



Serial PlugIn

How to integrate your own device to IPEmotion

Felix Ottofueling 20.12.2012

Contents

1. Introduction
2. Overview of the architecture
3. IPEmotion interface of the Serial PlugIn
4. Visual Studio 2010 C++ template
5. Examples
 1. Metrix Multimeter MX 556

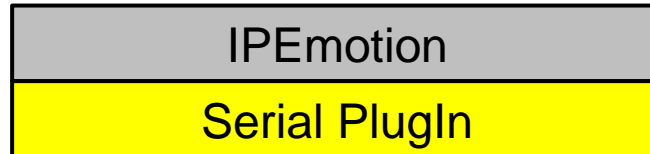
- ▶ The serial PlugIn is a very flexible PlugIn which supports data communication to many serial devices.
- ▶ Anybody can integrate his own device by developing the device specific “acquisition.dll”.
- ▶ Supported interface of this PlugIn version is based on RS232.
- ▶ The Visual Studio 2010 solution template for the C++ interface to handle the data communication is available.
- ▶ The PlugIn supports only to read data. You cannot write or send values from IPEmotion to the serial device.

Overview of the architecture

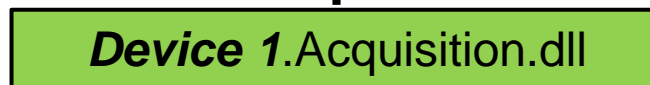


IPEMOTION

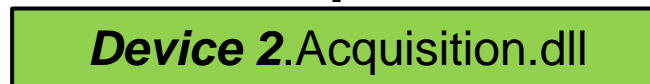
Test & Measurement Software



+



+



+

... n other device

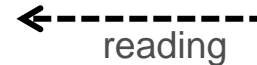
RS232
reading



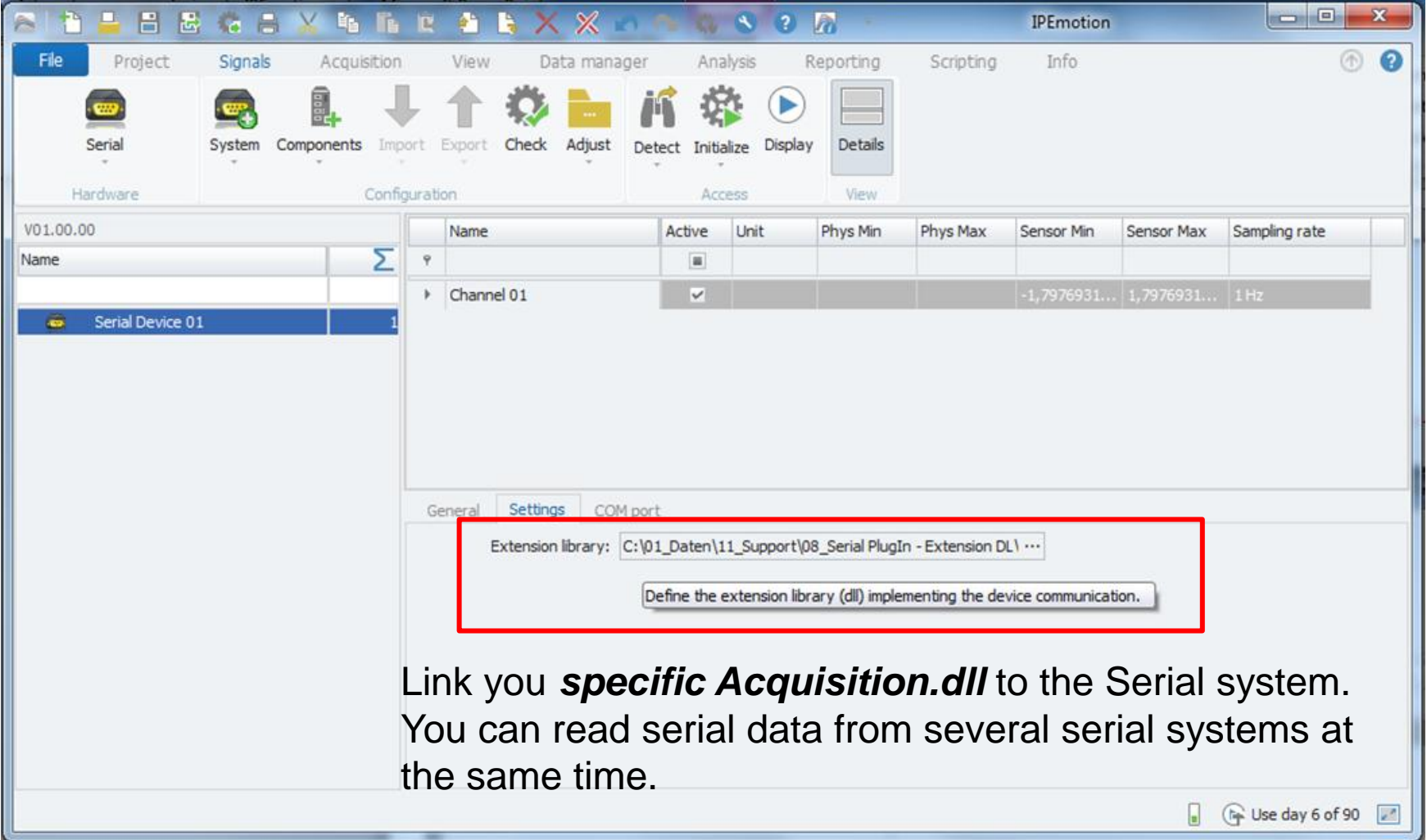
RS232
reading



RS232
reading



IPEmotion interface 1



The screenshot displays the IPEmotion software interface. The top menu bar includes File, Project, Signals, Acquisition, View, Data manager, Analysis, Reporting, Scripting, and Info. Below the menu is a toolbar with icons for Serial, System, Components, Import, Export, Check, Adjust, Detect, Initialize, Display, and Details. The main workspace is divided into several sections. On the left, under 'Hardware', there is a list of devices, with 'Serial Device 01' selected. In the center, a table lists the configuration for 'Channel 01'. The table has columns for Name, Active, Unit, Phys Min, Phys Max, Sensor Min, Sensor Max, and Sampling rate. The 'Active' column for 'Channel 01' is checked. At the bottom, the 'Settings' tab is active, showing the 'Extension library' field with the path 'C:\01_Daten\11_Support\08_Serial PlugIn - Extension DL\ ...'. A red box highlights this field and the text below it: 'Define the extension library (dll) implementing the device communication.'

Name	Active	Unit	Phys Min	Phys Max	Sensor Min	Sensor Max	Sampling rate
Channel 01	<input checked="" type="checkbox"/>				-1,7976931...	1,7976931...	1 Hz

Extension library: C:\01_Daten\11_Support\08_Serial PlugIn - Extension DL\ ...

Define the extension library (dll) implementing the device communication.

Link your **specific Acquisition.dll** to the Serial system. You can read serial data from several serial systems at the same time.

IPEmotion interface 2

The screenshot displays the IPEmotion software interface. The top menu bar includes File, Project, Signals, Acquisition, View, Data manager, Analysis, Reporting, Scripting, and Info. Below the menu is a toolbar with icons for Hardware (Serial), Configuration (System, Components, Import, Export, Check, Adjust), Access (Detect, Initialize, Display), and View (Details). The main workspace is divided into two panes. The left pane shows a tree view with 'V01.00.00' and 'Serial Device 01'. The right pane displays a table of channels.

Name	Active	Unit	Phys Min	Phys Max	Sensor Min	Sensor Max	Sampling rate
Channel 01	<input checked="" type="checkbox"/>				-1,7976931...	1,7976931...	1 Hz

Below the table, the 'COM port' settings are visible, highlighted by a red rectangle:

- Port number: COM-1
- Baud rate: 9,6 kBd
- Data bits: 8
- Parity: NONE
- Stop bits: 1
- Flow control: NONE

Configure the COM port settings as specified in the technical manual of the serial device.

At the bottom right, there is a status bar showing 'Use day 6 of 90'.

Visual Studio template of Acquisition.dll



```
#pragma region Copyright 2012 IPETRONIK GmbH & Co.KG
//
// All rights are reserved. Reproduction or transmission in whole or in part, in
// any form or by any means, electronic, mechanical or otherwise, is prohibited
// without the prior written consent of the copyright owner.
//
// KGI
//
#pragma endregion

////////////////////////////////////
/// <summary>
/// include
/// </summary>
#include "Extension.h"

////////////////////////////////////
/// <summary>
/// lib
/// </summary>
#pragma comment(lib, "Version.lib")

////////////////////////////////////
/// <summary>
/// Global member: callback functions
/// </summary>
PFN_EXT_SET_PACKET_DATA_FLOAT64_FUNC g_pfnDataFloat64 = NULL; // transfer packet data of double type to the plugin
PFN_EXT_SEND_DATA_FUNC g_pfnSendData = NULL; // send data via the systems COM port
PFN_EXT_READ_INPUT_BUFFER g_pfnReadInputBuffer = NULL; // read new data received via the
// systems COM port since last call of g_pfnReadInputBuffer
PFN_EXT_RESET_INPUT_BUFFER g_pfnResetInputBuffer = NULL; // reset the input buffer of the systems COM port

PFN_EXT_TRACE_MESSAGE g_pfnTraceInformation = NULL; // Set a information message to the trace log. Message is
// a null terminated WCHAR string with maximum length of MAX_WCHAR (currently 256)
PFN_EXT_TRACE_MESSAGE g_pfnTraceWarning = NULL; // Set a warning message to the trace log. Message is
// a null terminated WCHAR string with maximum length of MAX_WCHAR (currently 256)
PFN_EXT_TRACE_MESSAGE g_pfnTraceError = NULL; // Set a error message to the trace log. Message is
// a null terminated WCHAR string with maximum length of MAX_WCHAR (currently 256)

#ifdef __cplusplus
extern "C" {
#endif

////////////////////////////////////
/// <summary>
/// Called when the driver is being initialized. This function is called once at the end of the PlugIn function "InitDriver".
/// InitDriver is called directly after SetCallback.
/// </summary>
/// <param name="pwszErrorMessage">
/// reserved for future use -
/// pointer to buffer for an error message, propagated to
/// IPEmotion when function returns E_FAIL
/// </param>
```

- ▶ The Visual Studio 2010 C++ template is available.
- ▶ The integration of new devices is therefore more easy.
- ▶ The Acquisition.dll includes the C code to handle the device specific commands to read the data.

For any questions get in touch
with the IPEmotion SW
development team!

Examples

1. Metrix Multimeter MX 556

Is reading the display value via RS232

