


| | | |
|---|---|---------------------------|
|  Matthews Marking Systems™ | Document name Matthews MPERIA Command Protocol Manual | Page 1 (85) |
| | Document number 795 247 | Date 2017-05-23 |

Matthews MPERIA Command Protocol Manual

for MPERIA 6.1.x



| | | |
|---|---|---------------------------|
|  Matthews Marking Systems™ | Document name Matthews MPERIA Command Protocol Manual | Page 2 (85) |
| | Document number 795 247 | Date 2017-05-23 |

Table of Contents

| | |
|-------------------------------------|----|
| 1 - Overview..... | 4 |
| 2 - Protocol..... | 5 |
| 2.1 - Availability..... | 5 |
| 2.2 - Commands and responses..... | 5 |
| 2.3 - Installation context..... | 6 |
| 2.4 - ASCII control characters..... | 6 |
| 3 - Command overview..... | 7 |
| 3.1 - Installations..... | 7 |
| 3.2 - Messages..... | 9 |
| 3.3 - Counters..... | 9 |
| 3.4 - Database tables..... | 10 |
| 3.5 - Variables..... | 10 |
| 3.6 - L-series support..... | 10 |
| 3.7 - T-series support..... | 11 |
| 3.8 - V-series support..... | 11 |
| 3.9 - I/O control..... | 11 |
| 3.10 - Miscellaneous..... | 11 |
| 4 - Command reference..... | 12 |
| 4.1 - CNT_DEL..... | 12 |
| 4.2 - CNT_GET..... | 13 |
| 4.3 - CNT_GET_I..... | 14 |
| 4.4 - CNT_GET_R..... | 15 |
| 4.5 - CNT_GET_RC..... | 16 |
| 4.6 - CNT_LIST..... | 17 |
| 4.7 - CNT_RESET..... | 18 |
| 4.8 - CNT_SET..... | 19 |
| 4.9 - CNT_SET_I..... | 20 |
| 4.10 - CNT_SET_R..... | 21 |
| 4.11 - CNT_SET_RC..... | 22 |
| 4.12 - CNT_UPDATE..... | 23 |
| 4.13 - DB..... | 24 |
| 4.14 - DB_ADD_ROW..... | 25 |
| 4.15 - DB_ADD_TABLE..... | 26 |
| 4.16 - DB_DEL_ROW..... | 27 |
| 4.17 - DB_DEL_TABLE..... | 28 |
| 4.18 - DB_GET..... | 29 |
| 4.19 - DB_INC..... | 30 |
| 4.20 - DB_INIT_TABLE..... | 31 |
| 4.21 - DB_SET..... | 32 |
| 4.22 - ENC_BLOCK..... | 33 |
| 4.23 - ENC_RESET_REVERSE..... | 34 |
| 4.24 - GOOD..... | 35 |
| 4.25 - INPUT_GET..... | 36 |
| 4.26 - INST..... | 37 |
| 4.27 - L_DECAP..... | 38 |
| 4.28 - L_DECAP_MARKER..... | 39 |
| 4.29 - L_DECAP_TABLE..... | 40 |

| | | |
|---|---|---------------------------|
|  Matthews Marking Systems™ | Document name Matthews MPERIA Command Protocol Manual | Page 3 (85) |
| | Document number 795 247 | Date 2017-05-23 |

| | |
|----------------------------------|----|
| 4.30 - LOG..... | 41 |
| 4.31 - MSG_DEL..... | 42 |
| 4.32 - MSG_LIST..... | 43 |
| 4.33 - MARKER_DIR..... | 44 |
| 4.34 - MARKER_MARGIN..... | 45 |
| 4.35 - MARKER_UPSIDEDOWN..... | 46 |
| 4.36 - MARK_GAP..... | 47 |
| 4.37 - MARK_INTERVAL..... | 48 |
| 4.38 - NONCRITICAL..... | 49 |
| 4.39 - OFFLINE..... | 50 |
| 4.40 - ONLINE..... | 51 |
| 4.41 - OUTPUT_SET..... | 52 |
| 4.42 - PEEKMSG..... | 53 |
| 4.43 - PLUGIN_ACTION..... | 54 |
| 4.44 - PRINT_CNT..... | 55 |
| 4.45 - PRINT_CNT_RESET..... | 56 |
| 4.46 - QUEUE..... | 57 |
| 4.47 - QUEUE_LEN..... | 58 |
| 4.48 - QUEUE_RESET..... | 59 |
| 4.49 - SEL..... | 60 |
| 4.50 - SELDB..... | 61 |
| 4.51 - SELMSG..... | 62 |
| 4.52 - SET_SPEED..... | 63 |
| 4.53 - START..... | 64 |
| 4.54 - STATE..... | 65 |
| 4.55 - STATUS..... | 66 |
| 4.56 - STOP..... | 69 |
| 4.57 - T_CLEAN..... | 70 |
| 4.58 - T_CLEAN_MARKER..... | 71 |
| 4.59 - TRIG..... | 72 |
| 4.60 - TRIG_ACTIVE..... | 73 |
| 4.61 - V_FLUSH..... | 74 |
| 4.62 - V_FLUSH_MARKER..... | 75 |
| 4.63 - V_SET_CLEANER..... | 76 |
| 4.64 - V_SET_CLEANER_MARKER..... | 77 |
| 4.65 - VAR_DEL..... | 78 |
| 4.66 - VAR_GET..... | 79 |
| 4.67 - VAR_LIST..... | 80 |
| 4.68 - VAR_SET..... | 81 |
| 4.69 - VAR_SET2..... | 82 |
| 4.70 - VERSION..... | 83 |
| 4.71 - SET_CLOCK..... | 84 |
| 5 - Revision history..... | 85 |

1 Overview

The MPERIA platform is designed from the ground-up to be an automation controller for marking/coding, packaging and warehouse/distribution control systems. Real-time inkjet and laser printing, and labeling all require data input from a host. Most marking and printing technologies have specific formats that need to be sent by a host.

MPERIA supports three different protocols that can be used by remote systems. Each protocol has it's own strengths and weaknesses, which makes it important to pick the right protocol to use.

| | |
|-----------------------------|--|
| Line Action Protocol | <p>Advantages:</p> <ul style="list-style-type: none"> • Simple line based protocol. Just send the line of text. • The GUI is used to configure what the received line is used for, e.g. selecting a message or setting a variable. <p>Disadvantages:</p> <ul style="list-style-type: none"> • Can only do one thing per port, e.g. select a message, but a single port cannot set a variable and a counter. • Can only do simple things. No support for triggering, performing maintenance, reading status, etc. |
| XML Protocol | <p>Advantages:</p> <ul style="list-style-type: none"> • Based on the XML standard, which means that most high level systems have tools to generate and parse XML. • Supports advanced functions such as creating messages. <p>Disadvantages:</p> <ul style="list-style-type: none"> • Requires XML support. • Cannot be used on serial ports. |
| Command Protocol | <p>Advantages:</p> <ul style="list-style-type: none"> • Easy to use. • Extremely easy to parse. Responses are single bytes. • Low overhead. <p>Disadvantages:</p> <ul style="list-style-type: none"> • While more powerful than the Line Action Protocol, it cannot for example create messages. |

2 Protocol

2.1 Availability

The Command Protocol is always available on TCP/IP ports 10005 and 10006. The difference between the two ports is that port 10006 does not send any responses back.

The Command Protocol can also be enabled for serial ports using the Protocol Configuration screen.

The Command Protocol was introduced in MPERIA 2.5.

2.2 Commands and responses

The Command Protocol is a command/response protocol. Commands are sent from the remote host to the Controller. Every command causes a response to be sent from the Controller back to the remote host.

All commands sent to the Controller has the following structure:

| |
|---|
| <code><ESC>command<FS>arg0<FS>arg1<FS> ... <FS>argN<ETB></code> |
|---|

If a command does not have any arguments, the structure simply becomes

`<ESC>command<ETB>`

A command with a single argument becomes

`<ESC>command<FS>arg0<ETB>`

A command with two argument becomes

`<ESC>command<FS>arg0<FS>arg1<ETB>`

Every command sent to the Controller causes the Controller to send a single byte back to indicate whether the command was successfully executed or not. If the command was successful, an **<ACK>** is sent back, and if the command failed, a **<NAK>** is sent back.

Some commands also provide data after the **<ACK>**. Such data always end with a single **<ETB>** byte. If the data contains several values, they are separated by **<FS>** bytes.

The protocol uses UTF-8 to encode Unicode strings.

It is recommended to set the NoDelay property on the socket when using the Command Protocol over TCP/IP and not use the Nagle algorithm to reduce communication delay.

2.3 Installation context

Several commands, e.g. STOP, START and SELMSG, operate on Installations. These commands do not take names of Installations as arguments. Instead, the INST command is used to specify which Installations later commands operate on. This has several advantages:

- Single Installation Controllers never need to execute an INST command.
- Multiple operations on a single Installation (or a set of Installations) do not need to repeat the Installation name for each command, which makes the communication more compact.

2.4 ASCII control characters

The Command Line Protocol uses the following ASCII control characters:

| ASCII | Hex | Dec | Description |
|-------|-----|-----|--|
| <ACK> | 06 | 6 | Acknowledge. Sent from the Controller to indicate the success of a command sent to the Controller. |
| <NAK> | 15 | 21 | Negative acknowledge. Sent from the Controller to indicate the failure of a command sent to the Controller. |
| <ETB> | 17 | 23 | End of Transmission Block. Used as the final character in all commands sent to the Controller, and as the end of data sent from the Controller as a response to a command. |
| <ESC> | 1B | 27 | Escape. Used as the starting character in all commands sent to the Controller. |
| <FS> | 1C | 28 | Unit Separator. Separates the command and arguments sent to the Controller, and multiple responses sent from the Controller as a response to a command. |
| <GS> | 1D | 29 | Group Separator. Separates components of command arguments into groups. |
| <RS> | 1E | 30 | Record separator used to separate groups into records. |
| <US> | 1F | 31 | Unit separator used to separate records into units. |

3 Command overview

3.1 Installations

Commands to manage Installations:


| Command | Description |
|-------------|---|
| OFFLINE | Set Installations in offline mode. |
| ONLINE | Set Installations in online mode. |
| STOP | Stop Installations. |
| START | Start Installations. |
| STATE | Get the state of a single Installation. |
| STATUS | Get the status of a Installations. |
| GOOD | Test if Installations are good, meaning that they do not require service and have no critical problems. |
| NONCRITICAL | Test if Installations are having any critical problems preventing them from printing. |

Commands to manage message selection:

| Command | Description |
|---------|---|
| SELMSG | Select message and database row for printing. |
| SELDB | Select database row for printing. |
| SEL | Select message and database row for printing. |
| PEEKMSG | Get the name of the currently selected message. |

Commands to manage printing:

| Command | Description |
|-----------------|---|
| TRIG | Trigger printing. |
| TRIG_ACTIVE | Get the current trigger status (active/inactive). |
| QUEUE | Fixate and queue the selected message. |
| QUEUE_RESET | Reset the print queue. |
| QUEUE_LEN | Get the print queue length. |
| MARKER_DIR | Set the marker print direction. |
| MARKER_MARGIN | Set the marker margin. |
| SET_SPEED | Set the fixed speed. |
| PRINT_CNT | Get the print count for messages and Installations. |
| PRINT_CNT_RESET | Reset the print count for messages and Installations. |

| | | |
|---|---|---------------------------|
|  Matthews Marking Systems™ | Document name Matthews MPERIA Command Protocol Manual | Page 8 (85) |
| | Document number 795 247 | Date 2017-05-23 |

| Command | Description |
|-------------------|--------------------------------------|
| ENC_BLOCK | Blocks encoder. |
| ENC_RESET_REVERSE | Resets the encoder's backward state. |

3.2 Messages

Commands used to manage messages:

| Command | Description |
|----------|------------------------------|
| MSG_DEL | Delete a message. |
| MSG_LIST | List all available messages. |

3.3 Counters

Commands used to manage counters:

| Command | Description |
|------------|---|
| CNT_LIST | List all available counters. |
| CNT_GET | Get the current value of a counter. |
| CNT_SET | Set the current value of a counter. |
| CNT_RESET | Reset the current value of a counter to the start value. |
| CNT_DEL | Delete a counter. |
| CNT_GET_RC | Get the current repeat iteration of a counter. |
| CNT_SET_RC | Set the current repeat iteration of a counter. |
| CNT_GET_I | Get the increment of a counter, which is by how much it is incremented when it is incremented. |
| CNT_SET_I | Set the increment of a counter, which is by how much it is incremented when it is incremented. |
| CNT_GET_R | Get the repeat count of a counter, which is by how many times a counter value is used before it is incremented. |
| CNT_SET_R | Set the repeat count of a counter, which is by how many times a counter value is used before it is incremented. |
| CNT_UPDATE | Update the counter value |

3.4 Database tables

Commands used to manage database tables:

| Command | Description |
|--------------|--|
| DB | Select the database table to operate on. |
| DB_ADD_TABLE | Create a new database table. |
| DB_DEL_TABLE | Delete a database table. |

Commands used to manage the selected database table:

| Command | Description |
|---------------|---|
| DB_INIT_TABLE | Initialize the database table. |
| DB_ADD_ROW | Add a row to the database table. |
| DB_DEL_ROW | Delete one or more rows from the database table. |
| DB_GET | Get values from a row in the database table. |
| DB_SET | Set values in a row in the database table. |
| DB_INC | Increment a value in a row in the database table. |

3.5 Variables

Commands used to manage variables:

| Command | Description |
|----------|---|
| VAR_LIST | List all available variables. |
| VAR_GET | Get the value of a variable. |
| VAR_SET | Set the value of a variable. |
| VAR_SET2 | Set the value of one or more variables. |
| VAR_DEL | Delete a variable. |

3.6 L-series support

Commands specific to L-series print heads:

| Command | Description |
|----------------|--|
| L_DECAP | Decap all L-series cartridges in the selected Installations. |
| L_DECAP_MARKER | Decap all L-series cartridges in the selected Marker. |
| L_DECAP_TABLE | Set the decap configuration table for a L-series ink. |

3.7 T-series support

Commands specific to T-series print heads:

| Command | Description |
|----------------|---|
| T_CLEAN | Clean all T-series print heads in the selected Installations. |
| T_CLEAN_MARKER | Clean all T-series print heads in the selected Marker. |

3.8 V-series support

Commands specific to T-series print heads:

| Command | Description |
|----------------------|--|
| V_FLUSH | Flush all V-series print heads in the selected Installations. |
| V_FLUSH_MARKER | Flush all V-series print heads in the selected Marker. |
| V_SET_CLEANER | Enable cleaner for all V-series print heads in the selected Installations. |
| V_SET_CLEANER_MARKER | Enable cleaner for all V-series print heads in the selected Marker. |

3.9 I/O control

Command for controlling external I/Os:

| Command | Description |
|------------|-------------------|
| INPUT_GET | Get input state. |
| OUTPUT_SET | Set output state. |

3.10 Miscellaneous

Miscellaneous commands:

| Command | Description |
|---------------|---|
| VERSION | Get the version of the MPERIA Controller. |
| LOG | Create a log entry in the MPERIA log. |
| PLUGIN_ACTION | Run a plugin function. |

4 Command reference

4.1 CNT_DEL

Description:

Delete a counter. Counters can only be deleted if they are not in use. Attempting to delete a counter that is in used causes a <NAK> to be returned.

Command:

```
<ESC>CNT_DEL<FS>name<ETB>
```

| Field | Description |
|-------------|----------------------|
| <i>name</i> | Name of the counter. |

Successful response:

```
<ACK>
```

Example:

To delete the counter MYCOUNTER, send:

```
<ESC>CNT_DEL<FS>MYCOUNTER<ETB>
```

Example response:

```
<ACK>
```

4.2 CNT_GET

Description:

Get the current value of a counter.

If this command is used to set a non-global counter, the following rules apply:

- If the *INST* command has been executed with an empty list of Installations, the global value of the counter will be returned.
- If the *INST* command was executed with a list of Installations, only the local value of the first Installation will be returned.

Note: Use the command INST to select which installations to operate on.

Command:

```
<ESC>CNT_GET<FS>name<ETB>
```

| Field | Description |
|-------------|----------------------|
| <i>name</i> | Name of the counter. |

Successful response:

```
<ACK>value<ETB>
```

| Field | Description |
|--------------|-----------------------|
| <i>value</i> | Value of the counter. |

Example:

To get the value of the counter MYCOUNTER, send:

```
<ESC>CNT_GET<FS>MYCOUNTER<ETB>
```

Example response:

```
<ACK>42<ETB>
```

4.3 CNT_GET_I

Description:

Get the increment of a counter, which is by how much it is incremented when it is incremented.

Command:

```
<ESC>CNT_GET_I<FS>name<ETB>
```

| Field | Description |
|-------------|----------------------|
| <i>name</i> | Name of the counter. |

Successful response:

```
<ACK>value<ETB>
```

| Field | Description |
|--------------|--------------------------|
| <i>value</i> | The counter's increment. |

Example:

To get the increment of the counter MYCOUNTER, send:

```
<ESC>CNT_GET_I<FS>MYCOUNTER<ETB>
```

Example response:

```
<ACK>1<ETB>
```

4.4 CNT_GET_R

Description:

Get the repeat count of a counter, which is by how many times a counter value is used before it is incremented.

Command:

```
<ESC>CNT_GET_R<FS>name<ETB>
```

| Field | Description |
|-------------|----------------------|
| <i>name</i> | Name of the counter. |

Successful response:

```
<ACK>value<ETB>
```

| Field | Description |
|--------------|-----------------------------|
| <i>value</i> | The counter's repeat count. |

Example:

To get the repeat count of the counter MYCOUNTER, send:

```
<ESC>CNT_GET_R<FS>MYCOUNTER<ETB>
```

Example response:

```
<ACK>1<ETB>
```

4.5 CNT_GET_RC

Description:

Get the current repeat iteration of a counter. The returned value is zero based, so if the repeat is set to N , the repeat iteration counts from zero to $N-1$.

If this command is used to set a non-global counter, the following rules apply:

- If the *INST* command has been executed with an empty list of Installations, the global repeat iteration of the counter will be returned.
- If the *INST* command was executed with a list of Installations, only the local repeat iteration of the first Installation will be returned.

Note: Use the command INST to select which installations to operate on.

Command:

```
<ESC>CNT_GET_RC<FS>name<ETB>
```

| Field | Description |
|-------------|----------------------|
| <i>name</i> | Name of the counter. |

Successful response:

```
<ACK>value<ETB>
```

| Field | Description |
|--------------|---------------------------------|
| <i>value</i> | The counter's repeat iteration. |


Example:

To get the current repeat iteration of the counter MYCOUNTER, send:

```
<ESC>CNT_GET_RC<FS>MYCOUNTER<ETB>
```

Example response:

```
<ACK>0<ETB>
```


| | | |
|--|--|--------------------|
|  Matthews Marking Systems™ | Document name Matthews MPERIA Command Protocol Manual | Page 17 (85) |
| | Document number 795 247 | Date 2017-05-23 |

4.6 CNT_LIST

Description:

List all available counters.

Command:

```
<ESC>CNT_LIST<ETB>
```

Successful response:

```
<ACK>name<FS>...<ETB>
```

Example:

To list all available counters:

```
<ESC>CNT_LIST<ETB>
```

Expected response:

```
<ACK>BATCH<FS>MYCOUNTER<FS>My Pretty Counter<ETB>
```

4.7 CNT_RESET

Description:

Reset the current value of a counter to the start value.

If this command is used to set a non-global counter, the following rules apply:

- If the *INST* command has been executed with an empty list of Installations, all instances of the counter will be set to the start value.
- If the *INST* command was executed with a list of Installations, only the local values for the listed Installations will be set to the start value.

Note: Use the command INST to select which installations to operate on.

Command:

```
<ESC>CNT_SET<FS>name<ETB>
```

| Field | Description |
|-------|----------------------|
| name | Name of the counter. |

Successful response:

<ACK>

Example:

To reset the value of the counter MYCOUNTER, send:

```
<ESC>CNT_RESET<FS>MYCOUNTER<ETB>
```

Expected response:

<ACK>

4.8 CNT_SET

Description:

Set the current value of a counter.

If this command is used to set a non-global counter, the following rules apply:

- If the *INST* command has been executed with an empty list of Installations, all instances of the counter will be set to the specified value.
- If the *INST* command was executed with a list of Installations, only the local values for the listed Installations will be set to the specified value.

Note: Use the command INST to select which installations to operate on.

Command:

```
<ESC>CNT_SET<FS>name<FS>value<ETB>
```

| Field | Description |
|--------------|---------------------------|
| <i>name</i> | Name of the counter. |
| <i>value</i> | New value of the counter. |

Successful response:

<ACK>

Example:

To set the value of the counter MYCOUNTER to 42, send:

```
<ESC>CNT_SET<FS>MYCOUNTER<FS>42<ETB>
```

Expected response:

<ACK>

4.9 CNT_SET_I

Description:

Set the increment of a counter, which is by how much it is incremented when it is incremented.

Command:

```
<ESC>CNT_SET_I<FS>name<FS>value<ETB>
```

| Field | Description |
|--------------|-------------------------------|
| <i>name</i> | Name of the counter. |
| <i>value</i> | New increment of the counter. |

Successful response:

```
<ACK>
```

Example:

To set the increment of the counter MYCOUNTER to 42, send:

```
<ESC>CNT_SET_I<FS>MYCOUNTER<FS>42<ETB>
```

Expected response:

```
<ACK>
```

4.10 CNT_SET_R

Description:

Set the repeat count of a counter, which is by how many times a counter value is used before it is incremented.

Command:

```
<ESC>CNT_SET_R<FS>name<FS>value<ETB>
```

| Field | Description |
|--------------|-----------------------------|
| <i>name</i> | Name of the counter. |
| <i>value</i> | The counter's repeat count. |

Successful response:

```
<ACK>
```

Example:

To set the repeat count of the counter MYCOUNTER to 10, send:

```
<ESC>CNT_SET_R<FS>MYCOUNTER<FS>10<ETB>
```

Expected response:

```
<ACK>
```

4.11 CNT_SET_RC

Description:

Set the current repeat iteration of a counter. The value is zero based, so if the repeat is set to N , the repeat iteration counts from zero to $N-1$.

If this command is used to set a non-global counter, the following rules apply:

- If the *INST* command has been executed with an empty list of Installations, the repeat iteration of all instances of the counter will be set to the specified value.
- If the *INST* command was executed with a list of Installations, only the repeat iteration for the listed Installations will be set to the specified value.

Note: Use the command INST to select which installations to operate on.

Command:

```
<ESC>CNT_SET_RC<FS>name<FS>value<ETB>
```

| Field | Description |
|--------------|---------------------------------|
| <i>name</i> | Name of the counter. |
| <i>value</i> | The counter's repeat iteration. |

Successful response:

```
<ACK>
```

Example:

To set the current repeat iteration of the counter MYCOUNTER to 0, send:

```
<ESC>CNT_SET_RC<FS>MYCOUNTER<FS>0<ETB>
```

Expected response:

```
<ACK>
```

4.12 CNT_UPDATE

Description:

Causes the counter to update to next value

If this command is used to set a non-global counter, the following rules apply:

- If the *INST* command has been executed with an empty list of Installations, the value of all instances of the counter will be updated.
- If the *INST* command was executed with a list of Installations, only the listed Installations will be updated to a new value.

Note: Use the command INST to select which installations to operate on.

Command:

```
<ESC>CNT_UPDATE<FS>name<ETB>
```

| Field | Description |
|-------|----------------------|
| name | Name of the counter. |

Successful response:

```
<ACK>
```

Example:

To make the counter MYCOUNTER count up, send:

```
<ESC>CNT_UPDATE<FS>MYCOUNTER<ETB>
```

Expected response:

```
<ACK>
```

4.13 DB

Description:

Select the database table that following database commands will operate on.

Command:

<ESC>DB<FS>name<ETB>

| Field | Description |
|-------------|---|
| <i>name</i> | Name of the database table. An empty name selects the default database table. If the name is not empty, and the database table does not exist, the command will fail. |

Successful response:

<ACK>

Example:

To select the database table “My table”, send:

<ESC>DB<FS>My table<ETB>

Expected response:

<ACK>

To select the default database table, send:

<ESC>DB<FS><ETB>

Expected response:

<ACK>

4.14 DB_ADD_ROW

Description:

Add a row to the database table selected by the DB command.

Command:

<ESC>DB_ADD_ROW<FS>columns<ETB>

| Field | Description |
|----------------|---|
| <i>columns</i> | Columns separated by <GS>. The command will fail if the number of columns to be added is greater than the number of columns in the table. |

Successful response:

<ACK>

Example:

Assume table with the three columns “SKU”, “Name” and “Weight”. To add a new row to the table with SKU 42, name “Apples” and weight “10 kg”, send:

<ESC>DB_ADD_ROW<FS>42<GS>Apples<GS>10 kg<ETB>

Expected response:

<ACK>

4.15 DB_ADD_TABLE

Description:

Create a new database table.

Command:

```
<ESC>DB_ADD_TABLE<FS>name<ETB>
```

| Field | Description |
|-------------|--|
| <i>name</i> | Name of the new database table. The command will fail if a table by this name already exist. |

Successful response:

```
<ACK>
```

Example:

To create a new database table called “My table”, send:

```
<ESC>DB_ADD_TABLE<FS>My table<ETB>
```

Expected response:

```
<ACK>
```

4.16 DB_DEL_ROW

Description:

Delete rows from the database table selected by the DB command..

Command:

```
<ESC>DB_DEL_ROW<FS>selector<ETB>
```

| Field | Description |
|-----------------|---|
| <i>selector</i> | A list of column/value pairs, separated by <GS>. Each column/value pair is separated by <RS>. |

All rows with a matching value will be deleted.

Successful response:

```
<ACK>
```

Example 1:

To delete all rows where the column “Name” is “Apples”, send:

```
<ESC>DB_DEL_ROW<FS>Name<RS>Apples<ETB>
```

Expected response:

```
<ACK>
```

Example 2:

To delete all rows where the column “Name” is “Apples” and the column “Origin” is “Italy”, send:

```
<ESC>DB_DEL_ROW<FS>Name<RS>Apples<GS>Origin<RS>Italy<ETB>
```

Expected response:

```
<ACK>
```

4.17 DB_DEL_TABLE

Description:

Delete a database table.

Command:

```
<ESC>DB_DEL_TABLE<FS>name<ETB>
```

| Field | Description |
|-------------|---|
| <i>name</i> | Name of the database table to be deleted. |

Successful response:

```
<ACK>
```

Example:

To delete a database table called “My table”, send:

```
<ESC>DB_DEL_TABLE<FS>My table<ETB>
```

Expected response:

```
<ACK>
```

4.18 DB_GET

Description:

Get values from the database table selected by the DB command..

Command:

```
<ESC>DB_GET<FS>selector<FS>columns<ETB>
```

| Field | Description |
|-----------------|---|
| <i>selector</i> | A list of column/value pairs, separated by <GS>. Each column/value pair is separated by <RS>. |
| <i>columns</i> | Columns for which to get the values, separated by <GS>. |

Successful response:

```
<ACK>columns<ETB>
```

| Field | Description |
|----------------|------------------------------------|
| <i>columns</i> | List of values, separated by <GS>. |

Example 1:

Given the following database table:

| SKU | Name | Origin |
|-----|---------|---------|
| 13 | Kiwi | Israel |
| 42 | Apples | Italy |
| 88 | Oranges | Morocco |

To get the value of the column “Origin” from the first row where the column “Name” is “Apples”, send:

```
<ESC>DB_GET<FS>Name<RS>Apples<FS>Origin<ETB>
```

Expected response:

```
<ACK>Italy<ETB>
```

Example 2:

Given the database table from example 1, to get the values of the columns “SKU” and “Origin” from the first row where the column “Name” is “Apples”, send:

```
<ESC>DB_GET<FS>Name<RS>Apples<FS>SKU<GS>Origin<ETB>
```

Expected response:

```
<ACK>42<GS>Italy<ETB>
```

4.19 DB_INC

Description:

Increment a numeric value in a cell in the database table selected by the DB command..

Command:

1. <ESC>DB_INC<FS>selector<FS>column<ETB>
2. <ESC>DB_INC<FS>selector<FS>column<FS>increment<ETB>

| Field | Description |
|------------------|---|
| <i>selector</i> | A list of column/value pairs, separated by <GS>. Each column/value pair is separated by <RS>. |
| <i>column</i> | Column to increment. |
| <i>increment</i> | Optional value to increment with. Equals to one if not supplied. |

All rows with a matching value will be deleted.

Successful response:

<ACK>

Example 1:

Given the following database table:

| SKU | Name | Count |
|-----|---------|-------|
| 13 | Kiwi | 10 |
| 42 | Apples | 7 |
| 88 | Oranges | 0 |

To increment the Count column by one in the rows where the column “Name” is “Apples”, send:

<ESC>DB_INC<FS>Name<RS>Apples<FS>Count<ETB>

This will change the Count from 7 to 8.

Example 2:

Given the same table as in Example 1, to subtract four from Count column in the rows where the column “SKU” is 13, send:

<ESC>DB_INC<FS>SKU<RS>13<FS>Count<FS>-4<ETB>

This will change the Count from 10 to 6.

4.20 DB_INIT_TABLE

Description:

Initialize the database table selected by the DB command.

Command:

1. <ESC>DB_INIT_TABLE<FS>columns<ETB>
2. <ESC>DB_INIT_TABLE<FS>columns<FS>defcol<ETB>

| Field | Description |
|---------|--|
| columns | Columns separated by <GS>. |
| defcol | Default column index, where zero refers to the first column. |

Successful response:

<ACK>

Example 1:

To initialize the current database table to have three columns “SKU”, “Name” and “Weight”, send:

<ESC>DB_INIT_TABLE<FS>SKU<GS>Name<GS>Count<ETB>

Example 2:

To initialize the current database table to have three columns “SKU”, “Name” and “Weight”, and make “Name” the default column in the Controller's user interface, send:

<ESC>DB_INIT_TABLE<FS>SKU<GS>Name<GS>Count<FS>1<ETB>

4.21 DB_SET

Description:

Set the value in a cell in the database table selected by the DB command.

Command:

```
<ESC>DB_SET<FS>selector<FS>values<ETB>
```

| Field | Description |
|-----------------|---|
| <i>selector</i> | A list of column/value pairs, separated by <GS>. Each column/value pair is separated by <RS>. |
| <i>values</i> | A list of column/value pairs, separated by <GS>. Each column/value pair is separated by <RS>. |

Successful response:

```
<ACK>
```

Example 1:

Given the following database table:

| SKU | Name | Origin |
|-----|---------|---------|
| 13 | Kiwi | Israel |
| 42 | Apples | Italy |
| 88 | Oranges | Marocco |

To change the Origin of SKU 42 (Apples) to Sweden, send:

```
<ESC>DB_SET<FS>SKU<RS>42<FS>Origin<RS>Sweden<ETB>
```

Example 2:

Given the following database table:

| SKU | Name | Origin | Color | Price |
|-----|---------|---------|--------|-------|
| 13 | Kiwi | Israel | Green | 100 |
| 42 | Apples | Italy | Green | 200 |
| 43 | Apples | Norway | Red | 220 |
| 88 | Oranges | Morocco | Yellow | 150 |

To change the Color of Norwegian apples to Green and its price to 210, send:

```
<ESC>DB_SET
<FS>Origin<RS>Norway<GS>Name<RS>Apples
<FS>Color<RS>Green<FS>Price<RS>210
<ETB>
```

Note: The above should be sent as a single line without extra spaces. It has been split into lines to make it more readable.

4.22 ENC_BLOCK

Description:

Blocks the encoder for all Installations selected using the INST command from detecting motion. Locking their position to the same value even if the physical encoder signals motion. If a failure to lock any Installation's encoder was detected **NAK** will be returned.

This command can not be used for Installations using fixed speed.

Command:

```
<ESC>ENC_BLOCK<FS>state<ETB>
```

| Field | Description |
|-------|--|
| state | True if the encoder should be blocked, false to unblock it and start detecting motion. |

Successful response:

```
<ACK>
```

Example:

To block the encoder for Installation "a" from detecting motion:

```
<ESC>INST<FS>Production Line 1<ETB>
<ESC>ENC_BLOCK<FS>true<ETB>
```

Example response:

```
<ACK>
<ACK>
```

| | | |
|--|--|--------------------|
|  Matthews Marking Systems™ | Document name Matthews MPERIA Command Protocol Manual | Page 34 (85) |
| | Document number 795 247 | Date 2017-05-23 |

4.23 ENC_RESET_REVERSE

Description:

Resets the encoder's back filter for all Installations selected by the INST command, making the next forward motion detectable. If the call to reset failed for any Installation **NAK** will be returned.

This does not work when the Installation is configured for fixed speed.

Command:

```
<ESC>ENC_RESET_REVERSE<ETB>
```

Successful response:

```
<ACK>
```


Example:

To reset the back state for all encoders on Installations:

```
<ESC>INST<ETB>
<ESC>ENC_RESET_REVERSE<ETB>
```

Example response:

```
<ACK>
<ACK>
```

| | | |
|--|--|--------------------|
|  Matthews Marking Systems™ | Document name Matthews MPERIA Command Protocol Manual | Page 35 (85) |
| | Document number 795 247 | Date 2017-05-23 |

4.24 GOOD

Description:

Test if all is good about all Installations selected using the INST command, meaning that they do not require service and have no critical problems.

Command:

```
<ESC>GOOD<ETB>
```

Successful response:

```
<ACK>
```

Example:

To get the status of the selected Installation, send:

```
<ESC>GOOD<ETB>
```

Example response if all is well:

```
<ACK>
```

4.25 INPUT_GET

Description:

Get the current state of an input pin.

Command:

```
<ESC>INPUT_GET<FS>io<ETB>
```

| Field | Description |
|-----------|---|
| <i>io</i> | Name of the I/O as defined in the External I/O configuration. |

Successful response:

```
<ACK>state<ETB>
```

Examples:

To get the state of the input pin defined as I0:

```
<ESC>INPUT_GET<FS>I0<ETB>
```

Example response if all is well:

```
<ACK>0<ETB>
```

4.26 INST

Description:

Define the Installation or Installations on which other command operate.

Command:

1. <ESC>INST<ETB>
2. <ESC>INST<FS>*installation₁* ... <FS>*installation_N*<ETB>

| Field | Description |
|---------------------------------|---|
| <i>installation_k</i> | Name or numeric ID of the Installation. |

Successful response:

<ACK>

Examples:

To make subsequent command operate on all Installations:

<ESC>INST<ETB>

To make subsequent command operate only on Installation “Production Line 1”:

<ESC>INST<FS>Production Line 1<ETB>

To make subsequent command operate only on the Installations “Production Line 2” and “Production Line 3”:

<ESC>INST<FS>Production Line 2<FS>Production Line 3<ETB>

4.27 L_DECAP

Description:

Decap all L-series cartridges in all Installations selected using the INST command.

Command:

```
<ESC>L_DECAP<FS>count<ETB>
```

| Field | Description |
|--------------|---|
| <i>count</i> | Number of times to decap. Must be less than 1024. |

Successful response:

```
<ACK>
```

Example:

To decap 100 times of ink with all L-series cartridges in the selected Installations, send:

```
<ESC>L_DECAP<FS>100<ETB>
```

Expected response:

```
<ACK>
```

4.28 L_DECAP_MARKER

Description:

Decap all L-series cartridges in the specified Marker in all Installations selected using the INST command.

Command:

```
<ESC>L_DECAP_MARKER<FS>marker<FS>count<ETB>
```

| Field | Description |
|---------------|---|
| <i>marker</i> | Name of the Marker. |
| <i>count</i> | Number of times to decap. Must be less than 1024. |

Successful response:

```
<ACK>
```

Example:

To decap 100 times of ink with all L-series cartridges in the Marker “Operator Side” in the selected Installations, send:

```
<ESC>L_DECAP_MARKER<FS>Operator Side<FS>100<ETB>
```

Expected response:

```
<ACK>
```

4.29 L_DECAP_TABLE

Description:

Set the decap configuration table for a L-series ink and reconfigure all Installations selected using the INST command.

Note: Use the command INST to select which installations the command shall have immediate effect on.

Command:

```
<ESC>L_DECAP_TABLE<FS>ink<FS>configuration<ETB>
```

| Field | Description |
|----------------------|---|
| <i>ink</i> | Name of the ink. |
| <i>configuration</i> | A list of time/count pairs, separated by <GS>. Each time/count pair is separated by <RS>. |

Time is in seconds since the last printed mark and count is the number of times to decap. Count must be less than 1024.

Successful response:

```
<ACK>
```

Example:

To set the decap configuration table for ink LS-6020 so that it will decap 30 times after 10 seconds and 110 times after 200 seconds, send:

```
<ESC>L_DECAP_TABLE<FS>LS-6020<FS>10<RS>30<FS>200<RS>110<ETB>
```

Expected response:

```
<ACK>
```


4.30 LOG

Description:

Create a log entry in the MPERIA log.

Command:

```
<ESC>LOG<FS>severity<FS>text<ETB>
<ESC>LOG<FS>severity<FS>text<FS>info<ETB>
```

| Field | Description |
|-----------------|--|
| <i>severity</i> | Severity. Must be one of NONE, DEBUG, INFO, SERVICE, CRITICAL. |
| <i>text</i> | Text to add to the log. Must be a single line. |
| <i>info</i> | Optional extra information. Can be multiple lines. |

Successful response:

```
<ACK>
```

Example:

To add a service log message, send:

```
<ESC>LOG<FS>SERVICE<FS>Replace whozz-o-matic now!<ETB>
```

Expected response:

```
<ACK>
```

4.31 MSG_DEL

Description:

Delete message from the message database. If the message can not be found **NAK** is returned.

Command:

1. `<ESC>MSG_DEL<FS>message<ETB>`

| Field | Description |
|----------------|---------------|
| <i>message</i> | Message name. |

Successful response:

`<ACK>`


Example 1:

To delete the message “My Message”, send:

`<ESC>MSG_DEL<FS>My Message<ETB>`

Expected response:

`<ACK>`

| | | |
|--|--|--------------------|
|  Matthews Marking Systems™ | Document name Matthews MPERIA Command Protocol Manual | Page 43 (85) |
| | Document number 795 247 | Date 2017-05-23 |

4.32 MSG_LIST

Description:

List all messages from the message database.

Command:

1. `<ESC>MSG_LIST<ETB>`

Successful response:

`<ACK>`

Example 1:

To delete the message “My Message”, send:

`<ESC>MSG_LIST<ETB>`

Expected response:

`<ACK>`


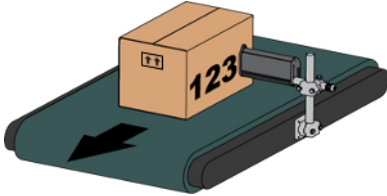
4.33 MARKER_DIR

Description:

Set the print direction of the specified Marker in all Installations selected using the INST command.

Command:

```
<ESC>MARKER_DIR<FS>marker<FS>direction<ETB>
```

| Field | Description |
|------------------|---|
| <i>marker</i> | Name of the Marker. |
| <i>direction</i> | Print direction: <ul style="list-style-type: none"> 0 = Left to Right:  <ul style="list-style-type: none"> 1 = Right to Left:  |

Successful response:

```
<ACK>
```

Example:

To set the print direction of the Marker “Operator Side” in the selected Installations to Left-to-Right, send:

```
<ESC>MARKER_DIR<FS>Operator Side<FS>1<ETB>
```

Expected response:

```
<ACK>
```

4.34 MARKER_MARGIN

Description:

Set the margin of the specified Marker in all Installations selected using the INST command.

Command:

```
<ESC>MARKER_MARGIN<FS>marker<FS>margin<ETB>
```

| Field | Description |
|------------------|------------------------|
| <i>marker</i> | Name of the Marker. |
| <i>direction</i> | Margin in millimeters. |

Successful response:

```
<ACK>
```

Example:

To set the margin of the Marker “Operator Side” in the selected Installations to 25.4 mm, send:

```
<ESC>MARKER_MARGIN<FS>Operator Side<FS>25.4<ETB>
```

Expected response:

```
<ACK>
```

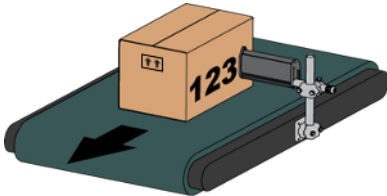

4.35 MARKER_UPSIDEDOWN

Description:

Set whether the specified Marker in all Installations selected using the INST command shall print upside down or not.

Command:

```
<ESC>MARKER_UPSIDEDOWN<FS>marker<FS>upsideup<ETB>
```

| Field | Description |
|-----------------|--|
| <i>marker</i> | Name of the Marker. |
| <i>upsideup</i> | Print upside down: <ul style="list-style-type: none"> 0 = Normal:  <ul style="list-style-type: none"> 1 = Upside down:  |

Successful response:

```
<ACK>
```

Example:

To make the Marker “Operator Side” in the selected Installations print upside down, send:

```
<ESC>MARKER_MARGIN<FS>Operator Side<FS>1<ETB>
```

Expected response:

```
<ACK>
```

4.36 MARK_GAP

Description:

Set the gap between repeated marks for the specified Marker in all Installations selected using the INST command.

Command:

```
<ESC>MARK_GAP<FS>marker<FS>gap<ETB>
```

| Field | Description |
|---------------|--------------------------|
| <i>marker</i> | Name of the Marker. |
| <i>gap</i> | Mark gap in millimeters. |

Successful response:

```
<ACK>
```

Example:

To set the mark gap for the Marker “Operator Side” in the selected Installations to 25.4 mm, send:

```
<ESC>MARKER_GAP<FS>Operator Side<FS>25.4<ETB>
```

Expected response:

```
<ACK>
```

4.37 MARK_INTERVAL

Description:

Set the interval between repeated marks for the specified Marker in all Installations selected using the INST command.

Command:

```
<ESC>MARK_INTERVAL<FS>marker<FS>interval<ETB>
```

| Field | Description |
|-----------------|-------------------------------|
| <i>marker</i> | Name of the Marker. |
| <i>interval</i> | Mark interval in millimeters. |

Successful response:

```
<ACK>
```


Example:

To set the mark interval for the Marker “Operator Side” in the selected Installations to 25.4 mm, send:

```
<ESC>MARKER_INTERVAL<FS>Operator Side<FS>25.4<ETB>
```

Expected response:

```
<ACK>
```


| | | |
|--|--|--------------------|
|  Matthews Marking Systems™ | Document name Matthews MPERIA Command Protocol Manual | Page 49 (85) |
| | Document number 795 247 | Date 2017-05-23 |

4.38 NONCRITICAL

Description:

Test if all Installations selected using the INST command do not have critical problems preventing them from printing.

Command:

```
<ESC>NONCRITICAL<ETB>
```

Successful response:

```
<ACK>
```


Example:

To test if there are no critical errors in the selected Installations, send:

```
<ESC>NONCRITICAL<ETB>
```

Example response if all is well:

```
<ACK>
```

| | | |
|--|--|--------------------|
|  Matthews Marking Systems™ | Document name Matthews MPERIA Command Protocol Manual | Page 50 (85) |
| | Document number 795 247 | Date 2017-05-23 |

4.39 OFFLINE

Description:

Set all Installations selected by the INST command in offline mode.

Command:

```
<ESC>OFFLINE<ETB>
```

Successful response:

```
<ACK>
```

Example:

To set all Installations in offline mode, send:

```
<ESC>INST<ETB>
<ESC>OFFLINE<ETB>
```

Example response if all is well:


```
<ACK>
<ACK>
```

To set Installations “First Installation” and “Second Installation” in offline mode, send:

```
<ESC>INST<FS>First Installation<FS>Second Installation<ETB>
<ESC>OFFLINE<ETB>
```

Example response if all is well:

```
<ACK>
<ACK>
```

| | | |
|--|--|--------------------|
|  Matthews Marking Systems™ | Document name Matthews MPERIA Command Protocol Manual | Page 51 (85) |
| | Document number 795 247 | Date 2017-05-23 |

4.40 ONLINE

Description:

Set all Installations selected by the INST command in online mode.

Command:

```
<ESC>ONLINE<ETB>
```

Successful response:

```
<ACK>
```

Example:

To set all Installations in online mode, send:

```
<ESC>INST<ETB>
<ESC>ONLINE<ETB>
```

Example response if all is well:

```
<ACK>
<ACK>
```

To set Installations “First Installation” and “Second Installation” in online mode, send:

```
<ESC>INST<FS>First Installation<FS>Second Installation<ETB>
<ESC>ONLINE<ETB>
```

Example response if all is well:

```
<ACK>
<ACK>
```

4.41 OUTPUT_SET

Description:

Set the value of an external I/O.

Command:

```
<ESC>OUTPUT_SET<FS>io<FS>state<FS>delay<FS>onTime<FS>offTime<FS>
numTransitions<ETB>
```

| Field | Description |
|-----------------------|---|
| <i>io</i> | I/O pin configuration name specified under the External I/O interface. |
| <i>state</i> | Initial state to set. |
| <i>delay</i> | Time before the state specified by the state attribute gets set the first time. |
| <i>onTime</i> | Time spent in positive state specified in ms. |
| <i>offTime</i> | Time spent in negative state specified in ms. |
| <i>numTransitions</i> | Number of state transitions after the initial value has been set. |

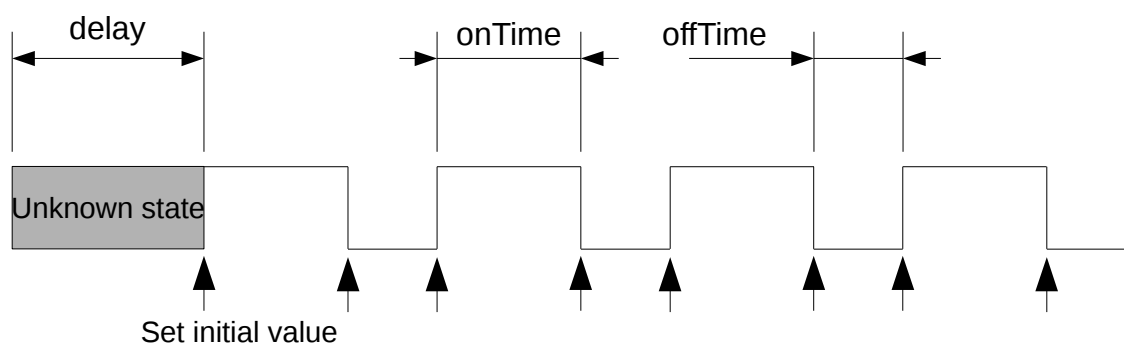
Successful response:

```
<ACK>
```

Example:

The request below for a pin with an negative initial state will give a result described by the following picture. Transitions are marked as arrows at the bottom of the picture.

```
<ESC>OUTPUT_SET<FS>O0<FS>1<FS>1200<FS>1000<FS>600<FS>7<ETB>
```



4.42 PEEKMSG

Description:

Get the name of the currently selected message on the Installation selected using the INST command. This command fails if more than one Installation is selected.

Command:

```
<ESC>PEEKMSG<ETB>
```

Successful response:

```
<ACK>name<ETB>
```

| Field | Description |
|-------------|--|
| <i>name</i> | Name of the currently selected message, or an empty value if no message is selected. |

Example:

To get the name of the currently selected message, send:

```
<ESC>PEEKMSG<ETB>
```

Example response:

```
<ACK>My Message<ETB>
```

4.43 PLUGIN_ACTION

Description:

Run the *plugin_action* entry point in the specified Lua plugin. See the Lua plugin manual for more information.

Command:

1. <ESC>PLUGIN_ACTION<FS>*plugin name*<ETB>
2. <ESC>PLUGIN_ACTION<FS>*plugin name*<FS>*input*<ETB>
3. <ESC>PLUGIN_ACTION<FS>*plugin name*<FS>*input*<FS>*config*<ETB>

| Field | Description |
|--------------------|--|
| <i>plugin name</i> | The name of the plugin. |
| <i>input</i> | Input to the plugin_action function. |
| <i>config</i> | Configuration to the plugin_action function. |

Successful response:

<ACK>*response*<ETB>

| Field | Description |
|-----------------|---|
| <i>response</i> | The returned string value from the plugin function. |


Example:

To call *example_plugin.lua*, send:

<ESC>PLUGIN_ACTION<FS>*example_plugin.lua*<FS>Hello World!<ETB>

Example response:

<ACK>

| | | |
|--|--|--------------------|
|  Matthews Marking Systems™ | Document name Matthews MPERIA Command Protocol Manual | Page 55 (85) |
| | Document number 795 247 | Date 2017-05-23 |

4.44 PRINT_CNT

Description:

Get the print count for all messages, or the for a specific message, on all Installations selected using the INST command. Note that if the INST command is used without arguments to select all Installation, the print count will also include Installations that are no longer available.

Command:

1. <ESC>PRINT_CNT<ETB>
2. <ESC>PRINT_CNT<FS>*message*<ETB>

Successful response:

<ACK>*count*<ETB>

Example:

To get the print count of all messages, send:

<ESC>PRINT_CNT<ETB>

Example response:


<ACK>56723<ETB>

To get the total print count for the message “My Message”, send:

<ESC>PRINT_CNT<FS>MyMessage<ETB>

Example response:

<ACK>42<ETB>

| | | |
|--|--|--------------------|
|  Matthews Marking Systems™ | Document name Matthews MPERIA Command Protocol Manual | Page 56 (85) |
| | Document number 795 247 | Date 2017-05-23 |

4.45 PRINT_CNT_RESET

Description:

Reset the print count for all messages, or the for a specific message, on all Installations selected using the INST command. Note that if the INST command is used without arguments to select all Installation, the print count will also be reset for Installations that are no longer available.

Command:

1. <ESC>PRINT_CNT_RESET<ETB>
2. <ESC>PRINT_CNT_RESET<FS>message<ETB>

Successful response:

<ACK>

Example:

To reset the total print count of all messages, send:

<ESC>PRINT_CNT_RESET<ETB>

Example response:


<ACK>

To reset the total print count for the message “My Message”, send:

<ESC>PRINT_CNT_RESET<FS>MyMessage<ETB>

Example response:

<ACK>

| | | |
|--|--|--------------------|
|  Matthews Marking Systems™ | Document name Matthews MPERIA Command Protocol Manual | Page 57 (85) |
| | Document number 795 247 | Date 2017-05-23 |

4.46 QUEUE

Description:

Fixate and queue the selected message on all Installations selected using the INST command. The installations must have Automatic Message Fixation disabled for the QUEUE command to succeed.

Command:

```
<ESC>QUEUE<ETB>
```

Successful response:

```
<ACK>
```

Example:

To fixate and queue the selected message on all Installations, send:

```
<ESC>INST<ETB>
<ESC>QUEUE<ETB>
```

Example response:

```
<ACK>
<ACK>
```

4.47 QUEUE_LEN

Description:

Get the queue length on the Installation selected using the INST command. The command fails if more than one Installation is selected.

Command:

```
<ESC>QUEUE_LEN<ETB>
```

Successful response:

```
<ACK>length<ETB>
```

| Field | Description |
|---------------|---------------------------|
| <i>length</i> | The current queue length. |


Example:

To get the version of the MPERIA Controller, send:

```
<ESC>QUEUE_LEN<ETB>
```

Example response:

```
<ACK>5<ETB>
```

| | | |
|--|--|--------------------|
|  Matthews Marking Systems™ | Document name Matthews MPERIA Command Protocol Manual | Page 59 (85) |
| | Document number 795 247 | Date 2017-05-23 |

4.48 QUEUE_RESET

Description:

Reset the queue on all Installations selected using the INST command. The installations must have Automatic Message Fixation disabled for the QUEUE command to succeed.

Command:

```
<ESC>QUEUE_RESET<ETB>
```

Successful response:

```
<ACK>
```

Example:

To reset the queue on all Installations, send:

```
<ESC>INST<ETB>
<ESC>RESET<ETB>
```

Example response:

```
<ACK>
<ACK>
```

4.49 SEL

Description:

Select message and database row for printing on all Installations selected using the INST command. The database column and value must exist in each Installation's configured database table. If the message cannot be selected for an Installation, or the database cell cannot be found, that currently selected message is deselected, the database row becomes invalid, and **NAK** is returned.

Command:

1. `<ESC>SEL<FS>message<FS>DB column<FS>DB value<ETB>`
2. `<ESC>SEL<FS>message<FS>DB column<FS>DB value<FS>max print<ETB>`

| Field | Description |
|------------------|--|
| <i>message</i> | Message name. An empty value deselects the message. |
| <i>DB column</i> | The database column to search in. |
| <i>DB value</i> | The database value to search for. |
| <i>max print</i> | Optional maximum number of times the message can be printed without being deselected. A negative value (which is the default) means that the message will never be automatically deselected. |

Successful response:

`<ACK>`

Example:

To select the message “My Message” and the database row where the column “Origin” is “India”, send:

`<ESC>SEL<FS>My Message<FS>Origin<FS>India<ETB>`

Expected response:

`<ACK>`

4.50 SELDB

Description:

Select database row for printing on all Installations selected using the INST command. The database column and value must exist in each Installation's configured database table. If the database cell cannot be found, the database row becomes invalid, and **NAK** is returned.

Command:

1. `<ESC>SELDB<FS>DB column<FS>DB value<ETB>`
2. `<ESC>SELDB<FS>DB column<FS>DB value<FS>max print<ETB>`

| Field | Description |
|------------------|--|
| <i>DB column</i> | The database column to search in. |
| <i>DB value</i> | The database value to search for. |
| <i>max print</i> | Optional maximum number of times the message can be printed without being deselected. A negative value (which is the default) means that the message will never be automatically deselected. |

Successful response:

`<ACK>`

Example:

To select the database row where the column "Origin" is "India", send:

`<ESC>SELDB<FS>Origin<FS>India<ETB>`

Expected response:

`<ACK>`

4.51 SELMSG

Description:

Select message for printing on all Installations selected using the INST command. If the message cannot be selected for an Installation, that currently selected message is deselected, and **NAK** is returned.

Command:

1. `<ESC>SELMSG<FS>message<ETB>`
2. `<ESC>SELMSG<FS>message<FS>max print<ETB>`

| Field | Description |
|------------------|--|
| <i>message</i> | Message name. An empty value deselects the message. |
| <i>max print</i> | Optional maximum number of times the message can be printed without being deselected. A negative value (which is the default) means that the message will never be automatically deselected. |

Successful response:

`<ACK>`

Example 1:

To select the message “My Message”, send:

`<ESC>SELMSG<FS>My Message<ETB>`

Expected response:

`<ACK>`

Example 2:

To select the message “My Message” to be printed only once, send:

`<ESC>SELMSG<FS>My Message<FS>1<ETB>`

Expected response:

`<ACK>`

4.52 SET_SPEED

Description:

Set the fixed speed of all Installations selected by the INST command. Note that the installations must already be configured to use fixed speed for this command to succeed. This command will not change the configuration from using an encoder to using fixed speed.

Command:

```
<ESC>SET_SPEED<FS>speed<ETB>
```

| Field | Description |
|--------------|--|
| <i>speed</i> | Fixed speed in meters per second as a floating point number using period (.) as decimal separator. |

Successful response:

```
<ACK>
```


Example:

To set the fixed speed to 1.42 m/s, send:

```
<ESC>SET_SPEED<FS>1.42<ETB>
```

Expected response:

```
<ACK>
```

| | | |
|--|--|--------------------|
|  Matthews Marking Systems™ | Document name Matthews MPERIA Command Protocol Manual | Page 64 (85) |
| | Document number 795 247 | Date 2017-05-23 |

4.53 START

Description:

Start all Installations selected by the INST command.

Command:

```
<ESC>START<ETB>
```

Successful response:

```
<ACK>
```

Example:

To start all Installations, send:

```
<ESC>INST<ETB>
<ESC>START<ETB>
```

Example response if all is well:

```
<ACK>
<ACK>
```

To start Installations “First Installation” and “Second Installation”, send:

```
<ESC>INST<FS>First Installation<FS>Second Installation<ETB>
<ESC>START<ETB>
```

Example response if all is well:

```
<ACK>
<ACK>
```


4.54 STATE

Description:

Get the current state of a single Installation selected using the INST command.

Command:

```
<ESC>STATE<ETB>
```

Successful response:

```
<ACK>state<ETB>
```

The following states are possible:

| Status | Description |
|----------------------|--|
| <i>OFFLINE</i> | The Installation is stopped. |
| <i>INCOMPLETE</i> | The Installation is missing PTMs. |
| <i>STARTING</i> | The Installation is starting up. |
| <i>STARTED</i> | The Installation is started. |
| <i>PAUSED</i> | The Installation is stopped. |
| <i>STOPPING</i> | The Installation is stopping. |
| <i>RESTART_DELAY</i> | The Installation failed, and will be restarted after a delay. |
| <i>UNLICENSED</i> | The Installation cannot be started because of a license problem. |

Example:

To get the state of the selected Installation, send:

```
<ESC>STATE<ETB>
```

Example response:

```
<ACK>STARTED<ETB>
```

4.55 STATUS

Description:

Get the status of all Installations selected using the INST command.

Command:

```
<ESC>STATUS<ETB>
```

Successful response:

1. <ACK><ETB>
2. <ACK>status₁ ... <FS>status_N<ETB>

| Field | Description |
|---------------------|--------------------|
| status _k | Status descriptor. |


The first form is returned if all is good. The second form returns a list of status strings. The following status items are possible:

| Status | Description |
|--------------------------|---------------------------------|
| <i>missing</i> | PTM is missing |
| <i>updating</i> | PTM is being updated |
| <i>broken</i> | PTM communication error |
| <i>brokenMsg</i> | Message Error |
| <i>stolen</i> | PTM used by other Controller |
| <i>inkLow</i> | Low ink level |
| <i>inkEmpty</i> | Out of ink |
| <i>tempLow</i> | Low temperature |
| <i>crtUnknown</i> | Unknown cartridge |
| <i>crtMixed</i> | Mixed cartridges |
| <i>crtUnauthorized</i> | Cartridge not permitted |
| <i>crtUnsupported</i> | Unsupported cartridge |
| <i>bulkUnauthorized</i> | Bulk ink is not permitted |
| <i>crtMissing</i> | No cartridge |
| <i>crtBroken</i> | Broken cartridge |
| <i>crtOvertemp</i> | Overheating cartridge |
| <i>crtAlmostWornOut</i> | Bulk cartridge almost worn out. |
| <i>crtOther</i> | Other cartridge error |
| <i>inkLicenseWarning</i> | Ink license warning |
| <i>inkLicenseError</i> | Ink license error |

| | |
|-------------------------|--|
| <i>ptmLicenseError</i> | License for Number of PTMs |
| <i>crtLicenseError</i> | License for number of cartridges |
| <i>techLicenseError</i> | The license for number of technologies has been exceeded |
| <i>dodCleaner</i> | The marker is using cleaner instead of ink |
| <i>wrongState</i> | Wrong state |

The following status items are specific to Autolabel markers.

| Status | Description |
|---------------------------|---|
| paper_low | Paper low detected |
| low_air_pressure | Air pressure below set limit |
| ribbon_low | Ribbon low detected |
| rewinder_almost_full | The rewinder for backing paper is almost full |
| trig_outside_window | The trigger was activated outside the print window |
| trig_during_print | The trigger was activated during print |
| trig_without_data | The trigger was activated but no dynamic data was available |
| trig_open_head | The trigger was activated with the print head open |
| trig_righter_not_ready | The trigger was activated but the righter was not ready |
| trig_applicator_not_ready | The trigger was activated but the applicator was not ready |
| missing_trigger | Missing apply trig within the specified time after print trig |
| paper_out | Paper out detected |
| no_air_pressure | The air pressure is below the set limit |
| printhead_open | The print head is open |
| rewinder_full | The rewinder for backing paper is full |
| layout_not_found | The selected layout is not found in the internal database |
| rendering_error | There was an error when rendering the layout |
| ribbon_out | Out of thermal transfer ribbon |
| applicator_error | An error was detected when applying the label |
| applicator_stopped | Pallet applicator stop button is active |
| label_error | Label stuck/label lost detected when applying the label |
| adjusting_media | Adjusting label media after closing the print head |

| | | |
|---|---|---------------------------|
|  Matthews Marking Systems™ | Document name Matthews MPERIA Command Protocol Manual | Page 68 (85) |
| | Document number 795 247 | Date 2017-05-23 |

Example:

To get the status of the selected Installation, send:


<ESC>STATUS<ETB>

Example response if all is OK:

<ACK><ETB>

Example response if the ink level is low and a cartridge is missing:

<ACK>inkLow<FS>crtMissing<ETB>

| | | |
|--|--|--------------------|
|  Matthews Marking Systems™ | Document name Matthews MPERIA Command Protocol Manual | Page 69 (85) |
| | Document number 795 247 | Date 2017-05-23 |

4.56 STOP

Description:

Stop all Installations selected by the INST command.

Command:

```
<ESC>STOP<ETB>
```

Successful response:

```
<ACK>
```

Example:

To stop all Installations, send:

```
<ESC>INST<ETB>
<ESC>STOP<ETB>
```

Example response if all is well:

```
<ACK>
<ACK>
```

To stop Installations “First Installation” and “Second Installation”, send:

```
<ESC>INST<FS>First Installation<FS>Second Installation<ETB>
<ESC>STOP<ETB>
```

Example response if all is well:

```
<ACK>
<ACK>
```

4.57 T_CLEAN

Description:

Clean all T-series print heads in all Installations selected using the INST command.

Command:

```
<ESC>T_CLEAN<FS>duration<ETB>
```

| Field | Description |
|-----------------|---|
| <i>duration</i> | Optional duration in milliseconds. The default value is 300 ms. |

Successful response:

```
<ACK>
```

Example:

To clean all T-series print heads in the installation “Production Line 1”, send:

```
<ESC>INST<FS>Production Line 1<ETB>
<ESC>T_CLEAN<ETB>
```

Expected response:

```
<ACK>
<ACK>
```

4.58 T_CLEAN_MARKER

Description:

Clean all T-series print heads in the specified Marker in all Installations selected using the INST command.

Command:

```
<ESC>T_CLEAN<FS>name<FS>duration<ETB>
```

| Field | Description |
|-----------------|---|
| <i>name</i> | Marker name. |
| <i>duration</i> | Optional duration in milliseconds. The default value is 300 ms. |

Successful response:

```
<ACK>
```

Example:

To clean all T-series print heads in the marker “Operator Side” in Installation “Production Line 1”, send:

```
<ESC>INST<FS>Production Line 1<ETB>
<ESC>T_CLEAN_MARKER<FS>Operator Side<ETB>
```

Expected response:

```
<ACK>
<ACK>
```

4.59 TRIG

Description:

Soft trigger all Installations selected using the INST command.

Command:

<ESC>TRIG<FS>activate<ETB>

| Field | Description |
|-----------------|--|
| <i>activate</i> | Optional bool value indicating if the trigger shall be activated or deactivated. If true, the trigger is activated, if false, the trigger is deactivated. If not present the trigger will first be activated and then immediately deactivated. |

Successful response:

<ACK>


Example:

To soft trigger all installations, send:

<ESC>INST<ETB>
<ESC>TRIG<ETB>

Example response:

<ACK>
<ACK>

| | | |
|--|--|--------------------|
|  Matthews Marking Systems™ | Document name Matthews MPERIA Command Protocol Manual | Page 73 (85) |
| | Document number 795 247 | Date 2017-05-23 |

4.60 TRIG_ACTIVE

Description:

Get the current trigger state of the first Installation selected using the INST command.

Command:

```
<ESC>TRIG_ACTIVE<ETB>
```

Successful response:

```
<ACK>state<ETB>
```


Examples:

To get the state of the trigger:

```
<ESC>TRIG_ACTIVE<ETB>
```

Example response if the trigger is active:

```
<ACK>1<ETB>
```

| | | |
|--|--|--------------------|
|  Matthews Marking Systems™ | Document name Matthews MPERIA Command Protocol Manual | Page 74 (85) |
| | Document number 795 247 | Date 2017-05-23 |

4.61 V_FLUSH

Description:

Flush all V-series print heads in all Installations selected using the INST command.

Command:

```
<ESC>V_FLUSH<ETB>
```

Successful response:

```
<ACK>
```

Example:

To flush all V-series print heads in the installation “Production Line 1”, send:

```
<ESC>INST<FS>Production Line 1<ETB>
<ESC>V_FLUSH<ETB>
```

Expected response:

```
<ACK>
<ACK>
```

4.62 V_FLUSH_MARKER

Description:

Flush all V-series print heads in the specified Marker in all Installations selected using the INST command.

Command:

```
<ESC>V_FLUSH_MARKER<FS>name<ETB>
```

| Field | Description |
|-------------|--------------|
| <i>name</i> | Marker name. |

Successful response:

```
<ACK>
```

Example:

To flush all V-series print heads in the marker “Operator Side” in Installation “Production Line 1”, send:

```
<ESC>INST<FS>Production Line 1<ETB>
<ESC>V_FLUSH__MARKER<FS>Operator Side<ETB>
```

Expected response:

```
<ACK>
<ACK>
```

4.63 V_SET_CLEANER

Description:

Enable cleaner for all V-series print heads in all Installations selected using the INST command.

Command:

```
<ESC>V_SET_CLEANER<FS>enable<ETB>
```

| Field | Description |
|---------------|---------------------------------------|
| <i>enable</i> | 1 to enable cleaner, 0 to enable ink. |

Successful response:

```
<ACK>
```

Example:

To enable cleaner for all V-series print heads in the installation “Production Line 1”, send:

```
<ESC>INST<FS>Production Line 1<ETB>
<ESC>V_SET_CLEANER<FS>1<ETB>
```

Expected response:

```
<ACK>
<ACK>
```

4.64 V_SET_CLEANER_MARKER

Description:

Enable cleaner for all V-series print heads in the specified Marker in all Installations selected using the INST command.

Command:

```
<ESC>V_SET_CLEANER_MARKER<FS>name<FS>enable<ETB>
```

| Field | Description |
|---------------|---------------------------------------|
| <i>name</i> | Marker name. |
| <i>enable</i> | 1 to enable cleaner, 0 to enable ink. |

Successful response:

```
<ACK>
```

Example:

To enable cleaner for all V-series print heads in the marker “Operator Side” in Installation “Production Line 1”, send:

```
<ESC>INST<FS>Production Line 1<ETB>
```

```
<ESC>V_SET_CLEANER_MARKER<FS>Operator Side<FS>1<ETB>
```

Expected response:

```
<ACK>
```

```
<ACK>
```

4.65 VAR_DEL

Description:

Delete a variable. Variables can only be deleted if they are not in use. Attempting to delete a variable that is in used causes a <NAK> to be returned.

Command:

```
<ESC>VAR_DEL<FS>name<ETB>
```

| Field | Description |
|-------|-----------------------|
| name | Name of the variable. |

Successful response:

```
<ACK>
```

Example:

To delete the variable PRODUCT, send:

```
<ESC>VAR_DEL<FS>PRODUCT<ETB>
```

Example response:

```
<ACK>
```

4.66 VAR_GET

Description:

Get the value of a variable.

If this command is used to set a non-global variable, the following rules apply:

- If the *INST* command has been executed with an empty list of Installations, the global value of the variable will be returned.
- If the *INST* command was executed with a list of Installations, only the local value of the first Installation will be returned.

Note: Use the command INST to select which installations to operate on.

Command:

```
<ESC>VAR_GET<FS>name<ETB>
```

| Field | Description |
|-------------|-----------------------|
| <i>name</i> | Name of the variable. |

Successful response:

```
<ACK>value<ETB>
```

| Field | Description |
|--------------|------------------------|
| <i>value</i> | Value of the variable. |


Example:

To get the value of the variable PRODUCT, send:

```
<ESC>VAR_GET<FS>PRODUCT<ETB>
```

Example response:

```
<ACK>Bananas<ETB>
```

| | | |
|--|--|--------------------|
|  Matthews Marking Systems™ | Document name Matthews MPERIA Command Protocol Manual | Page 80 (85) |
| | Document number 795 247 | Date 2017-05-23 |

4.67 VAR_LIST

Description:

List all available variables.

Command:

```
<ESC>VAR_LIST<ETB>
```

Successful response:

```
<ACK>name<FS>...<ETB>
```

Example:

To list all available variables:

```
<ESC>VAR_LIST<ETB>
```

Expected response:

```
<ACK>BATCH<FS>OPERATOR<FS>My Pretty Variable<ETB>
```


4.68 VAR_SET

Description:

Set the value of a variable. If the variable does not exist, a new variable is created.

If this command is used to set a non-global variable, the following rules apply:

- If the *INST* command has been executed with an empty list of Installations, all instances of the variable will be set to the specified value.
- If the *INST* command was executed with a list of Installations, only the local values for the listed Installations will be set to the specified value.

Note: Use the command INST to select which installations to operate on.

Command:

```
<ESC>VAR_SET<FS>name<FS>value<ETB>
```

| Field | Description |
|--------------|----------------------------|
| <i>name</i> | Name of the variable. |
| <i>value</i> | New value of the variable. |

Successful response:

```
<ACK>
```

Example:

To set the value of the variable PRODUCT to Bananas, send:

```
<ESC>VAR_SET<FS>PRODUCT<FS>Bananas<ETB>
```

Expected response:

```
<ACK>
```

4.69 VAR_SET2

Description:

Set the values of one or more variables. If the variables does not exist, new variables are created.

If this command is used to set non-global variables, the following rules apply:

- If the *INST* command has been executed with an empty list of Installations, all instances of the variable will be set to the specified value.
- If the *INST* command was executed with a list of Installations, only the local values for the listed Installations will be set to the specified value.

Note: Use the command INST to select which installations to operate on.

Command:

```
<ESC>VAR_SET2<FS>name1<GS>value1 ... <FS>nameN<GS>valueN<ETB>
```

| Field | Description |
|--------------------------|--|
| <i>name_k</i> | Name of the k th variable. |
| <i>value_k</i> | New value of the k th variable. |

Successful response:

```
<ACK>
```

Example:

To set the value of the variable PRODUCT to Bananas, send:

```
<ESC>VAR_SET2<FS>PRODUCT<GS>Bananas<ETB>
```

Expected response:

```
<ACK>
```

To set the value of the variable PRODUCT to Oak, WEIGHT to 10, and SIZE to Large, send:

```
<ESC>VAR_SET2<FS>PRODUCT<GS>Oak<FS>WEIGHT<GS>10<FS>SIZE<GS>Large<ETB>
```

Expected response:

```
<ACK>
```

4.70 VERSION

Description:

Get the version of the MPERIA Controller.

Command:

```
<ESC>VERSION<ETB>
```

Successful response:

```
<ACK>version<ETB>
```

| Field | Description |
|----------------|----------------------------|
| <i>version</i> | MPERIA Controller version. |

Example:

To get the version of the MPERIA Controller, send:

```
<ESC>VERSION<ETB>
```

Example response:

```
<ACK>2.5.0<ETB>
```

4.71 SET_CLOCK

Description:

Set the system clock

Command:

```
<ESC>SET_CLOCK<FS>datetime<ETB>
```

Successful response:

```
<ACK>
```

| Field | Description |
|-----------------|--|
| <i>datetime</i> | A string with 12 or 14 digits of date and time in the format: yyyymmddHHMMss ss is optional, and will be set to 0 if omitted |

Example:

To set sytem time, send:

```
<ESC>SET_CLOCK<FS>201411260838<ETB>
```

Example response:

```
<ACK>
```

5 Revision history

| Revision | Date | Changes |
|----------|------------|--|
| 00 | 2013-12-10 | First revision. For 2.5.x. |
| 01 | 2014-05-16 | Updates for 3.0.x Added commands: L_DECAP_TABLE, SET_SPEED, CNT_RESET, MARKER_MARGIN, V_FLUSH, V_FLUSH_MARKER Updated description of fault handling for SEL, SELDB and SELMSG. |
| 02 | 2014-06-02 | Add optional activate argument to trigger command. |
| 03 | 2014-08-28 | Updates for 3.2.x Added commands: PRINT_CNT, PRINT_CNT_RESET, V_SET_CLEANER, V_SET_CLEANER_MARKER. |
| 04 | 2014-10-23 | Updates for 3.3.x Added commands: MARKER_UPSIDEDOWN Corrected MARKER_DIR arguments. |
| 05 | 2015-01-26 | Updates for 4.0.x Added commands: MARKER_UPSIDEDOWN, SET_CLOCK. |
| 06 | 2015-xx-xx | Updates for 5.0.x Added commands: MARK_GAP, MARK_INTERVAL, MSG_DEL, MSG_LIST |
| 07 | 2015-06-05 | Updates for 5.1.x Added Autolabel specific status codes. |
| 08 | 2015-10-07 | Updated for 5.2.x Added DEBUG level to the LOG command. |
| 09 | 2015-10-14 | Updated for 5.3.x Added commands: VAR_SET2. |
| 10 | 2016-03-03 | Added missing status codes: <i>crtOvertemp</i> , <i>crtAlmostWornOut</i> , <i>techLicenseError</i> , <i>dodCleaner</i> . |
| 11 | 2016-05-20 | Updated for 5.3.x Added commands: CNT_UPDATE. Fixed documentation of commands: NONCRITICAL. |
| 12 | 2016-11-09 | Updated for 6.0.x Added command: INPUT_GET, OUTPUT_SET, TRIG_ACTIVE. |
| 13 | 2016-12-23 | Updated for 6.1.x Added commands: ENC_BLOCK, ENC_RESET_REVERSE, OFFLINE, ONLINE |

Table 1: Revision history.