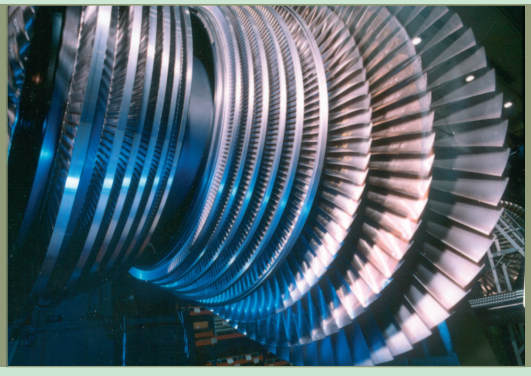
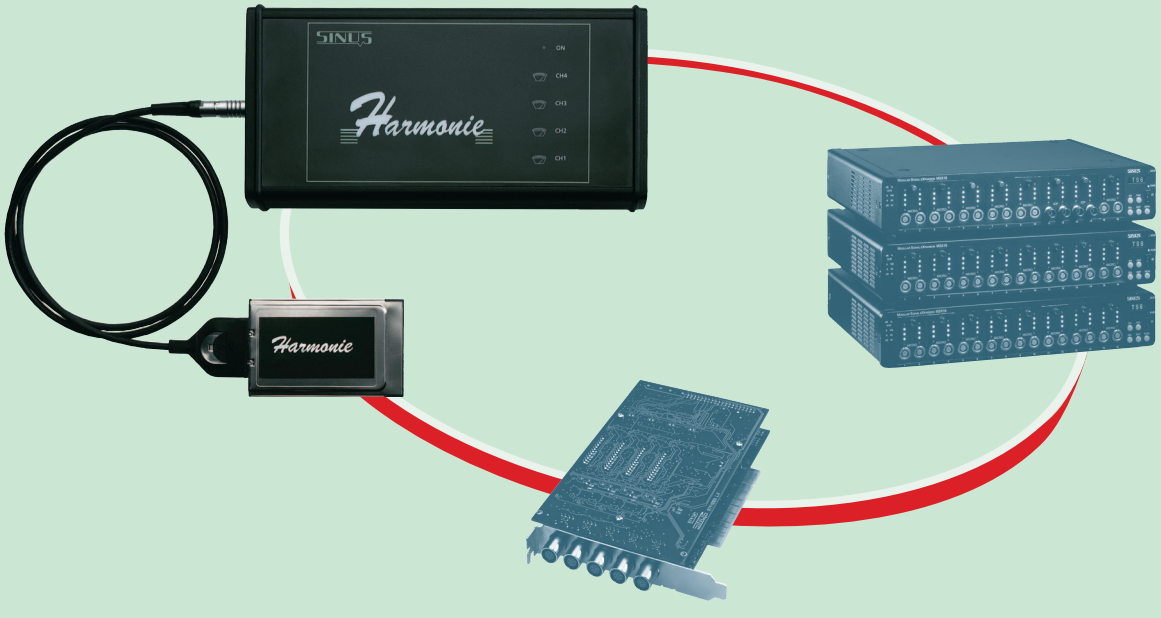


SINUS MATLAB™ Toolbox

- Reliable, stable basis for applications programming
- Operating system & driver independence
- Increased software reusability
- Accelerated development
- Simplified data handling
- MATLAB conventions
- Scaleable number of logical/physical channels



SINUS
Messtechnik GmbH

SMT

Open up the whole MATLAB world for your applications, projects and cooperations with universities and industry!

The SINUS MATLAB Toolbox not only allows applications to be developed faster and more cost-effectively - it also makes them "future-proof", protecting your valuable software investment.

Its stable definition, independent of the underlying operating system and device drivers, also increases modularity by ensuring significant compatibility between different device types (e.g. MSX16 and the HARMONIE™ family) as well as between "live" and "pre-recorded" data.

The application programmer is freed from burdens such as device-specific procedural interfaces, units conversions, channel separation and deblocking.

Installation is also simplified: the SMT itself will attempt to check the installation status and will report any problems found.

MATLAB Release 13 or higher is required.

Functions in the SMT would for example typically be called in the following sequence for a simple data capture application:

```
SMTQueryDevices;
harm = SMTFindDevice(`Type', `HARMONIE');
SMTSelectDevice(harm);
SMTOpenDevice; % (initialization)
SMTSetDevice('FileName', 'TestHarmonie');
ai2 = SMIFindChannel(`Type', `AnalogInput'
      `ChannelID', `IN2');
SMTSetChannel(ai2, `Enabled', 1, ,ICP', 1);
SMTStart(5); % (SMT logs data to file)
[data, time] = SMTGetData(ai2);
      % (data loaded into workspace)
SensorUnit = SMTGetChannel(ai2, `SensorUnit');
plot(time, data); xlabel(`s'); ylabel(SensorUnit);
SMTCloseDevice; % (resources freed)
```

Key Concepts:

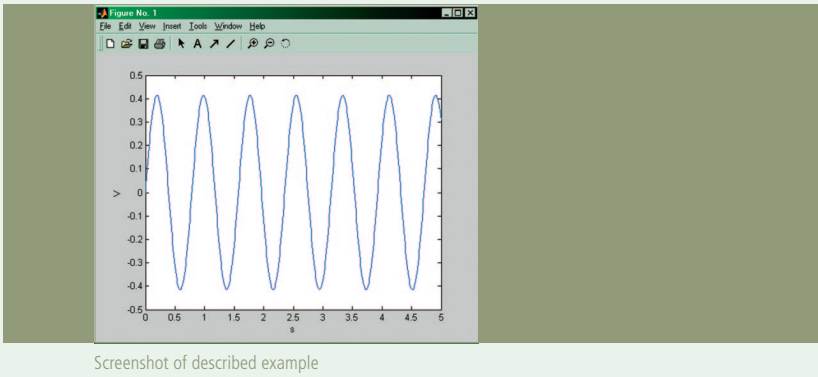
Devices may be physical devices (e.g. a HARMONIE PCI board or an MSX16 tower) or "virtual" devices used to access files previously recorded using the SMT.

Channels are the data streams received from or transmitted to a device (as MATLAB arrays).
SMT channels are "logical" channels, several of which may be based on a given physical channel (reflecting the capabilities of the physical device).

Properties of devices and channels define attributes which can be read and (where appropriate) set via the SMT; they are accessed in a manner familiar to all experienced MATLAB programmers.

As well as general properties available for all device types or channels, properties specific to particular device types and their (logical) channels allow applications to access hardware-specific functionality where necessary without impact on the device-independent portions of the application.

Error Handling in the SMT is performed in a consistent manner: all functions provide success/error/warning codes and messages.
Applications can make use of this in particular to exploit available hardware-specific features at run-time without loss of generality.



Screenshot of described example

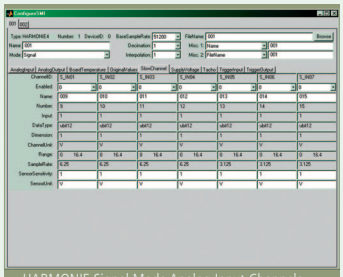
SINUS

Function Syntax Summary

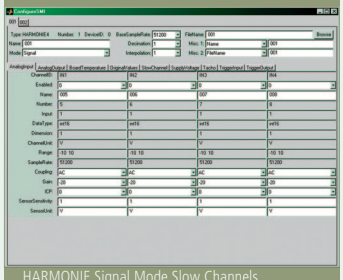
[DeviceNumbers, RC, RM] = SMTQueryDevices
[RC, RM] = SMTSelectDevice(DeviceNumber)
[RC, RM] = SMTOpenDevice
[Properties, RC, RM] = SMTSetDevice
[ValidValues, RC, RM] = SMTSetDevice(
PropertyName)
[RC, RM, ErrorProp] = SMTSetDevice(
PropertyName, PropertyValue, ...)
[Properties, RC, RM] = SMTGetDevice
[Value, RC, RM] = SMTGetDevice(PropertyName)
[DeviceNumbers, RC, RM] = SMTFindDevice(
PropertyName, PropertyValue, ...)
[Properties, RC, RM] = SMTSetChannel(Channel)
[ValidValues, RC, RM] = SMTSetChannel(
Channel, PropertyName)
[RC, RM, ErrorProp] = SMTSetChannel(
Channels,PropertyName, PropertyValue, ...)
[Properties, RC, RM] = SMTGetChannel(Channel)
[Value, RC, RM] = SMTGetChannel(
Channel,PropertyName)
[Channels, RC, RM] = SMTFindChannel(
PropertyName, PropertyValue, ...)
[RC, RM] = SMTSaveProperties(FileName)
[RC, RM] = SMTLoadProperties(FileName)
[RC, RM] = SMTLoadProperties(
FileName, `createdevice')
[RC, RM] = SMTStart
[RC, RM] = SMTStart(Duration)
[Data, Time, RC, RM] = SMTGetData(Channels)
[Data, Time, RC, RM] = SMTGetData(
Channels,`time', StartPos, EndPos)
[Data, Time, RC, RM] = SMTGetData(
Channels,`samples', StartPos, EndPos)
[Data, RC, RM] = SMTPeekData(Channels, Samples)
[RC, RM] = SMTPutData(Channels, Data)
[RC, RM] = SMTStop
[RC, RM] = SMTCloseDevice

(RC and RM are the Result Code and Result Message respectively)

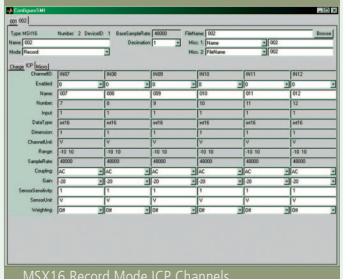
Example Applications



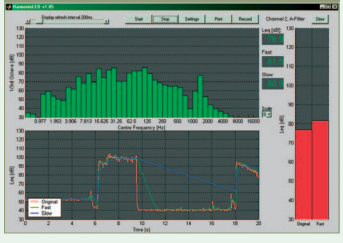
HARMONIE Signal Mode Analog Input Channels



HARMONIE Signal Mode Slow Channels



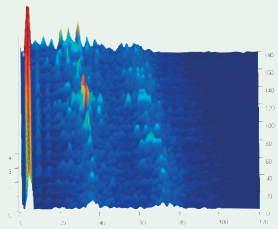
MSX16 Record Mode ICP Channels



Sound level meter application in MATLAB

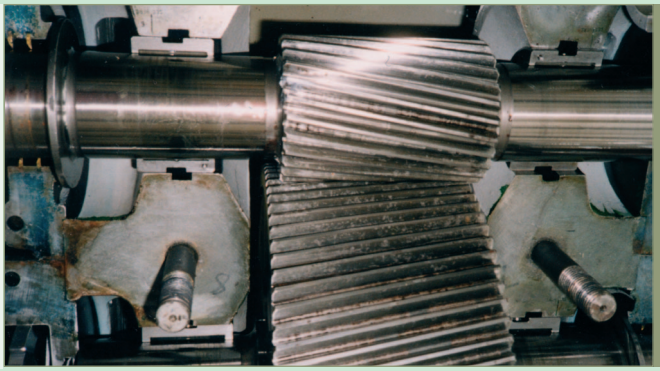
These screenshots show an example application which allows all properties of all available devices and channels to be displayed and set as appropriate - it could be extended to become a fully-fledged data recorder application.

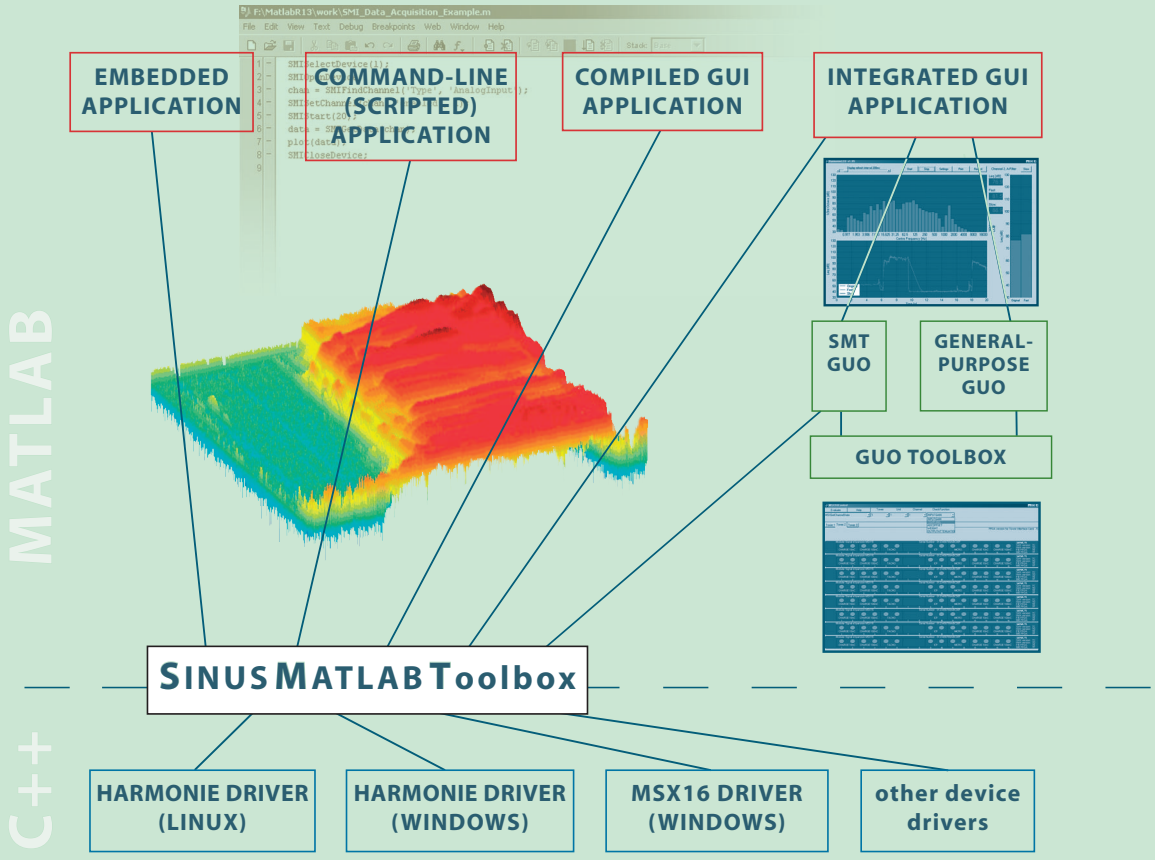
This application is fully SMT-based and contains no device-specific code - it can also be expected to work without change when further devices or properties are supported by the SMT!



Order tracking analysis of rotating machine components

SINUS MATLAB Architecture





SINUS Messtechnik GmbH
Foepplstrasse 13
D-04347 Leipzig
Tel.: +49 341 2 44 29-0
Fax.: +49341 2 4429-99
www.soundbook.de