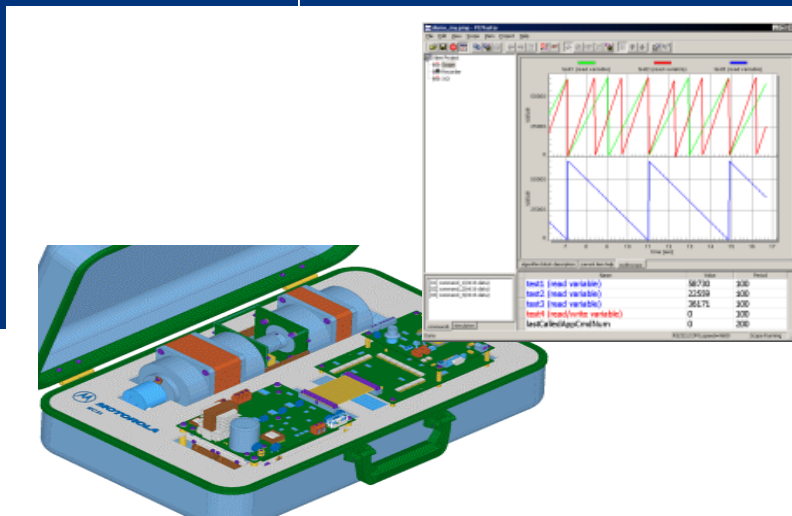


# Embedded PC Master Application



Overview

June 3, 2002

Rev. 0.2

MOTOROLA and the Stylized M Logo are registered in the US Patent & Trademark Office. All other product or service names are the property of their respective owners. © Motorola, Inc. 2002.



**MOTOROLA**  
intelligence everywhere™

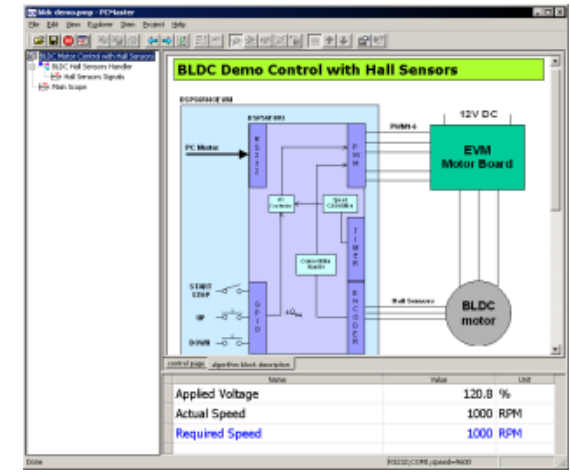
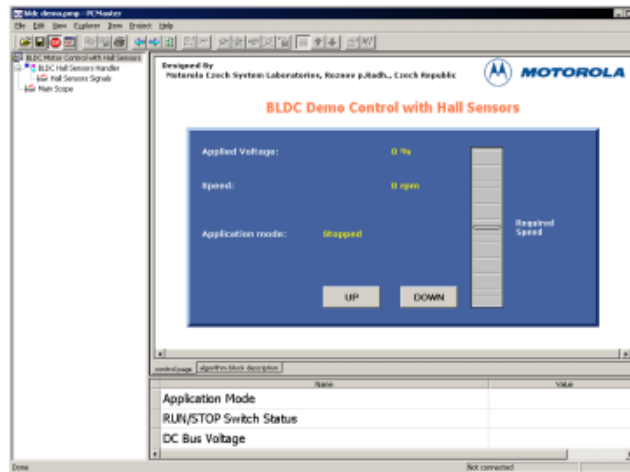
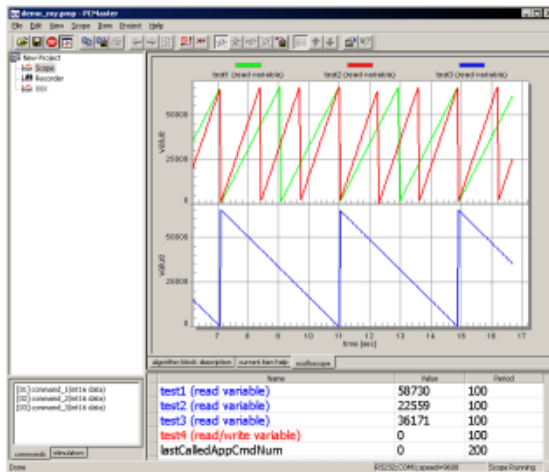
*digitaldna™*

# What is PC master software?

- Real Time Monitor
- Control Panel
- Demonstration Platform/  
Selling tool



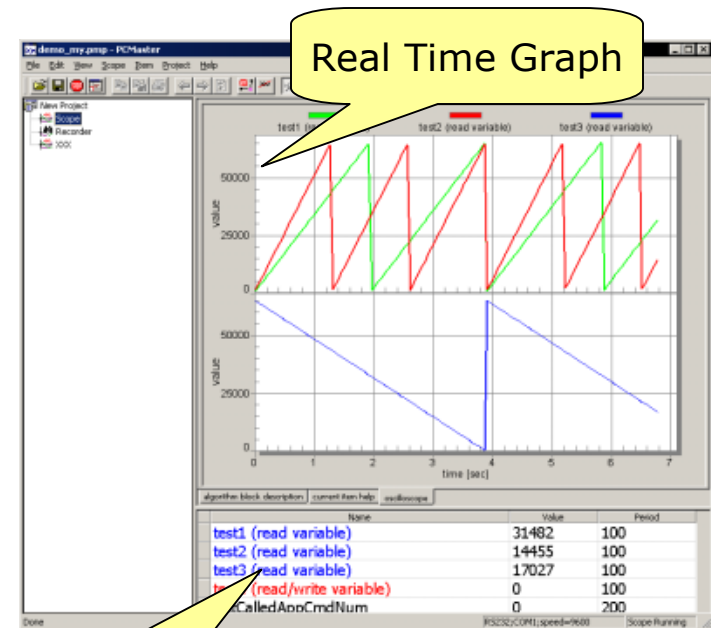
FOR YOUR  
EMBEDDED  
APPLICATION



# 1. Real Time Monitor

Watching on-board variables or memory locations in various formats:

- Text
  - name
  - value - hex, dec, real, ...
  - min, max
  - enumerated labels
- Real-time waveform (real-time oscilloscope)
- High-speed recorded data (on-board memory oscilloscope)

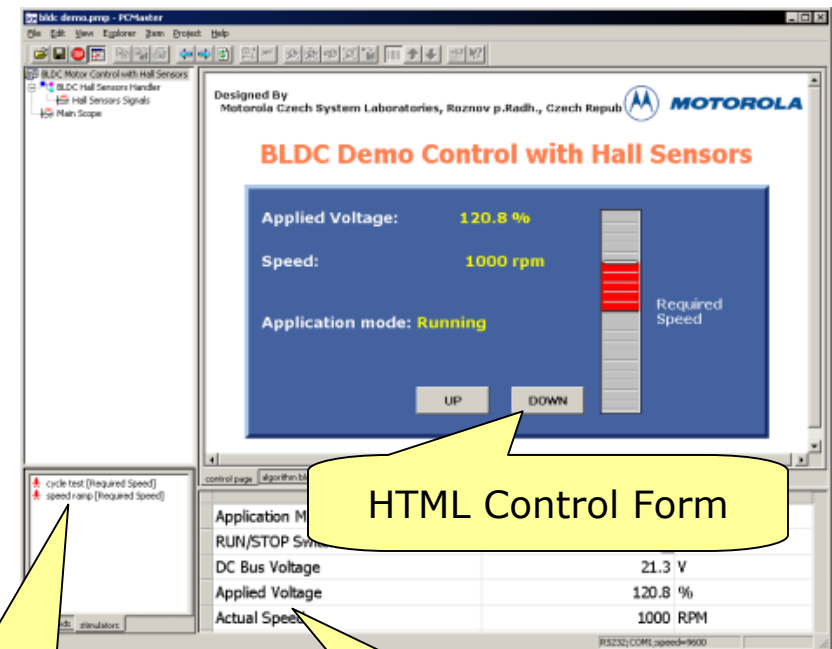


Variable Watch

## 2. Control Panel

Several ways how to control embedded applications:

- Direct setting of the variable value from the variable watch
- Time-table stimulation of the variable value
- User command / message control
- VisualBasic script-powered HTML Forms (push buttons, indicators, sliders, ...)
- By external application (e.g. Excel) embedded in GUI



Stimulators

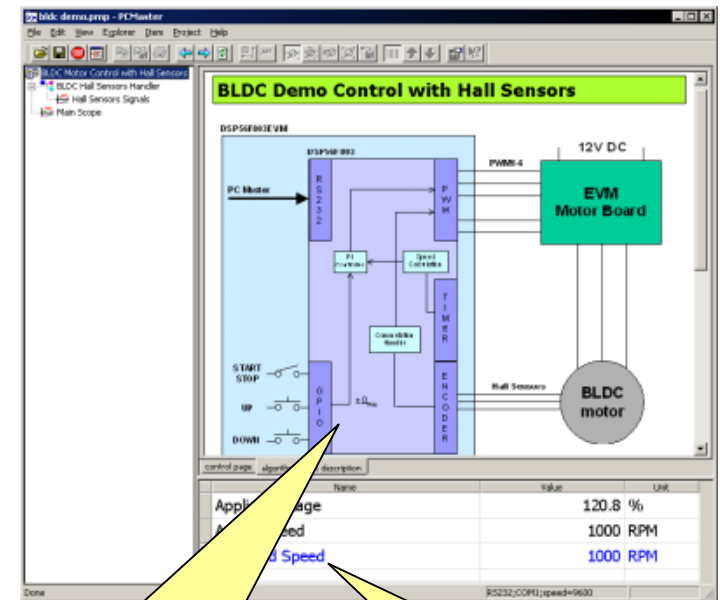
HTML Control Form

Variable Watch

### 3. Demonstration Platform

The features of an embedded application can be both described and demonstrated

- By HTML pages that can contain
  - text and pictures
  - sounds, video sequences
  - internet links
  - and any other web contents
- Simultaneous live data monitoring
- Browsing through the functional blocks of embedded application



Sell your work  
using this tool !

HTML Description Page

Variable Watch

# Application Window: Project Tree

The screenshot shows the PCMaster application window with the title bar 'demo\_my.pmp - PCMaster'. The menu bar includes File, Edit, View, Explorer, Item, Project, and Help. The toolbar contains various icons for file operations and project management. The main window is divided into three panes:

- Project Tree:** Located on the left, it displays a hierarchical structure of project items. The tree starts with 'New Project' and includes 'Block1' (containing 'Scope' and 'Recorder'), 'Block2' (containing 'Sub-Block' and 'Scope'), and another 'Scope' and 'Recorder' at the bottom.
- Variable Watch:** Located at the bottom left, it is currently empty.
- Detail View:** Located on the right, it displays information about the selected item in the Project Tree. It includes a list of variables (e.g., 'Show me w...'), a section for 'Other links' (e.g., 'Welcome Screen'), and a table for 'algorithm block description' with columns for 'Name' and 'Value'.

A green callout box explains the Project Tree:

The project tree contains the defined logical project structure. There are three kinds of tree items:

1. Block items - each block item is assigned a list of variables displayed in the "Variable Watch" when the item is selected. Block items can contain sub-items of any kind.
2. Oscilloscope item - displays the real-time graph of selected variable values. The graph is displayed in the Detail View when the oscilloscope item is selected in the tree.
3. Recorder item - displays the high-speed graph of selected variable values.

# Application Window: Variable Watch

**The Variable Watch displays immediate values of selected variables.**

**When the variable is write-enabled, the user is allowed to change the variable value directly in the watch grid.**

**Variable Watch**

Name	Value	Period
test1 (read variable)	64326	100
test2 (read variable)	31697	100
test3 (read variable)	32877	100

**Drop-down list with variable values**

variable\_04

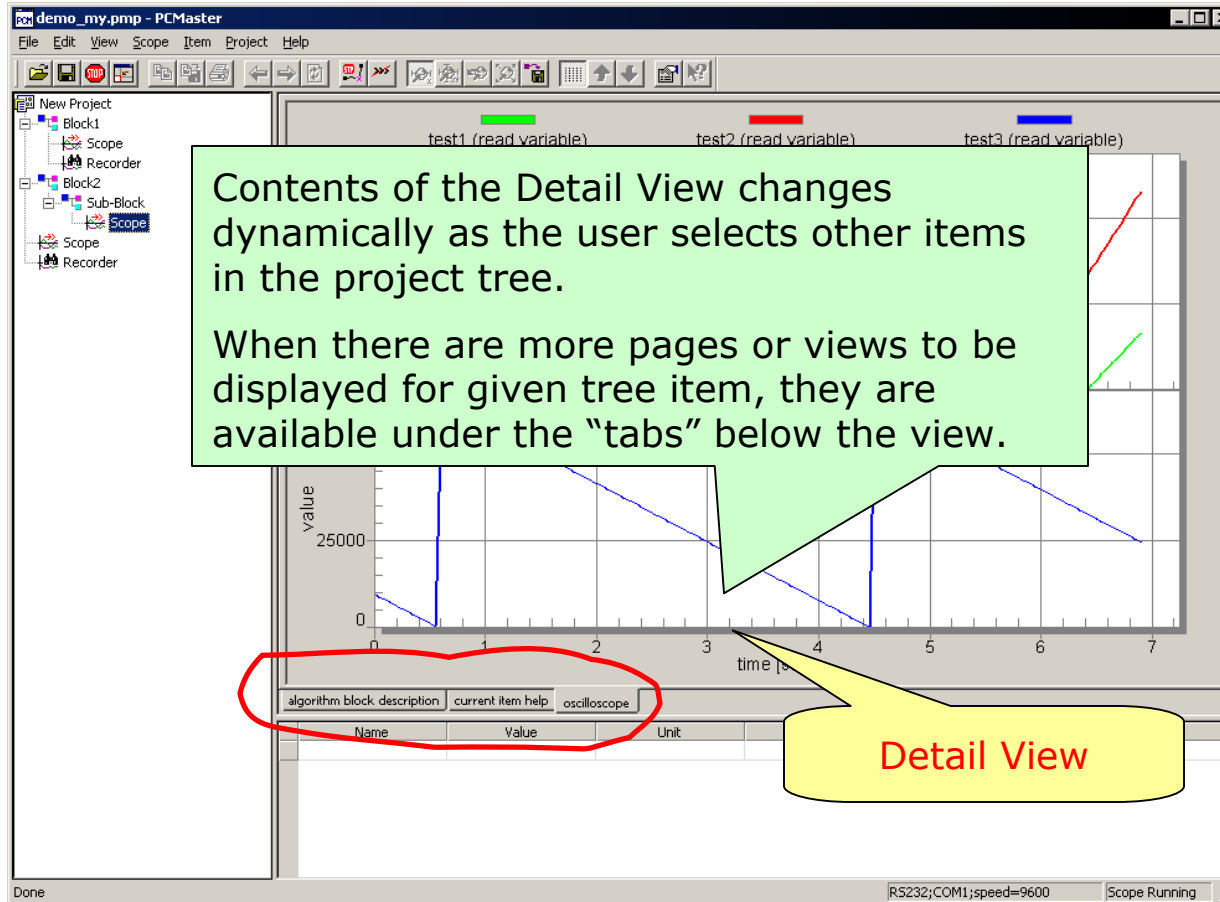
1
2
3
4
5
6
7

**Variable values with text labels assigned**

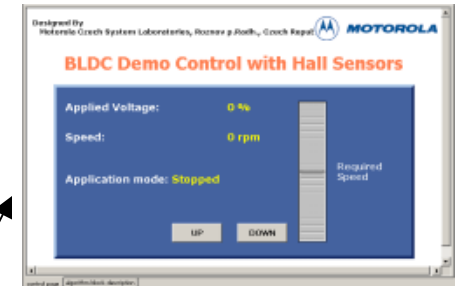
variable\_05

ON
OFF
ON

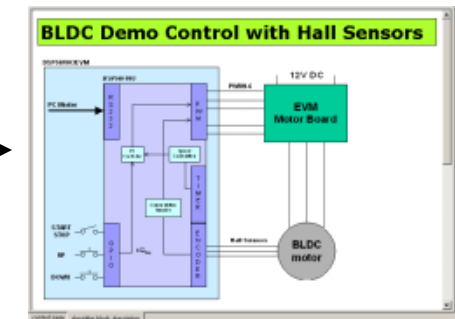
# Application Window: Detail View



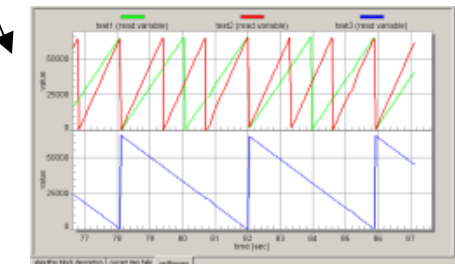
Control Page



Description Page

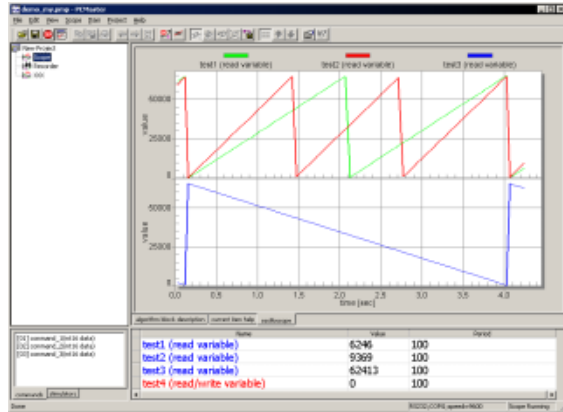


Scope or Recorder Page





# How it Works?: Communication



**PC Communication Library**

**Plug-in Module**

**RS232**

On the PC side, the application uses a separately loaded dynamic library to handle the communication with the target.

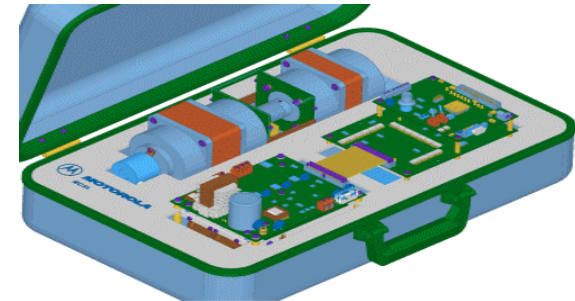
In the simplest scenario, the library uses built-in RS232 implementation to communicate with the target.

Both the communication protocol and the library are documented and may be used in custom applications.

On the embedded side, the communication and protocol is implemented in the object module, which has to be linked together with an embedded application.

Source code of the module is available.

RS232



**Embedded Application**

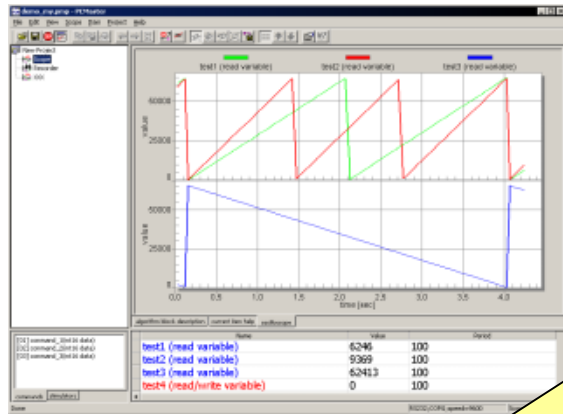
**Protocol Implementation**



**MOTOROLA**  
intelligence everywhere™

*digitaldna™*

# How it Works?: Comm. Plug-in Modules



PC Communication Library

Plug-in Module (any media)

A custom communication plug-in module can be used instead of the RS232 interface.

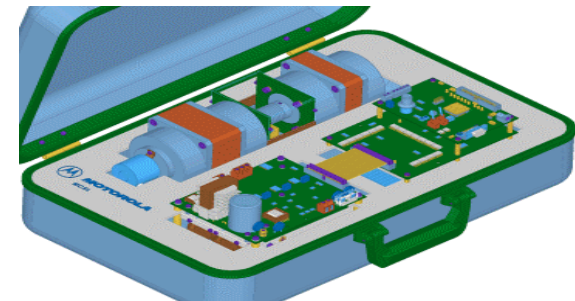
User can write own modules to implement the transport layer for the defined protocol.

Plug-in modules can be written in any programming language. They are distributed and used in a binary form of the ActiveX component (Microsoft COM technology).

USB, Ethernet, ...

Typically, the user would use or create a custom plug-in module to implement a connection via alternative interfaces.

For example: USB, CAN or Ethernet



Embedded Application

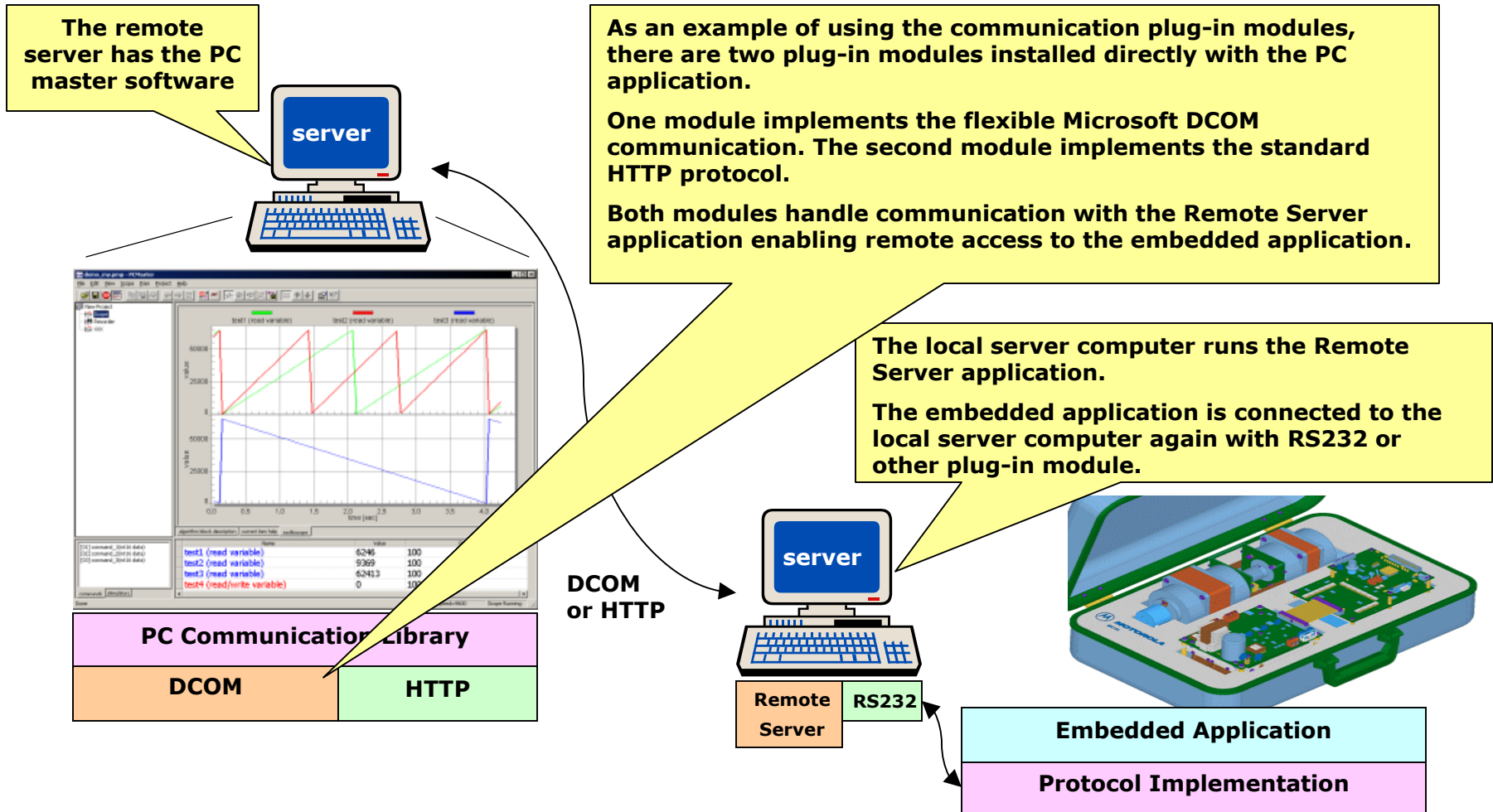
Protocol Implementation



**MOTOROLA**  
intelligence everywhere™

*digitaldna™*

# How it Works?: Remote Connection



**MOTOROLA**  
intelligence everywhere™

*digitaldna™*

# Variables

A variable represents a memory location in the embedded application. It is defined by:

- **name**
- **sampling frequency**
- **address (or symbol from ELF or MAP file)**
- **type**
- **size or bit field**
- **display format**
- **numeric transformation**
- **text labels assignment**

The image shows a 'Variable' configuration dialog box with two tabs: 'Definition' and 'Modifying'. The 'Definition' tab is active. The dialog contains the following fields and options:

- Variable name:** variable\_01
- Sampling period:** 1 sec
- Show as:** DEC
- Variable section:**
  - Address:** 0x0000
  - Type:** unsigned fixed point
  - Size:** 2 bytes
- Bit fields section:**
  - When the value is received:**
    - shift it: 0 bits right, and..
    - mask with: no mask (-1)
- Show section:**
  - ☒ val ☐ min ☐ max
  - Fixed digits:** (spin box)
  - ☐ Fill left zeroes
- Real type transformation:**
  - None** (dropdown)
  - Unit:** unit
  - No parameters** (button)
  - ☐ Use 'Moving Averages' filter
  - ☒ Reset history on manual modify
  - History time:** 5000 ms
- Text enumeration (after transform):**
  - ☐ Enumeration enabled ☐ Always show numeric value
  - default:** unknown
  - ☒ Show number
  - Edit..** (button)
  - Add..** (button)
  - Del** (button)

At the bottom are buttons for **OK**, **Cancel**, **Apply**, and **Help**.

# Application Commands

An application command is a message sent to the embedded application from the master

- Application command can be defined to have any number of numeric parameters, which are delivered to embedded application too
- Embedded application is notified about receiving the command by setting the dedicated flag bit
- Master application periodically polls the result value to determine when the command is processed
- Any text can be assigned to the command result value and displayed in GUI when the command is processed

# Project Deployment

It is very easy to deploy the GUI project to other users or customers:

- Whole project is saved to a single file
- Resource files (HTML code, scripts, images, etc.) are packed into the project file and unpacked in other computer's temporary space
- The project can be protected against changes by activating the “demo mode”
- The “demo mode” can be password-protected

