



M Y N A HSM

**Mettler Toledo Driver for DeltaV
Programmable Serial Interface Card
Series 2**

USER MANUAL

Rev. P1.55

April 28, 2009

DeltaV is a trademark of Emerson Process Management © Emerson Process Management 1998, 1999. All rights reserved.

Printed in the U.S.A.

While this information is presented in good faith and believed to be accurate, MYNAH Technologies does not guarantee satisfactory results from reliance upon such information. Nothing contained herein is to be construed as a warranty or guarantee, express or implied, regarding the performance, merchantability, fitness or any other matter with respect to the products, nor as a recommendation to use any product or process in conflict with any patent. MYNAH Technologies reserves the right, without notice, to alter or improve the designs or specifications of the products described herein.



1 INTRODUCTION

1.1 Scope

This document is the User Manual for the Mettler Toledo Weigh Scale serial communication driver firmware for the Emerson DeltaV Control System; it provides information required to install, configure, and maintain the Mettler Toledo driver firmware on the DeltaV Series 2 Programmable Serial Interface Card (PSIC). The reader should be familiar with Emerson's DeltaV controller system and the Mettler Toledo Equipment. Specific Mettler Toledo Weigh Scales supported are Lynx, Jaguar, Puma, Cougar, and ID3sTx.

The section *Document Format* briefly describes the contents of each section of this manual. *System Specifications* outlines hardware and software requirements for the Mettler Toledo Driver (P1.55) firmware. *Related Documents* lists other documents used to prepare this manual.

1.2 Document Format

This document is organized as follows:

Table 1 - Document Format

Introduction	Describes the scope and purpose of this document.
Theory of Operation	Provides a general functional overview of the Mettler Toledo Weigh Scale Driver.
Downloading Firmware	Describes downloading procedures for the Mettler Toledo Driver firmware on to the DeltaV PSIC.
PSIC Configuration	Describes procedures and guidelines for configuring the DeltaV PSIC.
Driver Communications	Describes Mettler Toledo commands used and DeltaV Registers containing Weight data.
Operational Check	Provides tips and assistance to ensure PSIC is properly setup and configured.
DeltaV - Mettler Toledo Electrical Interface	Describes the electrical interface between DeltaV and the Mettler Toledo Weigh Scale. Also describes the pin assignments for RS-232 communications.
Technical Support	Describes who to call if you need assistance.



1.3 System Specifications

The following table lists the minimum hardware requirements for the Mettler Toledo Weigh Scale Driver:

Table 2 - System Specifications

Firmware	Mettler Toledo Driver Firmware (P1.55)
Protocol Compatibility	<p>Mettler Toledo Protocol as defined in the documents listed below:</p> <ol style="list-style-type: none"> 1. <u>Cougar Terminal User's Guide, Appendix 2: Continuous Mode Output for Cougar Terminals with Standard Software.</u> 2. <u>Puma Terminal Technical Manual and Users Guide, Appendix 2: Serial Output Formats and Appendix 4: Host Mode Serial Interface.</u> 3. <u>M 8141 Installation and Operation Manual, Section 5.9, Host Interface.</u> 4. <u>IND780 Terminal Technical Manual, Appendix D: Multi-Continuous Out.</u>
Software Requirements	<p>DeltaV System Software (Release 4.2 or later) installed on a hardware-appropriate Windows NT workstation configured as a ProPlus for DeltaV</p> <p>Serial Interface Port License (VE4102)</p>
Minimum Hardware Requirements	<p>FRSI DeltaV PSIC Hardware PN: 12P2506X022</p> <p>FRSI DeltaV M3, M5, MD or Series 2 MD Controller, Power Supply and 2 wide controller carrier</p> <p>FRSI 8 wide I/O card carrier</p> <p>Mettler Toledo devices</p>

1.4 Revision History

Rev	Release Date	Revised By	Checked By	Description
1.17	11/07	NFW	NFW	Revision with driver toolkit v2.2
1.55	4/09	NFW	NFW	Revision with driver toolkit v3.01



2 THEORY OF OPERATION

As part of the serial interface port license, a standard Modbus protocol is installed on the DeltaV PSIC prior to customization. The PSIC needs to be flash upgraded from the Modbus protocol to the Mettler Toledo protocol before operation.

The RS-232 communication settings must be configured properly to ensure accurate communication between the PSIC and Mettler Toledo devices. RS-422/485 may be used if the Mettler Toledo devices support this electrical standard.

This driver functions either as a master or a slave. In master mode, the PSIC continuously sends weight and status read commands to the connected scale devices. The received responses are reported to DeltaV via dataset registers. When a user command is detected (commands are listed below), that command is sent out to the scale. The corresponding response Ack or Nak is reported back to the dataset.

In slave mode, this driver communicates with the Mettler Toledo Weigh Scales by simply receiving the continuous output of the scales. This output contains the weight and status information, which is reported to DeltaV via dataset registers.

In Multi-Continuous 1 or Multi-Continuous 2 mode, this driver receives continuous output from the scales and reports the weight and status information back to DeltaV. The driver also has the ability to send commands to the scales, using the CTPZ command set.

In general, the primary functions of the driver are listed below:

- Performs data and message handling between DeltaV and Mettler Toledo devices.
- Checks validity of messages received from the Mettler Toledo devices.
- Processes reply information and updates the corresponding dataset registers
- Update dataset register status and data block status to indicate the communication state.

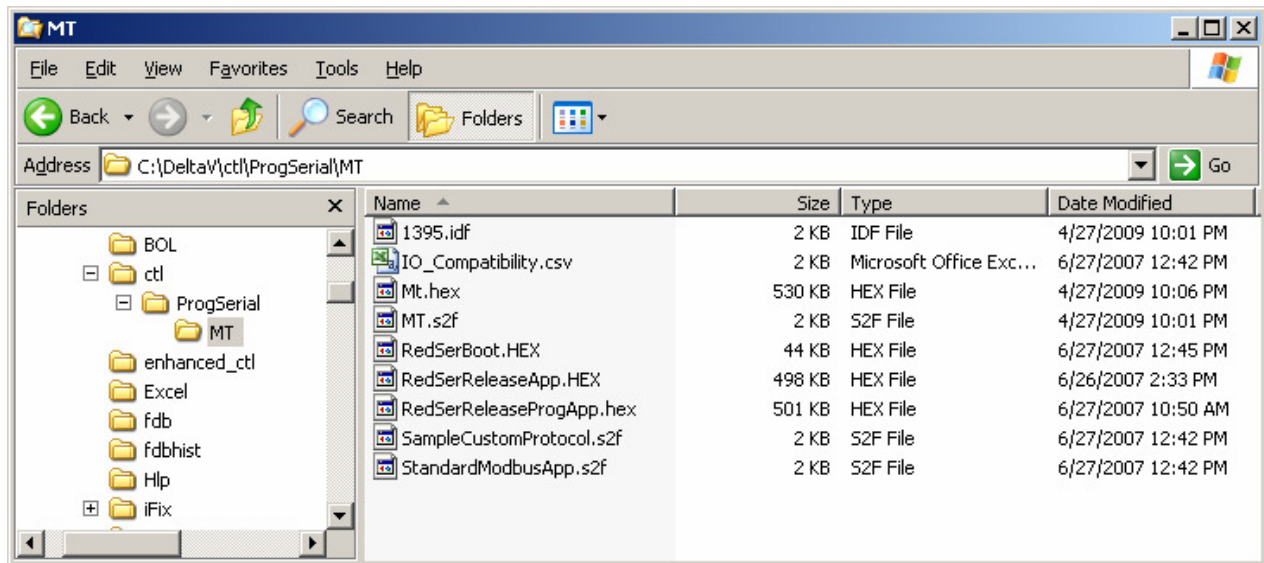


3 Downloading the firmware

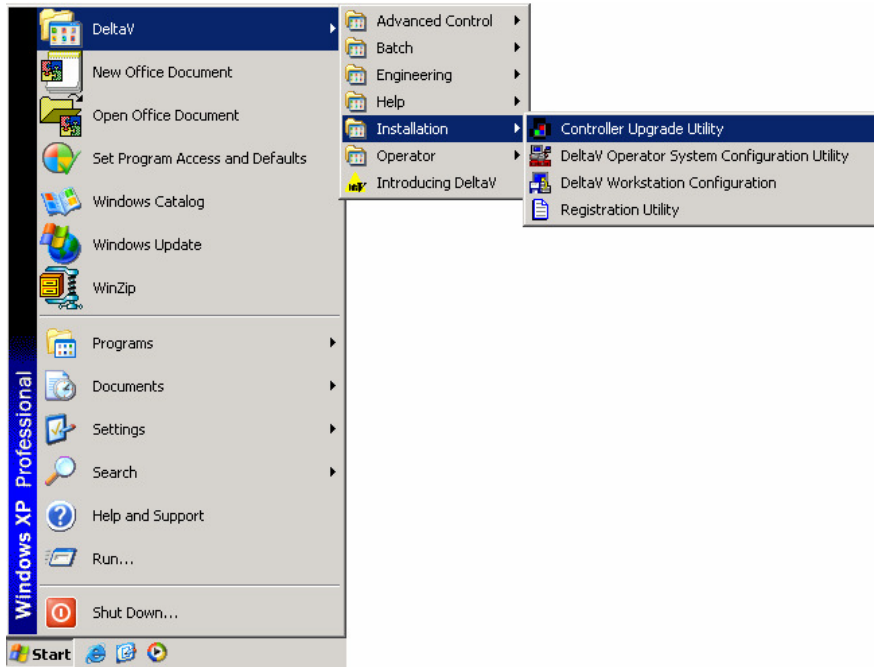
The driver software comprises 4 files, distributed on a mini-CD. These files must be copied to the DeltaV directory (you must create the directory first) on your ProPlus Workstation. The path is:

\DeltaV\ctl\ProgSerial\MT

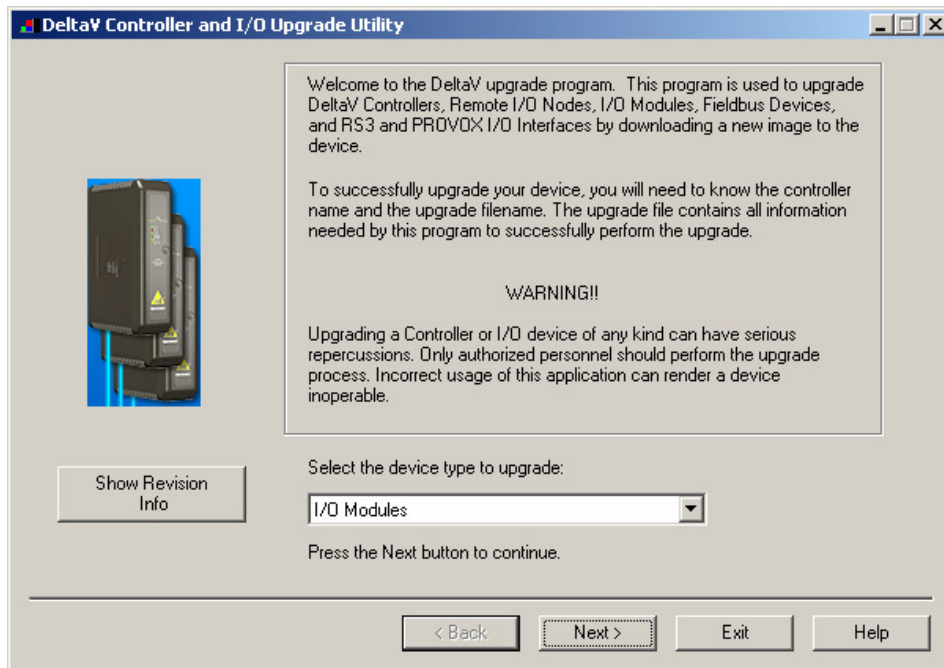
The following shows a completed copy operation:



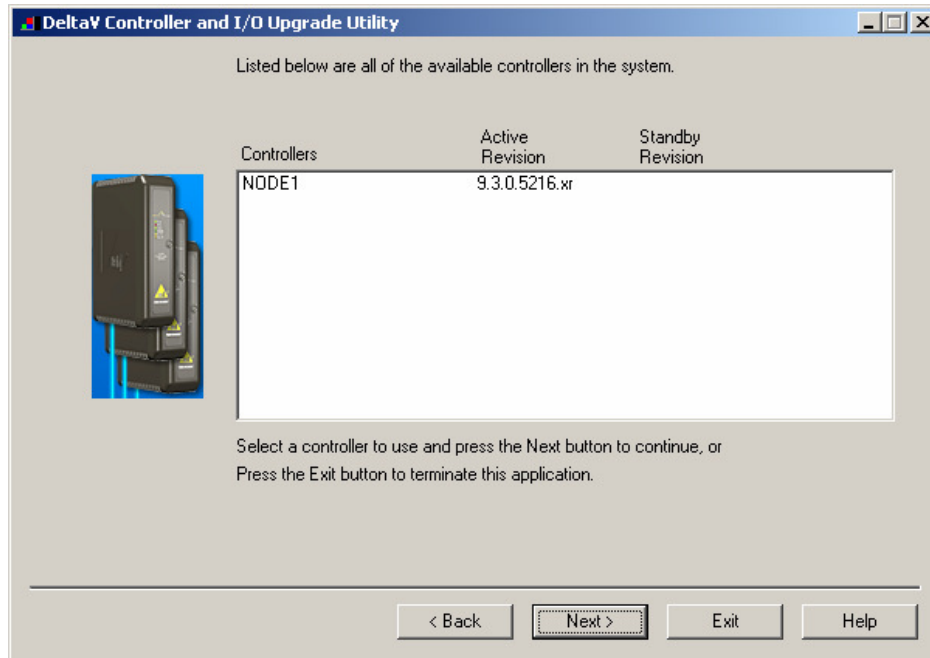
After copy completion, you are ready to program (or upgrade) the Programmable Serial Card with the supplied custom driver software. The steps are as follows:



1. Click on the Start button and select DeltaV-> Installation-> Controller Upgrade Utility as shown above, and the following dialog will appear:

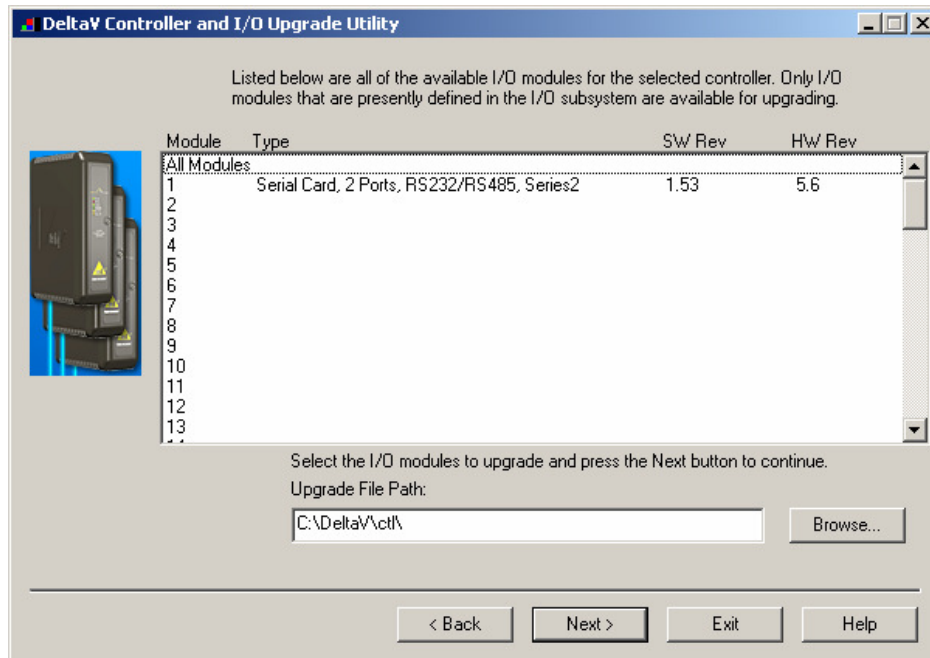


2. Choose Upgrade I/O Modules from the drop down menu and click Next.



3. The above dialog will appear, listing all the available Controllers in your network. From this dialog, select the appropriate Controller and then Click Next.

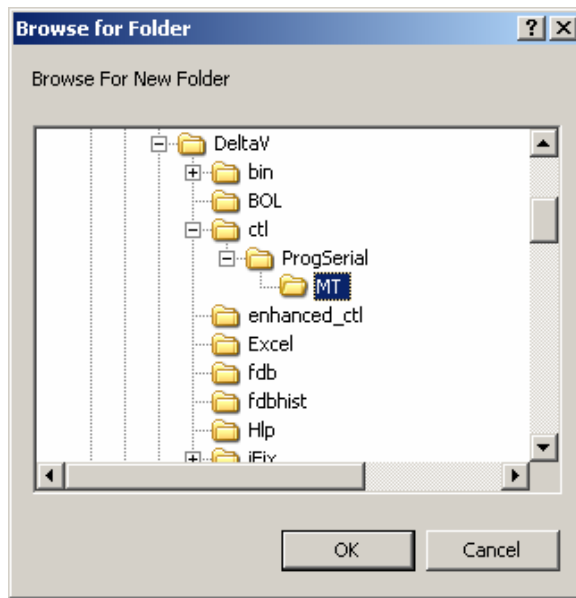
4. The following dialog will appear, listing all the I/O modules in your selected Controller. The shown list of I/O modules is an example only. Your list will be different.



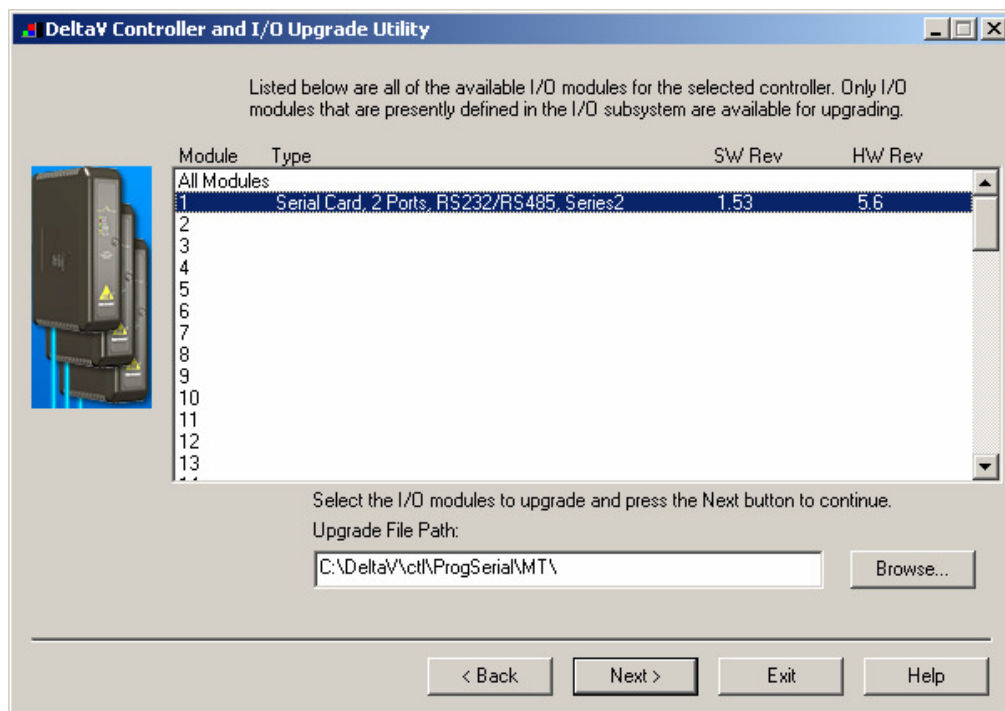


Note: The first time a standard Serial card is upgraded to the Mettler Toledo Driver, the dialog will be as shown below. When upgrading an existing Programmable Serial Card, skip Steps 4, 5 and 6, and go to Step 7.

5. Click the Browse button and select the DeltaV path as shown below, and then click Ok. Note that the disk drive could be C or D.

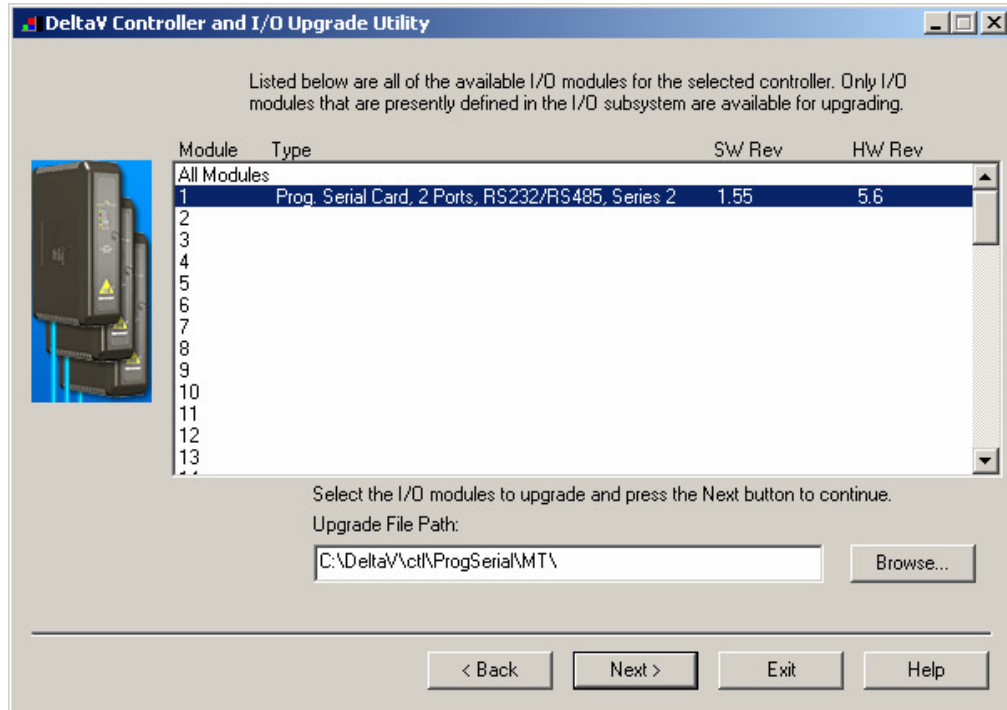


6. Select the I/O module again as shown below and then click Next. Go to Step 9.





7. If you are upgrading an existing Programmable Serial Card, the dialog will be as shown below. From this dialog, select the Programmable Serial Card I/O Module in the list.



For example, we will select I/O Module 1. This will give you a dialog, from which you will select the file path to where the driver software is located. This path will be:

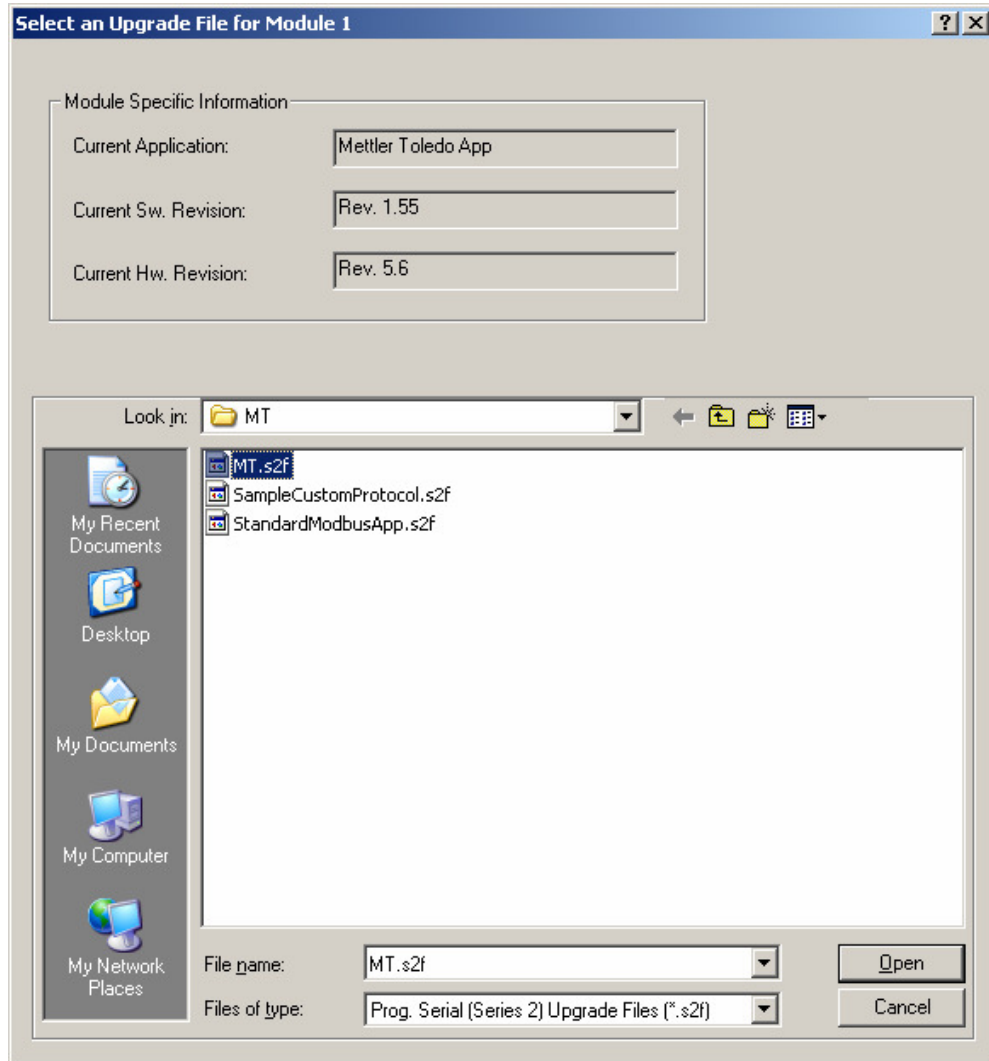
`\\DeltaV\ctl\ProgSerial\MT\`



Once you are in the specified directory, you will need to select the following file:

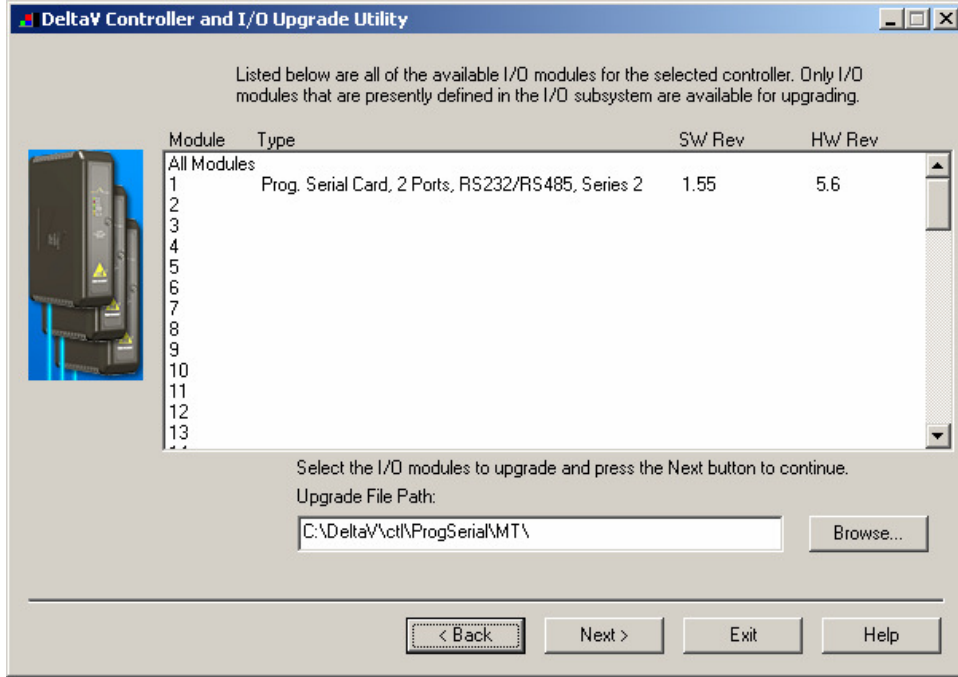
MT.S2F

This is shown in the following dialog.

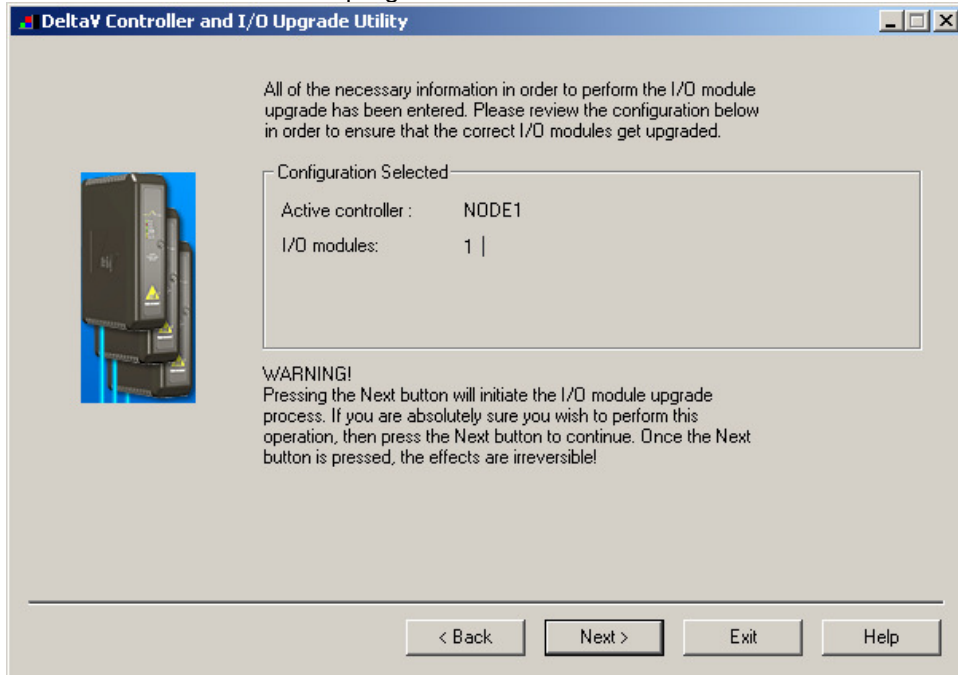




8. After selecting the .S2F file, Click on Open. This dialog will close and you will be back to the following:

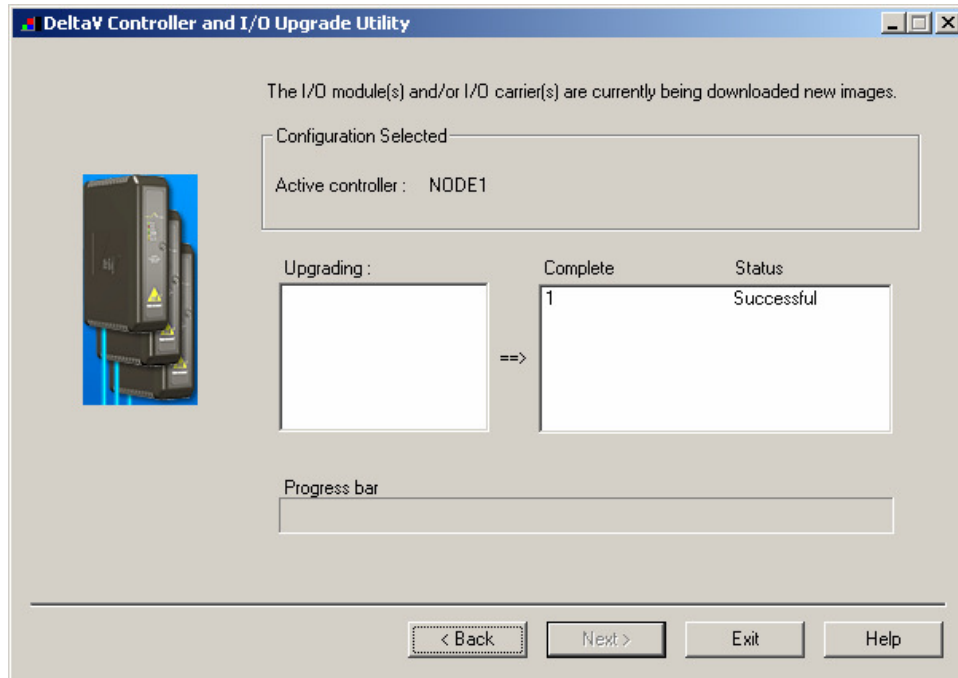


9. In this dialog, Click Next again. You will get the following dialog, confirming the Controller and I/O Module to program.





10. Click Next and the I/O Module upgrade process will begin. After completion, you will receive the following dialog, indicating success.



11. This completes the I/O Module upgrade process.



4 CONFIGURATION INFORMATION

This section describes the steps necessary to configure the DeltaV PSIC and the Mettler Toledo Weigh Scale to obtain proper communication.

4.1 Device and Dataset Configuration

The following paragraphs discuss some attributes in the device and dataset configuration:

4.1.1 Device Address

In Master mode, you can configure a maximum of 8 devices, each representing a weigh scale connected to a PSIC port. Note that you can multi-drop these 8 devices only if this is supported by the devices. Furthermore, if the multi-dropped devices are directly attached to the PSIC, then you must use RS-422/485. Alternatively, you can use RS-232 only if an intermediate converter device is being used which converts RS-232 to RS-422/485, or RS-232 to Fiber Optic.

Each device configured will by definition use 2 datasets. The device address attribute is located in the device configuration box under port. The device address is used as part of the commands sent out to the weigh scales. Under user configuration control, the device address for the weigh scales may be a dataset register value. In this case, the commands will use the dynamically specified address in communications.

In Master mode, communications can be Toledo, Masstron, Multi-Continuous 1, or Multi-Continuous 2.

In Slave mode, configure only one device. The device address is not used internally by the PSIC driver, but can be set to match the weigh scale address.

4.1.2 Output Mode

Two output modes are available in the DeltaV PSIC: block output (0) and single output (1). This value is not used by the driver. Leave this value at its default setting.

4.1.3 DeltaV Data Type

Each weigh scale uses 2 datasets. Dataset 1 must use DeltaV data type Floating point with status. Dataset 2 may be configured as DeltaV data type 8 bit uint w/ status or 16 bit uint w/ status. There are no other datasets defined for a device.



4.1.4 DeviceDataType

In Slave mode configure the Device Data Type as 0 for both datasets.

In Master mode, the Device Data Type value in dataset 1 determines the communication method used. Configure the value as follows:

Table 3 - Device Data Types

Device Data Type	Mettler Toledo Device Type
0	Toledo format for Puma and Cougar type devices
1	Masstron format for Puma and Cougar type devices
2	8142 format for Lynx type devices
3	Multi-Continuous 1 format
4	Multi-Continuous 2 format

In all modes, configure the Device Data Type as 0 for the second dataset.

4.1.5 Data Start Address and Number of Values

In Slave mode use the following values for the dataset data start address:

Table 4 - Dataset Properties for Slave Mode

Port	Device	Dataset	Start Address	Values
1	1	1	0	4
1	1	2	50	7
2	1	1	0	4
2	1	2	50	7



In Master mode use the following values for the dataset data start address:

Table 5 - Dataset Properties for Toledo, Masstron, and 8142 modes

Device Data Type	Device	Dataset	Start Address	Values
0	1 - 8	1	0	21
0	1	2	0	22
1	1 - 8	1	0	21
0	1	2	0	22
2	1 - 8	1	0	21
0	1	2	0	22

Table 6 - Dataset Properties for Multi-Continuous 1 and 2 mode

Device Data Type	Device	Dataset	Start Address	Values
3	1	1	0	16
0	1	2	0	40
4	1	1	0	16
0	1	2	0	40



4.1.6 Special Data 1-5

Under the Special data tab, only the Special data 1 field is used. It is a bit mask with values as follows:

Field Value	Bit	Usage
1	0	If this bit is set, the driver will perform checksum verification on received messages. You can disable or enable checksum generation in the Weigh Scale unit. Note: If the Weigh Scale unit is generating checksums, this bit must be set.
2	1	If this bit is set, the driver calculates the Gross or Net values based on data in the message. For example, if the message weight data has Net and Tare, the driver calculates Gross. Or if the message weight data has Gross and Tare, the driver calculates Net. If this bit is clear, the driver only receives the indicated Gross or Net values. Note: This field is used only in Slave mode.
4	2	If this bit is set, then this device does not use the configured device address. Instead, the user selects and writes the device number to be used into dataset 2, register 15.
8	3	This bit disables the command echo checking. If the weigh scale communication port is set to Loop Through mode then this bit must be 0. If the communication port is set to Star configuration, then this bit must be 1. Please refer to Mettler Toledo Host Mode Serial Interface configuration documentation for additional information.
16	4	This bit must be a 1 if a bypassed scale should be displayed in error. If this bit is 0, the bypassed scale will not show an error.
32	5	This bit is set if the driver should perform weight data scaling as dictated by the Scale Status bytes in Master or Slave Mode. If the bit is 0, no scaling is done. Note: Previous versions of this driver performed scaling in Slave mode all the time. This flag now allows users to select if this function is required. If your existing DeltaV configuration relies on scaled data, you will need to modify your dataset configuration and set this flag.
64	6	This bit is set if the weigh scale is in Loop Through mode, i.e., bit 3 above is 0. Setting this bit allows the driver to do message echo handling.



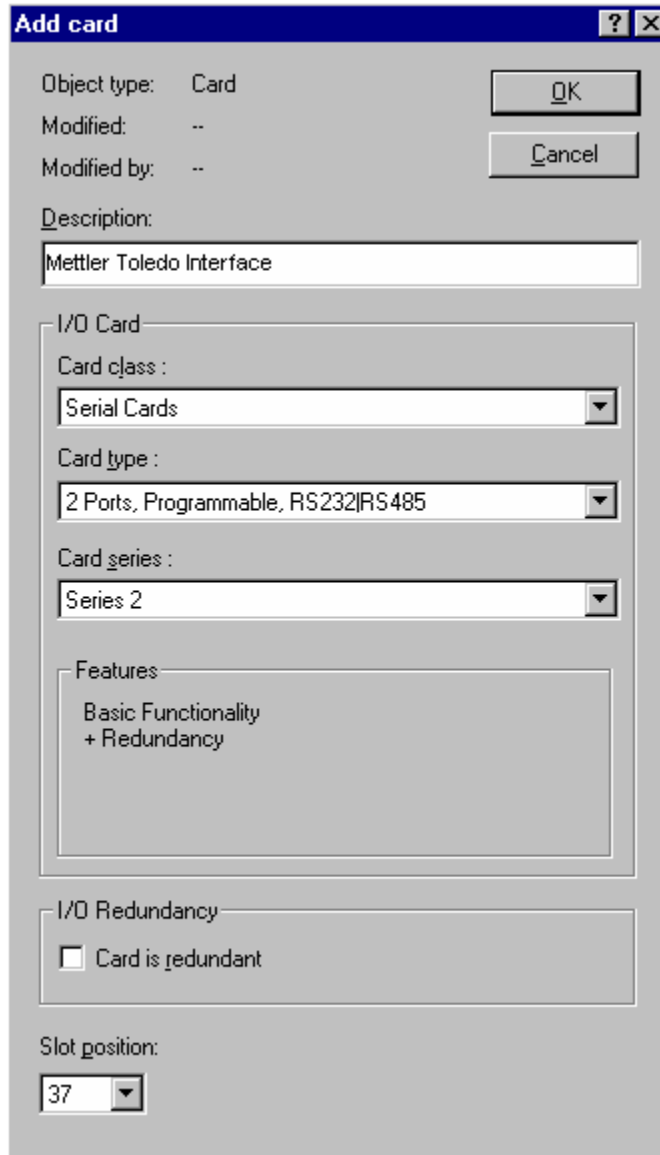
128	7	This bit is set if the weigh scale uses an ASCII encoded scale ID when in Multi Continuous 1 mode. If the bit is 0, numeric ID's are used.
-----	---	--

Add the bit values to get multiple options. For example, a value of 7 indicates that the first 3 bits are set.

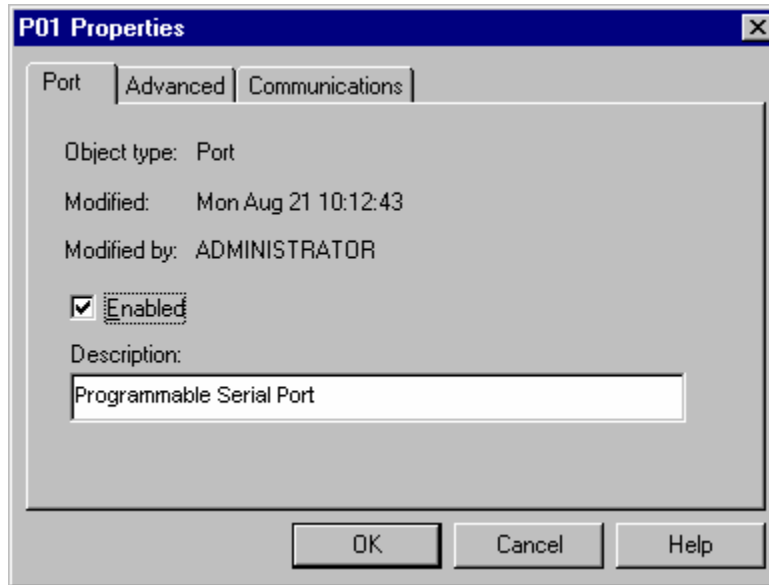
4.2 Dataset Configuration Display

To have the Programmable Serial Card communicate with the Weigh Scale, follow these steps:

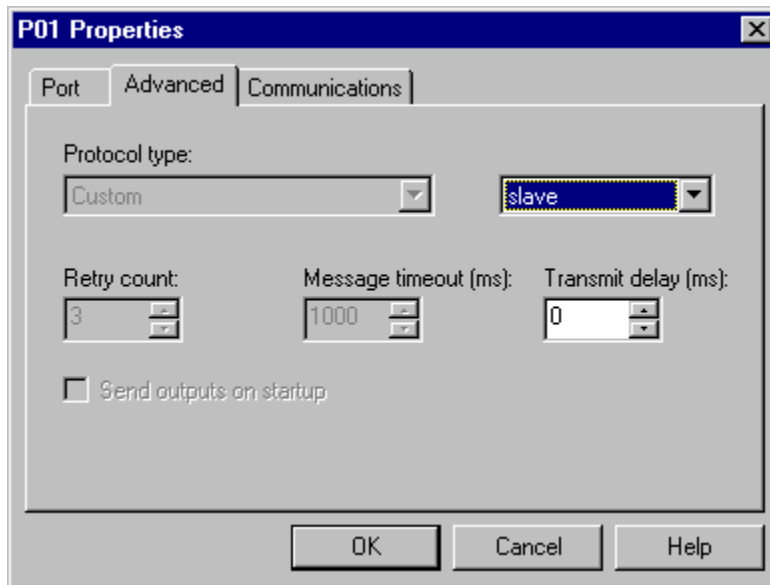
1. In DeltaV, configure the serial card. This will create a Programmable Serial Card and define 2 ports under it, P01 and P02. Do not select I/O Redundancy.



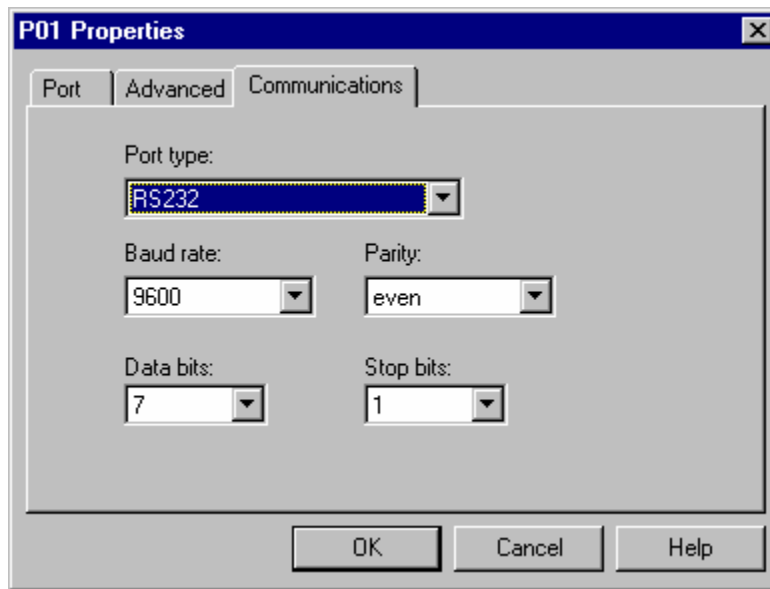
2. Right mouse click on Port 1. The following dialog will appear.



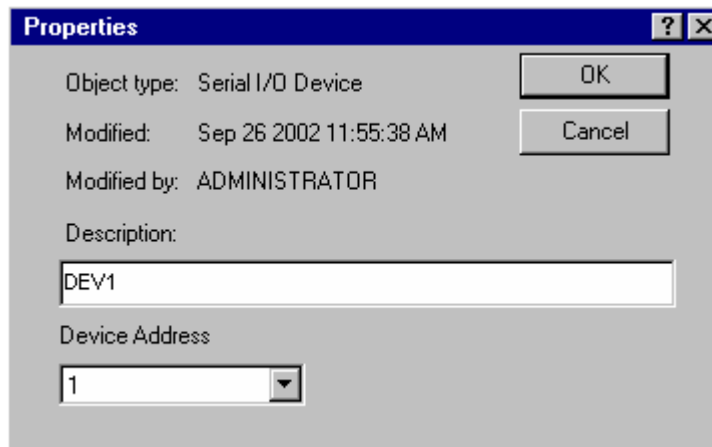
3. Click on the Enabled checkbox to enable the Port. Next select the Advanced tab. The following dialog will appear.



4. In this dialog, select communications parameters as shown. The DeltaV Serial port will be Master or Slave. In this example, the port is configured as Slave. Next click the Communications tab. The following dialog will appear.



5. Specify the Port type. The Port type will be RS-232 or RS-422/485 depending on the Weigh Scale. If scales are being multi-dropped directly, you must use RS-422/485. The Baud Rate, Parity, Data bits and Stop bits parameters must match the Weigh Scale configuration. Select the appropriate parameters and then click OK
6. Configure a Serial Device under the Port by doing a Right Mouse click and selecting New Serial Device. The following dialog will appear:



7. Specify the device address and description. In Slave mode, the device address is not used by the driver but can be defined as the Weigh Scale address. In Master mode, the device address uniquely identifies each scale connected to a port. Then click OK.



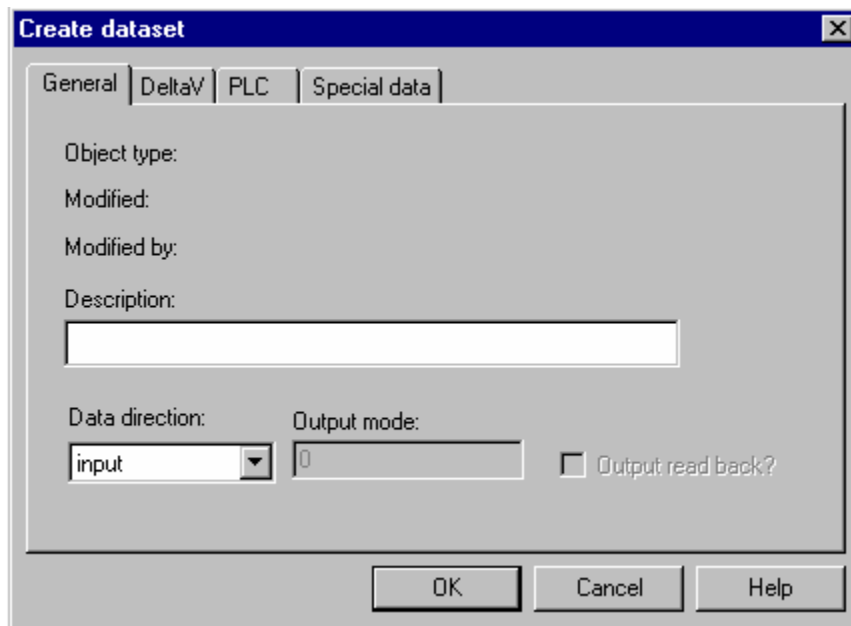
This will add the serial device. Only one device per port is supported.

8. Next, configure datasets in the Serial Device. For this application, each device must have 2 datasets under it. In Slave mode, both datasets will be of type Input. Dataset 1 will have 4 values of type Floating point with status. Dataset 2 will have 7 values of type 8 bit Unsigned Int with status.

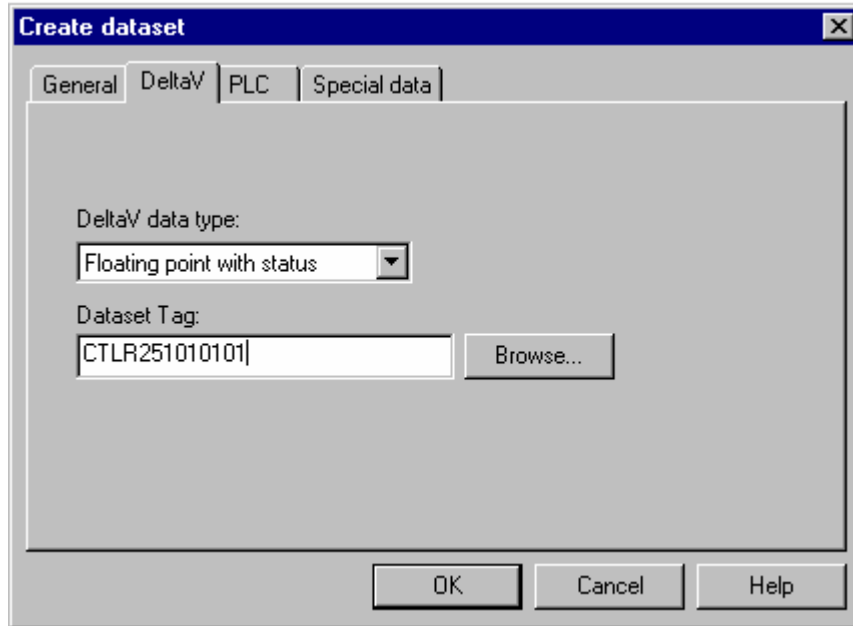
In Master mode (Toledo, Masstron, and 8142 format), both datasets will be of type Output. Dataset 1 will have 21 values of type Floating point with status. Dataset 2 will have 22 values of type 8 bit Unsigned Int with status or 16 bit Unsigned Int with status.

When in Multi-Continuous 1 or 2 master mode, Dataset 1 will have a direction of Input and have 16 values. Dataset 2 will be Output with 40 values.

To add a new dataset, right mouse click on the Serial Device and select New dataset. The following dialog will appear.

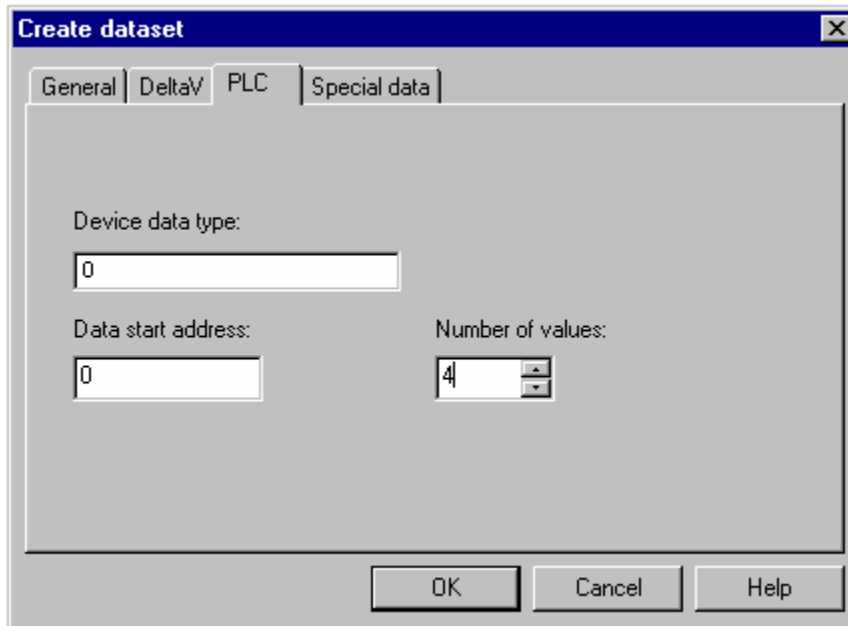


9. Configure the data direction to be input. Next, click on the DeltaV tab. The following dialog will appear.



10. In this dialog, configure the data type needed for DeltaV. You can see the available types by clicking on the drop down list. Remember for this application, the dataset data type is Floating point.

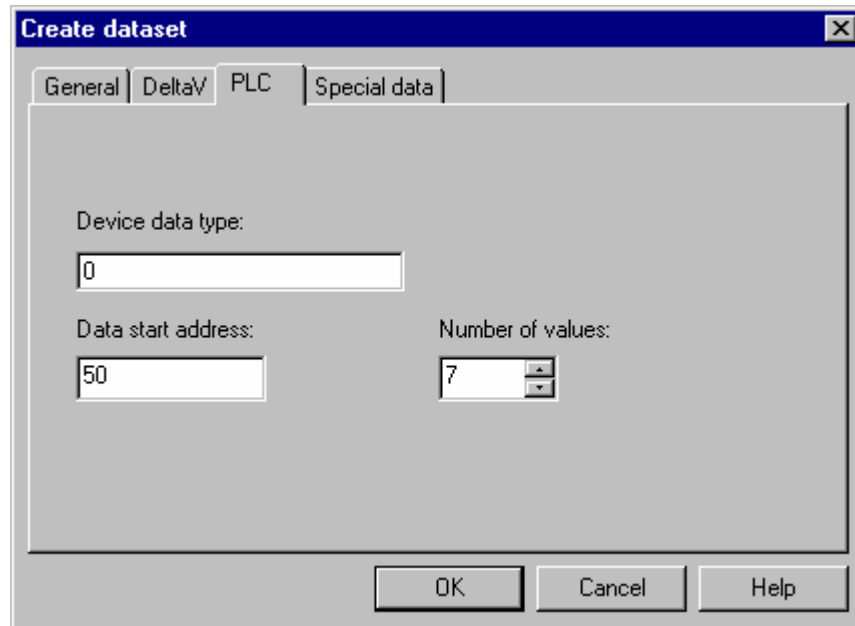
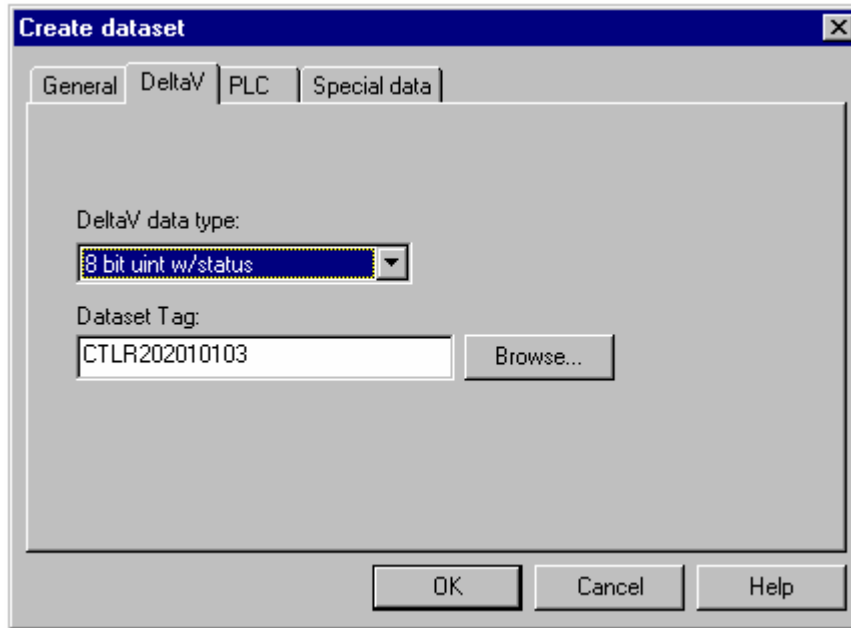
11. Next click the PLC tab. The following dialog will appear.



Select the parameters as shown above for Slave mode. In Master mode, the Number of values will be 21 or 16. The Device Data Type will be 0 for Toledo, 1 for Masstron, 2 for 8142, 3 for Multi-Continuous 1, or 4 for Multi-Continuous 2 mode.



For Dataset 2, select the parameters as shown below.



Configure the values as shown above for Slave mode. In Master mode, the Data start address will always be 0, and the Number of values will be 22 or 40. Device Data Type will be 0 for all modes.



4.3 Serial Driver Communications

4.3.1 Slave Mode

The driver will continuously communicate with the Weigh Scales. This is known as the “Continuous” Mode of the Weigh Scales (or Toledo Continuous).

Note: While setting up the Weigh Scale unit, select Continuous Mode output to occur at a user defined frequency. For periodic outputs from the Weigh Scale, the Serial card software has been tested to handle messages with a minimum interval of 10ms. Typically however, a message interval of 1 second provides sufficient throughput. Do not select Continuous Output with a frequency of A/D Sync.

For this application, the Serial Card does not send any commands to the Weigh Scale. The Serial Card simply receives the data packets, parses them out and makes the data available to DeltaV in the configured registers.

If there are communication errors, these are reported up to DeltaV. Errors are of two types:

1. Physical communication problems;
2. Status information received from the Weigh Scale.

Each weigh scale will have the registers configured as shown in Table 7.



Table 7 - Dataset Register Assignments for Slave Mode

Data Name	Data Type	Assigned Register
Gross Weight	Floating Point	Dataset1-R1
Net Weight	Floating Point	Dataset1-R2
Tare Weight	Floating Point	Dataset1-R3
Unused	Unused	Dataset1-R4
Gross Weight Units	Unsigned Integer 8	Dataset2-R1 1 – LBS 2 – KGS
Net Weight Units	Unsigned Integer 8	Dataset2-R2 1 – LBS 2 – KGS
Tare Weight Status	Unsigned Integer 8	Dataset2-R3 Not Used
Status SB1	Unsigned Integer 8	Dataset2-R4 See Table 8
Status SB2	Unsigned Integer 8	Dataset2-R5 See Table 11
Status SB3	Unsigned Integer 8	Dataset2-R6 See Table 12
Error Code	Unsigned Integer 8	Dataset2-R7 See Table 14



The Status byte SB1 will be a bit mask defined as follows:

Table 8 - Status Byte 1

Status Bit	Value	Description
0	1	Position of Decimal Point (Table 9)
1	2	Position of Decimal Point (Table 9)
2	4	Position of Decimal Point (Table 9)
3	8	Rounding (Table 10)
4	16	Rounding (Table 10)
5	32	Not used
6	64	Print Demand
7	128	Not Used

Position of Decimal Point will be determined as follows. The driver will move the decimal point as indicated.

Table 9 - Position of Decimal Point (SB1)

Bit Position			Indicated Position
2	1	0	
0	0	0	XXXX00.
0	0	1	XXXXX0.
0	1	0	XXXXXX.
0	1	1	XXXXX.X
1	0	0	XXXX.XX
1	0	1	XXX.XXX
1	1	0	XX.XXXX



Rounding of the weight (as described below) is strictly for status. The driver will not perform this function. It is assumed that the weigh scale has already rounded the values.

Table 10 - Rounding (SB1)

Bit Position		Rounding
4	3	
0	1	1
1	0	2
1	1	5

The Status byte SB2 will be a bit mask defined as follows:

Table 11 - Status Byte 2

Status Bit	Value	Description
0	1	1=Net 0=Gross
1	2	Sign 1=Negative 0=Positive
2	4	1=Out of Range
3	8	1=Motion
4	16	0=LB 1=KG
5	32	Always=1
6	64	1=In Power Up
7	128	Not Used



The Status byte SB3 will be a bit mask defined as follows:

Table 12 - Status Byte 3

Status Bit	Value	Description
0	1	Units (Table 13)
1	2	Units (Table 13)
2	4	Units (Table 13)
3	8	1=Print Request
4	16	1=Expand Data X 10
5	32	Always=1
6	64	Hand Tare (Metric Only)
7	128	Not Used

Units will be determined as follows.

Table 13 - Units (SB3)

Bit Position			Indicated Units
2	1	0	
0	0	0	LB or KG selected by SB2-bit 4
0	0	1	Grams
0	1	0	Metric Tons
0	1	1	Ounces
1	0	0	Troy Ounces
1	0	1	Penny Weight
1	1	0	Tons
1	1	1	Custom Units



The Error Code byte represents errors detected in the received messages. Most of these errors could be a result of mismatched communication parameters. These are defined as follows:

Table 14 - Dataset Error Codes

Error Code	Description
101	Invalid start of message detected. For continuous mode (RS-232), this should be a <STX>. Instead some other character was received.
102	Invalid or incomplete message received.
103	Invalid checksum received.



4.3.2 Master Mode

The driver will continuously send commands to read the weight and status from the Weigh Scales. The weight data read is for Net, Gross, Tare and Display. If there are communication errors, these are reported up to DeltaV. Each weigh scale will always have only the following registers configured:

Table 15 - Dataset 1 Register Assignments

Data Name	Assigned Register	Type
Net Weight	R1	Read from Scale
Gross Weight	R2	Read from Scale
Tare Weight	R3	Read from Scale
Over Capacity Weight	R4	Read from Scale
Hand Tare Weight	R5	Read from Scale
Under Capacity Weight	R6	Read from Scale
Unused	R7-R10	
Tare Value	R11	Written to Scale
SP1	R12	Read/Write
SP2	R13	Read/Write
SP3	R14	Read/Write
SP4	R15	Read/Write
SP1 Preact/Dribble	R16	Read/Write
SP2 Preact/Dribble	R17	Read/Write
SP3 Preact/Dribble	R18	Read/Write
SP4 Preact/Dribble	R19	Read/Write
SP1 Tolerance 1	R20	Read/Write
SP2 Tolerance 2	R21	Read/Write



Table 16 - Dataset 2 Register Assignments

Data Name	Assigned Register	Type
Net Units	R1	Read from Scale
Gross Units	R2	Read from Scale
Tare Units	R3	Read from Scale
Over Capacity Units	R4	Read from Scale
Hare Tare Units	R5	Read from Scale
Under Capacity Units	R6	Read from Scale
Status Byte 1	R7	Read from Scale
Status Byte 2	R8	Read from Scale
Status Byte 3	R9	Read from Scale
Status Byte 4	R10	Read from Scale
Status Byte 5	R11	Read from Scale
Status Byte 6	R12	Read from Scale
Scale Online Status	R13	0 – Scale is Offline 1 – Scale is Online
Scale Bypass	R14	User Supplied
Device ID	R15	User supplied
Unused	R16-R19	
Command	R20	Written to Scale
Command Parameter	R21	Written to Scale
Command Status	R22	Internal



The following Toledo commands are supported.

Table 17 - Supported Toledo Commands

Command	Description	Parameters	Additional Information
5	Auto Tare	0	
6	Keyboard Tare	1	Tare Value (DS1, R11) is sent out
7	Digital Zero	0	
10	SP Read	1	SP # in command parameter register (DS2, R21) to read. SP # must be between 0 and 9 (Table 18)
20	SP Write	2	SP # in command parameter register (DS2, R21) to written. SP # must be between 0 and 9. SP value is in RS1, R12-R21 (Table 18)
30	Mode	0	Set Mode to Net
31	Mode	0	Set Mode to KG
32	Mode	0	*Set Mode to Net
33	Mode	0	*Set Mode to Gross
40	KB Lock	0	Lock Keyboard of scale
41	KB Unlock	0	Unlock Keyboard of scale



Table 18 - SP Number

SP #	Description
0	SP 1
1	SP 2
2	SP 3
3	SP 4
4	SP 1 Preact/Dribble
5	SP 2 Preact/Dribble
6	SP 3 Preact/Dribble
7	SP 4 Preact/Dribble
8	SP 1 Tolerance 1
9	SP 2 Tolerance 2

*Some scales do not support these Modes. In these cases, the scale responds with a Nak.

The following Masstron commands are supported.

Table 19 - Supported Masstron Commands

Command	Description	Additional Information
1	Clear	Clear to gross Mode
2	Tare	Tare off the display weight
3	Zero	Zero the scales



The Status byte SB1 will be a bit mask defined as follows:

Table 20 - Status Byte 1

Status Bit	Value	Description
0	1	Position of Decimal Point (Table 21)
1	2	Position of Decimal Point (Table 21)
2	4	Position of Decimal Point (Table 21)
3	8	Rounding (Table 22)
4	16	Rounding (Table 22)

Table 21 – Decimal Point Position (SB1)

Bit Position			Indicated Position
2	1	0	
0	0	0	XXXX00.
0	0	1	XXXXX0.
0	1	0	XXXXXX.
0	1	1	XXXXX.X
1	0	0	XXXX.XX
1	0	1	XXX.XXX
1	1	0	XX.XXXX

Rounding of the weight (as described below) is strictly for status. The driver will not perform this function. It is assumed that the weigh scale has already rounded the values.

Table 22 - Rounding (SB1)

Bit Position		Rounding
4	3	
0	1	1
1	0	2
1	1	5



The Status byte SB2 will be a bit mask defined as follows:

Table 23 - Status Byte 2

Status Bit	Value	Description
0	1	1=Net; 0=Gross
1	2	1=LB; 0=KG/Alt
2	4	1=Alternate units programmed for non zero value
3	8	Always 0
4	16	Always 0
5	32	Always 0
6	64	1=Tare Enabled; 0=Tare disabled
7	128	Parity bit

The Status byte SB3 will be a bit mask defined as follows:

Table 24 - Status Byte 3

Status Bit	Value	Description
0	1	Always 0
1	2	Always 0
2	4	1=Motion
3	8	1=Center of Zero
4	16	Always 0
5	32	1=Over Capacity
6	64	1=LB/KG switching enabled
7	128	Parity bit



The Status byte SB4 will be a bit mask defined as follows:

Table 25 - Status Byte 4

Status Bit	Value	Description
0	1	1=Negative Weight
1	2	Always 0
2	4	1=Keyboard Tare entered
3	8	Always 0
4	16	Always 0
5	32	1=Keyboard locked out
6	64	1=Expanded mode Enabled
7	128	Parity bit

The Status byte SB5 will be a bit mask defined as follows:

Table 26 - Status Byte 5

Status Bit	Value	Description
0	1	Always 0
1	2	Always 0
2	4	Always 0
3	8	1=Power up
4	16	1=SP enabled
5	32	Always 0
6	64	Always 0
7	128	Parity bit



The Status byte SB6 will be a bit mask defined as follows:

Table 27 - Status Byte 6

Status Bit	Value	Description
0	1	0=SP1 feed on
1	2	0=SP2 feed on
2	4	0=SP3/SP1 fast feed on
3	8	0=SP4/SP2 fast feed on
4	16	0=SP1/Zero Tolerance 1 feed on
5	32	0=SP2/Zero Tolerance 2 feed on
6	64	Always 0
7	128	Parity bit



4.3.3 Multi-Continuous 1 Mode

In this mode the driver will read Multi-Continuous 1 messages from the device and have the ability to send CTPZ commands (Clear, Tare, and Zero only) to individual scales. To enter this mode the port must be in master mode and Dataset 1 must have a device data type equal to 3. In addition to Dataset 1, Dataset 2 must be in output mode in order to send commands.

Table 28 - Dataset 1 Register Map (Multi-Continuous 1)

Data Name	Data Type	Assigned Register	Scale ID
Gross Weight	Floating Point	R1	1
Net Weight	Floating Point	R2	1
Tare Weight	Floating Point	R3	1
Unused	Unused	R4	
Gross Weight	Floating Point	R5	2
Net Weight	Floating Point	R6	2
Tare Weight	Floating Point	R7	2
Unused	Unused	R8	
Gross Weight	Floating Point	R9	3
Net Weight	Floating Point	R10	3
Tare Weight	Floating Point	R11	3
Unused	Unused	R12	
Gross Weight	Floating Point	R13	4
Net Weight	Floating Point	R14	4
Tare Weight	Floating Point	R15	4
Unused	Unused	R16	



Table 29 - Dataset 2 Register Mapping (Multi-Continuous 1)

Data Name	Data Type	Assigned Register	Scale ID
Gross Weight Units	Unsigned Integer 8	R1 1 – LBS 2 – KGS	1
Net Weight Units	Unsigned Integer 8	R2 1 – LBS 2 – KGS	1
Tare Weight Status	Unsigned Integer 8	R3 Not Used	1
Status SB1	Unsigned Integer 8	R4 See Table 30	1
Status SB2	Unsigned Integer 8	R5 See Table 33	1
Status SB3	Unsigned Integer 8	R6 See Table 34	1
Error Code	Unsigned Integer 8	R7 See Table 36	1
Unused	Unused	R8	
Command	Unsigned Integer 8	R9 See Table 37	1
Command Status	Unsigned Integer 8	R10 0 – Command OK >0 - Error	1
Gross Weight Units	Unsigned Integer 8	R11 1 – LBS 2 – KGS	2
Net Weight Units	Unsigned Integer 8	R12 1 – LBS 2 – KGS	2
Tare Weight Status	Unsigned Integer 8	R13 Not Used	2
Status SB1	Unsigned Integer 8	R14 See Table 30	2



Status SB2	Unsigned Integer 8	R15 See Table 33	2
Status SB3	Unsigned Integer 8	R16 See Table 34	2
Error Code	Unsigned Integer 8	R17 See Table 36	2
Unused	Unused	R18	2
Command	Unsigned Integer 8	R19 See Table 37	2
Command Status	Unsigned Integer 8	R20 0 – Command OK >0 - Error	2
Gross Weight Units	Unsigned Integer 8	R21 1 – LBS 2 – KGS	3
Net Weight Units	Unsigned Integer 8	R22 1 – LBS 2 – KGS	3
Tare Weight Status	Unsigned Integer 8	R23 Not Used	3
Status SB1	Unsigned Integer 8	R24 See Table 30	3
Status SB2	Unsigned Integer 8	R25 See Table 34	3
Status SB3	Unsigned Integer 8	R26 See Table 34	3
Error Code	Unsigned Integer 8	R27 See Table 36	3
Unused	Unused	R28	3
Command	Unsigned Integer 8	R29 See Table 37	3
Command Status	Unsigned Integer 8	R30 0 – Command OK >0 - Error	3



Gross Weight Units	Unsigned Integer 8	R31 1 – LBS 2 – KGS	4
Net Weight Units	Unsigned Integer 8	R32 1 – LBS 2 – KGS	4
Tare Weight Status	Unsigned Integer 8	R33 Not Used	4
Status SB1	Unsigned Integer 8	R34 See Table 30	4
Status SB2	Unsigned Integer 8	R35 See Table 34	4
Status SB3	Unsigned Integer 8	R36 See Table 34	4
Error Code	Unsigned Integer 8	R37 See Table 36	4
Unused	Unused	R38	4
Command	Unsigned Integer 8	R39 See Table 37	4
Command Status	Unsigned Integer 8	R40 0 – Command OK >0 - Error	4



The Status byte SB1 will be a bit mask defined as follows:

Table 30 - Status Byte 1 (Multi-Cont 1)

Status Bit	Value	Description
0	1	Position of Decimal Point (Table 31)
1	2	Position of Decimal Point (Table 31)
2	4	Position of Decimal Point (Table 31)
3	8	Rounding (Table 32)
4	16	Rounding (Table 32)
5	32	Not used
6	64	Print Demand
7	128	Not Used

Position of Decimal Point will be determined as follows. The driver will move the decimal point as indicated.

Table 31 - Decimal Point Position (SB1)

Bit Position			Indicated Position
2	1	0	
0	0	0	XXXX00.
0	0	1	XXXXX0.
0	1	0	XXXXXX.
0	1	1	XXXXX.X
1	0	0	XXXX.XX
1	0	1	XXX.XXX
1	1	0	XX.XXXX



Rounding of the weight (as described below) is strictly for status. The driver will not perform this function. It is assumed that the weigh scale has already rounded the values.

Table 32 - Rounding (SB1)

Bit Position		Rounding
4	3	
0	1	1
1	0	2
1	1	5

The Status byte SB2 will be a bit mask defined as follows:

Table 33 - Status Byte 2 (Multi-Cont 1)

Status Bit	Value	Description
0	1	Type 1=Net 0=Gross
1	2	Sign 1=Negative 0=Positive
2	4	1=Out of Range
3	8	1=Motion
4	16	Unit 0=LB 1=KG
5	32	Always=1
6	64	1=In Power Up
7	128	Not Used



The Status byte SB3 will be a bit mask defined as follows:

Table 34 - Status Byte 3 (Multi-Cont 1)

Status Bit	Value	Description
0	1	Units (Table 35)
1	2	Units (Table 35)
2	4	Units (Table 35)
3	8	1=Print Request
4	16	1=Expand Data X 10
5	32	Always=1
6	64	Hand Tare (Metric Only)
7	128	Not Used

Units will be determined as follows.

Table 35 – Units (SB3)

Bit Position			Indicated Units
2	1	0	
0	0	0	LB or KG selected by SB2-bit 4
0	0	1	Grams
0	1	0	Metric Tons
0	1	1	Ounces
1	0	0	Troy Ounces
1	0	1	Penny Weight
1	1	0	Tons
1	1	1	Custom Units



The Error Code byte represents errors detected in the received messages. Most of these errors could be a result of mismatched communication parameters. These are defined as follows:

Table 36 - Dataset Error Codes (Multi-Cont 1)

Error Code	Description
101	Invalid start of message detected. For continuous mode (RS-232), this should be a <STX>. Instead some other character was received.
102	Invalid or incomplete message received.
103	Invalid checksum received.

The Command Register is used to send the following commands to a scale. The driver uses the register address to send the message to the correct scale.

Table 37 - Supported Commands (Multi-Cont 1)

Command Number	Description	Additional Information
1	Clear	Clear to gross Mode
2	Tare	Tare off the display weight
3	Zero	Zero the scales



4.3.4 Multi-Continuous 2 Mode

In this mode the driver will read Multi-Continuous 2 messages from the device and have the ability to send CTPZ commands (Clear, Tare, and Zero only) to individual scales. To enter this mode the port must be in master mode and Dataset 1 must have a device data type equal to 4. In addition to Dataset 1, Dataset 2 must be in output mode in order to send commands.

Table 38 - Dataset 1 Register Map (Multi-Continuous 2)

Data Name	Data Type	Assigned Register	Scale ID
Gross Weight	Floating Point	R1	1
Net Weight	Floating Point	R2	1
Tare Weight	Floating Point	R3	1
Unused	Unused	R4	
Gross Weight	Floating Point	R5	2
Net Weight	Floating Point	R6	2
Tare Weight	Floating Point	R7	2
Unused	Unused	R8	
Gross Weight	Floating Point	R9	3
Net Weight	Floating Point	R10	3
Tare Weight	Floating Point	R11	3
Unused	Unused	R12	
Gross Weight	Floating Point	R13	4
Net Weight	Floating Point	R14	4
Tare Weight	Floating Point	R15	4
Unused	Unused	R16	



Table 39 - Dataset 2 Register Mapping (Multi-Continuous 2)

Data Name	Data Type	Assigned Register	Scale ID
Gross Weight Units	Unsigned Integer 8	R1 1 – LBS 2 – KGS	1
Net Weight Units	Unsigned Integer 8	R2 1 – LBS 2 – KGS	1
Tare Weight Status	Unsigned Integer 8	R3 Not Used	1
Status SB1	Unsigned Integer 8	R4 See Table 40	1
Status SB2	Unsigned Integer 8	R5 See Table 43	1
Status SB3	Unsigned Integer 8	R6 See Table 44	1
Error Code	Unsigned Integer 8	R7 See Table 46	1
Unused	Unused	R8	
Command	Unsigned Integer 8	R9 See Table 47	1
Command Status	Unsigned Integer 8	R10 0 – Command OK >0 - Error	1
Gross Weight Units	Unsigned Integer 8	R11 1 – LBS 2 – KGS	2
Net Weight Units	Unsigned Integer 8	R12 1 – LBS 2 – KGS	2
Tare Weight Status	Unsigned Integer 8	R13 Not Used	2
Status SB1	Unsigned Integer 8	R14 See Table 40	2



Status SB2	Unsigned Integer 8	R15 See Table 43	2
Status SB3	Unsigned Integer 8	R16 See Table 44	2
Error Code	Unsigned Integer 8	R17 See Table 46	2
Unused	Unused	R18	2
Command	Unsigned Integer 8	R19 See Table 47	2
Command Status	Unsigned Integer 8	R20 0 – Command OK >0 - Error	2
Gross Weight Units	Unsigned Integer 8	R21 1 – LBS 2 – KGS	3
Net Weight Units	Unsigned Integer 8	R22 1 – LBS 2 – KGS	3
Tare Weight Status	Unsigned Integer 8	R23 Not Used	3
Status SB1	Unsigned Integer 8	R24 See Table 40	3
Status SB2	Unsigned Integer 8	R25 See Table 43	3
Status SB3	Unsigned Integer 8	R26 See Table 44	3
Error Code	Unsigned Integer 8	R27 See Table 46	3
Unused	Unused	R28	3
Command	Unsigned Integer 8	R29 See Table 47	3
Command Status	Unsigned Integer 8	R30 0 – Command OK >0 – Error	3



Gross Weight Units	Unsigned Integer 8	R31 1 – LBS 2 – KGS	4
Net Weight Units	Unsigned Integer 8	R32 1 – LBS 2 – KGS	4
Tare Weight Status	Unsigned Integer 8	R33 Not Used	4
Status SB1	Unsigned Integer 8	R34 See Table 40	4
Status SB2	Unsigned Integer 8	R35 See Table 43	4
Status SB3	Unsigned Integer 8	R36 See Table 44	4
Error Code	Unsigned Integer 8	R37 See Table 46	4
Unused	Unused	R38	4
Command	Unsigned Integer 8	R39 See Table 47	4
Command Status	Unsigned Integer 8	R40 0 – Command OK >0 - Error	4



The Status byte SB1 will be a bit mask defined as follows:

Table 40 - Status Byte 1 (Multi-Cont 2)

Status Bit	Value	Description
0	1	Position of Decimal Point (Table 41)
1	2	Position of Decimal Point (Table 41)
2	4	Position of Decimal Point (Table 41)
3	8	Rounding (Table 42)
4	16	Rounding (Table 42)
5	32	Not used
6	64	Print Demand
7	128	Not Used

Position of Decimal Point will be determined as follows. The driver will move the decimal point as indicated.

Table 41 - Decimal Point Position (SB1)

Bit Position			Indicated Position
2	1	0	
0	0	0	XXXX00.
0	0	1	XXXXX0.
0	1	0	XXXXXX.
0	1	1	XXXXX.X
1	0	0	XXXX.XX
1	0	1	XXX.XXX
1	1	0	XX.XXXX



Rounding of the weight (as described below) is strictly for status. The driver will not perform this function. It is assumed that the weigh scale has already rounded the values.

Table 42 - Rounding (SB1)

Bit Position		Rounding
4	3	
0	1	1
1	0	2
1	1	5

The Status byte SB2 will be a bit mask defined as follows:

Table 43 - Status Byte 2 (Multi-Cont 2)

Status Bit	Value	Description
0	1	Type 1=Net 0=Gross
1	2	Sign 1=Negative 0=Positive
2	4	1=Out of Range
3	8	1=Motion
4	16	Unit 0=LB 1=KG
5	32	Always=1
6	64	1=In Power Up
7	128	Not Used



The Status byte SB3 will be a bit mask defined as follows:

Table 44 - Status Byte 3 (Multi-Cont 2)

Status Bit	Value	Description
0	1	Scale ID (Table 35)
1	2	Scale ID (Table 35)
2	4	Scale ID (Table 35)
3	8	1=Print Request
4	16	1=Expand Data X 10
5	32	Always=1
6	64	Hand Tare (Metric Only)
7	128	Not Used

Units will be determined as follows.

Table 45 – Scale ID (SB3)

Bit Position			Indicated Scale ID
2	1	0	
0	0	0	Unused
0	0	1	Scale A
0	1	0	Scale B
0	1	1	Scale C
1	0	0	Scale E (Sum)
1	0	1	Unused
1	1	0	Unused
1	1	1	Unused



The Error Code byte represents errors detected in the received messages. Most of these errors could be a result of mismatched communication parameters. These are defined as follows:

Table 46 - Dataset Error Codes (Multi-Cont 2)

Error Code	Description
101	Invalid start of message detected. For continuous mode (RS-232), this should be a <STX>. Instead some other character was received.
102	Invalid or incomplete message received.
103	Invalid checksum received.

The Command Register is used to send the following commands to a scale. The driver uses the register address to send the message to the correct scale.

Table 47 - Supported Commands (Multi-Cont 2)

Command Number	Description	Additional Information
1	Clear	Clear to gross Mode
2	Tare	Tare off the display weight
3	Zero	Zero the scales



4.4 Steps for User Commands

1. For commands which do not require any parameters, simply write the command number into the Command Register (DS2, R20).
2. For commands which require parameters, write the parameter values first. Then write the command number.
3. The command execution will begin. The driver will determine if the command is valid. If an invalid command is found, the command register will be set to 0 and the command status will have the value 254.
4. The driver will then check for valid parameter numbers. If an invalid parameter number is found, the command register will be set to 0, and the command status will have the value 253.
5. If no errors are found, the driver will format the command and send it to the weigh scale.
6. If the command completes successfully, the command status will be set to 0 and the command register will be set to the command number plus 100.
7. If the scale returns a Nak, the command status will be set to 255.
8. In Master mode, you can bypass the scale scan. This is useful if you hookup temporary scales to the serial card. Such scales are typically also given a user defined ID. Bypassing a scale allows you to physically remove a scale without adversely affecting the scan time for other permanent scales. Set the scale bypass register (R14) of second dataset to 1 for bypass, 0 for normal scan.



5 Operational Check

5.1 Scope

The following sections provide some assistance to ensure the interface is working properly.

5.2 Verify Hardware and Software Version Number

The user can verify that the Mettler Toledo driver has been installed using the DeltaV Diagnostics tool. The Diagnostics tool will show the Hardware Revision No. (HwRev) and the Software Revision No. (SwRev).

To begin the DeltaV Diagnostic tool select Start-> DeltaV-> Operator-> Diagnostics. In the Diagnostics tool expand the Controller, I/O and then double click on the Programmable Serial Interface Card that has the Mettler Toledo driver installed.

The following information will be displayed:

HwRev	Hardware Revision	Rev 1.1	(or later)
SwRev	Software Revision	Rev 1.10	(or later)

5.3 Verify Configuration

- Verify port configuration: The serial port must be enabled. User needs to make sure communication settings such as baud rate, parity, and the number of data bits matches the Mettler Toledo device settings.
- Verify dataset configuration: The datasets configured must be as shown above.

5.4 Verify I/O Communication with Control Studio

User can create I/O modules in the control studio to verify correct values are read from the Mettler Toledo Weigh Scale and the PSIC. For input data, the values should be changed in the Mettler Toledo and verified that the new data are correctly reported.

5.5 Using Diagnostics

- Verify PSIC communication: Select the PSIC on Diagnostics and press the right mouse button. Select Display Real -Time Statistics from the drop down menu. If the Programmable Serial Interface Card is functioning then the user will see the Valid Responses counter and the Async and/or Sync Transactions counters incrementing. There will not be any error counting up.



- Verify port statistics: Select the Port on the Programmable Serial Interface Card and press the right mouse button. Then select Display Port Statistics from the drop down menu. Verify that the port communications statistics are being displayed properly and are counting as expected for the Mettler Toledo protocol's functionality.
- Verify dataset values: Select a dataset and press the right mouse button. Select View Dataset Registers from the Drop down window. Verify that the dataset values are displayed as expected.

5.6 LED Indication

The Yellow LED for the port should be on solid when all communications on that port are valid. The Yellow LED should be blinking if there is some valid communications and some communications with errors on that port. The Yellow LED should be OFF if there are no valid communications on that port.



6 DeltaV – Mettler-Toledo Electrical Interface

The electrical interface between DeltaV and the Mettler Toledo devices conforms to the RS-232 protocol. The RS-232 cable connecting Mettler Toledo and the DeltaV PSIC should not exceed 50 feet as specified by the EIA standard for RS-232 protocol. Section 6.1 shows the pin assignments for the PSIC serial terminal block for RS-232 protocol.

6.1 RS-232 Pin Assignments for DeltaV PSIC

Table 48 - RS-232 Pin Assignments

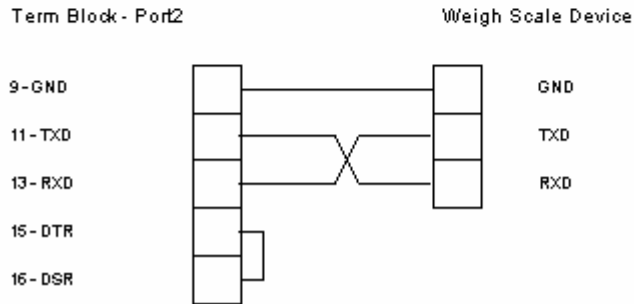
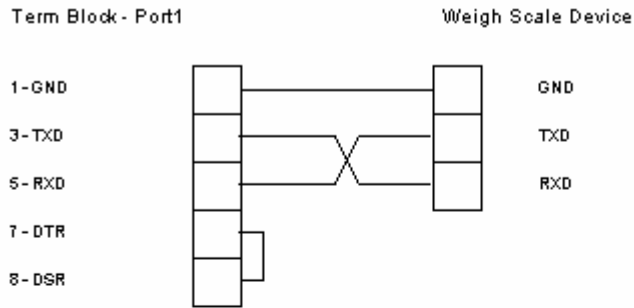
Terminal Number	Signal Description
1	Port 1 - Isolated Ground (GND)
2	Unused
3	Port 1 - Transmit Data (TXD)
4	Unused
5	Port 1 - Receive Data (RXD)
6	Unused
7	Port 1 - Data Terminal Ready (DTR)
8	Port 1 - Dataset Ready (DSR)
9	Port 2 - Isolated Ground (GND)
10	Unused
11	Port 2 - Transmit Data (TXD)
12	Unused
13	Port 2 - Receive Data (RXD)
14	Unused
15	Port 2 - Data Terminal Ready (DTR)
16	Port 2 - Dataset Ready (DSR)



6.2 Wiring Connections for RS-232 Communications

Five terminals need to be connected between the PSIC and the Mettler Toledo port. Pins 3 (TXD) and 5 (RXD) need to be crossed so that the Mettler Toledo TXD is connected to PSIC RXD, and the Mettler Toledo RXD is connected to PSIC TXD. Pins 7 (DTR) and 8 (DSR) also need to be crossed in the same manner between the PSIC and the Mettler Toledo.

In general, the following RS-232 cable pinout can be used.





M Y N A H™

Powerful Solutions for Digital Plants

7 Technical Support

For technical support or to report a defect, please give MYNAH Technologies a call at (636) 681-1555. If a defect is discovered, please document it in as much detail as possible and then fax your report to us at (636) 681-1660.

You can also send us your questions via e-mail. Our address is:

support@mynah.com

Thank you for using DeltaV.