

SMT - SINUS Measurement Toolbox for MATLAB®

© SINUS Messtechnik GmbH, Leipzig, Germany

August 9, 2012 Version 2.x

User Documentation

Contents

1	Introduction	3
1.1	General	3
1.2	Typical sequence of functions for a simple batch data capture application	3
1.3	Example applications	4
1.4	Compilation	4
1.4.1	MCC 3.0	4
1.4.2	MCC 4.0 and later	4
1.5	Restrictions	4
2	SMT Functions	5
2.1	SMTResultCode	5
2.2	SMTQueryDevices	6
2.3	SMTSelectDevice	6
2.4	SMTCalibrateDevice	6
2.5	SMTOpenDevice	7
2.6	SMT Property Functions	7
2.6.1	SMTGet	7
2.6.2	SMTSet	7
2.6.3	SMTSetDevice	7
2.6.4	SMTGetDevice	8
2.6.5	SMTFindDevice	8
2.6.6	SMTSetChannel	9
2.6.7	SMTGetChannel	9
2.6.8	SMTFindChannel	9
2.6.9	SMTSaveProperties	10
2.6.10	SMTLoadProperties	10
2.6.11	SMTGetMeasurementFiles	11
2.6.12	SMTHelp	11
2.7	SMTStart	11
2.8	SMTGetData	12
2.9	SMTPeekData	13
2.10	SMTPutData	14
2.11	SMTStop	14
2.12	SMTCloseDevice	14
3	General Properties	15
3.1	SMT Properties	15
3.2	Device Properties	15
3.3	Channel Properties	16
A	Result Code Names and Messages	17
A.1	Errors	18
A.2	Warnings	19
A.3	Harmonie-Specific Errors	19
A.4	Harmonie-Specific Warnings	19
A.5	MSX16-Specific Errors	20
A.6	MSX16-Specific Warnings	21

1 Introduction

1.1 General

This document describes the SINUS Measurement Toolbox (SMT) for MATLAB[®] from the viewpoint of the MATLAB programmer. The SMT offers a consistent and platform-independent* interface from MATLAB to the measuring devices produced by SINUS Messtechnik GmbH. Programs written on the basis of the SMT benefit from its stability and independence from operating system and driver implementations; in addition, it is possible to write code which is largely or even completely independent of the hardware device to be used. All these factors result in greatly increased software reusability.

The following concepts are fundamental to the SMT:

- Device:** e.g. a HARMONIE[™]/Soundbook[™], an MSX16[™] tower or a Virtual Device, which allows access to files previously recorded using the SMT (possibly on a different PC).
- Channel:** a (logical) data stream, e.g. AnalogInput or SquareSum values, delivered in defined physical units.
- Property:** an attribute of a device or channel (or of the SMT itself).

The number of channels and properties varies with the device type and the working mode. The SMT comes with an example program called *SINUSconfigure* which shows the devices and channels with their SMT properties in a convenient way.

The SMT requires MATLAB 6.5.1 (Release 13, Service Pack 1) or later; information about installation procedures and the level of support (devices and operating systems) currently offered can be obtained from the release notes. To compile SMT applications to stand alone programs the MatLab Compiler[™] has to be installed. The SMT itself will attempt to check the installation status of the software it requires and will report any problems found. Please address all correspondence regarding the SMT to smt.support@sinusmess.de; correspondence regarding the SINUS devices and their drivers may be addressed to drivers.support@sinusmess.de.

1.2 Typical sequence of functions for a simple batch data capture application

- SMTQueryDevices
- SMTSelectDevice
- SMTOpenDevice (*initialization*)
 - SMTSetDevice (*device configuration*)
 - SMTSetChannel (*channel configuration*)
 - SMTStart (*execution*)
 - data transfer*
 - SMTStop
 - SMTGetData (*load data into workspace*)
- SMTCloseDevice (*free resources*)

*Provided that device drivers for the desired platform exist

1.3 Example applications

The following applications are delivered as source code which can be freely modified and used as examples:

- `SINUSconfigure`
- `SINUSrecorder`
- `SINUSoscilloscope`
- `SINUSdataConverter`
- `SINUSMAT`

They are compilable, work for all SINUS device families, and can be used as they are or as components of custom-written applications. Each of these applications is not more than a few hundred lines of code long, most of which is for the graphical user interfaces; the source code is extensively commented. Two further functions complete this set of examples: `SINUSerrWarn` (used by the above four applications for systematic handling of errors and warnings returned by SMT functions) and `SINUSexamples`, which simply allows the applications to be started via pushbutton controls.

1.4 Compilation

To compile SMT applications, the MatLab Compiler is necessary. There are differences between the compilers present in R13 (mcc ver. 3.0) and those of later releases (mcc ver. 4+). See the MATLAB compiler documentation for further details.

1.4.1 MCC 3.0

Applications using the SMT can be compiled without access to the SMT source code: the C shared library `libsmt.dll` and the corresponding `libsmt.mlib` file are supplied for this purpose together with `libsmt.lib` and `libsmt.h` - all four files should be copied into the compilation directory, but of these, only `libsmt.dll` is required at run-time. The file `winiomatlab.dll` is also required at run-time, as is `sinusinterface.dll`. An application such as `SINUSexamples.m` can be compiled as follows:

```
mcc -l -g -m -B sgl SINUSexamples libsmc.mlib
```

Note that the file `libmatlb.dll` included with the SMT must be used for compiled applications instead of the standard version included in MATLAB Release 13 (Service Pack 1): this cures a MATLAB bug which prevented the SMT timer from executing in compiled code and resulted in no data being acquired.

1.4.2 MCC 4.0 and later

For these mcc versions the MLIB feature is no longer supported. To compile a standalone application using the SMT, make sure that the p-files of the SMT as well as the driver files `sinusinterface.dll` and `winiomatlab.dll` are on the MATLAB path or in the compilation directory and type:

```
mcc -g -m application_name
```

The compiler generates an .exe-file and a .ctf-file with the application name which are needed at runtime. `libsmt.h`, and `libsmt.ctf` are supplied for use in C/C++ projects.

1.5 Restrictions

The SMT defines a global variable (structure) `gSMT`, which must not be changed in any way other than by the SMT itself. The results of accessing this structure are undefined and unsupported: the SMT developers reserve the right to change this structure without notice in the interests of technical progress.

When checking for specific error and warning codes (other than 0 for 'Success'), the function `SMTResultCode` (section 2.1) should always be used to translate result code names (see Appendix A) to their numeric values: these numeric values may change in a future release of the SMT.

2 SMT Functions

The SMT provides a set of MATLAB functions to control SINUS devices. Only the functions described in this document should be called: the remaining functions in the SMT directory are only intended for internal use by the SMT itself and may be changed or deleted without notice in a future version of the SMT.

Commonly occurring parameters:

Channel : name or number of a single logical channel
Channels : name or number of a single logical channel *or*
cell array of names or array of numbers of multiple logical channels
Device : name or number of a physical or virtual SINUS Device
PropertyName : a property name
PropertyValue : a property value

All SMT functions return a result code (RC) $\left\{ \begin{array}{l} < 0 \rightarrow \textit{warning} \\ 0 \rightarrow \textit{success} \\ > 0 \rightarrow \textit{error} \end{array} \right.$

and an error/warning message (RM).

For example, the use of any channel or device property not yet supported by a given implementation of the SMT will result in an appropriate result code and message.

If one or more errors occur, only the first detected error is returned. However, if multiple warnings occur without an error occurring, then an array of warnings is returned in RC; in this case, RM is a cell array of strings rather than a simple string. See also the function `SMTResultCode` (section 2.1) and Appendix A.

For detailed information about errors, warnings and other information the SMT 2.0 and higher is able to write a log-file (SMT.log). This file is located in the measurement path and contains messages from the SMT depending on a user-settable information depth called the verbosity level. Use `SMTSet` and `SMTGet` to learn about the supported verbosity levels. Because writing to a text file may slow down time-critical processes, it is possible to set the number of messages that should be collected before a write operation is performed. This parameter is called `FlushLogFileAfter`.

2.1 SMTResultCode

Return the value of a named SMT result code.

Syntax

```
[RC, RM] = SMTResultCode(RCName)
[RC, RCName] = SMTResultCode(RC)
[RCstruct, RMstruct] = SMTResultCode
```

Description

`[RC, RM] = SMTResultCode(RCName)` returns the result code RC and message RM corresponding to the (case-sensitive) string RCName. If RCName is supplied but is not a valid result code name, RC is returned as NaN (Not a Number) and RM is an appropriate message.

`[RC, RCName] = SMTResultCode(RC)` (where RC is numeric) returns the result code RC and the result code name string RCName corresponding to RC. If RC is not a valid result code, RC is returned as NaN (Not a Number) and RCName is replaced by an appropriate message, otherwise RC is returned unchanged. This form of the function is useful from the command line in order to find out the correct RCName for use in MATLAB code which checks for a specific result code identified during testing.

`[RCstruct, RMstruct] = SMTResultCode` returns structures RCstruct and RMstruct whose field names are all the valid result code names and whose values are the result codes and messages respectively.

The first form of this function should always be used when checking for specific result codes: their

numerical values are not guaranteed to remain constant between releases of the SMT (except that `SMTResultCode('Success')` always returns 0), but their names will remain valid. See Appendix A for a list of result code names and messages.

2.2 SMTQueryDevices

Return array of device numbers of all installed SINUS devices.

Syntax

```
[DeviceNumbers, RC, RM] = SMTQueryDevices
```

Description

`[DeviceNumbers, RC, RM] = SMTQueryDevices` returns an array of device numbers of all installed SINUS devices to `DeviceNumbers`. The array may also contain virtual devices (see section 2.6.10) which may be used for reading data previously recorded (possibly on another machine) without requiring the physical device to be present.

Remarks

After a SINUS device is added or removed, `SMTQueryDevices` must be called again, after which device numbers previously obtained should no longer be relied upon. Before adding or removing a SINUS device, be sure that all other SINUS devices are closed. SINUS devices can be closed with `SMTCloseDevice`. The first device in `DeviceNumbers` is selected as the current device by `SMTQueryDevices`.

2.3 SMTSelectDevice

Select current device.

Syntax

```
[RC, RM] = SMTSelectDevice(Device)
```

Description

`[RC, RM] = SMTSelectDevice(Device)` makes the (physical or virtual) device current, i.e. all further SMT functions will apply to this device.

Remarks

`DeviceNumbers` can be obtained by calling `SMTQueryDevices`; `Device` may alternatively be a device name.

2.4 SMTCalibrateDevice

Calibrate device.

Syntax

```
[RC, RM] = SMTCalibrateDevice
```

Description

`[RC, RM] = SMTCalibrateDevice` calibrates the current SINUS device, which must be a physical, not virtual device (see sections 2.2 and 2.3). Please see the appropriate hardware documentation for information on warm-up times, necessary connections etc.

2.5 SMTOpenDevice

Open and initialize device.

Syntax

```
[RC, RM] = SMTOpenDevice
```

Description

[RC, RM] = SMTOpenDevice opens and initializes the current SINUS device (see sections 2.2 and 2.3).

2.6 SMT Property Functions

2.6.1 SMTGet

Return property values of the SMT itself.

Syntax

```
[Properties, RC, RM] = SMTGet  
[Value, RC, RM] = SMTGet(PropertyName)
```

Description

[Properties, RC, RM] = SMTGet returns all properties of the SMT itself together with their values to **Properties** as a structure array whose field names are the property names and whose field values are the values of the corresponding properties.

[Value, RC, RM] = SMTGet(PropertyName) returns the value for **PropertyName** of the SMT to **Value**.

2.6.2 SMTSet

Configure SMT properties or return settable SMT properties / valid values.

Syntax

```
[Properties, RC, RM] = SMTSet  
[ValidValues, RC, RM] = SMTSet(PropertyName)  
[RC, RM, ErrorProp] = SMTSet(PropertyName, PropertyValue, ...)
```

Description

[Properties, RC, RM] = SMTSet returns the settable properties and possible values for the SMT to **Properties** as a structure array whose field names are the property names and whose field values are the possible values of the corresponding properties.

[ValidValues, RC, RM] = SMTSet(PropertyName) returns an array of possible values for the named property for the SMT to **ValidValues**. If the possible values are strings, **ValidValues** is returned as a cell array of strings.

[RC, RM, ErrorProp] = SMTSet(PropertyName, PropertyValue, ...) sets one or more property values of the SMT. If an error or warning is encountered, the name of the property on which the error or warning was detected is returned to **ErrorProp**.

2.6.3 SMTSetDevice

Configure device properties or return settable device properties / valid values.

Syntax

```
[Properties, RC, RM] = SMTSetDevice
[ValidValues, RC, RM] = SMTSetDevice(PropertyName)
[RC, RM, ErrorProp] = SMTSetDevice(PropertyName, PropertyValue, ...)
```

Description

[Properties, RC, RM] = SMTSetDevice returns the settable properties and possible values for the current device to **Properties** as a structure array whose field names are the property names and whose field values are the possible values of the corresponding properties.

[ValidValues, RC, RM] = SMTSetDevice(PropertyName) returns an array of possible values for the named property for the current device to **ValidValues**. If the possible values are strings, **ValidValues** is returned as a cell array of strings.

[RC, RM, ErrorProp] = SMTSetDevice(PropertyName, PropertyValue, ...) sets one or more property values of the current device. If an error or warning is encountered, the name of the property on which the error or warning was detected is returned to **ErrorProp**.

2.6.4 SMTGetDevice

Return property values of the current device.

Syntax

```
[Properties, RC, RM] = SMTGetDevice
[Value, RC, RM] = SMTGetDevice(PropertyName)
```

Description

[Properties, RC, RM] = SMTGetDevice returns all device properties and their values to **Properties** as a structure array whose field names are the property names and whose field values are the values of the corresponding properties.

[Value, RC, RM] = SMTGetDevice(PropertyName) returns the value for **PropertyName** of the current device to **Value**.

2.6.5 SMTFindDevice

Find devices with given properties.

Syntax

```
[DeviceNumbers, RC, RM] = SMTFindDevice
[DeviceNumbers, RC, RM] = SMTFindDevice(PropertyName, PropertyValue, ...)
```

Description

[DeviceNumbers, RC, RM] = SMTFindDevice returns an array of all devices (logical device numbers) to **DeviceNumbers**.

[DeviceNumbers, RC, RM] = SMTFindDevice(PropertyName, PropertyValue, ...) returns an array of device numbers identifying devices which have the given **PropertyValue** for **PropertyName** to **DeviceNumbers**. If more than one **PropertyName/PropertyValue** pair is supplied, only those devices matching all the pairs are returned.

Remarks

In contrast to `SMTQueryDevices`, `SMTFindDevice` does not search for new hardware, does not change the currently selected device, and does not require that all devices are closed.

Examples

`DeviceNumbers = SMTFindDevice('Type', 'MSX16')` returns an array of all devices of type MSX16.

`DeviceNumbers = SMTFindDevice('DeviceID', 2)` returns the logical (SMT) device number of the device with physical ID 2 (e.g. as used by lower-level driver functions).

2.6.6 SMTSetChannel

Configure channel properties or return settable channel properties and valid values.

Syntax

```
[Properties, RC, RM] = SMTSetChannel(Channel)
[ValidValues, RC, RM] = SMTSetChannel(Channel, PropertyName)
[RC, RM, ErrorProp] = SMTSetChannel(Channels, PropertyName, PropertyValue, ...)
```

Description

`[Properties, RC, RM] = SMTSetChannel(Channel)` returns the user-settable properties and possible values for `Channel` to `Properties` as a structure array whose field names are the property names and whose field values are the possible values of the corresponding properties.

`[ValidValues, RC, RM] = SMTSetChannel(Channel, PropertyName)` returns an array of possible values for the named property for `Channel` to `ValidValues`. If the possible values are strings, `ValidValues` is returned as a cell array of strings.

`[RC, RM, ErrorProp] = SMTSetChannel(Channels, PropertyName, PropertyValue, ...)` sets one or more properties of one or more channels. `Channels` is the name or number of a single channel or an array of numbers or a cell array of names of multiple channels. If an error or warning is encountered, the name of the property on which the error or warning was detected is returned to `ErrorProp`.

2.6.7 SMTGetChannel

Return channel property values.

Syntax

```
[Properties, RC, RM] = SMTGetChannel(Channel)
[Value, RC, RM] = SMTGetChannel(Channels, PropertyName)
```

Description

`[Properties, RC, RM] = SMTGetChannel(Channel)` returns all channel properties and their values to `Properties` as a structure array whose field names are the property names and whose field values are the values of the corresponding properties. `Channel` is the name or number of a single channel.

`[Value, RC, RM] = SMTGetChannel(Channel, PropertyName)` returns the value of `PropertyName` for `Channels` to `Value`. `Channels` is the name or number of a single channel or an array of numbers or a cell array of names of multiple channels.

2.6.8 SMTFindChannel

Find channels with given properties.

Syntax

```
[Channels, RC, RM] = SMTFindChannel  
[Channels, RC, RM] = SMTFindChannel(PropertyName, PropertyValue, ...)
```

Description

[Channels, RC, RM] = SMTFindChannel returns an array of all channels (logical channel numbers) to Channels.

[Channels, RC, RM] = SMTFindChannel(PropertyName, PropertyValue, ...) returns an array of channels (logical channel numbers) which have the given PropertyValue for PropertyName to Channels. If more than one PropertyName/PropertyValue pair is supplied, only those channels matching all the pairs are returned.

Examples

Channels = SMTFindChannel('Enabled', 1) returns an array of all enabled channels.

Channels = SMTFindChannel('ChannelID', 'IN1') returns an array of all (logical) channels which correspond to the physical channel IN1.

2.6.9 SMTSaveProperties

Save device and channel properties to a MAT-file.

Syntax

```
[RC, RM] = SMTSaveProperties(FileName)
```

Description

[RC, RM] = SMTSaveProperties(FileName) saves device and channel properties for the currently selected device to FileName. If no extension is specified for FileName, the default extension .SMT is used.

2.6.10 SMTLoadProperties

Load device and channel properties from a MAT-file.

Syntax

```
[RC, RM] = SMTLoadProperties  
[RC, RM] = SMTLoadProperties(FileName)  
[RC, RM] = SMTLoadProperties(FileName, 'createdevice')
```

Description

[RC, RM] = SMTLoadProperties(FileName) loads device and channel properties for the currently selected and open device from FileName. If no extension is specified for FileName, the default extension .SMT is used. If FileName is empty or completely omitted, the default properties for the current device (and its default channels) are loaded. The properties loaded are checked for compatibility with the current device unless it is a virtual device.

[RC, RM] = SMTLoadProperties(FileName, 'createdevice') creates and selects a new virtual device which may be used for reading data previously recorded (possibly on another machine) without requiring the physical device to be present.

2.6.11 SMTGetMeasurementFiles

List the file names of all files belonging to a SMT measurement for managing purposes.

Syntax

```
[Files, RC, RM] = SMTGetMeasurementFiles
[Files, RC, RM] = SMTGetMeasurementFiles(filename)
```

Description

[Files, RC, RM] = SMTGetMeasurementFiles returns a list of the file names belonging to the current/last measurement recorded with the currently selected device in the cell array **Files**.

[Files, RC, RM] = SMTGetMeasurementFiles(filename) returns a list of the file names belonging to the SMT measurement stored in the file **filename**. If **filename** is not a valid SMT file, then **Files** is an empty cell array.

Remarks

The example program SINUSMAT (Measurement Administration Tool) is a GUI-based managing tool using this function.

2.6.12 SMTHelp

Return information about SMT, driver or device properties.

Syntax

```
[out, RC, RM] = SMTHelp(Name)
```

Description

[out, RC, RM] = SMTHelp(Name) returns help information about the property **Name**. The input may be any channel/ device name, property or setting name. If help is available, the variable **out** contains a short string that can be used for instance as tooltip string. If used in the MATLAB environment, **Name** might also be the name of a SMT function. This is the same as **out = help(SMTFunctionName)**. In case no help is available, **out** is empty, and **RC** and **RM** contain error messages.

2.7 SMTStart

Start data transfer and optionally set the duration (in seconds) of the data transfer.

Syntax

```
[RC, RM] = SMTStart
[RC, RM] = SMTStart(Duration)
```

Description

[RC, RM] = SMTStart starts the data transfer, which continues until SMTStop is called; SMTStart returns immediately.

[RC, RM] = SMTStart(Duration) stops the data transfer automatically after at least **Duration** (seconds); SMTStart(Duration) returns after the data transfer is complete.

Remarks

After `SMTStart` is called, the samples acquired from each enabled input channel are logged to disk unless the `FileOutput` channel property is set to 0. A header file containing the current properties and one data file for each such input channel will be created:

- `FileName.smt` contains device properties, channel properties and a time stamp (absolute time of the beginning of the recording). `FileName` is the value of the `FileName` device property.
- `FileNamexxx.dat` contains data acquired (in original format) from the channel with the file-name xxx.
- `FileNameDataLost.dat` contains data loss information; may also be empty.

These files will be overwritten when `SMTStart` is called again if the `FileName` device property has not been changed in the meantime.

Example

```
SMTSelectDevice(1);           select and
SMTOpenDevice;               open device 1
SMTSetDevice('FileName', 'Main'); set output file name
SMTSetChannel(3:4, 'Enabled', 1); enable (input) channels 3 and 4
SMTSetChannel(4, 'FileName', 'Mic1'); specify filename for channel 4
SMTStart;                    start acquisition
```

causes the following files to be created:

```
- Main.smt           device and channel properties
- Main003.dat        channel 3 data (003 is default filename for this ch.)
- MainMic1.dat       channel 4 data
- MainDataLost.dat   data loss info
```

2.8 SMTGetData

Return data (acquired samples) and relative time values (in seconds) for the samples.

Syntax

```
[Data, Time, RC, RM] = SMTGetData(Channels)
[Data, Time, RC, RM] = SMTGetData(Channels, 'time', StartPos, EndPos)
[Data, Time, RC, RM] = SMTGetData(Channels, 'samples', StartPos, EndPos)
```

Description

`[Data, Time, RC, RM] = SMTGetData(Channels)` returns all acquired samples of `Channels` to `Data` as an m-by-n array and the relative time values (in seconds) to `Time` as an m-by-1 array.

`[Data, Time, RC, RM] = SMTGetData(Channels, 'time', StartPos, EndPos)` returns all acquired samples of `Channels` from `StartPos` (in seconds) to `EndPos` (in seconds) to `Data` as an m-by-n array and the relative time values (in seconds) to `Time` as an m-by-1 array.

`[Data, Time, RC, RM] = SMTGetData(Channels, 'samples', StartPos, EndPos)` returns all acquired samples of `Channels` from `StartPos` (in samples) to `EndPos` (in samples) to `Data` as an m-by-n array and the relative time values (in seconds) to `Time` as an m-by-1 array.

Remarks

For one-dimensional channels, `Data` is returned as an m-by-n array where m is the number of samples and n is the number of channels.

`Time` is always returned as an m-by-1 array where m is the number of samples.

Channels must all be sampled at the same rate and simultaneously. **SMTLoadProperties** may be used to open previously recorded files (see section 2.6.10). In the event of data loss (e.g. due to overflow of a device driver's ring-buffers), the lost data are substituted with NaN (Not a Number) values.

Example:

```
[Data, Time] = SMTGetData([3 4], 'samples', 21, 25);
```

$$\text{Data} = \begin{pmatrix} S_{21_3} & S_{21_4} \\ S_{22_3} & S_{22_4} \\ S_{23_3} & S_{23_4} \\ S_{24_3} & S_{24_4} \\ S_{25_3} & S_{25_4} \end{pmatrix} \quad \text{Time} = \begin{pmatrix} t_{21} \\ t_{22} \\ t_{23} \\ t_{24} \\ t_{25} \end{pmatrix}$$

2.9 SMTPeekData

Preview data (acquired samples) during a data acquisition.

Syntax

```
[Data, RC, RM] = SMTPeekData(Channels, Samples)
```

Description

[Data, RC, RM] = SMTPeekData(Channels, Samples) returns at most **Samples** (number) of the samples most recently acquired from **Channels** to **Data**. **Channels** is the name or number of a single channel or an array of numbers or a cell array of names of multiple channels.

Remarks

For one-dimensional channels **Data** is returned as an m-by-n array where m (\leq **Samples**) is the number of samples and n is the number of channels.

Channels must all be sampled at the same rate and simultaneously.

This function is intended for efficient monitoring/previewing of acquired data. Data already returned by **SMTPeekData** are not returned by subsequent calls to **SMTPeekData**. The maximum amount of data which a given call to **SMTPeekData** can deliver is determined by the sample frequency of the channels and the time elapsed since the SMT last fetched data, which may vary.

Applications which require all data to be obtained with **SMTPeekData** must make use of the **NewDataCallback** property. The application sets this device property to the name of an M-file function which is called by the SMT when new data are available for at least one channel of the device; this callback function should use **SMTPeekData** to obtain the data. The function will not be called again while it is still executing; any new data arriving during this time are buffered and first made available in a subsequent call to the function. The **SINUSoscilloscope** example application demonstrates the use of the **NewDataCallback** property.

2.10 SMTPutData

Provide data for output.

Syntax

```
[RC, RM] = SMTPutData(Channels, Data)
```

Description

[RC, RM] = SMTPutData(Channels, Data) allocates Data for output on Channels. Data must consist of a column of data for each channel of Channels.

Remarks

Data is an m-by-n array where m is the number of samples and n is the number of channels. Channels must be of the same type and sampled at the same rate. The data output starts after SMTStart is called.

Example

```
SMTSelectDevice(1);           select and  
SMTOpenDevice;              open device 1  
SMTSetChannel(7, 'enabled', 1); enable (output) channel 7  
SMTPutData(7, data);        provide data for output  
SMTStart;                   start output
```

2.11 SMTStop

Stop data transfer.

Syntax

```
[RC, RM] = SMTStop
```

Description

[RC, RM] = SMTStop stops the data transfer for the current device.

2.12 SMTCloseDevice

Shut down device and free resources.

Syntax

```
[RC, RM] = SMTCloseDevice
```

Description

[RC, RM] = SMTCloseDevice closes the current SINUS device.

Remarks

Device and channel properties are lost. Before closing the device, device and channel properties can be saved by SMTSaveProperties.

3 General Properties

3.1 SMT Properties

The following properties are available for the SMT itself.

Property Name	Description	Default PropertyValue
Version (read only)	SMT version number (as string)	–
VerbosityLevel	depth of information to be logged in SMT.log	Error
FlushLogFileAfter	number of lines in the log file buffer before writing to SMT.log; helps controlling the file access for performance	10

3.2 Device Properties

The following properties are available for all device types. Other properties are depending on the specific device. Use `SMTGetDevice` and `SMTSetDevice` as well as `SMTHelp` to learn more about other device features.

Property Name	Description	Default PropertyValue
Number (read only)	logical device number	–
Type (read only)	device type	–
Name	logical device name	<i>device number (3 digits)</i>
DeviceID (read only)	physical device ID <i>empty for virtual device</i>	–
FileName	output file name	<i>same as Name</i>
Mode	operating mode of the device <i>modes define types and numbers of channels</i>	<i>device-specific</i>
BaseSampleRate	base sample rate in Hz	<i>device-specific</i>
Open (read only)	device currently open (0 or 1)	–
Started (read only)	data transfer currently started (0 or 1)	–
NewDataCallback	function name - see section 2.9	–
StartTime (read only)	date/time of the latest <code>SMTStart</code> for the device <i>in date vector format as returned by the CLOCK function in MATLAB: [year month day hour minute seconds]</i>	–
BlockCount	data buffer size for the driver; increase if data loss occurs	<i>device-specific</i>
Description	device description from the device manager	–

3.3 Channel Properties

The following properties are available for all (logical) channels. Other properties are depending on the specific channel. Use `SMTGetChannel` and `SMTSetChannel` as well as `SMTHelp` to learn more about which channels exist and what their specifications are.

Property Name	Description	Valid Values	Channels
Number (read only)	logical channel number	–	all channels
Type (read only)	channel type	–	
Name	logical channel name	(user-supplied string) <i>default: channel number (3 digits)</i>	
Enabled	enabled status	0 (disabled) <i>default</i> 1 (enabled)	
FileOutput	channel data written to file (necessary for <code>SMTGetData</code>)	0 (no file output) 1 (file output) <i>default</i>	
ChannelID (read only)	physical channel ID(s)	–	
SampleRate (read only)	effective sample rate in Hz (depending on <code>BaseSampleRate</code> & <code>Decimation</code>)	–	
Input (read only)	<code>SMTGetData</code> and <code>SMTPeekData</code> allowed	–	
Output (read only)	<code>SMTPutData</code> allowed	–	
DataType (read only)	data format e.g. <code>uint16</code> , <code>float32</code>	–	
Dimension (read only)	number of values which make up a single sample (normally 1)	–	
ChannelUnit (read only)	units of the logical channel data, e.g. 'Pa ² ' for <code>SquareSum</code> channels	–	
PhysicalChannelUnit (read only)	SI units of the physical channel, e.g. 'V'	–	
SensorUnit	SI units of the sensor, e.g. 'Pa'	(user-supplied string)	
SensorSensitivity	sensitivity of the sensor in <code>PhysicalChannelUnit/SensorUnit</code> , e.g. for a 50mV/Pa microphone: <code>SensorUnit</code> = 'Pa' <code>PhysicalChannelUnit</code> = 'V' <code>SensorSensitivity</code> = 0.05	(user-supplied double)	

A Result Code Names and Messages

The names listed in these tables may be passed as the `RCName` (string) parameter to `SMTPResultCode` in order to obtain a result-code value for comparison. Please see Sections 2 and 2.1 where the general principles of SMT error-handling are described.

A.1 Errors

Name	Message
DeviceNotOpen	Device must be opened first.
DeviceNotClosed	Device ... is not closed.
NoDevice	Device ... does not exist.
NoChannels	There are no channels.
NoChannel	Channel ... does not exist.
NoDeviceProperty	Property ... does not exist for the current device.
NoChannelProperty	Property ... does not exist for channel
NoEnabledChannels	There are no enabled channels.
NoInputChannel	Channels must be input channels.
NoOutputChannel	Channels must be output channels.
NoMultiChannel	Channel ... cannot be combined with other channels.
PropertyNotSettable	Property ... is not settable.
PropertyNotSettableAfterStart	Property ... is not settable during data transfer.
DeviceNotSupported	Device is not yet supported by the SMT.
WrongDataType	Wrong data type for
WrongNumberOfInputArguments	Wrong number of input arguments.
WrongNumberOfChannels	Wrong number of channels.
WrongInputParameter	Wrong input parameter
FileNotFound	... not found.
InvalidValue	Invalid value for
ParameterNotSpecified	... must be specified.
FailStartVirtualDevice	SMTStart is not allowed for virtual devices.
FailOutputVirtualDevice	SMTPutData is not allowed for virtual devices.
FailLoadVirtualDevice	Default properties of virtual devices cannot be set.
FailSaveVirtualDevice	Properties of virtual devices cannot be saved.
FailSelectVirtualDevice	Closed virtual device cannot be selected.
DataTransferNotStarted	Operation is only allowed during data transfer.
DataTransferNotStopped	Operation is not allowed during data transfer.
NoEqualSampleRates	Channels must be sampled at the same rate.
NoEqualTimeOffset	Channels must have the same time offset.
WrongDimension	Channels with dimension > 1 cannot be combined.
ChannelAndDataDoNotAgree	Number of channels and number of columns in data do not agree.
NoSMTPProperty	Property ... does not exist for the SMT itself.
PropertiesIncompatible	One or more properties are incompatible with current device.
DriverFilesOutOfDate	At least one driver file is outdated. Please update the driver.
SMTFileExists	A measurement file of this name (...) already exists.
PropertyNotImplemented	Property not yet implemented.
InvalidCombination	Invalid combination of Mode and BaseSampleRate!
LoadSynchronisedError	Properties not applicable for current device configuration!
FailCalibrateDeviceType	Devices of type ... cannot be calibrated.
SINUSInterfaceError	General Error using the SINUS Interface (...).
SINUSInterfaceNotEnabledError	SINUS Interface is disabled!
SMTLogError	SMTLog functionality not available!
NoHelpAvailable	No help available for
InvalidLicence	Licence Validation Check failed!
UnexpectedError	Unexpected error when executing SMT function: ...

A.2 Warnings

Name	Message
ValueAlreadyExists	PropertyValue already exists for another
DeviceNameRemoved	Device name was removed because it already exists for another device.
ValueRounded	Value was rounded to the nearest valid value.
DeviceAlreadyOpen	Device is already open.
DeviceAlreadyClosed	Device is already closed.
DataTransferAlreadyStarted	Data transfer has already started.
DataTransferAlreadyStopped	Data transfer has already stopped.
FailOpenVirtualDevice	Virtual devices cannot be opened.
FailCloseVirtualDevice	Virtual devices cannot be closed.
NoData	No data is available for channel
NoNewData	No new data is available.
RequestedNumberOfSamplesNotAvailable	Requested number of samples is greater than number of samples available.
InvalidCalibrationData	Wrong ... calibration data for actual device. Use defaults.
NoCalibrationData	No calibration data for Use defaults.
NoDevices	There are no devices present.
PropertiesIgnored	One or more properties were ignored.
OtherChannelsAffected	The property ... has also been changed for one or more other channels.
DriverFilesUntested	At least one driver file is newer then expected and may not have been tested.
DataLostDetected	DataLost has been detected for this measurement!

A.3 Harmonie-Specific Errors

Name	Message
HUML_Error	Error ... was returned from function
FailCalibrateHarmonie	Harmonie calibration failed (HUMLCalibration returned: ...) - please see hardware documentation.

A.4 Harmonie-Specific Warnings

Name	Message
WarnCalibrateHarmonie	Harmonie calibration incomplete (HUMLCalibration returned: ...) - please see hardware documentation. For example, clock calibration requires a 1Hz TTL signal on the TACHO input, and output calibration requires the outputs to be connected to the correspondingly numbered inputs.

A.5 MSX16-Specific Errors

Name	Message
MSX16Error	Error ... was returned from function
MSX16e102	MSX16 Unit ... / e102: Critical error during device mode change. Switch the device off and on again and repeat the procedure. If the error remains, please consult an authorised service partner.
MSX16e103	MSX16 Unit ... / e103: One or more components were not found (main board / power supply). Switch the device off and on again and repeat the procedure. If the error remains, please consult an authorised service partner.
MSX16e104	MSX16 Unit ... / e104: The device was started in Stand-Alone Mode and then started again via the PC. Switch off the device using its main power switch, and only start the device via the PC.
MSX16e109	MSX16 Unit ... / e109: Device IDs are incorrect. Switch the device off and on again. If the error remains, please consult an authorised service partner.
MSX16e110	MSX16 Unit ... / e110: The flash sector is write-protected. Repeat the procedure (e.g. calibration) or please consult an authorised service partner.
MSX16e111	MSX16 Unit ... / e111: The module ID is incorrect (module 1). Switch the device off, remove the affected module from its slot and please consult an authorised service partner.
MSX16e112	MSX16 Unit ... / e112: The module ID is incorrect (module 2). Switch the device off, remove the affected module from its slot and please consult an authorised service partner.
MSX16e113	MSX16 Unit ... / e113: The module ID is incorrect (module 3). Switch the device off, remove the affected module from its slot and please consult an authorised service partner.
MSX16e114	MSX16 Unit ... / e114: The module ID is incorrect (module 4). Switch the device off, remove the affected module from its slot and please consult an authorised service partner.
MSX16e115	MSX16 Unit ... / e115: The module ID is incorrect (module 5). Switch the device off, remove the affected module from its slot and please consult an authorised service partner.
MSX16e116	MSX16 Unit ... / e116: The module ID is incorrect (module 6). Switch the device off, remove the affected module from its slot and please consult an authorised service partner.
MSX16e117	MSX16 Unit ... / e117: The module ID is incorrect (module 7). Switch the device off, remove the affected module from its slot and please consult an authorised service partner.
MSX16e118	MSX16 Unit ... / e118: The module ID is incorrect (module 8). Switch the device off, remove the affected module from its slot and please consult an authorised service partner.
FailCalibrateMSX16	MSX16 calibration failed (MSXUtil returned: ...) - please see hardware documentation.

A.6 MSX16-Specific Warnings

Name	Message
MSX16w100	MSX16 Unit ... / w100: Conflict between bandwidth and weighting filter. The weighting filters are only available with full bandwidth. Either switch the weighting filter off or do not use decimation.
MSX16w102	MSX16 Unit ... / w102: Error from temperature monitoring. Switch the device off immediately, otherwise it may overheat and possibly be damaged.
MSX16w104	MSX16 Unit ... / w104: Under-voltage or power failure. Ensure a correct power supply (see handbook).
MSX16w105	MSX16 Unit ... / w105: User settings not available: factory settings loaded. Save the settings in the device again.
MSX16w106	MSX16 Unit ... / w106: Calibration data could not be loaded. Standard values will be used - calibrate the device.
MSX16w107	MSX16 Unit ... / w107: Temperature > 80°C. Ensure adequate ventilation for the device (see operating instructions).
MSX16w108	MSX16 Unit ... / w108: The flash memory could not be written to. Repeat the procedure.
MSX16w109	MSX16 Unit ... / w109: The flash memory could not be deleted. Repeat the procedure.
MSX16w111	MSX16 Unit ... / w111: No input modules found. Plug one or more modules into the slots of the device in order to be able to record data. If you only wish to play-back data, please ignore this warning.
MSX16w112	MSX16 Unit ... / w112: Temperature > 82°C, "silent mode" off. Ensure adequate ventilation for the device (see operating instructions).
MSX16w113	MSX16 Unit ... / w113: Channel settings not set. Choose valid values for the required settings
MSX16w114	MSX16 Unit ... / w114: ICP was switched-on, but unavoidably switched from DC- to AC-coupling . DC coupling is not appropriate for ICP supply
MSX16w115	MSX16 Unit ... / w115: Input coupling not set. The input coupling can not be switched to DC when ICP is on. Switch off the ICP supply and try again.
MSX16w128	MSX16 Unit ... / w128: Module 1 swapped. Recalibrate the device.
MSX16w129	MSX16 Unit ... / w129: Module 2 swapped. Recalibrate the device.
MSX16w130	MSX16 Unit ... / w130: Module 3 swapped. Recalibrate the device.
MSX16w131	MSX16 Unit ... / w131: Module 4 swapped. Recalibrate the device.
MSX16w132	MSX16 Unit ... / w132: Module 5 swapped. Recalibrate the device.
MSX16w133	MSX16 Unit ... / w133: Module 6 swapped. Recalibrate the device.
MSX16w134	MSX16 Unit ... / w134: Module 7 swapped. Recalibrate the device.
MSX16w135	MSX16 Unit ... / w135: Module 8 swapped. Recalibrate the device.
MSX16w136	MSX16 Unit ... / w136: Unknown module type (module 1). Request a firmware update from the manufacturer. Modules of unknown type are treated as ICP modules by default.
MSX16w137	MSX16 Unit ... / w137: Unknown module type (module 2). Request a firmware update from the manufacturer. Modules of unknown type are treated as ICP modules by default.
MSX16w138	MSX16 Unit ... / w138: Unknown module type (module 3). Request a firmware update from the manufacturer. Modules of unknown type are treated as ICP modules by default.
MSX16w139	MSX16 Unit ... / w139: Unknown module type (module 4). Request a firmware update from the manufacturer. Modules of unknown type are treated as ICP modules by default.
MSX16w140	MSX16 Unit ... / w140: Unknown module type (module 5). Request a firmware update from the manufacturer. Modules of unknown type are treated as ICP modules by default.
MSX16w141	MSX16 Unit ... / w141: Unknown module type (module 6). Request a firmware update from the manufacturer. Modules of unknown type are treated as ICP modules by default.

Name	Message
MSX16w142	MSX16 Unit ... / w142: Unknown module type (module 7). Request a firmware update from the manufacturer. Modules of unknown type are treated as ICP modules by default.
MSX16w143	MSX16 Unit ... / w143: Unknown module type (module 8). Request a firmware update from the manufacturer. Modules of unknown type are treated as ICP modules by default.
MSX16w144	MSX16 Unit ... / w144: Error during zero-check for output channel 1. Recalibrate the device.
MSX16w145	MSX16 Unit ... / w145: Error during zero-check for output channel 2. Recalibrate the device.
MSX16w146	MSX16 Unit ... / w146: Error during zero-check for output channel 3. Recalibrate the device.
MSX16w147	MSX16 Unit ... / w147: Error during zero-check for output channel 4. Recalibrate the device.
MSX16w148	MSX16 Unit ... / w148: Error during zero-check for output channel 5. Recalibrate the device.
MSX16w149	MSX16 Unit ... / w149: Error during zero-check for output channel 6. Recalibrate the device.
MSX16w150	MSX16 Unit ... / w150: Error during zero-check for output channel 7. Recalibrate the device.
MSX16w151	MSX16 Unit ... / w151: Error during zero-check for output channel 8. Recalibrate the device.
MSX16w152	MSX16 Unit ... / w152: Error during zero-check for output channel 9. Recalibrate the device.
MSX16w153	MSX16 Unit ... / w153: Error during zero-check for output channel 10. Recalibrate the device.
MSX16w154	MSX16 Unit ... / w154: Error during zero-check for output channel 11. Recalibrate the device.
MSX16w155	MSX16 Unit ... / w155: Error during zero-check for output channel 12. Recalibrate the device.
MSX16w156	MSX16 Unit ... / w156: Error during zero-check for output channel 13. Recalibrate the device.
MSX16w157	MSX16 Unit ... / w157: Error during zero-check for output channel 14. Recalibrate the device.
MSX16w158	MSX16 Unit ... / w158: Error during zero-check for output channel 15. Recalibrate the device.
MSX16w159	MSX16 Unit ... / w159: Error during zero-check for output channel 16. Recalibrate the device.
MSX16w160	MSX16 Unit ... / w160: Error during gain-check for output channel 1. Recalibrate the device.
MSX16w161	MSX16 Unit ... / w161: Error during gain-check for output channel 2. Recalibrate the device.
MSX16w162	MSX16 Unit ... / w162: Error during gain-check for output channel 3. Recalibrate the device.
MSX16w163	MSX16 Unit ... / w163: Error during gain-check for output channel 4. Recalibrate the device.
MSX16w164	MSX16 Unit ... / w164: Error during gain-check for output channel 5. Recalibrate the device.
MSX16w165	MSX16 Unit ... / w165: Error during gain-check for output channel 6. Recalibrate the device.
MSX16w166	MSX16 Unit ... / w166: Error during gain-check for output channel 7. Recalibrate the device.
MSX16w167	MSX16 Unit ... / w167: Error during gain-check for output channel 8. Recalibrate the device.

Name	Message
MSX16w168	MSX16 Unit ... / w168: Error during gain-check for output channel 9. Recalibrate the device.
MSX16w169	MSX16 Unit ... / w169: Error during gain-check for output channel 10. Recalibrate the device.
MSX16w170	MSX16 Unit ... / w170: Error during gain-check for output channel 11. Recalibrate the device.
MSX16w171	MSX16 Unit ... / w171: Error during gain-check for output channel 12. Recalibrate the device.
MSX16w172	MSX16 Unit ... / w172: Error during gain-check for output channel 13. Recalibrate the device.
MSX16w173	MSX16 Unit ... / w173: Error during gain-check for output channel 14. Recalibrate the device.
MSX16w174	MSX16 Unit ... / w174: Error during gain-check for output channel 15. Recalibrate the device.
MSX16w175	MSX16 Unit ... / w175: Error during gain-check for output channel 16. Recalibrate the device.
MSX16w176	MSX16 Unit ... / w176: The tacho module is not in slot 8. Plug the tacho module into slot 8 (channel 15/16) (only for hardware version 03)
MSX16w177	MSX16 Unit ... / w177: Error during zero-check for input channel 1. Recalibrate the device.
MSX16w178	MSX16 Unit ... / w178: Error during zero-check for input channel 2. Recalibrate the device.
MSX16w179	MSX16 Unit ... / w179: Error during zero-check for input channel 3. Recalibrate the device.
MSX16w180	MSX16 Unit ... / w180: Error during zero-check for input channel 4. Recalibrate the device.
MSX16w181	MSX16 Unit ... / w181: Error during zero-check for input channel 5. Recalibrate the device.
MSX16w182	MSX16 Unit ... / w182: Error during zero-check for input channel 6. Recalibrate the device.
MSX16w183	MSX16 Unit ... / w183: Error during zero-check for input channel 7. Recalibrate the device.
MSX16w184	MSX16 Unit ... / w184: Error during zero-check for input channel 8. Recalibrate the device.
MSX16w185	MSX16 Unit ... / w185: Error during zero-check for input channel 9. Recalibrate the device.
MSX16w186	MSX16 Unit ... / w186: Error during zero-check for input channel 10. Recalibrate the device.
MSX16w187	MSX16 Unit ... / w187: Error during zero-check for input channel 11. Recalibrate the device.
MSX16w188	MSX16 Unit ... / w188: Error during zero-check for input channel 12. Recalibrate the device.
MSX16w189	MSX16 Unit ... / w189: Error during zero-check for input channel 13. Recalibrate the device.
MSX16w190	MSX16 Unit ... / w190: Error during zero-check for input channel 14. Recalibrate the device.
MSX16w191	MSX16 Unit ... / w191: Error during zero-check for input channel 15. Recalibrate the device.
MSX16w192	MSX16 Unit ... / w192: Error during zero-check for input channel 16. Recalibrate the device.
MSX16w193	MSX16 Unit ... / w193: Error during gain-check for input channel 1. Recalibrate the device.
MSX16w194	MSX16 Unit ... / w194: Error during gain-check for input channel 2. Recalibrate the device.

Name	Message
MSX16w195	MSX16 Unit ... / w195: Error during gain-check for input channel 3. Recalibrate the device.
MSX16w196	MSX16 Unit ... / w196: Error during gain-check for input channel 4. Recalibrate the device.
MSX16w197	MSX16 Unit ... / w197: Error during gain-check for input channel 5. Recalibrate the device.
MSX16w198	MSX16 Unit ... / w198: Error during gain-check for input channel 6. Recalibrate the device.
MSX16w199	MSX16 Unit ... / w199: Error during gain-check for input channel 7. Recalibrate the device.
MSX16w200	MSX16 Unit ... / w200: Error during gain-check for input channel 8. Recalibrate the device.
MSX16w201	MSX16 Unit ... / w201: Error during gain-check for input channel 9. Recalibrate the device.
MSX16w202	MSX16 Unit ... / w202: Error during gain-check for input channel 10. Recalibrate the device.
MSX16w203	MSX16 Unit ... / w203: Error during gain-check for input channel 11. Recalibrate the device.
MSX16w204	MSX16 Unit ... / w204: Error during gain-check for input channel 12. Recalibrate the device.
MSX16w205	MSX16 Unit ... / w205: Error during gain-check for input channel 13. Recalibrate the device.
MSX16w206	MSX16 Unit ... / w206: Error during gain-check for input channel 14. Recalibrate the device.
MSX16w207	MSX16 Unit ... / w207: Error during gain-check for input channel 15. Recalibrate the device.
MSX16w208	MSX16 Unit ... / w208: Error during gain-check for input channel 16. Recalibrate the device.
MSX16w255	MSX16 Unit ... / w255: Buffer for errors and warnings is full. Acknowledge the warnings from the device.