

E201-9Q Matlab Interface

E2019Q.m file is intended for communication between Matlab and the E201-9Q USB encoder interface. It supports almost a complete command set, which is extensively described in [E201 USB encoder interface data sheet](#).

Matlab functions are defined as methods of a common class called E2019Q. This allows several different functions to be stored in a single Matlab ».m« file. Methods are intended to:

- establish connection between Matlab and E201-9Q,
- read status of E201-9Q and encoder,
- check and control encoder power supply,
- read encoder position in different formats.

To establish a connection, user must firstly define what Virtual COM port was assigned to E201-9Q interface.

Below is a brief description of all functions which are supported by E2019Q.m interface.

1. Open COM port:

```
% Function call:
E2019Q_ID = E2019Q.Open_COM_Port('COM49');|

% E2019Q_ID is here used as a name for the serial port object and could be chosen freely.
% This object will be used as an input parameter for other functions.
% When calling Open_COM_Port function, COM number (COM49 in upper case) has to be placed
% between single quotes.
% The actual port number depends on how many COM ports are already in use on the PC.
% In Windows 7 this can be found under:
%     Control Panel > System > Device Manager > Ports (COM & LPT)
```

2. Close COM port:

```
% Function call:
E2019Q.Close_COM_Port(E2019Q_ID);

% User has to close COM port with this function before physically disconnecting USB cable
% from a PC.
```

3. Read E201-9Q software version:

```
% Function call:
SW_Version = E2019Q.GetSoftwareVersion(E2019Q_ID);

% Return value is a string (version + CR).
```

Example of returned value: SW_Version = E201-9Q V1.19

4. Read E201-9Q serial number:

```
% Function call:
Serial_Num = E2019Q.GetSerialNumber(E2019Q_ID);

% Return value is a string (aaaaaaaa : bbbbbbbb : cccccccc + CR).
```

Example of returned value: Serial_Num = 05d9ff35 : 39365041 : 43226728

5. Read encoder supply status, voltage and current consumption:

```
% Function call:
Enc_Supply = E2019Q.GetEncSupply(E2019Q_ID);

% Return value is a string (s : a.aaa V : bbbb mA + CR), where "s" represents
% power supply status (1 or 0), "a.aaa" represents voltage and "bbbb"
% represents current consumption.
```

Example of returned value: Enc_Supply = 1 : 4.889 V : 0089 mA

6. Read status of hardware input pins on interface:

```
% Function call:
Pin_Status = E2019Q.GetInputPinStatus(E2019Q_ID);

% Return value is a string (abz + CR).
```

Example of returned value: Pin_Status = 110

7. Turn off power supply to encoder:

```
% Function call:
Power_Supply = E2019Q.EncSupply_OFF(E2019Q_ID);

% Return value is a string (OFF + CR).
```

Example of returned value: Power_Supply = OFF

8. Turn on power supply to encoder:

```
% Function call:
Power_Supply = E2019Q.EncSupply_ON(E2019Q_ID);

% Return value is a string (ON + CR).
```

Example of returned value: Power_Supply = ON

9. Read encoder position (string, decimal):

```
% Function call:
Enc_Position = E2019Q.GetEncPosition(E2019Q_ID);

% Return value is a string (nnnn:rrrr:ssss + CR), where "n" represents
% encoder count, "r" represents count value when reference/index was last
% seen, "s" represents status (status value of 1 shows that a reference was
% already detected).
```

Example of returned value: Enc_Position = 198460: 175852: 1

10. Read encoder position (string, decimal) with position timestamp in µs:

```
% Function call:
Enc_Position = E2019Q.GetEncPosition_Timestamp(E2019Q_ID);

% Return value is a string (nnnn:rrrr:ssss:tttt + CR), where "n" represents
% encoder count, "r" represents count value when reference/index was last
% seen, "s" represents status (status value of 1 shows that a reference was
% already detected), "t" represents position timestamp in microseconds.

% Note: available in E201 interface version 1.18 (and later)
```

Example of returned value: Enc_Position = 198455: 175852: 1: 1098036264

11. Read encoder position (string, HEX):

```
% Function call:
Enc_Position = E2019Q.GetEncPositionHEX(E2019Q_ID);

% Return value is a string (nnnnnnnnrrrrrrrrrrsssssss + CR), where "n" represents
% encoder count (signed 32 bit) in HEX format, "r" represents count value when reference/index
% was last seen (signed 32 bit) in HEX format, "s" represents status (status value of 1 shows
% that a reference was already detected).
```

Example of returned value: Enc_Position = 000307370002aeec00000001

12. Read encoder position (string, HEX) with position timestamp in µs:

```
% Function call:
Enc_Position = E2019Q.GetEncPositionHEX_Timestamp(E2019Q_ID);

% Return value is a string (nnnnnnnnrrrrrrrrrrsssssssTTTTTTTT + CR), where "n" represents
% encoder count (signed 32 bit) in HEX format, "r" represents count value when reference/index
% was last seen (signed 32 bit) in HEX format, "s" represents status (status value of 1 shows
% that a reference was already detected), "t" represents position timestamp in microseconds in
% HEX format.

% Note: available in E201 interface version 1.18 (and later)
```

Example of returned value: Enc_Position = 00068db80003c1f40000000101de39a6

13. Clear reference status flag:

```
% Function call:
E2019Q.ClearReferenceFlag(E2019Q_ID);

% Return value - none. Function clears reference status flag and leaves
% encoder count and reference mark intact.
```

14. Set current count value to zero:

```
% Function call:
E2019Q.ResetCurrentCount(E2019Q_ID);

% Return value - none. Function resets encoder count value to 0. This also affects the
% reference mark.
```

15. Clear zero offset value stored by “ResetCurrentCount” function:

```
% Function call:  
E2019Q.ClearZeroOffset(E2019Q_ID);  
  
% Return value - none. Function clears offset value stored by  
% "ResetCurrentCount" function.
```

16. Read encoder count in double precision format:

```
% Function call:  
Enc_Count = E2019Q.GetEncCountDOUBLE(E2019Q_ID);  
  
% Return value is an encoder count value in double precision format.
```

Example of returned value: Enc_Count = 206849

17. Read encoder reference mark in double precision format:

```
% Function call:  
Enc_Reference = E2019Q.GetEncReferenceDOUBLE(E2019Q_ID);  
  
% Return value is an encoder reference mark in double precision format.
```

Example of returned value: Enc_Reference = 43400

18. Read timestamp of position in double precision format:

```
% Function call:  
Pos_Timestamp = E2019Q.GetTimestampDOUBLE(E2019Q_ID);  
  
% Return value is a position timestamp in double precision format.
```

Example of returned value: Pos_Timestamp = 51804753

All functions which return any value have integrated timeout set to 3 seconds. If COM port reading is not completed during that time, reading procedure is terminated and »Timeout occurs while reading COM port« is displayed in Command Window.

Head office

RLS merilna tehnika d.o.o.

Poslovna cona Žeje pri Komendi
Pod vrbami 2
SI-1218 Komenda
Slovenia

T +386 1 5272100

F +386 1 5272129

E mail@rls.si

www.rls.si

Document issues

Issue	Date	Page	Corrections made
1	27. 6. 2016	-	New document

This product is not designed or intended for use outside the environmental limitations and operating parameters expressly stated on the product's datasheet. Products are not designed or intended for use in medical, military, aerospace, automotive or oil & gas applications or any safety-critical applications where a failure of the product could cause severe environmental or property damage, personal injury or death. Any use in such applications must be specifically agreed to by seller in writing, and is subject to such additional terms as the seller may impose in its sole discretion. Use of products in such applications is at buyer's own risk, and buyer will indemnify and hold harmless seller and its affiliates against any liability, loss, damage or expense arising from such use. Information contained in this datasheet was derived from product testing under controlled laboratory conditions and data reported thereon is subject to the stated tolerances and variations, or if none are stated, then to tolerances and variations consistent with usual trade practices and testing methods. The product's performance outside of laboratory conditions, including when one or more operating parameters is at its maximum range, may not conform to the product's datasheet. Further, information in the product's datasheet does not reflect the performance of the product in any application, end-use or operating environment buyer or its customer may put the product to. Seller and its affiliates make no recommendation, warranty or representation as to the suitability of the product for buyer's application, use, end-product, process or combination with any other product or as to any results buyer or its customer might obtain in their use of the product. Buyer should use its own knowledge, judgment, expertise and testing in selecting the product for buyer's application, end-use and/or operating environment, and should not rely on any oral or written statement, representation, or samples made by seller or its affiliates for any purpose. EXCEPT FOR THE WARRANTIES EXPRESSLY SET FORTH IN THE SELLER'S TERMS AND CONDITIONS OF SALE, SELLER MAKES NO WARRANTY EXPRESS OR IMPLIED WITH RESPECT TO THE PRODUCT, INCLUDING ANY WARRANTY OF MERCHANTABILITY OR FITNESS FOR ANY PARTICULAR PURPOSE, WHICH ARE DISCLAIMED AND EXCLUDED. All sales are subject to seller's exclusive terms and conditions of sale which, where the seller is (a) RLS merilna tehnika d.o.o., are available at <https://www.rls.si/customer-service>, (b) Renishaw, Inc., are available at <http://www.renishaw.com/Shop/legal/en/-42186>, or (c) another person, are available on request, and in each case, are incorporated herein by reference, and are the exclusive terms of sale. No other terms and conditions apply. Buyer is not authorized to make any statements or representations that expand upon or extend the environmental limitations and operating parameters of the products, or which imply permitted usage outside of that expressly stated on the datasheet or agreed to in writing by seller.

RLS merilna tehnika d.o.o. has made considerable effort to ensure the content of this document is correct at the date of publication but makes no warranties or representations regarding the content. RLS merilna tehnika d.o.o. excludes liability, howsoever arising, for any inaccuracies in this document. © 2018 RLS d.o.o.