DemoBox user manual

Disclaimer

VeryLogic GmbH provides this product and the document on an "as is" basis without warranty of any kind, either expressed or implied, including, but not limited to, the implied warranties of merchantability or fitness for a particular purpose. No claims will be accepted for damages howsoever arising as a result of use or failure of this product. Your statutory rights are not affected. This product or any variant of it is not intended for use in any appliance, device or system in which the failure of the product might reasonably be expected to result in personal injury.

This document could contain technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in later editions of this document. VeryLogic GmbH is under no obligation to notify any person of the changes.

0 Abbreviations

DC	Direct Current
FPGA	Field Programmable Gate Array
GUI	Graphical User Interface
HSR	High Availability Seamless Redundancy
IP	Intellectual Property
LED	Light-Emitting Diode
MAC	Media Access Control
PC	Personal Computer
PCB	Printed Circuit Board
PHY	Physical layer device
PRP	Parallel Redundancy Protocol
Tcl/Tk	Tool command language / Toolkit
USB	Universal Serial Bus

1 DemoBox

1.1 Hardware overview

The DemoBox hardware platform was developed for the evaluation of the FPGA IP cores offered by VeryLogic GmbH. The DemoBox hardware consists of a small low-cost Altera FPGA, a quad-port 10/100/1000 MBit Ethernet PHY transceiver, an USB controller and a DC/DC-converter soldered on a PCB (Figure 1) and enclosed in aluminum housing. Several connectors are used to connect the electronics with the external devices. In conjunction with the PC software the USB interface is used to monitor and access the register set of the evaluated core. The DC power connector connects a 5 Volt to 12 Volt power supply (center pin is positive). The BNC connector and the DIP switches are not used at the moment and are reserved for future use.

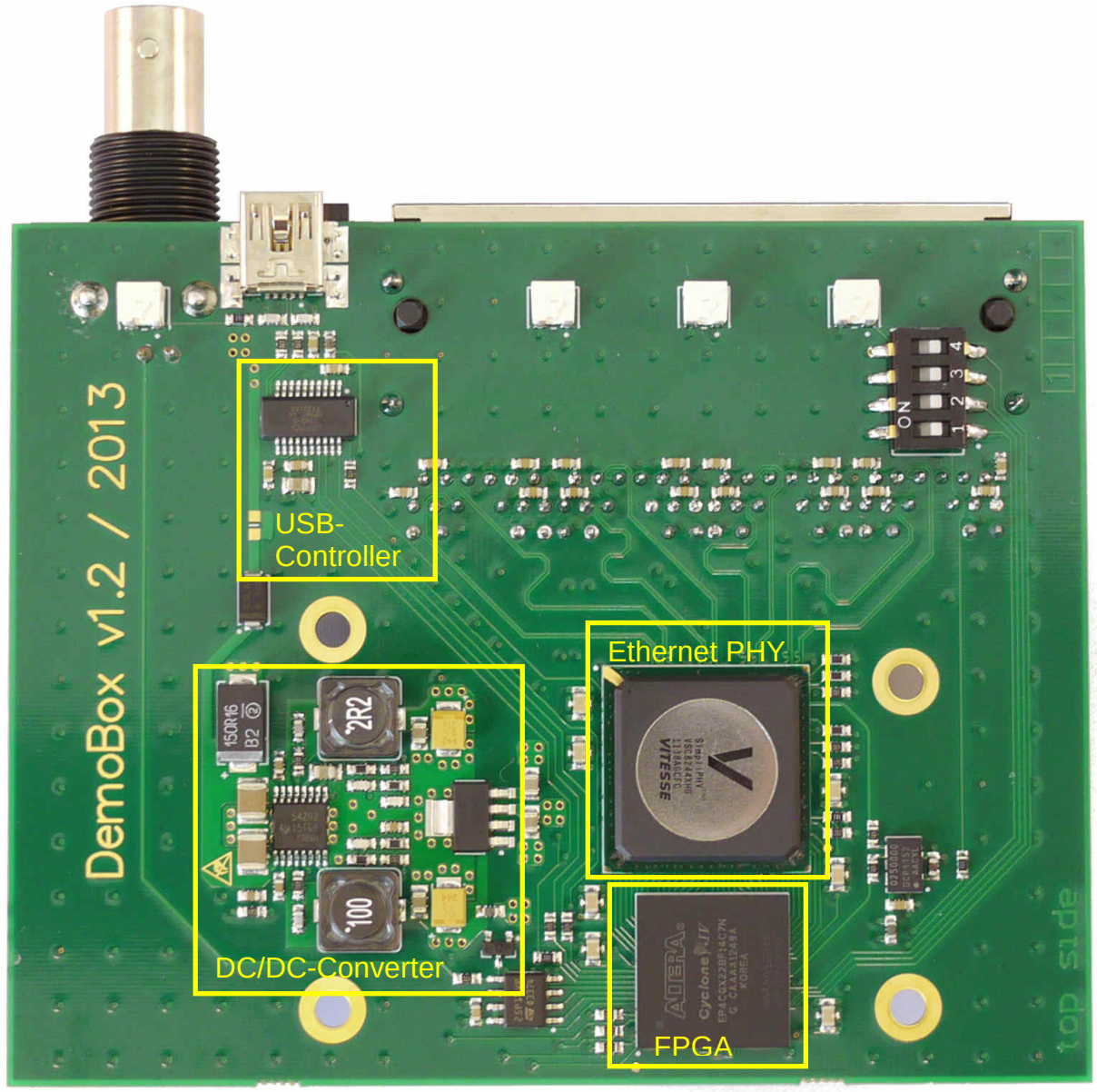

Figure 1: DemoBox PCB

The blue and green Light-Emitting Diodes (LEDs) provide status information of the DemoBox hardware (Table 1). The yellow and red LED provides status information of the IP-Core instantiated on the FPGA. If one of both core LEDs is active, the user should use USB-Monitor to inspect the register set for details.

LED	Function
Blue	Power supply status
Green	Active if the FPGA was successfully configured
Yellow	Active if a non-critical and recoverable issue was detected in the register set (e. g. frame drop)
Red	Active if a critical and non-recoverable error was detected in the register set (e. g. FIFO overflow).

Table 1: LED function description

The function of the quad RJ45 Ethernet connector is core-specific (Table 2).

LAN port	PRP-Core	HSR/PRP-Core
A	LAN A	ring port A / LAN A
B	LAN B	ring port B / LAN B
C	non-redundant node(s)	non-redundant node(s)
D	not used	not used

Table 2: Core-specific Ethernet port mapping

1.2 IP core monitoring and control

The USB Monitor software tool is provided with the DemoBox hardware to monitor and control the evaluated IP cores. The USB Monitor visualizes the register set data, which is collected and sent by a dedicated RegSetToUSB component inside the FPGA (Figure 2). This component reads the IP cores register set periodically and converts the binary register values to the human readable hexadecimal strings. Each register value is put in its own string terminated by a new line character. Each string contains the register address and the register content as an 8-bit respective a 32-bit hexadecimal value (Figure 5). The strings are sent to the FTDI USB controller and then consequently to the connected PC.

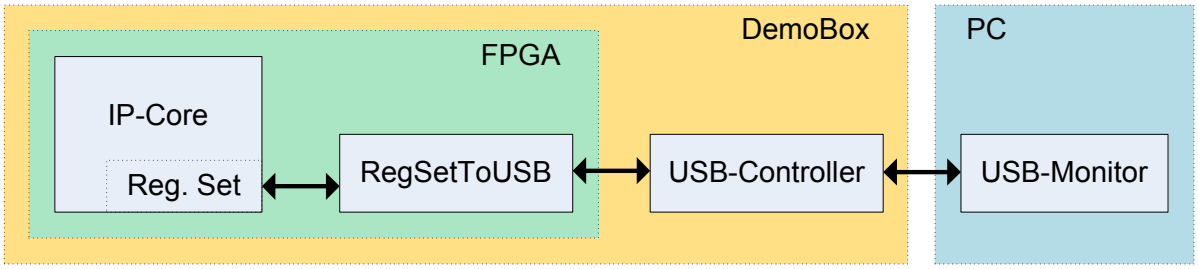

Figure 2: Register set access path

The USB Monitor can also modify the register set content by sending the new register value in the opposite direction to the RegSetToUSB component. The FPGA component writes then the new value into the IP-Cores register.

2 Software installation

2.1 USB driver and Tcl/Tk interpreter installation

To monitor the register set of the DemoBox the FTDI Virtual COM port driver and ActiveState Tcl/Tk language interpreter should be installed first. Both software packages can be downloaded directly from the Internet:

- FTDI Virtual COM driver download page,

Note: FTDI provides also driver installation guides.

- ActiveState ActiveTcl download page.

Note$_1$: Register the "*.tcl" file extension to the Tcl/Tk interpreter during the installation process for direct script execution.

Note$_2$: The USB Monitor script supports Microsoft Windows operation systems only.

After proper installation of the FTDI Virtual COM port driver, an additional COM port appears in the Windows Device Manager when the DemoBox is plugged to the PC (Figure 3).

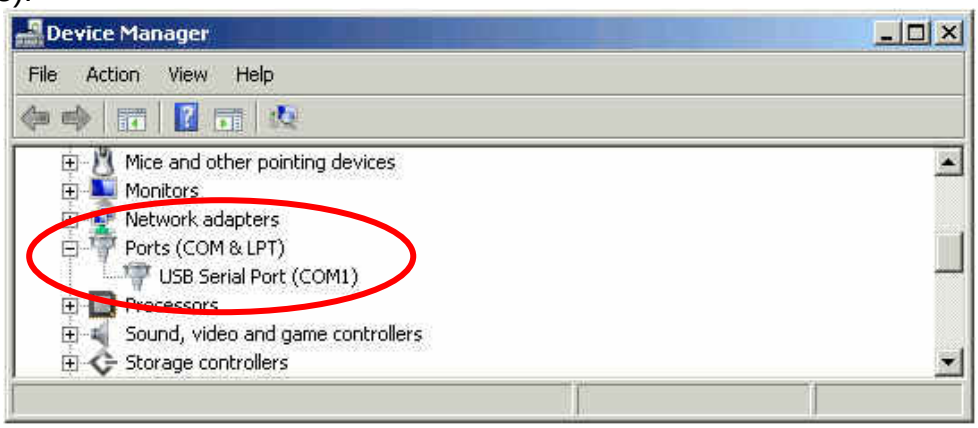

Figure 3: Additional COM port in the Windows Device Manager

During startup the USB Monitor retrieves the information of all COM ports available at the current system. Therefore the USB Monitor should be started *after* the DemoBox was connected to the PC and the Virtual COM-Port was detected. If "*.tcl" file extension was assigned to the Tcl/Tk interpreter, then the USB Monitor can be directly started by a double-click on the script icon. Otherwise the script should be started in the Wish console executing the "source full_path\USB_Montor.tcl" command. The "full_path" should be replaced with the file system path showing to the USB Monitor script location. If the USB Monitor was successfully started showing the Virtual COM port in the COM port combo box (Figure 7), then we can proceed with the USB Monitor description. Otherwise consult the next section dedicated to USB connection troubleshooting.

2.2 USB connection troubleshooting

If establishing the USB connection fails, first of all the presence of the USB Serial Converter and of the Virtual COM port should be checked in the Windows Device Manager. Sometimes the FTDI USB controller fails to initialize the USB connection properly, when the DemoBox is plugged to the PC. The Device Manager shows then

an “Unknown Device” in the USB Controllers group (Figure 4). In such a case a simple disconnect and re-connect of the DemoBox fixes the problem.

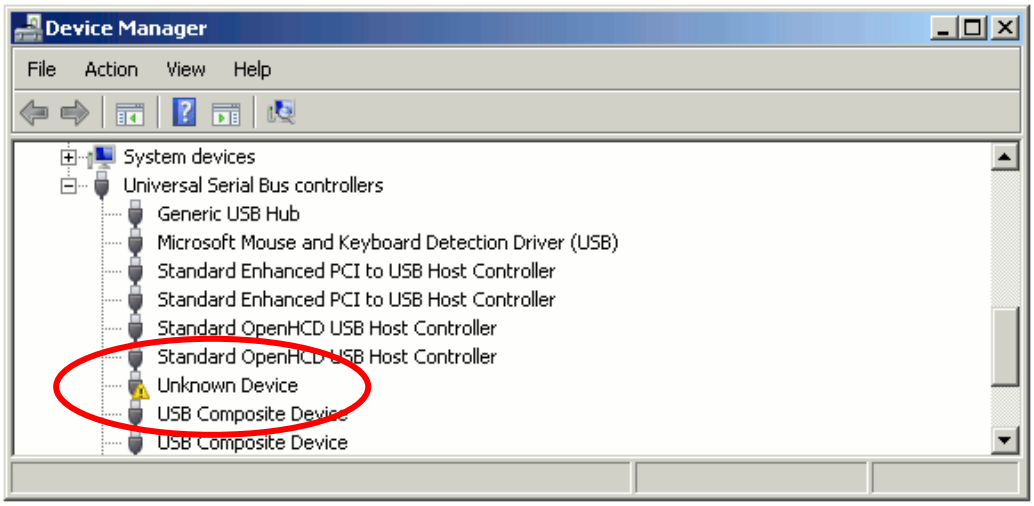

Figure 4: Unrecognized device in USB Controller Group

If the additional COM port is present in the Windows Device Manager, the data transfer can be checked by using any serial terminal like HyperTerminal or TeraTerm (Figure 5). The COM port settings are 115,200 Baud without hardware handshake. The DemoBox starts the transmission of the register contents once the start char (‘s’) is received from the PC. After pressing the pause key (‘p’) the transmission should be stopped.

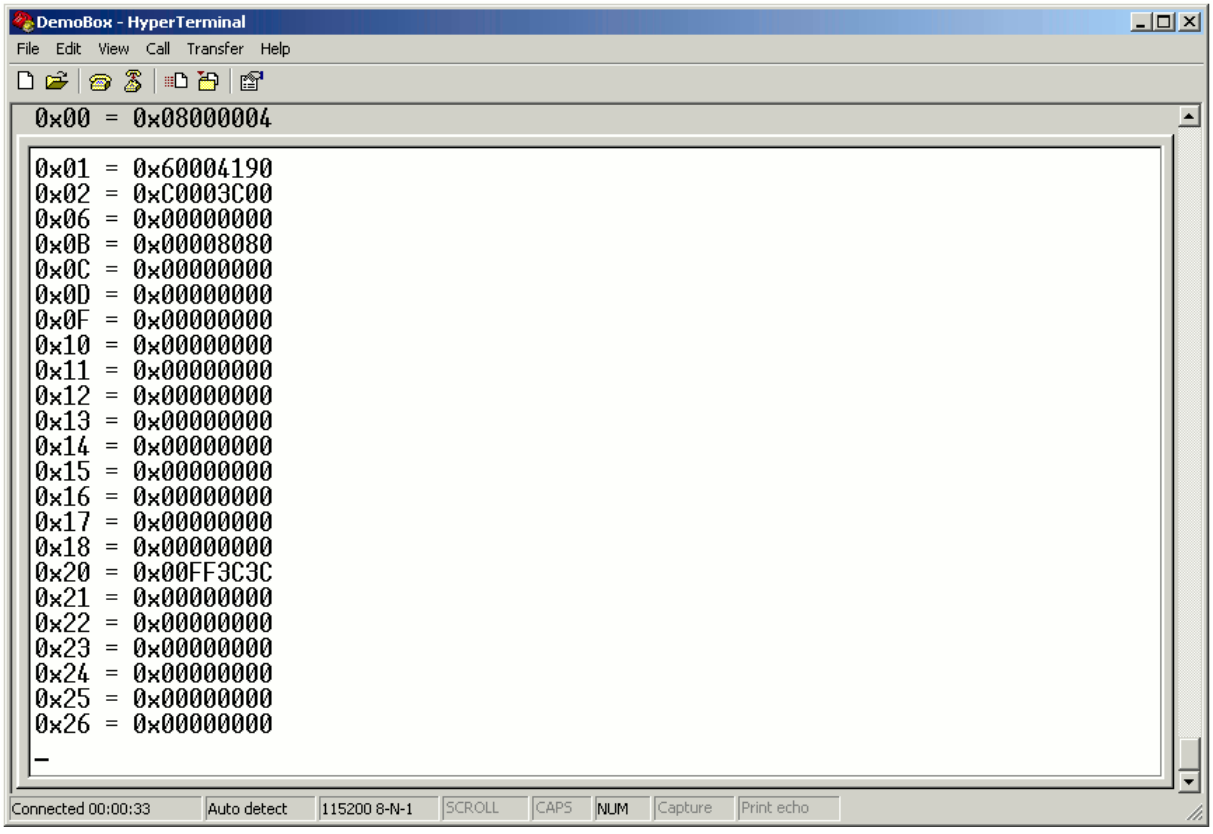

Figure 5: Direct data output in the HyperTerminal

The USB Monitor accepts only COM ports smaller than COM10. If your system maps the Virtual COM port to COM10 or higher, you need to remap it. This can be done in the Windows Device Manager (Figure 6)

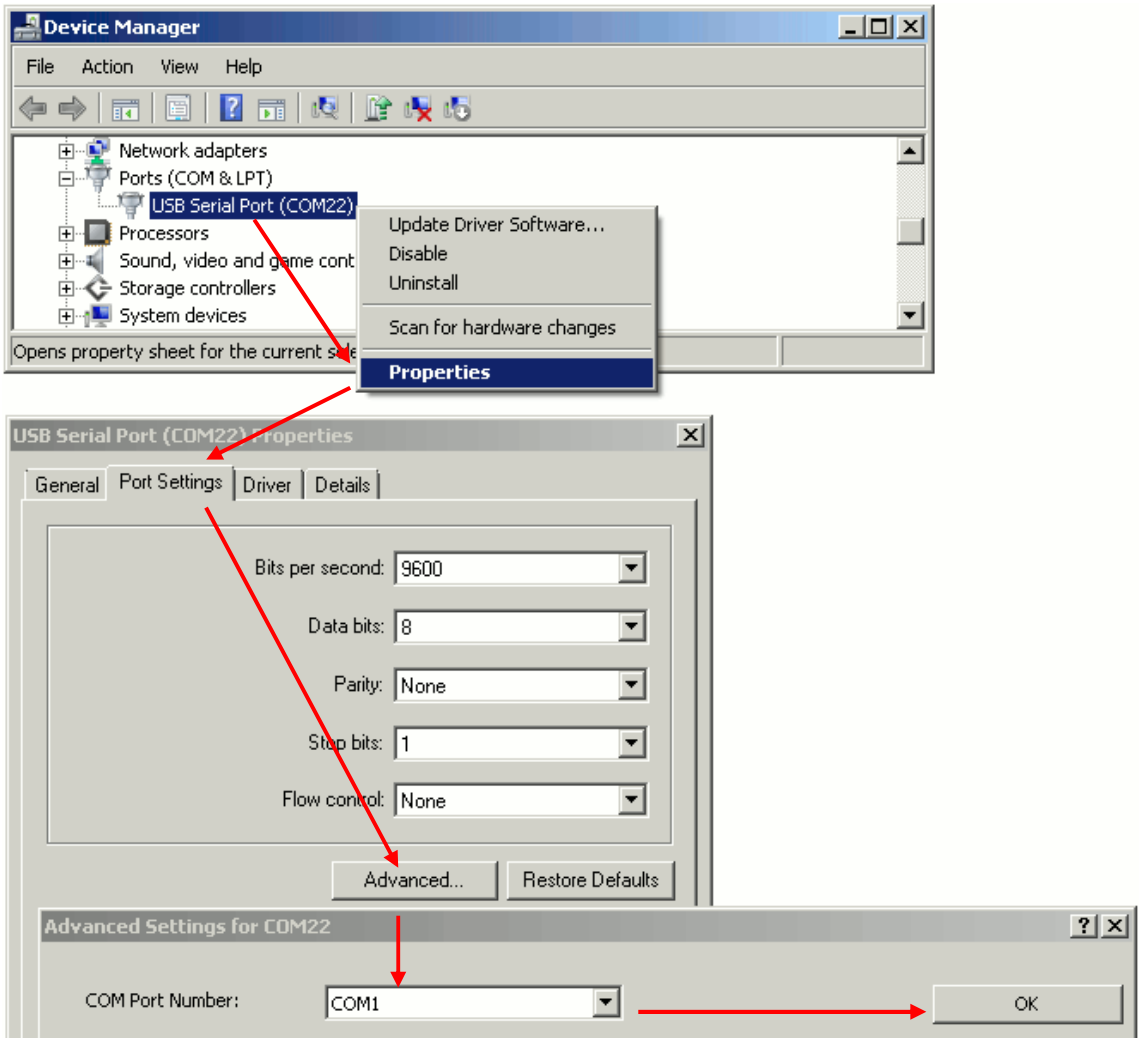

Figure 6: COM port remapping

3 USB Monitor description

The USB Monitor is intended to support all IP cores offered with DemoBox hardware for the evaluation. Therefore USB Monitor always shows the Graphical User Interface (GUI) tailored to monitor and control a specific IP core delivered with the DemoBox hardware. The following section describes general USB Monitor features by the example of the HSR/PRP-Cores GUI. The user is strongly encouraged to use specific register set description in the IP cores user manual when working with the USB Monitor.

3.1 Core Version Register notebook

The Core Version Register (CVER) notebook (Figure 7) contains general information about the IP core like Core ID, Version fields and the information which optional core features (e.g. Nodes Table) are present. The CVER notebook is the active notebook shown immediately after the USB Monitor start. Once the USB Monitor was started, the connection to the DemoBox can be established by selection of the Virtual COM port dedicated to the DemoBox and pressing the “Connect” button. The register

values are then periodically updated and as long the USB Monitor receives valid data from the DemoBox, the blue box in the right bottom corner is slowly moving. Hence the fields in Core Version Register are read only, none of the values can be modified.

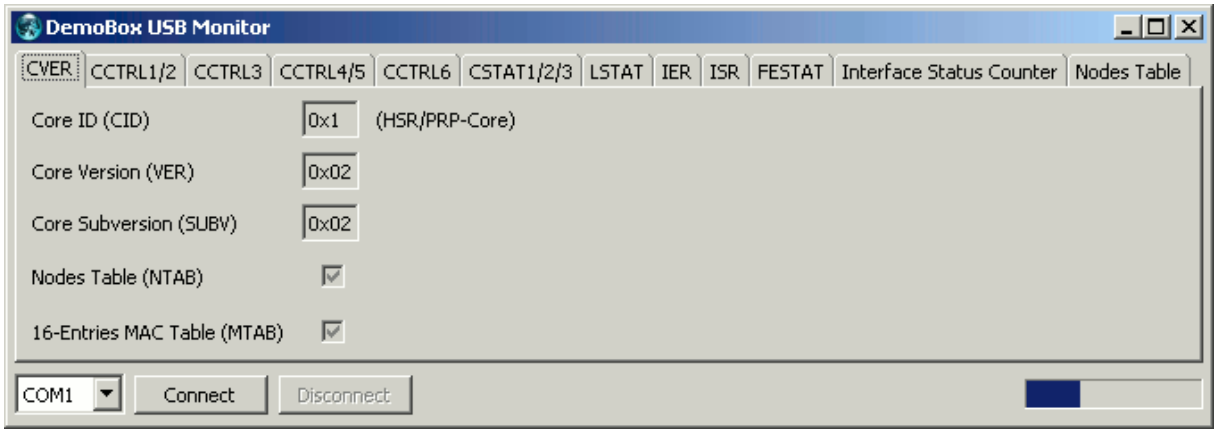

Figure 7: Core Version Register notebook

3.2 Core Control Register notebooks

The Core Control Registers (CCTRLx) contain fields which directly influence the behavior of the core (Figure 8). The single bit fields are mapped to check buttons. So if the user unchecks the Tag button, the new register value is written into register set and the HSR/PRP-Core stops tagging of the outgoing frames. Some register fields like Status Counter Snapshot (SCSS) are self-clearing. In such a case the check button is automatically unchecked if the user checks it.

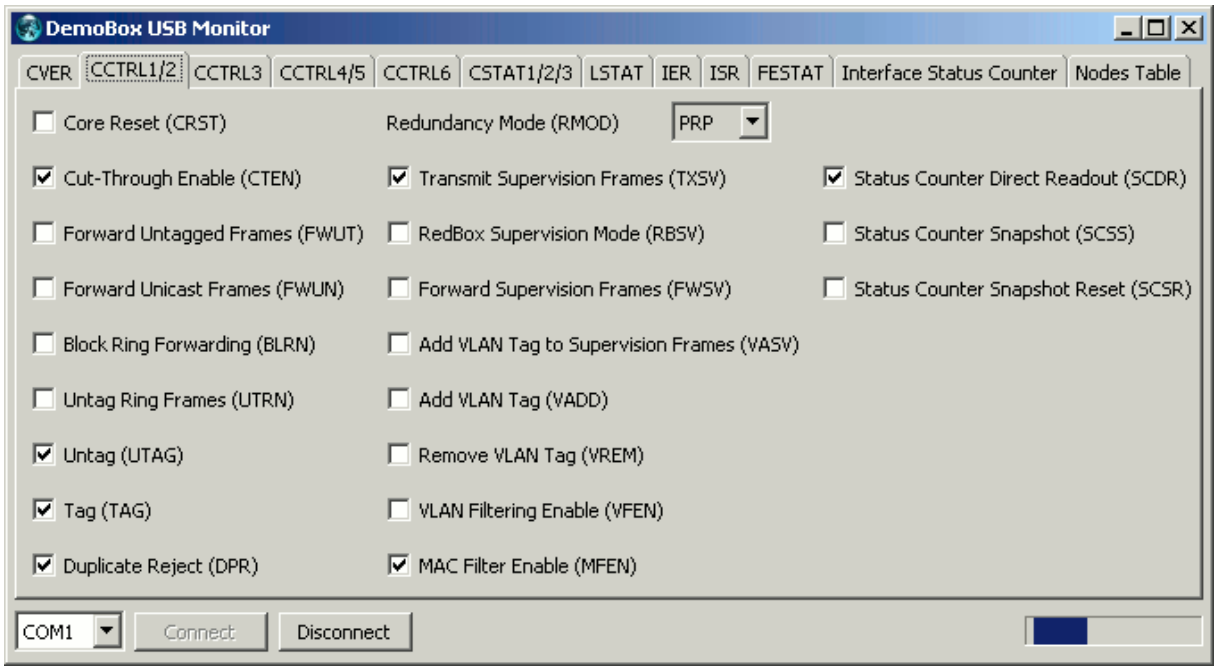

Figure 8: Core Control Register #1/2 notebook

The Redundancy Mode (RMOD) setting respective Core Reset (CRST) field require special user attention. Once one of the fields is changed, the USB connection to the DemoBox is disconnected due to reset of the core. After the reset user shall re-connect the USB connection by pressing the “Connect” button. If the connecting fails, user should try to unplug and plug the USB cable and try to re-connect again.

The multi-bit register fields are represented by the text entry fields with hexadecimal values inside. Some multi-bit fields are narrower than a multiple of 4-bit nibble needed for correct hexadecimal representation. In such a case, the unused top bits should be zeroed. If the user tries to write a larger value to a register field, than can be represented by this field (e.g. 0x90 to the 6-bit MAC Forget Time field), an error message appears (Figure 9) hence the USB Monitor checks plausibility of the entered values.

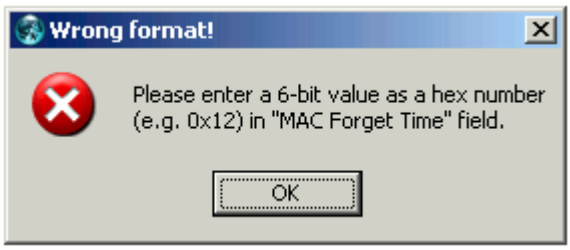

Figure 9: Wrong format error message

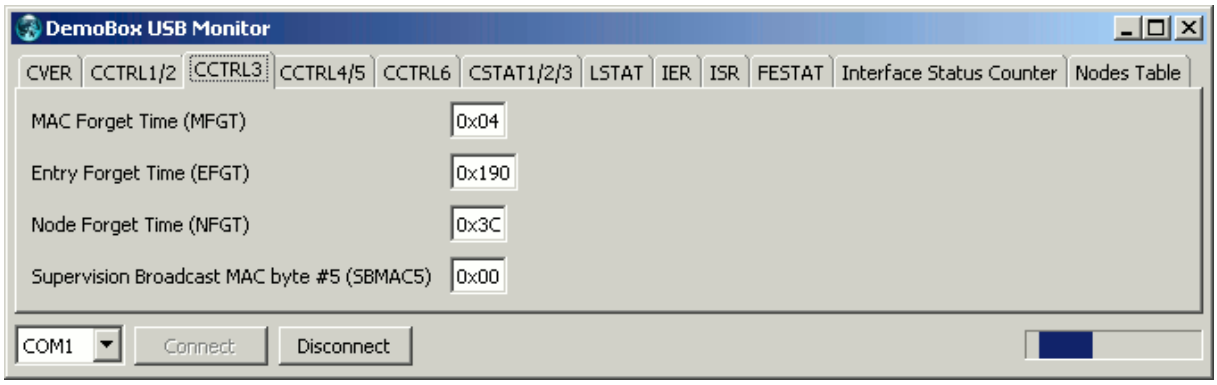

Figure 10: Core Control Register #2 notebook

The RedBox MAC Address fields in the CCTRL4 and CCTRL5 registers contain the MAC address (Figure 11) which is embedded into the Supervision frames if the core is used in RedBox mode. The four upper bytes are stored in the CCTRL4 and the both lower bytes are stored in the CCTRL5 register. The concatenated RedBox MAC address according to the Figure 11 is “00:12:34:56:78:90”.

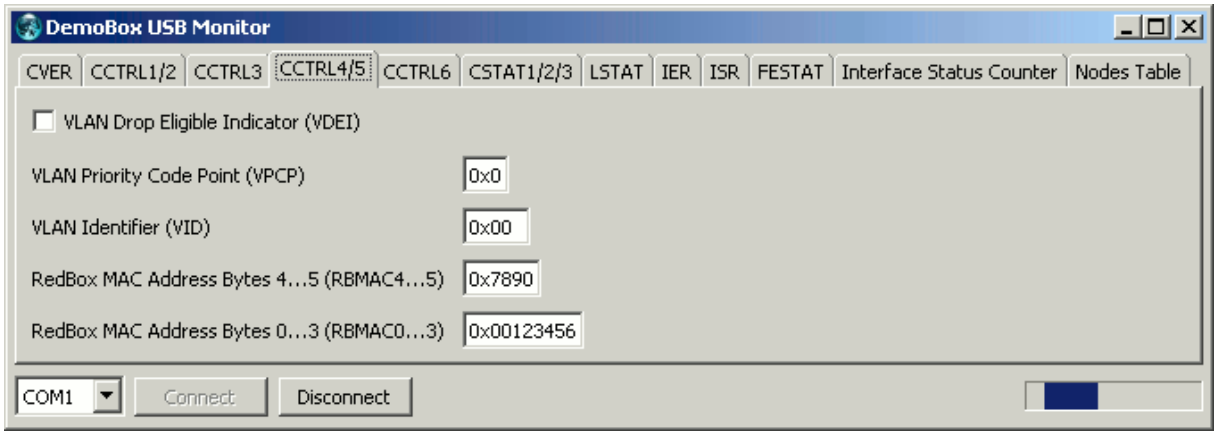

Figure 11: Core Control Registers #4/5 notebook

The CCTRL6 register provides priority control of frames forwarded to the ring relatively to VLAN-tagged frames transmitted from Port C to the ring. Furthermore it defines the Frame Buffer fill level threshold for Clear-To-Send line.

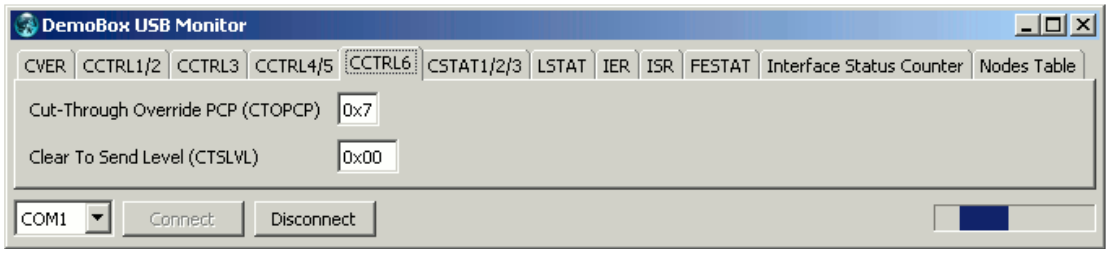

Figure 12: Core Control Register #6 notebook

3.3 Core Status Registers notebook

The Core Status Registers (CSTAT1/2/3) contain information related to the entry usage in the internal tables and Frame Buffers segment allocation (Figure 13). The according register fields are read-only and thus can't be changed.

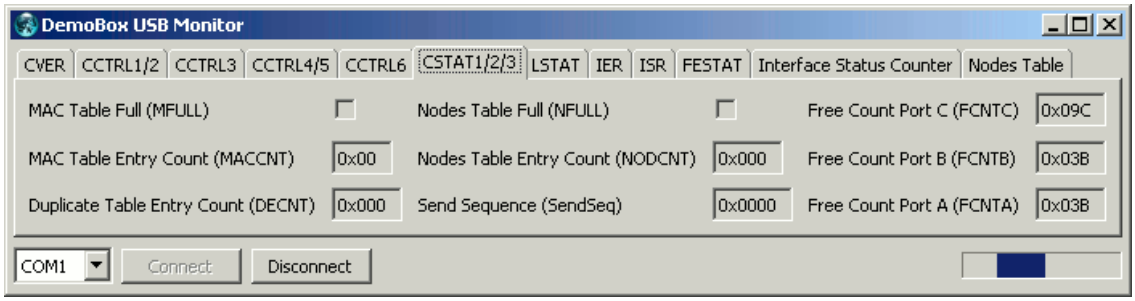

Figure 13: Core Status Registers #1/2/3 notebook

3.4 Link Status Register notebook

The Link Status Register (LSTAT) represents the link status of the Ethernet connection which is periodically retrieved from the PHYs via the SMI bus (Figure 14). If the PHY addresses are changed, then the link status information is not correct anymore, hence only the preset values are valid for the DemoBox hardware. If the Release SMI (RELS) check button is set, then the PRP-Core releases the control over the SMI bus to an external SMI controller (doesn't exist at the DemoBox hardware). The released SMI bus is then indicated by the SMI Idle (SIDLE) bit set.

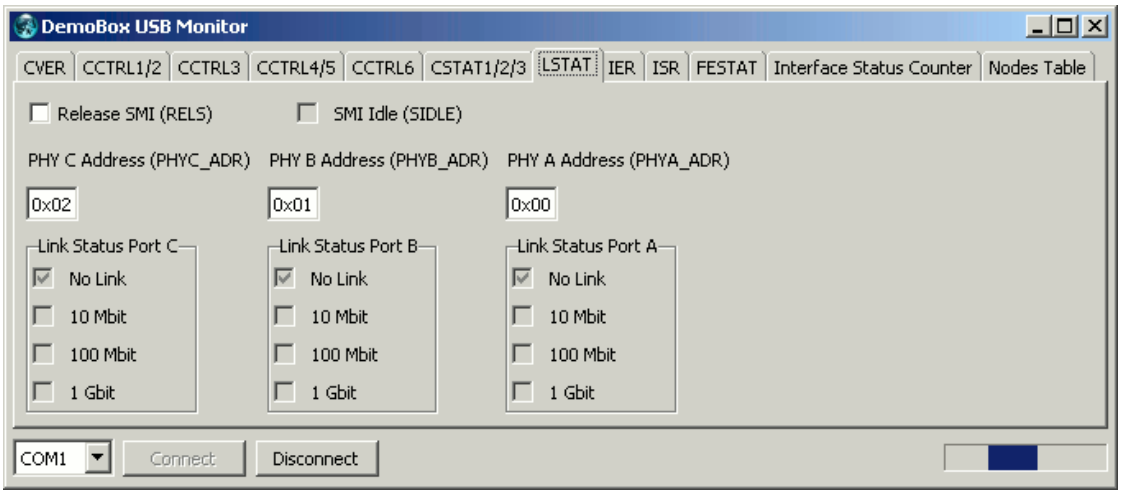

Figure 14: Link Status Register notebook

3.5 Interrupt Enable and Status Registers notebooks

The Interrupt Enable Register (IER) is used to enable the interrupt sources (Figure 15). The USB Monitor supports the processing of the active interrupts, which can be easily shown by the example of the Link Status Interrupt. To enable the Link Interrupt, the LSIE bit should be set like shown in the figure below. Then a notification window appears any time one of the Ethernet cables is plugged to or unplugged from the DemoBox (Figure 16).

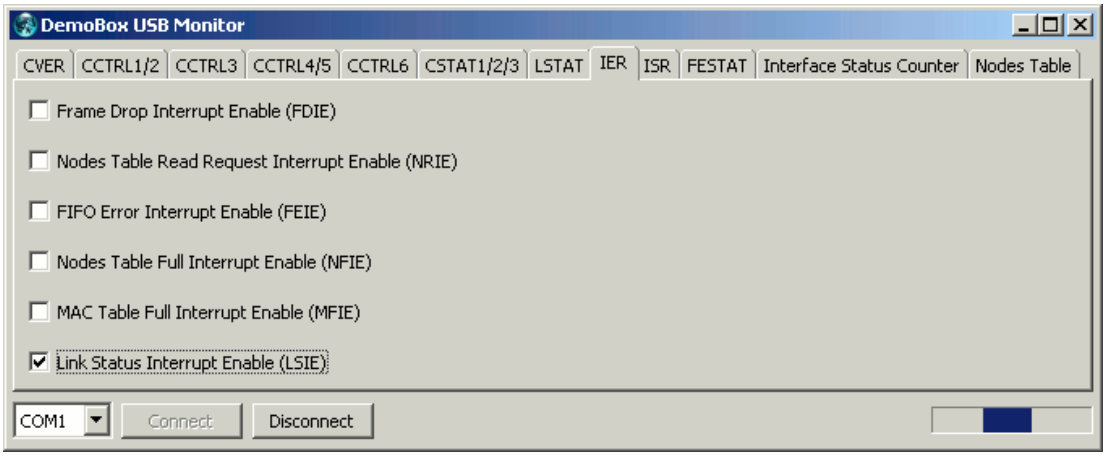

Figure 15: Interrupt Enable Register notebook

Figure 16: Interrupt notification message

The last interrupt source(s) can then be identified in the Interrupt Status Register (ISR) notebook (Figure 17).

Note: In contrary to the register set behavior, the values in the ISR notebook are only updated when a new interrupt occurs. Thus, the last interrupt source is permanently visible. The register set of the IP cores behaves differently: the ISR bits are cleared with every read access to the ISR register.

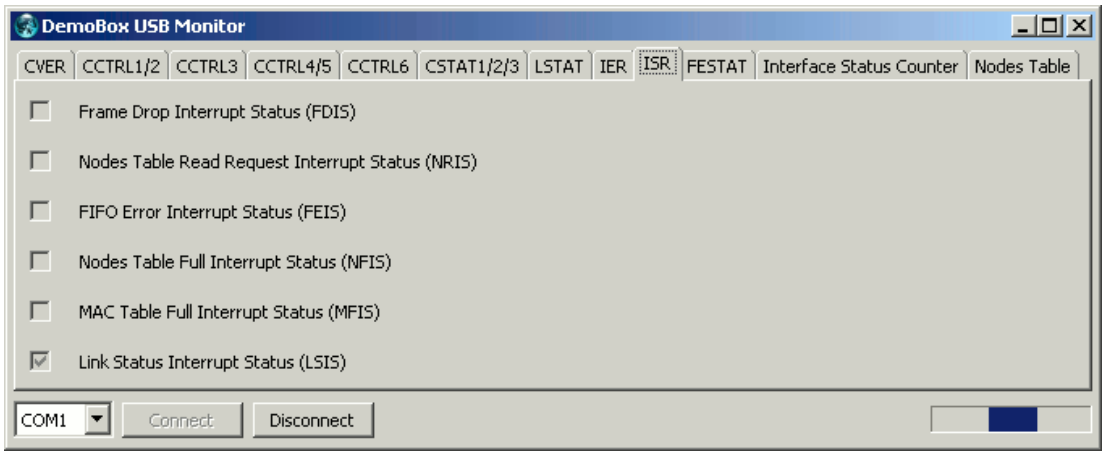

Figure 17: Interrupt Status Register notebook

3.6 FIFO Error Status Register notebook

The FIFO Error Status Register (FESTAT) notebook provides information on erroneous FIFO over- and underflow states. Such non-recoverable FIFO states are avoided in the IP cores by design, thus the user should never experience a non-zero value in this register. However if a non-zero value is read from the FESTAT register, then the PRP-Core has experienced a non-recoverable malfunction and should be reset.

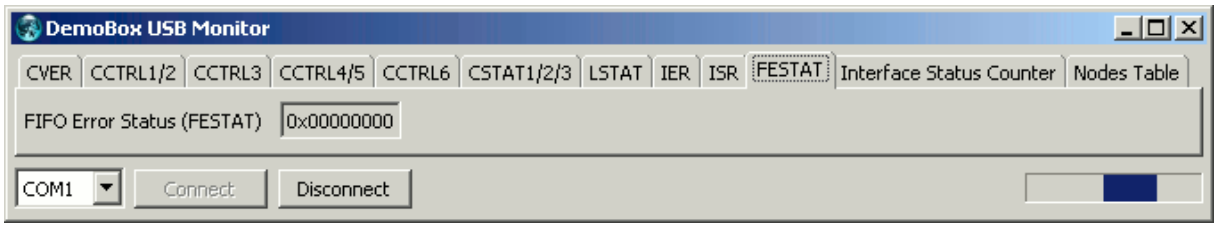

Figure 18: Frame Error Status Register notebook

3.7 Interface Status Counters notebook

The Interface Status Counters notebook shows the values of various status counters (Figure 19, Figure 20). They provide port-related frame and error counts, Frame Buffer-related frame drop counts and other helpful information. The values of the counters can be either directly read (SCDR bit set) or a snapshot of the counter values can be done to the snapshot registers by setting the SCSS bit (see CCTRL2 register).

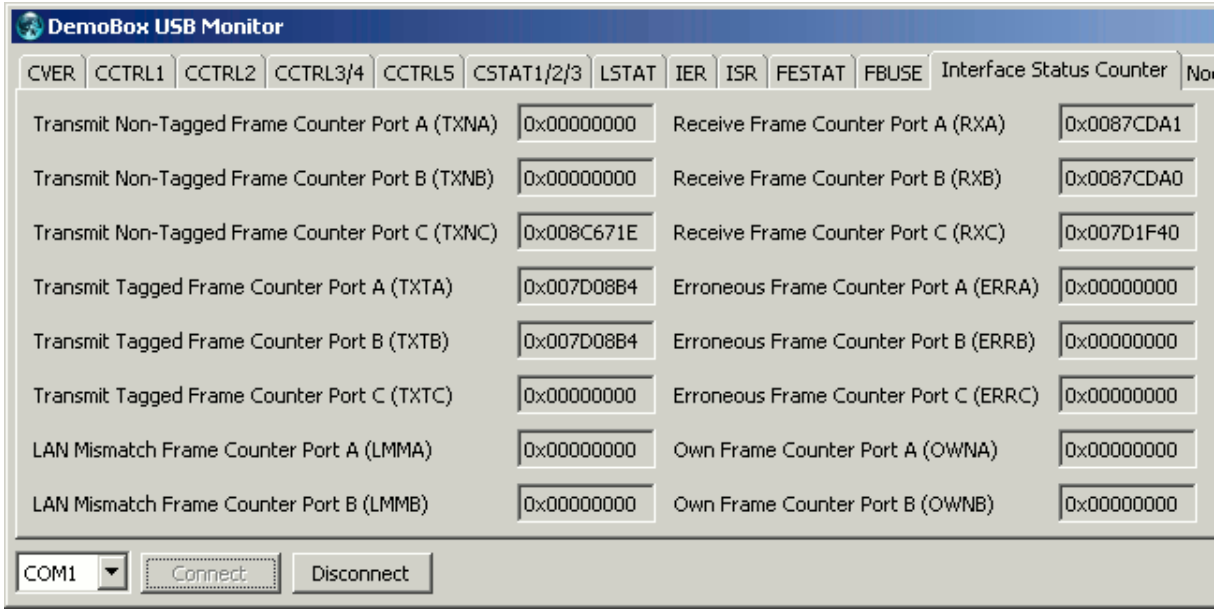

Figure 19: Frame Drop Register notebook (part I)

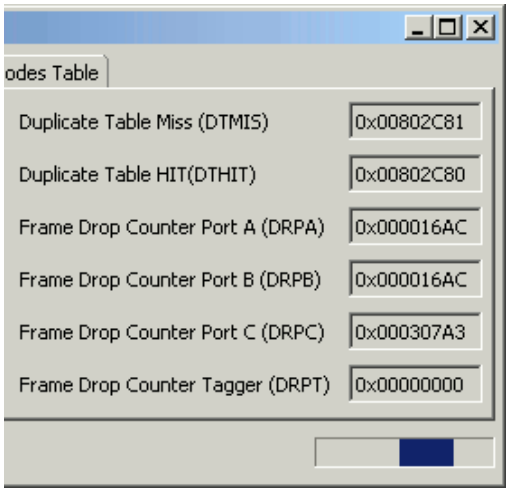

Figure 20: Frame Drop Register notebook (part II)

3.8 Nodes Table notebook

The Nodes Table notebook makes the content of the Nodes Table visible (Figure 21). Each non-empty row in the notebook represents a valid node entry. The readout of the Nodes Table is managed by the RegSetToUSB component in the FPGA, therefore Nodes Table Read Request (NREQ, Figure 10) switch is not available. Furthermore, the user shouldn't set the Nodes Table Read Request Interrupt Enable (NRIE, Figure 15) to avoid permanent interrupt messages.

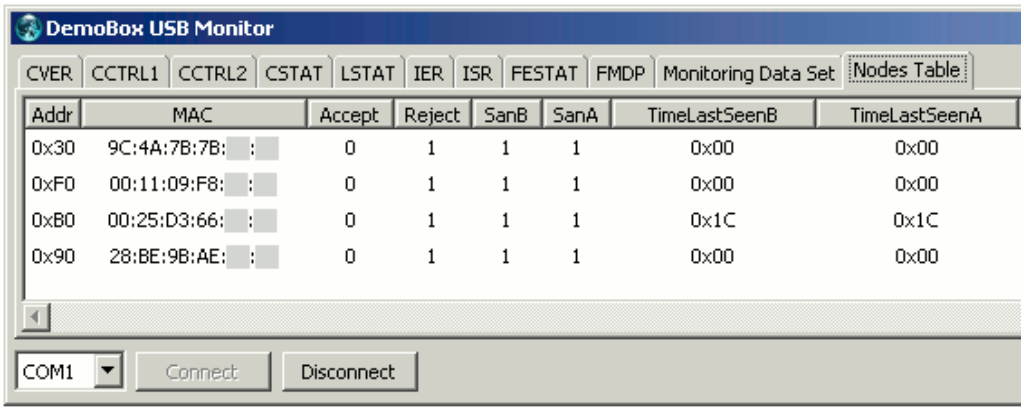

Figure 21: Nodes Table notebook (part I)

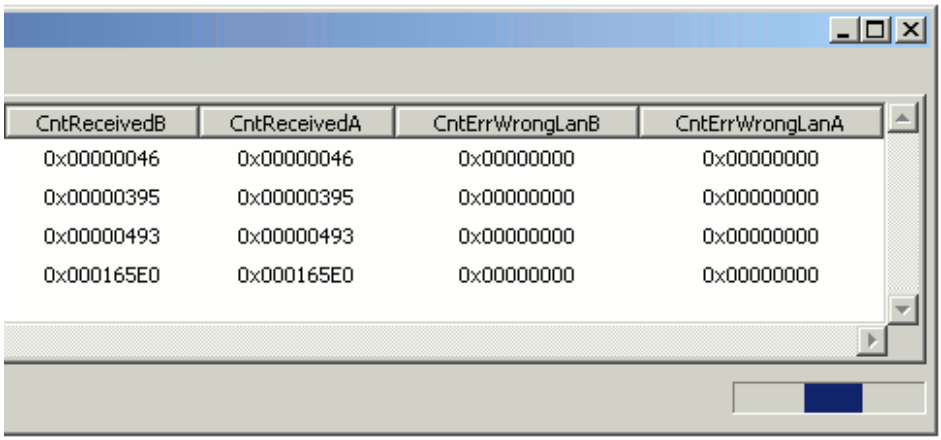

Figure 22: Nodes Table notebook (part II)

4 Appendix

4.1 Document history

Version	Date	Author	Comment
1.03	2015-07-20	VM	USB Monitor screen shots and overall description updated for the HSR/PRP-Core v2.02.
1.02	2015-02-16	VM	USB Monitor screen shots and overall description updated for the HSR/PRP-Core v1.04.
1.01	2014-08-02	VM	Copyright notice update
1.00	2014-05-17	VM	Initial version