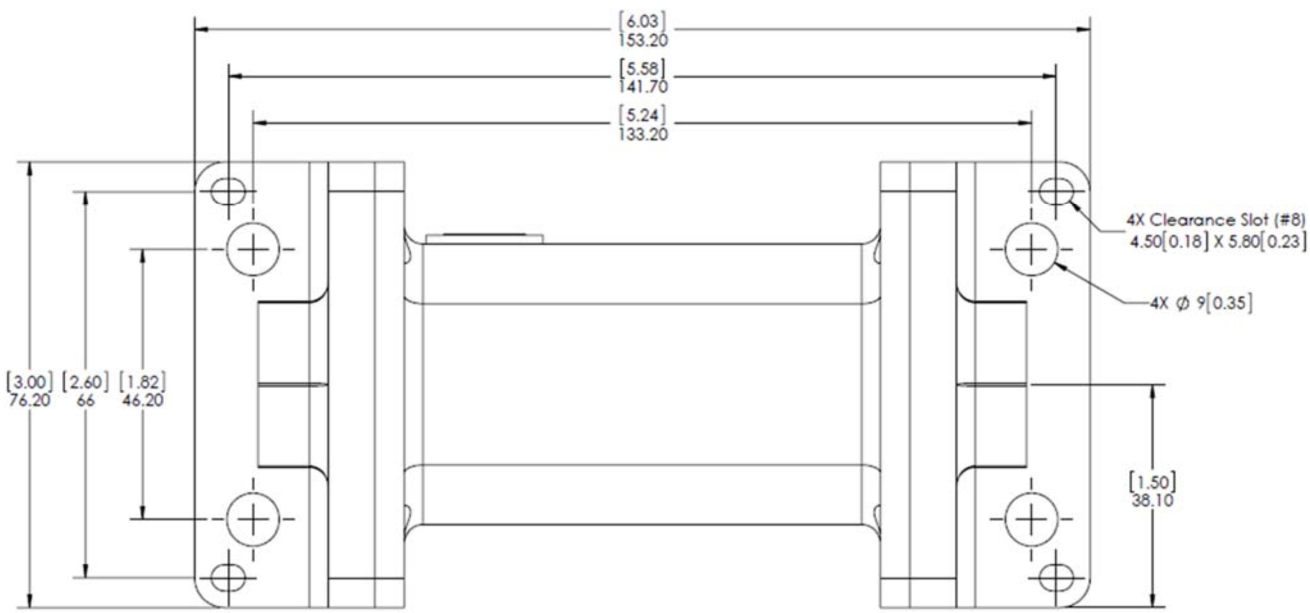

Figure 3: DM4500 Mounting Pattern

Note: To avoid false positive or reduced sensitivity to small particles, mount the sensor in a location that minimizes exposure to vibration.

The DM4600 is designed to mount between the CJC^{TM} Fine Filter's pump and filter assembly. Extended replacement mounting bolts and replacement o-rings are shipped for the target system.

4.3 PLUMBING

Before plumbing the DM4500 sensor into the flow, be sure to follow the guidelines set forth in 4.1 regarding location selection.

The DM4500 device uses ½" ORB (per SAE J1926/1) fluid ports to allow for compatibility with a wide variety of fluid systems. Use Swagelok fittings to adapt the DM4500 sensor to the available port or tubing type. To directly adapt the sensor to tubing, the following fittings are available:

Table 2: Tube Adapters

Tube OD	Manufacturer	Part Number
1/4"	Swagelok	SS-400-1-8ST
3/8"	Swagelok	SS-600-1-8ST
1/2"	Swagelok	SS-810-1-8ST
5/8"	Swagelok	SS-1010-1-8ST
3/4"	Swagelok	SS-1210-1-8ST
10mm	Swagelok	SS-10M0-1-8ST
12mm	Swagelok	SS-12M0-1-8ST

Similar fittings may be used to adapt other line sizes or fittings to a compatible interface.

When plumbing the DM4500 into a system take note of the following:

1. Carry out all pipe bending or hose routing and perform an alignment check prior to making any connections. If using tubing, ensure the mating tube is cut square and free of burrs. If using hoses, verify that the maximum bend radius will not be exceeded.
2. Lubricate tube/fitting adapter threads and o-ring before assembling. Do not apply sealing tape or compounds to this interface.
3. Hand tighten the tube/fitting adapter onto the DM4500 unit and verify the adapter's o-ring contacts the port face.
4. Tighten the tube/fitting adapter onto the DM4500 using the recommended 39-43 ft-lbs torque per SAE J1926-3 to ensure o-ring compression.

4.4 CABLING

4.4.1 Connector Pinout

All electrical connections to the Trident DM wear debris sensor are made through the single female I/O connector (Binder 09 0132 00 12). This connector is shown in Figure 4; the pinout and descriptions are shown in Table 3. Interface cables should be constructed using one of the mating connectors defined in Table 4.

Figure 4: I/O Connector Pinout

Table 3: Trident DM I/O Connector Pinout

Pin	Name	Description
B	V+	Power input
A	GND	Power return
K	RS 485 A[1,2]	RS 485 non-inverting pin
J	RS 485 B[1,2]	RS 485 inverting pin
H	COM GND[3,4]	Communications ground
G	COM GND[3,4]	Communications ground
F	RS 232 TX[1]	RS 232 Transmit
E	RS 232 RX[1]	RS 232 Receive
D	Relay+	Positive relay connection
C	Relay-	Negative relay connection
L	N/C	No connect
M	N/C	No connect

Note 1: Sensor communications are supported using the MODBUS standard over a 2-wire (3 including the ground line) half duplex RS-485 implementation or a standard RS-232 implementation.

Note 2: It is recommended to use shielded twisted pair cables for RS-485 data lines. Termination resistors matching the characteristic impedance of the cable (120 Ohms recommended) should be used at both ends of the data cable to improve bus performance.

Note 3: It is recommended that an isolated RS-485 adapter is used to connect the sensor to the host system.

Note 4: COM GND should not be connected via a shield or drain wire unless an isolated RS-485 adapter is used.

Table 4: Mating Connector Options

Manufacturer	Part Number	Description
Binder	99 5629 15 12	Male plug, straight, shielded
Binder	99 5629 75 12	Male plug, right angle, shielded

4.4.2 Trident DM Interface Cables

Poseidon offers a wide variety of interface cables in varying lengths and terminations. Contact Poseidon to discuss cabling needs.

4.4.3 Communications Adapters

While any RS485 or RS232 adapter which conforms with the associated standard should be compatible with the Trident DM sensor, the following communications adapters have been tested and confirmed to be compatible.

The SerialGear USB COMi-SI RS485 to USB adapter has been tested and confirmed to be working with the TA100 RS485 interface. The adapter ships in 4-wire RS485 mode from the manufacturer. The adapter will have to be configured for 2-wire RS485 mode via its configurable jumpers.

Figure 5: USB COMi-SI RS485 to USB adapter

The FTDI US232R series USB to DB9 Serial Adapter Cables have been tested and confirmed to be working with the Trident DM RS232 interface.

Figure 6: FTDI RS232 to USB adapter

Note: Some devices are manufactured with local echo enabled by default, meaning transmitted data is echoed back to the receiver. This may be problematic for host applications that are not designed to ignore echoed characters. Check your device and host application settings if you experience problems with a new adapter.

5 MODBUS INTERFACE DESCRIPTION

The specifications for the Modbus interface in this section apply to the RS-485 and RS-232 connections.

5.1 MODBUS COM SETTINGS

The default settings of the serial communications on the Trident DM are as follows:

Setting	Value
Baud rate	115,200 baud
Data bits	8
Parity	None
Stop bits	1
Mobus slave address	1

5.2 MODBUS HOLDING REGISTERS

The Trident DM exposes the following Modbus holding registers:

Register	Description	Writeable?	Length in 16-bit words	Units
21	Device's Modbus address	Yes, update takes effect after next device reset	1	N/a
22	Baud rate	Yes, update takes effect after next device reset	1	0 = 9600 baud 1 = 19200 baud 2= 38400 baud 3 = 57600 baud 4 = 115200 baud
23	Vendor ID		1	2 = Poseidon Systems
24	Product ID		1	11 = DM4500/4600
25	Serial number		2	N/A
27	Firmware version		2	N/A
29	Hardware revision		1	N/A
32	Electronics board temperature		1	Degrees Celsius
43	Flow rate estimate		1	Lpm * 128
44	Ferrous concentration estimate		1	mg/L * 128
45	Nonferrous concentration estimate		1	mg/L * 128
46	Number of ferrous particle bins	Yes—must accompany the ferrous particle bin	1	Number of bins

Register	Description	Writeable?	Length in 16-bit words	Units
		upper endpoints		
47	Ferrous particle bin upper endpoints	Yes—must accompany the number of ferrous particle bins	19	Microns
66	Ferrous particle counts		20	Number of particles
86	Number of nonferrous particle bins	Yes—must accompany the nonferrous particle bin upper endpoints	1	Number of bins
87	Nonferrous particle bin upper endpoints	Yes—must accompany the number of nonferrous particle bins	19	Microns
106	Nonferrous particle counts		20	Number of Particles
126	Total ferrous mass		2	Total ferrous mass since last read
128	Total nonferrous mass		2	Total nonferrous mass since last read
275	Clear bins	Yes	1	Write "1" to clear binned count values
276	Cumulative count write interval	Yes	1	0 = cumulative count off 600 to 65535 = write time interval in seconds, cumulative count on
277	Termination resistance (RS485)	Yes	1	1 = 120 ohm, 0 = none
278	Pulse output enable	Yes, update takes effect after next device reset	1	1 = enable pulse output 0 = disable pulse output
279	Micrograms per pulse	Yes, update takes effect after next	1	Number of micrograms each pulse represents

Register	Description	Writeable?	Length in 16-bit words	Units
		device reset		
280	Pulse on time (pulse high time)	Yes, update takes effect after next device reset	1	5ms multiplier Range = 1 to 200
281	Pulse off time (time between pulses when multiple pulses are output at once)	Yes, update takes effect after next device reset	1	5ms multiplier Range = 1 to 200
1238	Reset/Reboot sensor	Yes, reboots immediately	1	Write "10" to reboot the sensor

5.3 FUNCTION CODES

Function codes are the encoding of the type of Modbus command. The Trident DM Wear Debris Monitor supports the following function codes:

Function code		Description
03	0x03	Read holding registers
04	0x04	Read input registers
06	0x06	Write single holding register
16	0x10	Write multiple holding registers
43	0x2B	Read device identification
109	0x6D	Trident DM-specific sub-function
CODE+128	CODE+0x80	Modbus exception

Trident DM-specific sub-functions are:

Function code		Description
64	0x40	Get particle bins
65	0x41	Set particle bins
66	0x42	Get particle counts
67	0x43	Get particle history

5.3.1 Modbus Exception Codes

Trident DM supports the following codes, though not all are used:

Exception code		Description
1	0x01	Illegal function
2	0x02	Illegal data address
3	0x03	Illegal data value
4	0x04	Illegal slave device failure
5	0x05	Exception acknowledge
6	0x06	Slave device busy
8	0x08	Memory parity error
10	0x0A	Gateway path unavailable
11	0x0B	Gateway target failed

5.4 MESSAGE FORMATS

Multi-byte integer values (viz., 16-bit and 32-bit integers, signed or unsigned) are always transmitted big endian (most significant byte first) as required by the "MODBUS Application Protocol Specification" §4.2. The one exception to this is the Modbus 16-bit CRC which is always transmitted little endian (least significant byte first).

Particles (or particle bin endpoints) are encoded in 16 bits where the low order 15 bits specify the particle size in microns and the high order bit indicates the particle type (0 = nonferrous, 1 = ferrous). For example, 0x8123 is a 291-micron ferrous particle and 0x0321 is an 801-micron nonferrous particle.

5.5 READ HOLDING REGISTERS

Trident DM implements the "read holding registers" command as specified in §6.3 of the "MODBUS Application Protocol Specification", version 1.1b. Trident DM supports only the holding registers set forth in the table in the section "MODBUS Holding Registers" above.

Request:

Destination address	Function code = 0x03	Start register number (MSB)	Start register number (LSB)	Register count (MSB)	Register count (LSB)	Modbus CRC (LSB)	Modbus CRC (MSB)

Response:

Destination address	Function code = 0x03	Byte count	First register value (MSB)
First register value (LSB)	Second register value (MSB)	Second register value (LSB)	•••
•••			
Last register value (MSB)	Last register value (LSB)	Modbus CRC (LSB)	Modbus CRC (MSB)

Example:

To read the bank of 11 holding registers starting at holding register 21:

Client transmits	01 03 00 15 00 0B 15 C9
Client receives	01 03 16 00 01 00 04 00 01 00 04 00 74 00 68 00 06 00 06 00 01 00 00 00 00 01 A4

5.6 WRITE SINGLE HOLDING REGISTER

Trident DM implements the "write single register" command as specified in §6.6 of the "MODBUS Application Protocol Specification", version 1.1b. Only certain holding registers support writes as specified in the table in the section "MODBUS Holding Registers" above.

Request:

Destination address	Function code = 0x06	Register number (MSB)	Register number (LSB)	Value (MSB)	Value (LSB)	Modbus CRC (LSB)	Modbus CRC (MSB)

Response:

Destination address	Function code = 0x06	Register number (MSB)	Register number (LSB)	Value (MSB)	Value (LSB)	Modbus CRC (LSB)	Modbus CRC (MSB)

Example:

To set the serial line baud rate (holding register 22) to 19200 baud (encoded as 0x01):

Client transmits	01 06 00 16 00 01 A9 CE
Client receives	01 06 00 16 00 01 A9 CE

5.7 WRITE MULTIPLE HOLDING REGISTERS

Trident DM implements the "write multiple registers" command as specified in §6.12 of the "MODBUS Application Protocol Specification", version 1.1b. Only certain holding registers support writes as specified in the table in the section "MODBUS Holding Registers" above.

Request:

Destination address	Function code = 0x10	Start register number (MSB)	Start register number (LSB)
Register count (MSB)	Register count (LSB)	Byte count	First register value (MSB)
First register value (LSB)	Second register value (MSB)	Second register value (LSB)	•••
•••			
Last register value (MSB)	Last register value (LSB)	Modbus CRC (LSB)	Modbus CRC (MSB)

Response:

Destination address	Function code = 0x10	Start register number (MSB)	Start register number (LSB)	Register count (MSB)	Register count (LSB)	Modbus CRC (LSB)	Modbus CRC (MSB)

Example:

If the desired ferrous particle bins (holding registers 46 through 65) for the Trident DM device (Modbus address 1) are:

Ferrous bin
0 – 135
136 – 270
271 – 405
406 – 540
541 – 675
> 675 (implied)

Client transmits	01 10 00 2E 00 05 0A 00 87 01 0E 01 95 02 1C 02 A3 CA 87
Client receives	01 10 00 2E 00 05 60 03

5.8 READ DEVICE IDENTIFICATION

The Trident DM device implements the "read device identification" function code as specified in §6.21 of the "MODBUS Application Protocol Specification", version 1.1b. Trident DM supports basic, regular, and specific object identification. There are no extended category objects. Trident DM returns values for seven distinct objects: vendor name, product code, version, vendor URL, product name, model name, and user application name.

Request:

Destination address	Function code = 0x2B	Modbus encapsulated interface type = 0x0E	Read device id (see "device id" table, below)	Object id (see "object id" table, below)	Modbus CRC (LSB)	Modbus CRC (MSB)

Device id	Description	Object id
0x01	Basic device identification	Ignored—all objects in the "basic" category are returned
0x02	Regular device identification	Ignored—all objects in the "regular" category are returned
0x04	Specific object identification	See "object id" table, below

Object id	Description	Category
0x00	Vendor name	Basic
0x01	Product code	
0x02	Version	
0x03	Vendor URL	Regular
0x04	Product name	
0x05	Model name	
0x06	User application name	

Response:

Destination address	Function code = 0x2B	Modbus encapsulated interface type = 0x0E	Read device id (see "device id" table, above)
Conformity level = 0x82 (regular identification (stream access and individual access))	More follows = 0x00 (false)	Next object id = 0x00	Number of objects
First object id	First object length (in octets)	First object first octet	First object second octet
…			
Second object id	Second object length (in octets)	Second object first octet	Second object second octet
…			
Last object id	Last object length (in octets)	Last object first octet	Last object second octet
…			
Modbus CRC (LSB)	Modbus CRC (MSB)		

Example:

Retrieving the basic object set:

Client transmits	01 2B 0E 01 00 70 77
Client receives	01 2B 0E 01 82 00 00 03 00 0A 46 6C 6F 77 74 6F 6E 69 63 73 01 0E 53 6D 61 72 74 4D 6F 6E 44 65 62 72 69 73 02 1A 36 2E 36 2E 36 20 4E 6F 76 20 20 35 20 32 30 31 30 20 31 35 3A 32 34 3A 34 35 10 44

Retrieving the regular object set:

Client transmits	01 2B 0E 02 00 70 87

5.9 GET PARTICLE BINS

This command retrieves the upper endpoints for the particle bins as configured on the Trident DM device. The bin lower endpoints are inferred from the upper endpoint of the next smaller bin and there are no gaps. The bin for the smallest size particles always has a lower endpoint of zero. The endpoints for the ferrous particle bins are given before those for the nonferrous particles and bins are always ordered from smallest endpoint to largest. There is also an implied bin for all particles larger than the largest stated upper endpoint.

Particle bin endpoints are encoded in 16 bits where the low order 15 bits specify the particle size in microns and the high order bit indicates the particle type (0 = nonferrous, 1 = ferrous).

Request:

Destination address	Function code = 0x6D	Sub-function code = 0x40	Modbus CRC (LSB)	Modbus CRC (MSB)

Response:

<table>
<tr><td>Destination address</td><td>Function code = 0x6D</td><td>First ferrous bin upper endpoint in microns (MSB)</td><td>First ferrous bin upper endpoint in microns (LSB)</td></tr>
<tr><td>Second ferrous bin upper endpoint in microns (MSB)</td><td>Second ferrous bin upper endpoint in microns (LSB)</td><td colspan="2">…</td></tr>
<tr><td colspan="4">…</td></tr>
<tr><td>Last ferrous bin upper endpoint in microns (MSB)</td><td>Last ferrous bin upper endpoint in microns (LSB)</td><td>First nonferrous bin upper endpoint in microns (MSB)</td><td>First nonferrous bin upper endpoint in microns (LSB)</td></tr>
<tr><td>Second nonferrous bin upper endpoint in microns (MSB)</td><td>Second nonferrous bin upper endpoint in microns (LSB)</td><td colspan="2">…</td></tr>
<tr><td colspan="4">…</td></tr>
<tr><td>Last nonferrous bin upper endpoint in microns (MSB)</td><td>Last nonferrous bin upper endpoint in microns (LSB)</td><td>Modbus CRC (LSB)</td><td>Modbus CRC (MSB)</td></tr>
</table>

Example:

If the bins configured for the Trident DM device (Modbus address 1) were:

Ferrous bin	Nonferrous bin
0 – 200	0 – 333
201 – 400	334 – 666
401 – 600	667 – 999
601 – 800	> 999 (implied)
801 – 1000	
> 1000 (implied)	

Client transmits	01 6D 40 0D 60
Client receives	01 6D 80 C8 81 90 82 58 83 20 83 E8 01 4D 02 9A 03 E7 A7 C4

5.10 SET PARTICLE BINS

This command configures particle bins on the Trident DM device with the specified upper endpoints. The bin lower endpoints are inferred from the upper endpoint of the next smaller bin and there are no gaps. The bin for the smallest size particles is always interpreted as having a lower endpoint of zero. The endpoints for the ferrous particle bins must be given before those for the nonferrous particles and bins must be ordered from smallest endpoint to largest. The Trident DM device also infers a bin for all particles larger than the largest stated upper endpoint, so clients must specify no more than 19 endpoints.

Particle bin endpoints are encoded in 16 bits where the low order 15 bits specify the particle size in microns and the high order bit indicates the particle type (0 = nonferrous, 1 = ferrous).

Request:

Destination address	Function code = 0x6D	Sub-function code = 0x41	First ferrous bin upper endpoint in microns (MSB)
First ferrous bin upper endpoint in microns (LSB)	Second ferrous bin upper endpoint in microns (MSB)	Second ferrous bin upper endpoint in microns (LSB)	…
…			
Last ferrous bin upper endpoint in microns (MSB)	Last ferrous bin upper endpoint in microns (LSB)	First nonferrous bin upper endpoint in microns (MSB)	First nonferrous bin upper endpoint in microns (LSB)
Second nonferrous bin upper endpoint in microns (MSB)	Second nonferrous bin upper endpoint in microns (LSB)	…	
…			
Last nonferrous bin upper endpoint in microns (MSB)	Last nonferrous bin upper endpoint in microns (LSB)	Modbus CRC (LSB)	Modbus CRC (MSB)

Response:

Destination address	Function code = 0x6D	Modbus CRC (LSB)	Modbus CRC (MSB)

NOTE: Trident DM returns an illegal data value exception for the following conditions:

- The frame is too short (less than five octets)
- The frame is of even length
- The frame is too long (greater than 81 octets)
- More than 19 bins are specified for either ferrous or nonferrous particles (the 20th bin is implied—its upper endpoint is the size of the largest particle detectable by Trident DM)
- At least one bin endpoint is specified in non-increasing order within the ferrous or nonferrous bin specifier
- Any ferrous bin is specified after any nonferrous bin
- The sum of the number of encoded ferrous bin endpoints and the number of encoded nonferrous bin endpoints does not match the size of the payload

Example:

If the desired bins for the Trident DM device (Modbus address 1) are:

Ferrous bin	Nonferrous bin
0 – 200	0 – 333
201 – 400	334 – 666
401 – 600	667 – 999
601 – 800	> 999 (implied)
801 – 1000	
> 1000 (implied)	

Client transmits	01 6D 41 80 C8 81 90 82 58 83 20 83 E8 01 4D 02 9A 03 E7 CE 7C
Client receives	01 6D C1 CD

5.11 GET PARTICLE COUNTS

This command retrieves the binned particle counts for all configured bins on the Trident DM device. As a side effect, the Trident DM device clears the counts upon retrieval.

Note that a particle count for the implied bin (for all particles larger than the largest stated bin upper endpoint) is returned

Request:

Destination address	Function code = 0x6D	Sub-function code = 0x42	Modbus CRC (LSB)	Modbus CRC (MSB)

Response:

Destination address	Function code = 0x6D	First encoded particle (MSB)	First encoded particle (LSB)
Second encoded particle (MSB)	Second encoded particle (LSB)	…	
…			
Last encoded particle (MSB)	Last encoded particle (LSB)	Modbus CRC (LSB)	Modbus CRC (MSB)

Example:

If the particle counts on the Trident DM device (Modbus address 1) were:

Ferrous bin	Count	Nonferrous bin	Count
0 – 200	0	0 – 333	0
201 – 400	1	334 – 666	3
401 – 600	1	667 – 999	0
601 – 800	4	> 999	0
801 – 1000	0		
> 1000	0		

Client transmits	01 6D 42 8C A1
Client receives	01 6D 80 00 80 01 80 01 80 04 80 00 80 00 00 00 00 03 00 00 00 00 37 28

5.12 GET PARTICLE HISTORY

Request:

Destination address	Function code = 0x6D	Sub-function code = 0x43	Modbus CRC (LSB)	Modbus CRC (MSB)

Response:

Destination address	Function code = 0x6D	First encoded particle (MSB)	First encoded particle (LSB)
Second encoded particle (MSB)	Second encoded particle (LSB)	…	
Last encoded particle (MSB)	Last encoded particle (LSB)	Modbus CRC (LSB)	Modbus CRC (MSB)

Example:

If the particle queue on the Trident DM device (Modbus address 1) contains:

- Fe 257 microns, Fe 514 microns, Fe 771 microns, Fe 1028 microns, NFe 773 microns, Fe 518 microns, Fe 263 microns

Client transmits	01 6D 43 4D 61
Client receives	01 6D 81 01 82 02 83 03 84 04 03 05 82 06 81 07 2B D0

5.13 CYCLIC REDUNANCY CHECK (CRC)

Cyclic Redundancy Check (or CRC) is used to prevent the processing of corrupted messages by Trident DM . It is designed to detect accidental changes to messages on the Modbus network.

The Trident DM device uses the CRC algorithm set forth in Appendix B of the "MODBUS over Serial Line Specification and Implementation Guide", version 1.02. Trident DM's implementation is given below. Note that CRCs are computed across both the message header and payload.

```
/* Table of CRC values for high-order octet */
static const uint8_t s_crcHi[] =
{
    0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0, 0x80, 0x41,
    0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40,
    0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0, 0x80, 0x41,
    0x00, 0xC1, 0x81, 0x40, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41,
    0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0, 0x80, 0x41,
    0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40,
    0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40,
    0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40,
    0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0, 0x80, 0x41,
    0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40,
    0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0, 0x80, 0x41,
    0x00, 0xC1, 0x81, 0x40, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41,
    0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0, 0x80, 0x41,
    0x00, 0xC1, 0x81, 0x40, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41,
    0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41,
    0x00, 0xC1, 0x81, 0x40, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41,
    0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0, 0x80, 0x41,
    0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40,
    0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0, 0x80, 0x41,
    0x00, 0xC1, 0x81, 0x40, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41,
    0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0, 0x80, 0x41,
    0x00, 0xC1, 0x81, 0x40
};

/* Table of CRC values for low-order octet */
static const uint8_t s_crcLo[] =
{
    0x00, 0xC0, 0xC1, 0x01, 0xC3, 0x03, 0x02, 0xC2, 0xC6, 0x06, 0x07, 0xC7,
    0x05, 0xC5, 0xC4, 0x04, 0xCC, 0x0C, 0x0D, 0xCD, 0x0F, 0xCF, 0xCE, 0x0E,
    0x0A, 0xCA, 0xCB, 0x0B, 0xC9, 0x09, 0x08, 0xC8, 0xD8, 0x18, 0x19, 0xD9,
    0x1B, 0xDB, 0xDA, 0x1A, 0x1E, 0xDE, 0xDF, 0x1F, 0xDD, 0x1D, 0x1C, 0xDC,
    0x14, 0xD4, 0xD5, 0x15, 0xD7, 0x17, 0x16, 0xD6, 0xD2, 0x12, 0x13, 0xD3,
    0x11, 0xD1, 0xD0, 0x10, 0xF0, 0x30, 0x31, 0xF1, 0x33, 0xF3, 0xF2, 0x32,
    0x36, 0xF6, 0xF7, 0x37, 0xF5, 0x35, 0x34, 0xF4, 0x3C, 0xFC, 0xFD, 0x3D,
    0xFF, 0x3F, 0x3E, 0xFE, 0xFA, 0x3A, 0x3B, 0xFB, 0x39, 0xF9, 0xF8, 0x38,
    0x28, 0xE8, 0xE9, 0x29, 0xEB, 0x2B, 0x2A, 0xEA, 0xEE, 0x2E, 0x2F, 0xEF,
    0x2D, 0xED, 0xEC, 0x2C, 0xE4, 0x24, 0x25, 0xE5, 0x27, 0xE7, 0xE6, 0x26,
    0x22, 0xE2, 0xE3, 0x23, 0xE1, 0x21, 0x20, 0xE0, 0xA0, 0x60, 0x61, 0xA1,
```

```
    0x63, 0xA3, 0xA2, 0x62, 0x66, 0xA6, 0xA7, 0x67, 0xA5, 0x65, 0x64, 0xA4,
    0x6C, 0xAC, 0xAD, 0x6D, 0xAF, 0x6F, 0x6E, 0xAE, 0xAA, 0x6A, 0x6B, 0xAB,
    0x69, 0xA9, 0xA8, 0x68, 0x78, 0xB8, 0xB9, 0x79, 0xBB, 0x7B, 0x7A, 0xBA,
    0xBE, 0x7E, 0x7F, 0xBF, 0x7D, 0xBD, 0xBC, 0x7C, 0xB4, 0x74, 0x75, 0xB5,
    0x77, 0xB7, 0xB6, 0x76, 0x72, 0xB2, 0xB3, 0x73, 0xB1, 0x71, 0x70, 0xB0,
    0x50, 0x90, 0x91, 0x51, 0x93, 0x53, 0x52, 0x92, 0x96, 0x56, 0x57, 0x97,
    0x55, 0x95, 0x94, 0x54, 0x9C, 0x5C, 0x5D, 0x9D, 0x5F, 0x9F, 0x9E, 0x5E,
    0x5A, 0x9A, 0x9B, 0x5B, 0x99, 0x59, 0x58, 0x98, 0x88, 0x48, 0x49, 0x89,
    0x4B, 0x8B, 0x8A, 0x4A, 0x4E, 0x8E, 0x8F, 0x4F, 0x8D, 0x4D, 0x4C, 0x8C,
    0x44, 0x84, 0x85, 0x45, 0x87, 0x47, 0x46, 0x86, 0x82, 0x42, 0x43, 0x83,
    0x41, 0x81, 0x80, 0x40
};

EXPORT uint16_t cModbus_computeCrc(const uint8_t *src, size_t srcLen)
{
    uint8_t index;
    uint8_t crcHi = 0xff;
    uint8_t crcLo = 0xff;
    clog_assert(src);
    /* this implementation is taken from the Modbus over
    serial line spec. */
    while(srcLen--)
    {
        index = crcLo ^ *src++;
        crcLo = crcHi ^ s_crcHi[index];
        crcHi = s_crcLo[index];
    }
    /* Modbus CRCs are transmitted little-endian */
    return ((crcLo << 8) | crcHi);
}
```

6 RELAY INTERFACE

The Trident DM has a relay output that can be wired to generate pulses correlating to the detected particle masses. The threshold of mass accumulated before outputting a pulse is user defined. Pulse on time duration as well as pulse off time duration (in the case of multiple pulses needed to be output) is also user definable. Table 5 shows the relevant registers for defining the relay interface behavior. A sensor reset is required for any new settings in these control registers to take effect.

Table 5: Relay interface control registers

Register	Description	Value
278	Pulse output enable	0 = Disable, 1 = Enable
279	Micrograms per pulse	Micrograms represented by each pulse
280	Pulse on time	Pulse time high 5ms multiplier
281	Pulse off time	Pulse time low 5ms multiplier

Table 6 lists the absolute maximum ratings for the relay interface. The relay pins can be accessed through the communications connector as described in Table 3.

Table 6: Relay interface absolute maximum ratings

Parameter	Maximum rating	Unit
Load current (continuous)	300	mA_{DC}
Load voltage	100	V
On resistance	4	Ohms

Figure 7 shows an example circuit to use with the relay interface. This circuit will provide a voltage pulse between the supply voltage input and ground. Note that the relay used between pins C and D has an on state resistance of 4 Ohms and should be limited to a maximum current of 300 mA and maximum voltage of 30 Vdc.

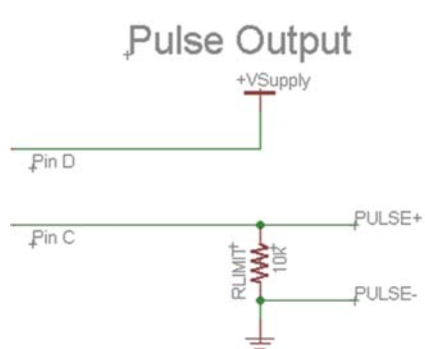

Figure 7: Example relay interface circuit

Figure 8 shows example output waveforms for the example circuit in Figure 7. Masses are calculated for each detected particle and a running total mass is kept. Once the total mass exceeds the value specified in register

279, pulses will be output. The total summed mass is divided by the value in register 279 to determine the number of pulses to output. The remainder is kept as the new total summed mass.

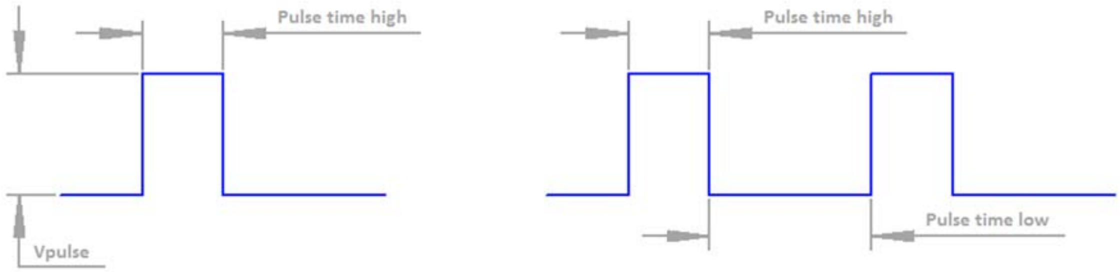

Figure 8: Example pulse output waveforms

The characteristics of the pulses are:

- $V_{pulse} = V_{supply}$
- Pulse time high = (Register 280) * 5ms
- Pulse time low = (Register 281) * 5ms