Open Protocol MT Focus 6000

Control and drive unit

Valid from Serial No. -

Instruction Supplement

MT Focus 6000	8432085100	

AtlasCopco

Table of Contents

Introduction

This document specifies all the MT Focus 6000 specific considerations when using the Open Protocol.

MicroTorque follows Atlas Copco Open Protocol 2.0 (where applicable) and the Atlas Copco Open Protocol specification is 2.6.

Communication

Open Protocol on the MT Focus 6000 controller can accept up to one RS232 and two TCP/IP connections simultaneously. The controller keeps a separate set of session data for each connection.

All connections are of type *Classic* as defined by Open Protocol.

Link level acknowledging and message sequence numbering as defined by Open Protocol is not supported at this time. This means that using Open Protocol over RS232 is less reliable than over TCP/IP.

RS232 considerations

Setup

Description	Value
Baud rate	115200 bit/s
Data bit	8
Parity	None
Stop bit	1
Handshake	None

Communication

There is a small communication difference between using Open protocol over RS232 and Ethernet. When using RS232 the protocol expects a 4 ASCII digit tag before the Start-of-Text [$02] and it expects an End-of-Text [$03] at the end. It expects this from all messages via RS232.

Example - Start communication:

(i) Eight spaces between 002000010060 and $00.

Ethernet: 002000010060 $00

RS232: **$07$09$07$09$02**002000010060 $00**$03**

Keep alive

OP via RS232 requires a keep alive signal [MID9999] every 15 seconds.

USB considerations

The MT Focus 6000 controller does not support Open Protocol over USB at this time.

TCP/IP considerations

The MT Focus 6000 controller only supports Open Protocol client connections to port 4545 at this time.

Open Protocol version 2.0 considerations

When sending or receiving messages containing generic PID fields, see notes on each MID for information on which parameter IDs that are used.

The following data node types are defined:

Node Type	Value
Pset	001 (Legacy 301)

The following data types are defined:

Data Type	Value
Integer	02
String	04
Datetime	05
Boolean	06
Scientific float	90

The scientific float datatype is a 12 character string containing the same format as that output from `printf_s("%012.*e", 3, value)`. For example, `00001.234e+2`.

The scientific float datatype is a MicroTorque specific extension of the Open Protocol standard.

The following unit types are defined:

Unit	Value
None	000
Newton meter (Nm)	001
Foot-pound force (ft-lbf)	002
Centi Newton meter (cNm)	003
Kilo Newton meter (kNm)	004 (New since fw v 1.18)
Mega Newton meter (MNm)	005 (New since fw v 1.18)
Inch-pound force (in-lbf)	006
Kilopond meter (kpm)	007 (New since fw v 1.18)
Ounce force inch (ozf-in)	010 (New since fw v 1.18)
Milli Newton meter (mNm)	012 (New since fw v 1.12) (See note on mNm below.)
Kilogram force centimeter (kgf-cm)	013
Gram force centimeter (gf-cm)	014
Ounce force foot (ft-ozf)	015
Degrees (°)	050
Millinewton meter MT (mNm)	090 (See note on mNm below.)
RPM	101
Seconds (s)	200
Degrees Celcius (°C)	251 (See note on Temperature below.)
Degrees Fahrenheit (F)	252 (New since fw v 1.18) (See note on Temperature below.)
Percent (%)	800 (See note on Percent below.)

ⓘ **Note on mNm unit:**

Firmware version 1.16 and earlier only supported 090 for mNm. Newer versions support both 090 (for backwards compatibility) and 012 (for OP compatibility). The controller will accept both unit types as input.

Output is configurable by controller parameter "Open Protocol Torque Unit", e.g. by using ToolsTalk MT. Default is 090 for backward compatibility, but newer implementations are recommended to switch to using 012 for better OP standard compatibility and compliance.

(i) **Note on Temperature unit:**

The temperature unit used by a controller is configurable using controller parameter "Open Protocol Temperature Unit", e.g. by using ToolsTalk MT. Default is 251 (°C).

(i) **Note on Percent unit:**

Changed from 009 to 800 in fw 1.20 and later, due to ACOP standard specification changes.

Special MT Focus 6000 considerations, general

All value strings returned by MT Focus 6000 Open Protocol are encoded as UTF-8. The MT Focus 6000 controller can internally store longer strings than what the Open Protocol supports. So if Open Protocol receives a string that exceeds the supported size, the string is truncated to the nearest whole UTF-8 character.

All value strings sent to the controller using MT Focus 6000 Open Protocol can be encoded as ASCII or UTF-8.

ⓘ The message header length field expresses length in bytes, not in characters.

No concept of programming control is implemented in MT Focus 6000 Open Protocol. All implemented protocol commands are available to all clients.

If the MT Focus 6000 controller receives a message that has a malformed header, the controller discards the entire packet. If the length field in the OP header is incorrect (due to the client sending erroneous data), it is possible to end up in a state where the controller has to discard the following packet entirely to reenter a functional session state. That is, after a malformed command is sent to the controller, depending on the exact nature of the malformed data, the controller can do one of the following:

- Discard the current message (not parse it as an OP message - no action, no ACK)

- Return a NAK (for example, **Unknown mid**)

- ACK the current message but discard the following message (for example, message is longer than length field indicated – this is discovered when the next message appears garbled).

Due to the last of these cases, it is recommended that mid 9999, keep alive, is sent at twice the minimum rate (every 7 seconds instead of every 15 seconds) as a keep alive message that follows some other, malformed, message, can be discarded, leading to the session closing due to timing out.

ⓘ All keep alive restrictions has been removed from firmware 1.10.0 when running OP and MTF6000 via Ethernet. RS232 still requires a keep alive message every 15 seconds.

Special MT Focus 6000 considerations, per mid

The following mids and revisions are supported by MT Focus 6000:

MID	Revisions	Name	Special considerations
0001	6	Application Communication Start	Only supports revision 6.
0002	6	Application Communication Start ACK	Only supports revision 6. The following fields are supported: • Controller name • Controller software version • Tool software version • Controller serial number • System type (004 = MicroTorque) • System Subtype (15 = IAM Basic Plus (available from fw v1.20), 20 = IAM Process, 30 = IAM Automation, 35 = IAM Smart Automation (available from fw v2.0), 40 = IAM QA (available from fw v2.0)) • Station ID • Station Name
0003	1	Application Communication Stop	N/A
0004	1	Application Communication NAK	Mid accepted field always contains the requested mid number – no special handling for mid 0008 and mid 0009.
0005	1	Application Communication ACK	Mid accepted field always contains the requested mid number – no special handling for mid 0008 and mid 0009.
0006	1	Application Data Message Request	Supports requests of the following mids: • 0067 (Tightening Result list) • 0215 (IO Device status) • 0217 (Relay Function) • 0221 (Digital Input function) • 0900 (Application Data Message Unsubscribe), supported from firmware version 1.16 • 1201 (Operation Result Overall Data), supported from firmware version 1.16 • 2501 (Tightening Program download)
0008	1	Application Data Message Subscribe	Supports subscription of the following mids: • 0015 (Program selected) • 0035 (Job information) • 0071 (Alarm, controller also sends 0074 if this subscription is active) • 0217 (Relay Function) • 0221 (Digital input function) • 0900 (Trace data, controller also sends 0901 if this subscription is active) • 1201 (Tightening result overall data, controller also sends 1202 if this subscription is active) • 1601 (New since fw v 1.20. Custom ID 1-4 results and scanned barcode data)
0009	1	Application Data Message Unsubscribe	Supports the same mids as for mid 0008
0010	1	Parameter Set ID Upload Request	N/A
0011	1	Parameter Set ID Upload Reply	N/A
0015	1	Parameter Set Selected	Selecting Pset 0 (no Pset selected) is permissible, and results in a notification if subscribed to. The content of all fields except Pset ID are undefined in this case.

MID	Revisions	Name	Special considerations
0018	1	Select Parameter Set	Only Pset numbers that are within the licensed range can be selected. Selecting Pset 0 (no Pset selected) is permissible.
0019	1-2	Set Parameter set batch size	Only Pset numbers that are within the licensed range can be selected. Selecting Pset 0 (no Pset selected) is permissible. Pset will also be deactivated after batch is completed.
0030	2	Job ID Upload Request	Only supports revision 2. The term *Job* is known as *Batch Sequence* in MT Focus 6000.
0031	2	Job ID Upload Reply	Only supports revision 2. The term *Job* is known as *Batch Sequence* in MT Focus 6000.
0035	2	Job Info	Is sent to subscribers once after each tightening, and once after the entire Job is completed. Does not support the Job batch mode field. Selecting Job 0 (no Job selected) is permissible, and results in a notification if subscribed to. The content of all fields except Job ID are undefined in this case. The term *Job* is known as *Batch Sequence* in MT Focus 6000.
0038	1-2	Select Job	Only Job numbers that are within the licensed range can be selected. Selecting Job 0 (no Job selected) is permissible. The term *Job* is known as *Batch Sequence* in MT Focus 6000.
0040	5	Tool Data Upload Request	Only supports revision 5.

MID	Revisions	Name	Special considerations
0041	5 & 9	Tool Data Upload Reply	The following fields are supported in revision 5: • Serial number • Number of tightenings • Calibration date • Software version • Max Torque (expressed in mNm * 100) • Gear ratio • Max speed • Tool model The following fields and all fields supported in revision 5 are supported in revision 9: • Tool number • Tool article number • Stroke length • Min speed • Max speed • Min torque • Mid torque • Min speed at min torque • Min speed at max torque • Max speed at min torque • Max speed at mid torque • Max speed at max torque • Temperature limit • Number of OK tightenings • Number of NOK tightenings • Lifetime OK tightenings • Lifetime NOK tightenings • Calibration interval • Calibration cycle interval • Last torque check • Cycles since last torque check
0071	2-3	Alarm	Supports Revision 2 and 3. ⓘ Revision 3 is supported from fw v 1.18. The following fields are supported: • Error code • ControllerReadyStatus • ToolReadyStatus • Time The term *Alarm* is known as *Event* in MT Focus 6000.
0072	1	Alarm Acknowledge	The term *Alarm* is known as *Event* in MT Focus 6000.
0074	1	Alarm Acknowledged On Controller	The term *Alarm* is known as *Event* in MT Focus 6000.
0076	2	Alarm Status	The term *Alarm* is known as *Event* in MT Focus 6000.
0078	1	Acknowledge Alarm Remotely On Controller	The term *Alarm* is known as *Event* in MT Focus 6000.
0080	1	Read Time Upload Request	N/A

MID	Revisions	Name	Special considerations
0081	1	Read Time Upload Reply	N/A
0082	1	Set Time	N/A
0150	1	Identifier Download Request	Special considerations: See *Notes on mid 0150 Identifier Download Request [Page 15]*
0157	1	Reset All Identifiers	N/A
0200	1	Externally monitored relays	N/A
0215	2	IO Device Status Reply	Special considerations: See *Notes on mid 0215 I/O Device Status [Page 15]*
0217	1	Relay Function	The term *Relay Function* is known as *Digital Output Function* in MT Focus 6000.
0221	1	Digital Input function	Special considerations: See *Notes on mid 0221 Digital Input Function [Page 16]*
0224	1	Set Digital Input Function	Special considerations: See *Notes on mid 0224 Set Digital Input Function [Page 16]*
0225	1	Reset Digital Input Function	N/A
0900	1-3	Trace Curve Data Message	Special considerations: See *Notes on mid 0900 Trace curve data message [Page 18]*
0901	1-3	Traces Plot Parameters Message	Special considerations: See *Notes on mid 0901 Traces Plot Parameters Message [Page 18]*
1201	1-3	Operation Result Overall Data	Special considerations: See *Notes on mid 1201 Operation result Overall data [Page 18]*
1202	1-2	Operation Result Object Data	Special considerations: See *Notes on mid 1202 Operation result object data [Page 19]*
1601	1	Dynamic identifier	Special considerations: See *Notes on mid 1601 Dynamic identifier [Page 27]*
1602	1	Dynamic identifier acknowledge	N/A
2500	1	Tightening Program Message Download	Supported from firmware version 1.16. Special considerations: See *Notes on mid 2500 Tightening Program Download [Page 27]*
2501	1	Tightening Program Message Upload	Special considerations: See *Notes on mid 2501 Tightening Program Upload [Page 28]*
9999	1	Keep Alive	Special considerations: See *Notes on mid 9999 Keep Alive [Page 34]*

Notes on mid 0006 Application data request

When requesting mid 1201, the controller responds with one mid 1201 message and one or more mid 1202 messages. Likewise, when requesting mid 0900, the controller responds with one mid 0900 message and one or more mid 0901 messages.

Notes on mid 0008 Application data message subscribe

When subscribing to mid 0035, the controller immediately responds with a mid 0005 ACK and a mid 0035 Job Information.

When subscribing to mid 0900, two separate mid 0900 messages is sent for each tightening result, one for angle data and one for torque data. The messages are immediately followed by one mid 0901 message.

When subscribing to mid 1201, only alternative 0 (new data) is available. Therefore, when using revision 1, all inserted values are interpreted as 0. Mid 1201 is followed by exactly one instance of mid 1202 containing the object data from one single controller/tool.

The MTF6000 controller does not need to receive an acknowledge from the integrator after sending a subscription result. The controller will ignore the ACK message if one is sent.

The following ACK messages are going to be ignored by the MTF6000 controller:

- MID 0016 - New parameter set selected acknowledge
- MID 0036 - Job info Acknowledge

- MID 0072 - Alarm acknowledge
- MID 0075 - Alarm acknowledged on controller acknowledge
- MID 0077 - Alarm status acknowledge
- MID 0212 - Status externally monitored inputs acknowledge
- MID 1203 - Operation result data acknowledge
- MID 1602 - Dynamic identifier data acknowledge

Notes on mid 0071 Alarm

Only those events that are saved in the event log on the controller result in a mid 0071 alarm message.

(i) A subscription on 0071 also results in 0074 Alarm Acknowledged on Controller being sent to the integrator.

The following error codes are supported:

Code	Explanation	Recommended Action
101	Supply voltage exceeded	Check power supply
102	Supply voltage too low	Check power supply
103	Internal 24V exceeded	Disconnect external equipment
104	Internal 24V too low	Disconnect external equipment
105	Internal 12V exceeded	Disconnect tool if connected
106	Internal 12V too low	Disconnect tool if connected
107	Internal 5V exceeded	Disconnect tool if connected
108	Internal 5V too low	Disconnect tool if connected
109	External 24V error	Disconnect external equipment
110	DC/DC temperature error	Replace the controller
111	I/O expanders reinitialized	Press OK to continue
112	Failed to reinitialize I/O	Restart controller
113	Power fault	Check power supply.
120	Zero offset error, Current	Replace the controller
121	Zero offset error, Sensor	Replace the tool transducer or the tool
122	Motor current too low	Replace the tool
123	Motor current exceeded	Replace the tool
124	Motor driver voltage too low	Replace the tool
125	Motor driver short circuit	Replace the tool
126	Motor driver temperature exceeded	Replace the tool
127	Motor driver open load	Replace the tool.
128	Motor driver fault	Restart the controller.
130	EEPROM read error	Replace the controller
131	EEPROM write error	Replace the controller
132	Internal hardware error	Replace the controller
133	File system error	Replace IAM MT
134	USB host overcurrent	Disconnect USB devices
135	USB flash drive volume error	Check that the flash drive is formatted as FAT32
140	No tool connected	Connect tool
141	Tool not supported	Connect supported tool
142	Tool communication error	Replace the tool
143	Tool 5V error	Replace the tool
144	Tool 12V error	Replace the tool
150	Tool initializing error	Check that the tool is running freely during start up
151	Tool friction too high	Check that the tool is running freely during start up
152	Tool angle encoder error	Connect tool

Code	Explanation	Recommended Action
153	Tool direction error	Connect tool
154	Bit was blocked	Check that the tool is running freely during start up
155	Tool motor temperature exceeded	Allow time for tool to cool down
156	Tool not supported	Replace the tool
157	Failed to update tool software	Disconnect transducer
158	Tool initialization error	
159	Torque measurement warning	Calibrate the tool
160	Torque measurement error	Calibrate the tool
201	Runtime parameter error	Restart the controller
202	Invalid controller parameters	Replace IAM MT
203	Invalid tool parameters	Replace the tool
204	Invalid I/O parameters	Replace IAM MT
205	Invalid identifier parameters	Replace IAM MT
206	Invalid Pset parameters	Replace IAM MT
207	Invalid Batch sequence parameters	Replace IAM MT
208	Invalid password parameters	Replace IAM MT
209	Tool statistics error	Replace the tool
210	Unable to load fonts from IAM MT	Replace IAM MT
211	Invalid tool configuration parameters	Replace IAM MT
212	Invalid operator parameters	Replace IAM MT
220	Result database error	Replace IAM MT
221	Graph database error	Replace IAM MT
225	Reset result database	Result database was reset
226	Reset graph database	Graph database was reset
227	No results to export - database empty	Nothing - There are no results to export
228	Old results removed	
229	Old graphs removed	
230	Pset not configured for connected tool	Run another Pset or change tool
231	Select error source	Change select source. Can be changed in ToolsTalk MT Controller settings
232	Select error busy	Wait until tool is not running
240	Network configuration has an error	Change network configuration.
241	IP address and gateway not in same subnet	Change network configuration.
301	Tool connected	Press OK to initialize the tool
302	Tool initializing	Please wait
303	Tool software updating	Please wait
304	Tool software update error	Check tool cable or replace the tool
305	Tool calibration required	Calibrate the tool
306	Service required	Service the tool
307	Torque check required	
308	Torque check required	
310	Software has been updated	Press OK to continue
311	Software file was not found on USB	Copy software file to USB flash drive
312	Software update failed	Software file on USB may be broken. Update software file on USB and try again
313	Copying software	Please wait. Software is being copied to/from USB
314	Can only show first 20 .mtp or .fw files	Press OK to continue
315	Software update required	
320	Login expired	User was logged out
321	User has logged in	User was logged in

Code	Explanation	Recommended Action
322	User has logged out	User was logged out
350	Battery adapter hardware error	Contact AC Service
351	Battery adapter temperature exceeded	Let battery to cool down
352	Battery adapter charger fault	Change battery
353	Battery temperature error	Let battery to cool down
354	Battery discharge error	Connect PSU to charge battery
355	Battery charge critically low	Connect PSU to charge battery
356	Battery not supported	Change battery
357	Battery adapter charge fault	Change battery
358	Battery adapter charge fault	Let battery cool down or change battery
401	Tightening error	Press OK to continue
402	Batch sequence error	Press OK to continue
403	Batch sequence parameter error	Check controller to see which parameter is incorrect
404	Pset not supported	Select another Pset
405	Pset configuration error	Update Pset parameters
507	ToolsNet skipped result	
511	ToolsNet major error	
601	Controller communication error	Check USB Sync cable
602	Transducer communication error	Check transducer cable
603	Measurement data lost due to lost USB link	Check USB Sync cable
604	Transducer connected	
605	No transucer detected	Connect transducer and try again
606	Remote device is busy	Try again later
607	Transducer service required	Contact SC Service, time to service transducer
608	Zero offset error, Transducer	Check that the transducer cable is fastened correctly or replace cable/transducer
609	Operator ID required by active verification	Scan operator ID
610	Connected device is not compatible with QA	
701	Fieldbus not supported by controller HW	Replace controller with rev C or higher
702	Invalid fieldbus parameters	Replace IAM MT
703	Fieldbus module error	Check fieldbus cable or replace the fieldbus module, press OK to continue.
704	Fieldbus driver warning	Check for firmware update, press OK to continue
705	Fieldbus driver error	Check for firmware update
706	Fieldbus module in error state	Check fieldbus network status
707	Fieldbus module in exception state	Check for firmware update
720	Fieldbus module not connected	Connect fieldbus module, check fieldbus cable or replace the fieldbus module
721	Fieldbus module not supported, wrong module type	Connect supported fieldbus module
722	Fieldbus module not supported, wrong network type	Connect supported fieldbus module
723	Fieldbus module not supported, wrong provider ID	Connect supported fieldbus module
724	Fieldbus module not supported, FW version not approved	Connect supported fieldbus module
730	Fieldbus not configured for connected module	Change configuration of fieldbus type or connect correct module
731	Fieldbus parameter error	Update fieldbus parameters and press OK
732	Fieldbus configuration of network settings failed	
740	Max 128 submodules supported for Profinet	Update fieldbus parameters and press OK
741	Profinet station name syntax error	Update fieldbus parameters and press OK
901	IAM MT not present	Insert IAM MT
902	Failed to initialize IAM MT	Restart the controller or reconnect the IAM MT

Code	Explanation	Recommended Action
903	IAM MT read error	Try again or change IAM MT
904	IAM MT write error	Try again or change IAM MT
905	Invalid software on IAM MT	Update software on IAM MT (manually from PC)
906	Flash programming error	Try again or replace the controller
907	Invalid license on IAM MT	Update software on IAM MT (manually from PC)
908	Updating content of IAM	Please wait
990	External event	
991	External event	
992	External event	

Notes on mid 0074 Alarm Acknowledged on Controller

This message is sent to integrator if a subscription to mid 0071 is done.

Notes on mid 0150 Identifier Download Request

Identifiers downloaded to the controller is handled in the same manner as scanned identifiers. That is, through simple or advanced identifier scan settings. An identifier that does not match the identifier settings in the controller is not accepted as input during the batch sequence – however, the controller responds to the mid 0150 request with a mid 0005 ACK regardless. If the identifier exceeds 100 characters, the controller responds to the request with mid 0004 NACK.

Notes on mid 0215 I/O Device Status

(i) This message contains the status of the physical I/O pins, not the status of the I/O functions.

The I/O device status can be requested using mid 0006. The mid 0215 reply currently only contains the I/O status of the internal device, that is the state of the IO pins of the controller itself. The I/O device ID field will therefore currently always contain **00**.

Notes on mid 0217 Relay Function

All status changes for both tracking and non-tracking events will be sent.

The following output functions can be sent for mid 0217:

ID	Output function	Description
1	Ready	Ready to start
2	Busy	Busy performing an operation
3	Error	An error is preventing start
4	Tightening OK	Tightening OK
5	Tightening NOK	Tightening not OK
6	Batch	Batch OK
7	Active event	An event is active
8	Blocking event	Active event is blocking operation
9	Clearable event	Active event is acknowledgeable
10	Initializing tool	Tool is being initialized
11	Tool disabled	Tool is disabled
12	Vacuum pump	Vacuum pump enabled/disabled
13	Start signal	Mirror of tool start trigger signal
14	Loosening signal	Mirror of tool loosening trigger signal
15	Push to start signal	Mirror of tool push to start signal
16	Batch Sequence DO1	Available for use in batch sequence
17	Batch Sequence DO2	Available for use in batch sequence

ID	Output function	Description
18	Batch Sequence DO3	Available for use in batch sequence
19	Batch Sequence DO4	Available for use in batch sequence
20	Batch Sequence DO5	Available for use in batch sequence
21	Batch Sequence DO6	Available for use in batch sequence
22	Batch Sequence DO7	Available for use in batch sequence
23	Batch Sequence DO8	Available for use in batch sequence
24	Batch sequence complete	Batch sequence OK
25	Batch sequence error	Batch sequence error
26	Batch error	Batch error
27	Standby Active	Controller is in standby mode
28	Screw not aligned	Screw not aligned limit has been exceeded. Screw has been picked up but is not aligned.
29	Screw aligned	Screw aligned limit has been exceeded. Screw is aligned straight with the bit.
30	External monitored 1	External monitored signal
31	External monitored 2	External monitored signal
32	External monitored 3	External monitored signal
33	External monitored 4	External monitored signal
34	External monitored 5	External monitored signal
35	External monitored 6	External monitored signal
36	External monitored 7	External monitored signal
37	External monitored 8	External monitored signal
38	External monitored 9	External monitored signal
39	External monitored 10	External monitored signal
40	Vacuum clean	Vacuum clean function is active
41	Vacuum connected	Checks if Vacuum Pump MT (8432 0854 00) is connected. MTF6000 must have at least HW revision C for this signal to operate. Vacuum Pump VPX6 does not trigger this signal regardless of MTF6000 HW revision.
44	Verification complete	Verification is completed
45	Verification active	A Verification program is active
46	Service required	Service required for tool
47	Battery connected	Battery is connected
48	Battery low	Low battery (below 20%)
49	Verification OK	Verification is OK
50	Verification NOK	Verification is NOK
51	Unlock bit 0	Binary value of which bit to select when using bit select function
52	Unlock bit 1	Binary value of which bit to select when using bit select function
53	Unlock bit 2	Binary value of which bit to select when using bit select function
54	Unlock bit 3	Binary value of which bit to select when using bit select function
55	Torque check required	Torque check limit has been reached
56	Rotating CW	Bit is rotating clockwise.
57	Rotating CCW	Bit is rotating counterclockwise.
58	Tightening selected	ETD MT or ETD MC tool is in tightening mode (available from firmware 3.4.2.0 and onwards)
59	Loosening selected	ETD MT or ETD MC tool is in loosening mode (available from firmware 3.4.2.0 and onwards)

Notes on mid 0221 Digital Input Function

All status changes for both tracking and non-tracking events will be sent.

The enumeration of digital input functions is specific to MT Focus 6000.

See also: *Notes on mid 0224 Set Digital Input Function [Page 16]*.

Notes on mid 0224 Set Digital Input Function

The following input functions can be triggered using mid 0224:

ID	Input function	Description
1	Start tightening	Start tightening operation
2	Start tightening (hold)	Start tightening with stop on negative flank
3	Start loosening	Start loosening operation
4	Start loosening (hold)	Start loosening with stop on negative flank
5	Stop operation	Stop ongoing operation
6	Reset	Stop operation if busy, reset tightening result if idle
8	Disable tool	Disable tool while set to high
9	Initialize tool	Initialize tool without confirmation
11	Clear event	Clear active event
12	Clear all events	Clear all active events
22	Reset Batch sequence	Reset Batch sequence
23	Increment batch	Skip a tightening in a Batch
24	Decrement batch	Redo a tightening in a Batch
25	Reset batch	Reset batch
26	Batch sequence DI1	Available for use in Batch sequence
27	Batch sequence DI2	Available for use in Batch sequence
28	Batch sequence DI3	Available for use in Batch sequence
29	Batch sequence DI4	Available for use in Batch sequence
30	Batch sequence DI5	Available for use in Batch sequence
31	Batch sequence DI6	Available for use in Batch sequence
32	Batch sequence DI7	Available for use in Batch sequence
33	Batch seguence DI8	Available for use in Batch sequence
34	Batch sequence DI9	Available for use in Batch sequence
35	Batch sequence DI10	Available for use in Batch sequence
36	Batch sequence DI11	Available for use in Batch sequence
37	Batch sequence DI12	Available for use in Batch sequence
38	Standby	Put controller into standby, resume on I/O activity
39	Reboot	Reboot controller.
40	Increment Batch Sequence	Skip a step in a Batch Sequence
42	Enable Vacuum	Enable Vacuum
43	Enable clean	Revers airflow to clean tube
44	Wake up	Wakes controller up from standby
45	Tool guiding light	Starts the tool guiding light, available for MT tools
46	Tool status red	Turns tool status light red, available for all QMC/MT tools
47	Tool status green	Turns tool status light green, available for all QMC/MT tools
48	Tool status blue	Turns tool status light blue, available for all QMC/MT tools
49	Tool status white	Turns tool status light white, available on all QMC/QMT tools
50	External monitored 1	Used to manipulate digital input without being linked to a specific function
51	External monitored 2	Used to manipulate digital input without being linked to a specific function
52	External monitored 3	Used to manipulate digital input without being linked to a specific function
53	External monitored 4	Used to manipulate digital input without being linked to a specific function
54	External monitored 5	Used to manipulate digital input without being linked to a specific function
55	External monitored 6	Used to manipulate digital input without being linked to a specific function
56	External monitored 7	Used to manipulate digital input without being linked to a specific function
57	External monitored 8	Used to manipulate digital input without being linked to a specific function
59	Start measurement	Start measurement operation
60	Stop measurement	Stop ongoing measurement
61	Start measurement (hold)	Start measurement with stop on negative flank
64	Set transducer zero	Set zero offset of connected transducer
65	Reset verification	Reset verification count to 0

Notes on mid 0900 Trace curve data message

Each mid 0900 message contains the data for one trace type, currently supported types are angle and torque.

The message contains the PID 02213 OR the PID 02214 coefficient for converting sample points. The data value of both these PIDs is encoded as char[12], format is that output from `printf_s("%.*e", 3, value)`. For example, `00001.234e+2`.

The resolution fields in the message encodes the time between two consecutive samples as char[12], format is that output from `printf_s("%.*e", 3, value)`. For example, `00001.234e+2`, expressed in seconds.

The samples contained in the binary part are encoded as little endian 16 bit two complement signed integers, expressed as degrees for angle trace, and as mNm for torque trace.

(i) The controller settings influence which results are sent out through Open Protocol – only those tightening results and graphs that are stored locally on the controller are sent out.

Notes on mid 0901 Traces Plot Parameters Message

The following PIDs are defined for MT Focus 6000:

ID	Content	Data Representation
30400	Graph marker seating index	Value is char[5], integer; zero based index of graph sample point
30401	Graph marker seating detected index	Value is char[5], integer; zero based index of graph sample point
30402	Graph marker snug index	Value is char[5], integer; zero based index of graph sample point
30403	Graph marker snug detected index	Value is char[5], integer; zero based index of graph sample point
30404	Graph marker final peak torque index	Value is char[5], integer; zero based index of graph sample point
30405	Graph marker final clamp torque index	Value is char[5], integer; zero based index of graph sample point
30420	Graph marker seating value	Value is char[12], format is that output from `printf_s("%012.*e", 3, value)`. For example, `00001.234e+2`
30421	Graph marker seating detected value	Value is char[12], format is that output from `printf_s("%012.*e", 3, value)`. For example, `00001.234e+2`
30422	Graph marker snug value	Value is char[12], format is that output from `printf_s("%012.*e", 3, value)`. For example, `00001.234e+2`
30423	Graph marker snug detected value	Value is char[12], format is that output from `printf_s("%012.*e", 3, value)`. For example, `00001.234e+2`
30424	Graph marker final peak torque value	Value is char[12], format is that output from `printf_s("%012.*e", 3, value)`. For example, `00001.234e+2`
30425	Graph marker final clamp torque value	Value is char[12], format is that output from `printf_s("%012.*e", 3, value)`. For example, `00001.234e+2`
30900	Parameter trace source	• 0 = unknown • 1 = tightening • 2 = QA (measurement)

Notes on mid 1201 Operation result Overall data

When subscribing to mid 1201, only send alternative 0 is supported. Time stamps and indexes are ignored on subscription. Send object data field is ignored on subscription – object data (mid 1202) is always sent.

(i) The controller settings influence which results are sent out through Open Protocol – only those tightening results and graphs that are stored locally on the controller are sent out. If no results are available in the result database, the controller sends a mid 0004 NACK instead of a result.

Notes on mid 1202 Operation result object data

The following PIDs are defined for MT Focus 6000:

ID	Content	Data Representation
30200	Result identifier	value is char[11], integer
30201	Result type	value is char[1]: • 0 = undefined • 1 = tightening • 2 = loosening
30202	Result code	value is char[1]: • 0 = undefined • 1 = ok • 2 = cancel • 3 = error
30203	Tightening start time	value is char[19], OP datetime. For example, `2016-09-23:11:50:20`
30204	Error step number	value can be max char[3], integer
30205	Error code	value is char[2]: • 0 = undefined • 1 = ok • 2 = valuehigh • 3 = valuelow • 4 = triggerlost • 5 = bitslipdetected • 6 = rehitdetected • 7 = noseatingdetected • 8 = damagedthreaddetected • 9 = rescindingtorquelow
30206	Error value	value is char[2]: • 0 = undefined • 1 = none • 2 = totalTime • 3 = stepTime • 4 = torque • 5 = stepAngle • 6 = totalAngle • 7 = clampTorque • 8 = clampAngle • 9 = tighteningAngle • 10 = looseningTime • 11 = Angle range
30207	Controller serial number	value can be max char[12]
30208	Controller name	value can be max char[90], encoding is utf8
30209	Controller ID	value can be max char[11], integer
30210	Station name	value can be max char[90], encoding is utf8
30211	Station ID	value can be max char[11], integer
30212	Line name	value can be max char[90], encoding is utf8
30213	Line ID	value can be max char[11], integer
30214	Tool serial number	value can be max char[12]

ID	Content	Data Representation
30215	Tool name	value can be max char[30]
30216	Pset number	value can be max char[11], integer
30217	Pset name	value can be max char[90], encoding is utf8
30218	Pset revision	value is char[9], integer
30219	Pset created date	value is char[19], OP datetime
30220	Pset modified date	value is char[19], OP datetime
30221	Batch sequence number	value is char[5], integer
30222	Batch sequence name	value can be max char[90], encoding is utf8
30223	Batch sequence revision	value is char[9], integer
30224	Batch sequence created date	value is char[19], OP datetime
30225	Batch sequence modified date	value is char[19], OP datetime
30226	Batch sequence step count	value is char[5], integer
30227	Batch sequence step number	value is char[5], integer
30228	Batch size	value is char[3], integer
30229	Batch count	value is char[3], integer
30230	Peak torque	value is char[12], format is that output from `printf_s("%012.*e", 3, value)`. For example, `00001.234e+2`
30231	Total angle	value is char[12], expressed in degrees, format is that output from `printf_s("%012.*e", 3, value)`. For example, `00001.234e+2`
30232	Total duration	value is char[12], expressed in seconds, format is that output from `printf_s("%012.*e", 3, value)`. For example, `00001.234e+2`
30233	Tool temperature	value is char[12], format is that output from `printf_s("%012.*e", 3, value)`. For example, `00001.234e+2`
30234	Total angle status	value is char[1]: • 0 = undefined • 1 = ok • 2 = low • 3 = high
30235	Total duration status	value is char[1]: • 0 = undefined • 1 = ok • 2 = low • 3 = high
30236	Final torque type	value is char[1]: • 0 = undefined • 1 = peak torque • 2 = clamp torque • 3 = final torque • 4 = average torque
30237	Final torque	value is char[12], format is that output from `printf_s("%012.*e", 3, value)`. For example, `00001.234e+2`
30238	Final angle	value is char[12], expressed in degrees, format is that output from printf_s("%012.*e", 3, value). For example, `00001.234e+2`
30239	Final angle type	value is char[1]: • 0 = undefined • 1 = step angle • 2 = clamp angle • 3 = tightening angle • 4 = step range angle (New since fw v 1.20)

ID	Content	Data Representation
30240	Final report step	value is char[3], integer
30241	Final torque status	value is char[1]: • 0 = undefined • 1 = ok • 2 = low • 3 = high
30242	Final angle status	value is char[1]: • 0 = undefined • 1 = ok • 2 = low • 3 = high
30243	Torque tuning	value is char[12], format is that output from `printf_s("%012.*e", 3, value)`. For example, `00001.234e+2`
30244	Custom identifier key 1	value is char[48]
30245	Custom identifier value 1	value can be max char[100]
30246	Custom identifier key 2	value can be max char[48]
30247	Custom identifier value 2	value can be max char[100]
30248	Custom identifier key 3	value can be max char[48]
30249	Custom identifier value 3	value can be max char[100]
30250	Custom identifier key4	value can be max char[48]
30251	Custom identifier value 4	value can be max char[100]
30252	Screw pickup attempts	value is char[3]
30253	Total pickup time	value is char[12], expressed in seconds, format is that output from `printf_s("%012.*e", 3, value)`. For example, `00001.234e+2`
30254	Batch screw number	value is char[3]
30255	Final max torque limit	Value is char[12], format is that output from `printf_s("%012.*e", 3, value)`. For example, `00001.234e+2`
30256	Final min torque limit	Value is char[12], format is that output from `printf_s("%012.*e", 3, value)`. For example, `00001.234e+2`
30257	Final max angle limit	Value is char[12], expressed in degrees, format is that output from `printf_s("%012.*e", 3, value)`. For example, `00001.234e+2`
30258	Final min angle limit	Value is char[12], expressed in degrees, format is that output from `printf_s("%012.*e", 3, value)`. For example, `00001.234e+2`
30259	Final seating point	value is char[12], format is that output from `printf_s("%012.*e", 3, value)`. For example, `00001.234e+2`
30260	Not aligned limit	value is char[12], expressed in unit none, format is that output from `printf_s("%012.*e", 3, value)`. For example, `00001.234e+2`
30261	Aligned limit	value is char[12], expressed in unit none, format is that output from `printf_s("%012.*e", 3, value)`. For example, `00001.234e+2`
30262	Not aligned status	value is char[1]: • 0 = undefined • 1 = ok • 2 = low • 3 = high
30263	Aligned status	value is char[1]: • 0 = undefined • 1 = ok • 2 = low • 3 = high

ID	Content	Data Representation
30264	Pset type	value is char[1]: • 0 = Pset • 1 = Verification Program
30265	Verification size	value is char[3], integer
30266	Verification count	value is char[3], integer
30267	Tool calibration date	value is char[19], OP datetime. For example, `2016-09-23:11:50:20`
30268	Tool configuration revision	value is char[9], numerical
30269	Tool torque estimation type	value is char[1]: • 0 = unknown • 1 = current • 2 = transducer
30270	Angle range angle	value is char[12], expressed in degrees, format is that output from `printf_s("%012.*e", 3, value)`. For example, `00001.234e+2`
30271	Angle range status	value is char[1]: • 0 = undefined • 1 = ok • 2 = low • 3 = high
30272	Range angle start step	value is char[3], numerical
30273	Range angle stop step	value is char[3], numerical
30274	Pset Batch Size	value is char[3], numerical
30275	Pset Batch Count	value is char[3], numerical
30276	Final seating point angle	value is char[12], expressed in degrees, format is that output from `printf_s("%012.*e", value)`. For example, "1.234e+2"
30277	Final seating point angle start step	value is char[3], numerical.
30278	Clamp torque	value is char[12], expressed in mNm, format is that output from `printf_s("%012.*e", value)`. For example, "1.234e+2"
30279	Clamp angle	value is char[12], expressed in degrees, format is that output from `printf_s("%012.*e", value)`. For example, "1.234e+2"
30301	Step type	value is char[3]: • 1 = thread engagement • 2 = angle • 3 = torque • 4 = torque seating monitoring step • 5 = seating control step • 6 = friction control • 7 = input • 8 = output • 9 = smart thread engagement • 10 = smart clamp monitoring • 11 = smart clamp control
30302	Step peak torque	value is char[12], format is that output from `printf_s("%012.*e", 3, value)`. For example, `00001.234e+2`
30303	Step angle	value is char[12], expressed in degrees, format is that output from `printf_s("%012.*e", 3, value)`. For example, `00001.234e+2`
30304	Step duration	value is char[12], expressed in seconds, format is that output from `printf_s("%012.*e", 3, value)`. For example, `00001.234e+2`
30305	Step transition torque	value is char[12], format is that output from `printf_s("%012.*e", 3, value)`. For example, `00001.234e+2`

ID	Content	Data Representation
30306	Step tightening method	value is char[1]: • 0 = undefined • 1 = torque • 2 = angle
30307	Step tightening angle	value is char[12], expressed in degrees, format is that output from `printf_s("%012.*e", 3, value)`. For example, `00001.234e+2`
30308	Step clamp torque	value is char[12], format is that output from `printf_s("%012.*e", 3, value)`. For example, `00001.234e+2`
30309	Step clamp angle	value is char[12], expressed in degrees, format is that output from `printf_s("%012.*e", 3, value)`. For example, `00001.234e+2`
30310	Step torque status	value is char[1]: • 0 = undefined • 1 = ok • 2 = low • 3 = high
30311	Step angle status	value is char[1]: • 0 = undefined • 1 = ok • 2 = low • 3 = high
30312	Step duration status	value is char[1]: • 0 = undefined • 1 = ok • 2 = low • 3 = high
30313	Step tightening angle status	value is char[1]: • 0 = undefined • 1 = ok • 2 = low • 3 = high
30314	Step clamp torque status	value is char[1]: • 0 = undefined • 1 = ok • 2 = low • 3 = high
30315	Step clamp angle status	value is char[1]: • 0 = undefined • 1 = ok • 2 = low • 3 = high
30316	Step seating point	value is char[12], format is that output from `printf_s("%012.*e", 3, value)`. For example, `00001.234e+2`
30317	Step lowest torque	value is char[12], format is that output from `printf_s("%012.*e", 3, value)`. For example, `00001.234e+2`
30318	Step average torque	value is char[12], format is that output from `printf_s("%012.*e", 3, value)`. For example, `00001.234e+2`

ID	Content	Data Representation
30319	Step average torque status	value is char[1]: • 0 = undefined • 1 = ok • 2 = low • 3 = high
30320	Step start angle	value is char[12], expressed in degrees, format is that output from `printf_s("%012.*e", 3, value)`. For example, `00001.234e+2`
30321	Step start tightening angle	value is char[12], expressed in degrees, format is that output from `printf_s("%012.*e", 3, value)`. For example, `00001.234e+2`
30322	Step seating point angle	value is char[12], expressed in degrees, format is that output from `printf_s("%012.*e", 3, value)`. For example, `00001.234e+2`
30323	Step engagement angle	value is char[12], expressed in degrees, format is that output from `printf_s("%012.*e", 3, value)`. For example, `00001.234e+2`
30324	Step engagement attempts	value is char[3], integer
30500	Measurement ID	value is char[11], integer
30501	Measurement Type	value is char[1]: • 0 = undefined • 1 = tightening • 2 = loosening • 3 = measurement
30502	Measurement Code	value is char[1]: • 0 = undefined • 1 = ok • 2 = cancel • 3 = error
30503	Measurement Start time	value is char[19], OP datetime. For example, `2016-09-23:11:50:20`
30507	Measurement QA Controller Serial number	value can be max char[12]
30508	Measurement QA Controller name	value can be max char[90], encoding is utf8
30509	Measurement QA Controller ID	value can be max char[11], integer
30510	Measurement QA Station name	value can be max char[90], encoding is utf8
30511	Measurement QA Station ID	value can be max char[11], integer
30512	Measurement QA Line name	value can be max char[90], encoding is utf8
30513	Measurement QA Line ID	value can be max char[11], integer
30514	Measurement Transducer serial number	value can be max char[12]
30515	Measurement Transducer name	value can be max char[30], encoding is utf8
30516	Measurement Transducer model	value is char[1]: • 0 = undefined • 1 = static • 2 = rotary
30517	Measurement Transducer calibration date	value is char[19], OP datetime. For example, `2016-09-23:11:50:20`
30518	Measurement Traceability source	value is char[1]: • 0 = undefined • 1 = station barcode • 2 = USB Sync
30601	Measurement Tightening Controller Serial number	value can be max char[12]

ID	Content	Data Representation
30602	Measurement Tightening Controller name	value can be max char[90], encoding is utf8
30603	Measurement Tightening Controller ID	value can be max char[11], integer
30604	Measurement Tightening Station name	value can be max char[90], encoding is utf8
30605	Measurement Tightening Station ID	value can be max char[11], integer
30606	Measurement Tightening Line name	value can be max char[90], encoding is utf8
30607	Measurement Tightening Line ID	value can be max char[11], integer
30608	Measurement Tool Serial number	value can be max char[12]
30609	Measurement Tool Name	value can be max char[30]
30610	Measurement Tool Calibration date	value is char[19], OP datetime. For example, `2016-09-23:11:50:20`
30611	Measurement Tool Configuration revision	value can be max char[11], integer
30612	Measurement Tool Torque type	value is char[1]: • 0 = undefined • 1 = current • 2 = transducer
30613	Measurement Tool Cycle count (statistics)	value can be max char[11], integer
30614	Measurement Tool OK count (statistics)	value can be max char[11], integer
30615	Measurement Tool NOK count (statistics)	value can be max char[11], integer
30616	Measurement Tool Average Torque (statistics)	value is char[12], format is that output from `printf_s("%012.*e", 3, value)`. For example, `00001.234e+2`
30617	Measurement Tool Average Temperature (statistics)	value is char[12], format is that output from `printf_s("%012.*e", 3, value)`. For example, `00001.234e+2`
30618	Measurement Tool Average Max Speed (statistics)	value is char[12], format is that output from `printf_s("%012.*e", 3, value)`. For example, `00001.234e+2`
30619	Measurement Tool Average Torque over Angle (statistics)	value is char[12], format is that output from `printf_s("%012.*e", 3, value)`. For example, `00001.234e+2`
30620	Measurement Tool Average cycle duration (statistics)	value is char[12], expressed in seconds, format is that output from `printf_s("%012.*e", 3, value)`. For example, `00001.234e+2`
30621	Measurement Tool Lifetime Cycle count (statistics)	value can be max char[11], integer
30622	Measurement Tool Lifetime OK count (statistics)	value can be max char[11], integer
30623	Measurement Tool Lifetime NOK count (statistics)	value can be max char[11], integer
30624	Measurement Tool Lifetime average Torque	value is char[12], format is that output from `printf_s("%012.*e", 3, value)`. For example, `00001.234e+2`
30625	Measurement Tool Lifetime average Temperature	value is char[12], format is that output from `printf_s("%012.*e", 3, value)`. For example, `00001.234e+2`
30626	Measurement Tool Lifetime average Max Speed	value is char[12], format is that output from `printf_s("%012.*e", 3, value)`. For example, `00001.234e+2`
30627	Measurement Tool Lifetime average Torque over Angle	value is char[12], format is that output from `printf_s("%012.*e", 3, value)`. For example, `00001.234e+2`
30628	Measurement Tool Lifetime average cycle duration	value is char[12], expressed in seconds, format is that output from `printf_s("%012.*e", 3, value)`. For example, `00001.234e+2`
30519	Measurement Tool Operator ID	value is char[30], encoding is utf8
30520	Measurement Tool Target torque	value is char[12], format is that output from `printf_s("%012.*e", 3, value)`. For example, `00001.234e+2`

ID	Content	Data Representation
30521	Measurement Tool Control limit	value is char[12], format is that output from `printf_s("%012.*e", 3, value)`. For example, `00001.234e+2`
30522	Measurement Tool Reference peak torque	value is char[12], format is that output from `printf_s("%012.*e", 3, value)`. For example, `00001.234e+2`
30523	Measurement Tool Reference angle	value is char[12], expressed in degrees, format is that output from `printf_s("%012.*e", 3, value)`. For example, `00001.234e+2`
30524	Measurement Tool Duration	value is char[12], expressed in seconds, format is that output from `printf_s("%012.*e", 3, value)`. For example, `00001.234e+2`
30525	Measurement Tool Target deviation [%]	value is char[12], format is that output from `printf_s("%012.*e", 3, value)`. For example, `00001.234e+2`
30526	Measurement Tool Target deviation status	value is char[1]: • 0 = undefined • 1 = ok • 2 = low • 3 = high
30529	Measurement Tool Evaluation mode	value is char[1]: • 0 = none • 1 = target deviation • 2 = tool deviation • 3 = evaluate all
30530	Measurement Tool Verification mode	value is char[1]: • 0 = none • 1 = free run • 2 = verification
30700	Measurement Tool Verification program number	value is char[3], numerical
30701	Measurement Tool Verification program name	value is char[90], encoding is utf8
30702	Measurement Tool Verification program revision	value is char[11], numerical
30703	Measurement Tool Verification program created date	value is char[19], OP datetime. For example, `2016-09-23:11:50:20`
30704	Measurement Tool Verification program modified date	value is char[19], OP datetime. For example, `2016-09-23:11:50:20`
30705	Measurement Tool Verification size	value is char[5], numerical
30706	Measurement Tool Verification criteria	value is char[1]: • 0 = none • 1 = NOK count • 2 = CMK
30707	Measurement Tool Verification max allowed NOK	value is char[5], numerical
30708	Measurement Tool Verification min allowed CMK	value is char[12], format is that output from `printf_s("%012.*e", 3, value)`. For example, `00001.234e+2`
30709	Measurement Tool Verification count	value is char[5], numerical
30710	Measurement Tool Verification NOK count	value is char[5], numerical
30711	Measurement Tool Verification final result	value is char[1]: • 0 = unknown • 1 = OK • 2 = NOK

ID	Content	Data Representation
30800	Measurement Verification result average reference peak torque	value is char[12], format is that output from `printf_s("%012.*e", 3, value)`. For example, `00001.234e+2`
30801	Measurement Verification result average target deviation [%]	value is char[12], format is that output from `printf_s("%012.*e", 3, value)`. For example, `00001.234e+2`
30802	Measurement Verification result CMK	value is char[12], format is that output from `printf_s("%012.*e", 3, value)`. For example, `00001.234e+2`
30803	Measurement Verification result control limit [%]	value is char[12], format is that output from `printf_s("%012.*e", 3, value)`. For example, `00001.234e+2`
30804	Measurement Verification result average tool peak torque	value is char[12], format is that output from `printf_s("%012.*e", 3, value)`. For example, `00001.234e+2`
30805	Measurement Verification result average tool deviation [%]	value is char[12], format is that output from `printf_s("%012.*e", 3, value)`. For example, `00001.234e+2`

Notes on mid 1601 Dynamic identifier

Supported from fw version 1.20.

This mid can be used to extract the values of the four Custom IDs stored inside the controller. It is also possible to extract the currently scanned barcode value.

The MID1601 will automatically update each time a Custom ID or entered barcode is updated or cleared.

The controller supports Format Type 0 and Send alternatives 0 and 2. Send alternative 0 will send all 9 PID's listed below. Alternative 2 allows sending a selected subset of these.

The following nine PID's are supported:

- 30244 Custom identifier key 1 value is char[48]
- 30245 Custom identifier value 1 value can be max char[100]
- 30246 Custom identifier key 2 value is char[48]
- 30247 Custom identifier value 2 value can be max char[100]
- 30248 Custom identifier key 3 value is char[48]
- 30249 Custom identifier value 3 value can be max char[100]
- 30250 Custom identifier key 4 value is char[48]
- 30251 Custom identifier value 4 value can be max char[100]
- 00060 Scanned Barcode can be max char[512]

Notes on mid 2500 Tightening Program Download

ⓘ This mid can only be used with the Automation license (supported from version 1.16).

A tightening program (Pset) can be created or modified using mid 2500.

The contents of the message shall be structured in the following manner:

- There must be one root node, and zero or more child nodes.
- The PID fields set on the root node correspond to the global Pset parameters.
- The root node shall have one child node for each step in the Pset.
- The PID fields set on each child node correspond to the parameters for that step.

Root Node (global PSET parameters)

PID01000 – pset number
PID30001 – step count
(rest of PIDs)

Root node has two mandatory fields

Child Node 1 (step 1)

PID30100 – step type
(rest of PIDs)

Child nodes have one mandatory field

Child Node 2 (step 2)

PID30100 – step type
(rest of PIDs)

Illustration 1: Schema of a message containing a two step PSET

Some global PIDs are mandatory when using mid 2500. The *Pset number* and *Step count* PIDs must always be set. When updating global parameters, apart from the mandatory PIDs, only the desired PIDs need to be included in the message. For PIDs not included in the message, the old parameter value will remain, or, in the case of creating a new Pset, the default parameter values are set automatically.

When setting values in a Pset step, the **Step type** PID is always mandatory, and the **Step type** PID must be the first PID under the child node. When updating Pset step parameters, apart from the mandatory **Step type** PID, only the desired PIDs need to be included in the message. For PIDs not included in the message, the old parameter value will remain, or, in the case of creating a new Pset, the default parameter values are set automatically.

The applicable limits will be enforced on all parameters. The *Adjust to limits* PID (30000) can be used to control the behavior: If set to 0, any out-of-limits parameter value for the Pset will result in a NAK response to the mid 2500 message, and no changes in the controller. If set to 1, the controller will adjust all Pset parameters to the nearest allowed value if necessary, save the Pset in the controller and return ACK. Note that there is no mechanism for determining which parameter values were adjusted to limits, if any. All torque parameters will automatically be adjusted if out of range regardless of *Adjust to limits* setting.

The format for the mid 2500 message content is exactly the same as for mid 2501. This means that it is possible for the client to get a Pset a controller by using mid 2501 and saving it (for example, to file), and then sending the data back **to** a controller via mid 2500. Only Pset numbers that are within the licensed range can be down - or uploaded. If more than one client simultaneously tries to write to the same Pset, the data from client that sent the Pset **last** is saved to the controller.

Notes on mid 2501 Tightening Program Upload

When requesting a tightening program upload using mid 0006, the requested node type field is ignored.

The contents of the mid 2501 message is structured in the following manner: The PID fields at the root node correspond to the global Pset parameters. The root node will have one child node for each step in the Pset. Each child node will contain the PID fields relevant to that step.

There are two different step types:

- 30100 Step type—supports all license levels up to Smart process.
- 30199 Step type extended—supports all license levels.

(i) Only Pset numbers that are within the licensed range can be down- or uploaded.

The following PIDs are defined for MT Focus 6000:

ID	Content	Data representation
1000	Pset number	Value is char[3], integer
30001	Step count	Value is char[3], integer. For example, 003 for 3 steps
30002	Pset name	Value is char[91], encoding is utf8
30003	Pset revision	Value is char[9], integer

ID	Content	Data representation
30004	Pset created	Value is char[19] OP datetime. For example, `2016- 09-23:11:50:20`
30005	Pset modified	Value is char[19] OP datetime. For example, `2016- 09-23:11:50:20`
30006	Configured tool type	Value is char[5], integer
30007	Configured tool name	Value is char[90], encoding is utf8
30008	Minimum time	Value is char[12], format is that output from `printf_s("%012.*e", 3, value)`. For example, `00001.234e+2`, expressed in seconds
30009	Maximum time	Value is char[12], format is that output from `printf_s("%012.*e", 3, value)`. For example, `00001.234e+2`, expressed in seconds
30010	Min total angle	Value is char[12], format is that output from `printf_s("%012.*e", 3, value)`. For example, `00001.234e+2`, expressed in degrees
30011	Max total angle	Value is char[12], format is that output from `printf_s("%012.*e", 3, value)`. For example, `00001.234e+2`, expressed in degrees
30012	Loosening torque	Value is char[12], format is that output from `printf_s("%012.*e", 3, value)`. For example, `00001.234e+2`, expressed in mNm
30013	Loosening angle	Value is char[12], format is that output from `printf_s("%012.*e", 3, value)`. For example, `00001.234e+2`, expressed in degrees
30014	Loosening speed	Value is char[4] integer, expressed in rpm
30015	Loosening vacuum enabled	Value is char[1]: • 0 = no • 1 = yes
30016	Loosening graph enabled	Value is char[1]: • 0 = no • 1 = yes
30017	Loosening start delay	Value is char[12], format is that output from `printf_s("%012.*e", 3, value)`. For example, `00001.234e+2`, expressed in seconds
30018	Graph start step	Value is char[2], integer
30019	Torque tuning	Value is char[12], format is that output from `printf_s("%012.*e", 3, value)`. For example, `00001.234e+2`, expressed in percentage
30020	Final report step	Value is char[2], integer
30021	Final report torque type	Value is char[1]: • 0 = undefined • 1 = peak torque • 2 = clamp torque • 3 = final torque • 4 = average torque
30022	Final report angle type	Value is char[1]: • 0 = step • 1 = clamp • 2 = tightening • 3 = range
30023	Bit slip detection	Value is char[1]: • 0 = disabled • 1 = enabled
30024	Loosening max time	Value is char[1]: • 0 = disabled • 1 = enabled
30025	Trigger lost torque	Value is char[12], format is that output from `printf_s("%012.*e", 3, value)`. For example, `00001.234e+2`, expressed in mNm
30026	Custom field count	Value is char[1], integer

ID	Content	Data representation
30027	Custom field 1 type	Value is char[1]: • 0 = general • 1 = step
30028	Custom field 1 data	Value is char[2] integer
30029	Custom field 1 step number	Value is char[2] integer
30030	Custom field 1 step data	Value is char[2] integer
30031	Custom field 2 type	Value is char[1]: • 0 = general • 1 = step
30032	Custom field 2 data	Value is char[2] integer
30033	Custom field 2 step number	Value is char[2] integer
30034	Custom field 2 step data	Value is char[2] integer
30035	Custom field 3 type	Value is char[1]: • 0 = general • 1 = step
30036	Custom field 3 data	Value is char[2] integer
30037	Custom field 3 step number	Value is char[2] integer
30038	Custom field 3 step data	Value is char[2] integer
30039	Custom field 4 type	Value is char[1]: • 0 = general • 1 = step
30040	Custom field 4 data	Value is char[2] integer
30041	Custom field 4 step number	Value is char[2] integer
30042	Custom field 4 step data	Value is char[2] integer
30043	Pset GUID	Value is char[36]
30044	Screw pickup enabled	Value is char[1]: • 0 = disabled • 1 = enabled
30045	Screw pickup vacuum	Value is char[1]: • 0 = disabled • 1 = enabled
30046	Screw pickup rotation	Value is char[1]: • 0 = disabled • 1 = enabled
30047	Screw pickup timeout	Value is char[12], format is that output from `printf_s("%012.*e", 3, value)`. For example, `00001.234e+2`, expressed in seconds
30048	Damaged thread detection	Value is char[1]: • 0 = disabled • 1 = enabled
30049	Not aligned limit	Value is char[12], format is that output from `printf_s("%012.*e", 3, value)`. For example, `00001.234e+2`, expressed in unit none
30050	Aligned limit	Value is char[12], format is that output from `printf_s("%012.*e", 3, value)`. For example, `00001.234e+2`, expressed in unit none
30051	Screw pickup guiding	Value is char[1]: • 0 = disabled • 1 = enabled
30052	Pset type	value is char[1]: • 0 = Pset • 1 = Verification Program

ID	Content	Data representation
30053	Verification Program size	value is char[3], integer
30054	Control Limit	Value is char[12], format is that output from `printf_s("%012.*e", 3, value)`. For example, `00001.234e+2`, expressed in percentage
30056	On verification complete	Value is char[1]: • 0 = Disable Verification Program • 1 = Previous Pset/Batch • 2 = Startup Pset/Batch Sequence • 3 = Keep Verification Program
30057	Verification mode	Value is char[1]: • 0 = Verify results • 1 = CMK
30058	Measurement evaluation method	Value is char[1]: • 1 = Target vs Ref • 2 = Tool vs Ref
30059	Minimum CMK	Value is char[12], format is that output from printf_s("%012.*e", 3, value). For example, 00001.234e+2
30060	Operator ID required	Value is char[1]: • 0 = No • 1 = Yes
30061	Angle range start step	Value is char[2] integer. For example, 02 for start step 2
30062	Angle range stop step	Value is char[2] integer. For example, 04 for stop step 4
30063	Minimum angle range angle	Value is char[12], format is that output from `printf_s("%012.*e", 3, value)`. For example, `00001.234e+2`, expressed in degrees
30064	Maximum angle range angle	Value is char[12], format is that output from `printf_s("%012.*e", 3, value)`. For example, `00001.234e+2`, expressed in degrees
30065	Tool Rehit	Value is char[1]: • 0 = Controller settings • 1 = Disabled • 2 = Enabled
30066	Force Loosening on NOK	value is char[1]: • 0 = No • 1 = Yes
30100	Step type	Value is char[1]. Step type only supports license levels up to Smart process: ⓘ If using a smart step type, set this value to 0 and use PID 30199 instead. • 0 = see PID 30199 • 1 = thread engagement • 2 = angle • 3 = torque • 4 = torque seating monitoring step • 5 = seating control step
30101	Step speed	value is char[4] integer, expressed in rpm
30102	Step transition torque	Value is char[12], format is that output from `printf_s("%012.*e", 3, value)`. For example, `00001.234e+2`, expressed in mNm
30103	Step transition angle	Value is char[12], format is that output from `printf_s("%012.*e", 3, value)`. For example, `00001.234e+2`, expressed in degrees

ID	Content	Data representation
30104	Step direction	Value is char[1]: • 0 = CW • 1 = CCW
30105	Step start delay	Value is char[12], format is that output from `printf_s("%012.*e", 3, value)`. For example, `00001.234e+2`, expressed in seconds
30106	Min step time	Value is char[12], format is that output from `printf_s("%012.*e", 3, value)`. For example, `00001.234e+2`, expressed in seconds
30107	Max step time	Value is char[12], format is that output from `printf_s("%012.*e", 3, value)`. For example, `00001.234e+2`, expressed in seconds
30108	Step vacuum enabled	Value is char[1]: • 0 = no • 1 = yes
30109	Step target angle	Value is char[12], format is that output from `printf_s("%012.*e", 3, value)`. For example, `00001.234e+2`, expressed in degrees
30110	Step min torque limit	Value is char[12], format is that output from `printf_s("%012.*e", 3, value)`. For example, `00001.234e+2`, expressed in mNm
30111	Step max torque limit	Value is char[12], format is that output from `printf_s("%012.*e", 3, value)`. For example, `00001.234e+2`, expressed in mNm
30112	Step target torque	Value is char[12], format is that output from `printf_s("%012.*e", 3, value)`. For example, `00001.234e+2`, expressed in mNm
30113	Step min angle limit	Value is char[12], format is that output from `printf_s("%012.*e", 3, value)`. For example, `00001.234e+2`, expressed in degrees
30114	Step max angle limit	Value is char[12], format is that output from `printf_s("%012.*e", 3, value)`. For example, `00001.234e+2`, expressed in degrees
30115	Step tightening angle trigger	Value is char[12], format is that output from `printf_s("%012.*e", 3, value)`. For example, `00001.234e+2`, expressed in mNm
30116	Step min tightening angle limit	Value is char[12], format is that output from `printf_s("%012.*e", 3, value)`. For example, `00001.234e+2`, expressed in degrees
30117	Step max tightening angle limit	Value is char[12], format is that output from `printf_s("%012.*e", 3, value)`. For example, `00001.234e+2`, expressed in degrees
30118	Step seating angle displacement	Value is char[12], format is that output from `printf_s("%012.*e", 3, value)`. For example, `00001.234e+2`, expressed in degrees
30119	Step gradient trigger point	Value is char[12], format is that output from `printf_s("%012.*e", 3, value)`. For example, `00001.234e+2`
30120	Step min clamp torque limit	Value is char[12], format is that output from `printf_s("%012.*e", 3, value)`. For example, `00001.234e+2`, expressed in mNm
30121	Step max clamp torque limit	Value is char[12], format is that output from `printf_s("%012.*e", 3, value)`. For example, `00001.234e+2`, expressed in mNm
30122	Step min clamp angle limit	Value is char[12], format is that output from `printf_s("%012.*e", 3, value)`. For example, `00001.234e+2`, expressed in degrees
30123	Step max clamp angle limit	Value is char[12], format is that output from `printf_s("%012.*e", 3, value)`. For example, `00001.234e+2`, expressed in degrees
30124	Step final tightening method	Value is char[1]: • 0 = torque • 1 = angle
30125	Step clamp torque	Value is char[12], format is that output from `printf_s("%012.*e", 3, value)`. For example, `00001.234e+2`, expressed in mNm
30126	Step clamp angle	Value is char[12], format is that output from `printf_s("%012.*e", 3, value)`. For example, `00001.234e+2`, expressed in degrees
30127	Fast speed change	Value is char[1]: • 0 = disabled • 1 = enabled

ID	Content	Data representation
30128	Rescinding torque limit	Value is char[12], format is that output from `printf_s("%012.*e", 3, value)`. For example, `00001.234e+2`, expressed in mNm
30129	Step min average torque	Value is char[12], format is that output from `printf_s("%012.*e", 3, value)`. For example, `00001.234e+2`
30130	Step max average torque	Value is char[12], format is that output from `printf_s("%012.*e", 3, value)`. For example, `00001.234e+2`
30131	Step input signal	Value is char[3]: • 1 = External monitored 1 • 2 = External monitored 2 • 3 = External monitored 3 • 4 = External monitored 4 • 5 = External monitored 5 • 6 = External monitored 6 • 7 = External monitored 7 • 8 = External monitored 8
30132	Step input trigger	Value is char[1]: • 0 = undefined • 1 = Positive flank • 2 = Negative flank • 3 = Any flank • 4 = High level • 5 = Low level
30134	Step output signal	Value is char[3]: • 1 = External monitored 1 • 2 = External monitored 2 • 3 = External monitored 3 • 4 = External monitored 4 • 5 = External monitored 5 • 6 = External monitored 6 • 7 = External monitored 7 • 8 = External monitored 8 • 9 = External monitored 9 • 10 = External monitored 10
30135	Step output signal level	Value is char[1]: • 0 = undefined • 1 = high • 2 = low
30136	Step output signal mode	Value is char[1]: • 0 = undefined • 1 = set • 2 = duration
30137	Step output signal duration	Value is char[12], expressed in seconds, format is that output from `printf_s("%012.*e", 3, value)`. For example, `00001.234e+2`
30138	Step engagement torque	Value is char[12], format is that output from `printf_s("%012.*e", 3, value)`. For example, `00001.234e+2`
30139	Step validation angle	Value is char[12], format is that output from `printf_s("%012.*e", 3, value)`. For example, `00001.234e+2`, expressed in degrees

ID	Content	Data representation
30140	Step retry action	Value is char[2]: • 0 = None • 1 = All • 2 = Torque high • 3 = Angle high
30141	Step retry limit	Value is char[2], integer
30142	Step loosening angle method	Value is char[2]: • 0 = Automatic • 1 = User defined
30143	Step loosening angle	Value is char[12], format is that output from `printf_s("%012.*e", 3, value)`. For example, 00001.234e+2, expressed in degrees
30144	Step loosening torque	Value is char[12], format is that output from `printf_s("%012.*e", 3, value)`. For example, 00001.234e+2
30145	Step loosening speed	Value is char[4] integer, expressed in rpm
30146	Step retry signal	value is char[3]: • 0 = Undefined • 1 = External monitored 1 • 2 = External monitored 2 • 3 = External monitored 3 • 4 = External monitored 4 • 5 = External monitored 5 • 6 = External monitored 6 • 7 = External monitored 7 • 8 = External monitored 8 • 9 = External monitored 9 • 10 = External monitored 10
30199	Step type extended	Value is char[3]. Step type extended supports all license levels: • 1 = thread engagement • 2 = angle • 3 = torque • 4 = torque seating monitoring step • 5 = seating control step • 6 = friction control • 7 = digital input • 8 = digital output • 9 = smart engagement • 10 = smart torque seating monitoring • 11 = smart seating control

Notes on mid 9999 Keep Alive

As per the Open Protocol specification, the timeout value of the Open Protocol for MT Focus 6000 implementation is 15 seconds. This means, a client must send a message or a keep alive more often than each 15:th second. However, it is recommended that the client sends keep alive at least each 7:th second.

ⓘ From firmware version 1.10 and later, the demand for a keep alive message has been removed when using Open Protocol via Ethernet. All Open protocol connections run via RS232 must still send the keep alive message within 15 seconds.

Guide for previous users of Open Protocol

The Open Protocol for MT Focus 6000 implementation supports a subset of the full Atlas Copco Open Protocol. While not all mids of the full Open Protocol are supported, the functionality itself can often be reached by using a different mid. This is mostly due to the differences in the MT Focus 6000 platform versus other platforms.

The following is a short guide that suggest which mids that can be used to get the same function from the full Open Protocol.

I want to use…	For MTF6000 Open Protocol, instead use
Mid 0008, to request historical tightening results (send alternative 1-4)	Mid 0006, to request mid 1201, mid 900 for tightening results and trace curves
Mid 0012, 0013, parameter set data upload	Mid 2501, tightening program upload
Mid 0014, parameter set selected subscribe	Mid 0008, general subscribe
Mid 0017, parameter set selected unsubscribe	Mid 0009, general unsubscribe
Mid 0034, job info subscribe	Mid 0008, general subscribe
Mid 0037, job info unsubscribe	Mid 0009, general unsubscribe
Mid 0042, disable tool	Mid 0224 with input function 8
Mid 0043, enable tool	Mid 0225 with input function 8
Mid 0050, VIN download	Mid 150, identifier download
Mid 0060, last tightening result subscribe	Mid 0008, general subscribe
Mid 0063, last tightening result unsubscribe	Mid 0009, general unsubscribe
Mid 0070, alarm subscribe	Mid 0008, general subscribe
Mid 0073, alarm unsubscribe	Mid 0009, general unsubscribe
Mid 0127, abort job	Mid 224 with input function 22
Mid 0128, job batch increment	Mid 224 with input function 23
Mid 0129, job batch decrement	Mid 224 with input function 24
Mid 0216, relay function subscribe	Mid 0008, general subscribe
Mid 0219, relay function unsubscribe	Mid 0009, general unsubscribe
Mid 0220, digital input function subscribe	Mid 0008, general subscribe
Mid 0223, digital input function unsubscribe	Mid 0009, general unsubscribe
Mid 2505, to modify a parameter set	Mid 2500, and set the parameters that you want to change
Mid 0270, controller reboot request	Mid 0224 with input function 39

Original instructions

PAP 22