

# SIS3316 16 Channel VME Digitizer

## Ethernet UDP Addendum

SIS GmbH  
Harksheider Str. 102A  
22399 Hamburg  
Germany

Phone: ++49 (0) 40 60 87 305 0  
Fax: ++49 (0) 40 60 87 305 20

email: [info@struck.de](mailto:info@struck.de)  
<http://www.struck.de>

Version: SIS3316-M-0101-1-V106-ethernet\_addendum.doc as of  
12.08.2015

## Revision Table:

Revision	Date	Modification
1.02	22.01.2013	First official release
1.03	10.04.2013	release related to Firmware Version - VME FPGA: V3316-2003 (0x33162003)  modification: - <b>fully</b> licensed XILINX Soft Tri-Mode Ethernet MAC core, no more time restriction
1.04	20.03.2015	release related to Firmware Version - VME FPGA: V3316-2008 (0x33162008)  modification: - DHCP support - response on Ping-request - Jumbo Frame support - add "Read Last Packet again" command (Retry) - change Ethernet UDP protocol (add packet identifier byte) <b>Note:</b> software has to be adapted (see:sis3316_ethernet_access_class)
1.05	27.03.2015	release related to Firmware Version - VME FPGA: V3316-2008 (0x33162008)  modification: - upgrade Software documentation
1.06	12.08.2015	release related to Firmware Version - VME FPGA: V3316-200A (0x3316200A)  modification: - support for optical Ethernet

## Table of contents

1	SIS3316 Ethernet interface .....	4
1.1	Hardware prerequisite .....	4
2	Computer UDP-Socket connection .....	5
2.1	Static ARP entry example (manually IP assignment) .....	6
2.1.1	On a Windows computer .....	6
2.1.2	On a Linux/Mac-OS computer .....	7
2.2	DHCP (automatically IP assignment) .....	8
2.3	Optical Ethernet support .....	9
3	FPGA Ethernet UDP protocol implementation .....	10
3.1	VME/CTRL FPGA Link interface register space access protocol .....	12
3.1.1	Access protocol with VME FPGA versions V3316-2007 and lower .....	12
3.1.2	Access protocol with VME FPGA versions V3316-2008 and higher .....	12
3.2	SIS3316 (VME/CTRL and ADC FPGA) register space access protocol .....	13
3.2.1	Access protocol with VME FPGA versions V3316-2007 and lower .....	13
3.2.2	Access protocol with VME FPGA versions V3316-2008 and higher .....	14
3.3	SIS3316 memory (FIFO) space access protocol .....	15
3.3.1	Access protocol with VME FPGA versions V3316-2007 and lower .....	16
3.3.2	Access protocol with VME FPGA versions V3316-2008 and higher .....	17
4	SIS3316 Addressing .....	18
4.1	Address Map Overview .....	18
4.1.1	VME/CTRL FPGA Link interface register set .....	19
4.1.2	SIS3316 register / memory space .....	19
5	VME/CTRL FPGA Link interface Register Description .....	20
5.1	Control/Status Register(0x0, write/read) .....	20
5.2	Module Id. and Firmware Revision Register .....	22
5.2.1	Major revision numbers .....	22
5.3	UDP protocol configuration register (0x8) .....	23
5.4	Last UDP Acknowledge Status Register .....	24
5.5	Link Interface Access Arbitration Control Register .....	24
5.6	Error counters .....	25
5.7	Ethernet Speed test counter .....	26
5.8	Hardware Version Register .....	26
6	Software .....	27
6.1	Build the program "sis3316_eth_access_rate_test" .....	27
6.1.1	Windows 7 (Windows 8) .....	27
6.1.2	Linux (Scientific 6.6) .....	31
6.1.3	MAC OS (10.9.4) .....	33
6.2	Executable programs .....	34
6.2.1	sis3316_eth_access_rate_test .....	34
6.2.2	sis3316_eth_fpga_update .....	34
6.2.3	sis3316_set_dhcp_mode .....	34
7	Index .....	35

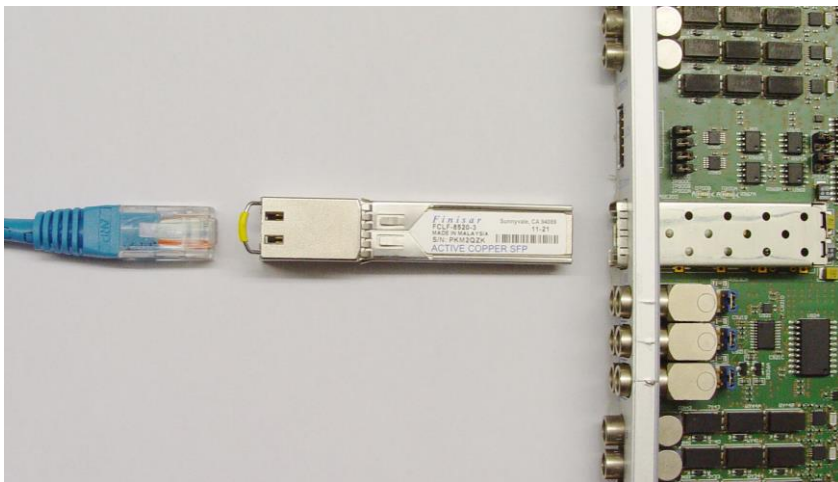
## 1 SIS3316 Ethernet interface

The SIS3316 provides several interface options (Ethernet, Optical and –upon request- VXS) besides the VME interface to access (control, write and read) on board resource. This manual describes the Ethernet interface.

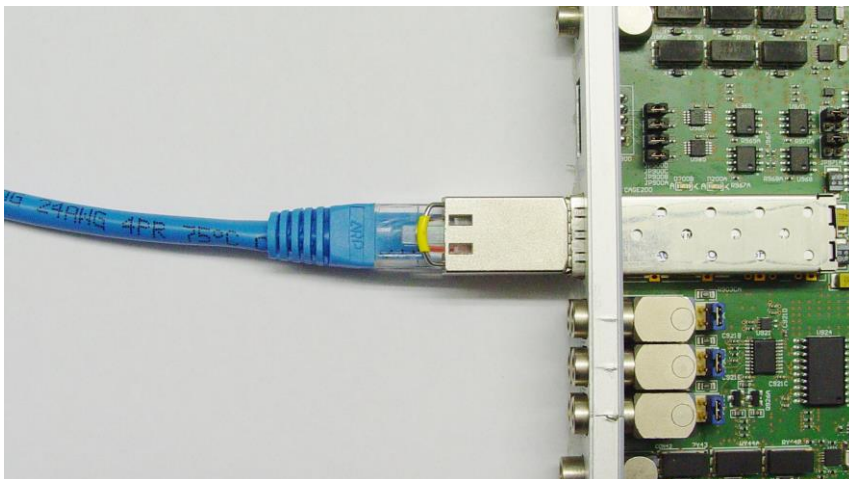


SIS3316-DT

### 1.1 Hardware prerequisite



A 1000BASE-T Copper SFP has to be installed into the SFP cage of the SIS3316.



## 2 Computer UDP-Socket connection

Each device (computer) on a network has an **IP address**, but the device's network interfaces have **MAC addresses**. The UDP socket library (Linux, MAC-OS and Windows) needs **IP addresses**, also, (own computer IP address and Server IP address) to establish a communication link between the computer and the SIS3316.

Therefore the SIS3316 MAC Address has to be assigned to an IP address on the network. This can be done manually with a **static ARP entry** on the computer, switch or router or automatically with a **DHCP (Dynamic Host Configuration Protocol)** request.

The MAC address of the SIS3316 is: **00:00:56:31:6x:xx** (xxx: hex. serial number)

The DHCP request mode can be enabled/disabled by the Switch SW80-4 and/or by settings of the DHCP option value in the One-Wire Eeprom (address offset 2) on the SIS3316.

SW80-4: refer to the SIS3316 User manual.

DHCP option value: refer to the project/program `sis3316_set_dhcp_mode`.

DHCP enable/disable table:

SW80-4	DHCP option value bit 1 (force disable)	DHCP option value bit 0 (force enable)	DHCP
off	0	0	disabled
off	0	1	enabled
off	1	0	disabled
off	1	1	disabled
on	0	0	enabled
on	0	1	enabled
on	1	0	disabled
on	1	1	enabled

Note: The LED 2 indicates after Power-Up and booting the FPGAs (LEDs blinking 1 sec. in the order of A, 2, U and 1) the Ethernet Link Status.

- blinking with 4 Hz (every 0.25 sec) indicates no Ethernet Link
- blinking with 0.8 Hz (every 1.25 sec) indicates DHCP request busy
- blinking with **ping** access

## 2.1 Static ARP entry example (manually IP assignment)

SIS3316 serial number: 25 (0x19):  
Desired SIS3316 IP address: 212.60.16.200 (SIS subnet)

**Note:** the used IP address must be locked on the DHCP-server !

### 2.1.1 On a Windows computer

To add a static ARP entry for a local UDP node with an IP address of **212.60.16.200** that resolves to a MAC address of **00:00:56:31:60:19**, type the following at an administrator command prompt:

```
>arp -s 212.60.16.200 00-00-56-31-60-19
```

This command will bind the default LAN-interface.

```
>arp -s 212.60.16.200 00-00-56-31-60-19 -N 212.60.16.24
```

This command will bind the own (further) LAN-interface with the IP-address 212.60.16.24.

The following command shows all ARP-entries:

```
>arp -a
```

Helpful utility commands: ipconfig and ping

Ping is a computer network administration software utility used to test the reachability of a host (SIS3316) on an Internet Protocol network.

```
>ping 212.60.16.200
```

The SIS3316 will answer and the LED U2 will flash up.

### 2.1.2 On a Linux/Mac-OS computer

To add a static ARP entry for a local UDP node with an IP address of **212.60.16.200** that resolves to a MAC address of **00:00:56:31:60:19**, type the following at an administrator command prompt:

```
#arp -i eth0 -s 212.60.16.200 00:00:56:31:60:19
```

This command will bind the default LAN-interface eth0.

The following command shows all ARP-entries:

```
$arp -a
```

Helpful utility commands: ipconfig and ping

Ping is a computer network administration software utility used to test the reachability of a host (SIS3316) on an Internet Protocol network.

```
$ping 212.60.16.200
```

The SIS3316 will answer and the LED U2 will flash up.

You will find shell-scripts like “**configure\_udp\_localNet\_eth0\_SerNo121**” or “**configure\_udp\_SisNet\_eth0\_SerNo121**” in the `./software` directory. Adapt one of these scripts and execute it as administrator:

```
#!/configure_udp_localNet_eth0_SerNo121
```

Example: “configure\_udp\_localNet\_eth0\_SerNo121”

```
#
ifconfig eth0 down
ifconfig eth0 192.168.1.1
ifconfig eth0 up
#
arp -i eth0 -s 192.168.1.100 00:00:56:31:60:79
#
#
# Network TCP/UDP tuning to support high-bandwidth applications
#
#sysctl -w net.core.rmem_max=33554432
#
sysctl -w net.core.rmem_max=8388608
sysctl -w net.core.wmem_max=8388608
sysctl -w net.core.rmem_default=65536
sysctl -w net.core.wmem_default=65536
#
sysctl -w net.ipv4.udp_mem='8388608 8388608 8388608'
#
sysctl -w net.ipv4.tcp_rmem='4096 87380 8388608'
sysctl -w net.ipv4.tcp_wmem='4096 65536 8388608'
sysctl -w net.ipv4.tcp_mem='8388608 8388608 8388608'
#
sysctl -w net.ipv4.route.flush=1
```

## 2.2 DHCP (automatically IP assignment)

If the DHCP option is enabled then the SIS3316 will request automatically an IP-address from the DHCP server. The LED 2 is blinking every 1.25 second while the DHCP request process is busy.

If a Name server is enabled it is possible to access the SIS3316 with its name and serial number:  
ping sis3316-nnnn (nnnn: four digits, decimal serial number)

Example with serial number 158:

> ping sis3316-0158

Windows:

```
C:\Users\th>ping sis3316-0158

Ping wird ausgeführt für sis3316-0158.intern [212.60.16.7] mit 32 Bytes Daten:
Antwort von 212.60.16.7: Bytes=32 Zeit<1ms TTL=128
Antwort von 212.60.16.7: Bytes=32 Zeit<1ms TTL=128
Antwort von 212.60.16.7: Bytes=32 Zeit<1ms TTL=128
Antwort von 212.60.16.7: Bytes=32 Zeit<1ms TTL=128

Ping-Statistik für 212.60.16.7:
    Pakete: Gesendet = 4, Empfangen = 4, Verloren = 0
    (0% Verlust),
    Ca. Zeitangaben in Millisek.:
        Minimum = 0ms, Maximum = 0ms, Mittelwert = 0ms

C:\Users\th>arp -a

Schnittstelle: 212.60.16.95 --- 0xb
Internetadresse      Physische Adresse      Typ
212.60.16.7          00-00-56-31-60-9e      dynamisch
212.60.16.98          00-10-00-04-62-00      dynamisch
```

Linux:

```
th@linux-th:~/sis3316_DT/software

Datei Bearbeiten Ansicht Suchen Terminal Hilfe

4 packets transmitted, 4 received, 0% packet loss, time 3000ms
rtt min/avg/max/mdev = 0.249/0.251/0.254/0.015 ms
[th@linux-th software]$ arp -a
sis3316-0158.intern (212.60.16.7) auf 00:00:56:31:60:9e [ether] auf eth0
SIS GmbH.intern (212.60.16.140) auf 00:a0:57:11:7a:27 [ether] auf eth0
[th@linux-th software]$
[th@linux-th software]$
[th@linux-th software]$
[th@linux-th software]$
[th@linux-th software]$
[th@linux-th software]$ ping sis3316-0158 -c 4
PING sis3316-0158.intern (212.60.16.7) 56(84) bytes of data.
64 bytes from sis3316-0158.intern (212.60.16.7): icmp_seq=1 ttl=64 time=0.267 ms
64 bytes from sis3316-0158.intern (212.60.16.7): icmp_seq=2 ttl=64 time=0.253 ms
64 bytes from sis3316-0158.intern (212.60.16.7): icmp_seq=3 ttl=64 time=0.250 ms
64 bytes from sis3316-0158.intern (212.60.16.7): icmp_seq=4 ttl=64 time=0.247 ms

--- sis3316-0158.intern ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3001ms
rtt min/avg/max/mdev = 0.247/0.254/0.267/0.013 ms
[th@linux-th software]$ arp -a
sis3316-0158.intern (212.60.16.7) auf 00:00:56:31:60:9e [ether] auf eth0
SIS GmbH.intern (212.60.16.140) auf 00:a0:57:11:7a:27 [ether] auf eth0
[th@linux-th software]$
```



## 2.3 Optical Ethernet support

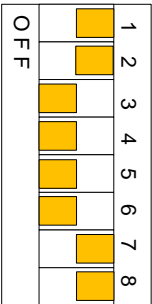
Optical decoupling of data acquisition devices from the readout system is mandatory in some applications (like installation on a platform under high voltage). In case of the SIS3316(-DT) 125MSPS 16-bit and 250 MSPS 14-bit digitizers optical decoupling can be established with readout over an optical Ethernet connection.

Starting with VME FPGA firmware revision V3316-200A Switch **SW80-3** is used to activate/deactivate auto-negotiation for the Ethernet link. The switch has to be in on position for an optical connection and in off position for a copper connection or an optical connection over a media converter (like TP-LINK part number MC220L).

The tests were performed with a DELOCK PCI Express 1x SFP Slot Gigabit LAN card (DELOCK part number 89368), two SFP link media (TP-LINK part number TL-SM311LM or Finisar FTLF8524P2BNV, Struck part number 03145) and a standard LC-LC multimode duplex50/125  $\mu\text{m}$  fiber.

## 2.4 SW80

SW80 is an 8 position dual inline switch. The functions are listed in the table below.

SW80	Off Function	On Function
	<b>Disable A32 slave addressing</b>	<b>Enable A32 slave addressing</b>
	<b>Analog power off</b>	<b>Analog power on</b>
	<b>Deactivate auto-negotiation</b>	<b>Activate auto-negotiation</b>
	<b>DHCP disable</b>	<b>DHCP enable</b>
	<b>Not used</b>	<b>Not used</b>
	<b>Not used</b>	<b>Not used</b>
	<b>Watchdog disable</b>	<b>Watchdog enable</b>
	<b>Disconnect VME SYSRESET from FPGA reset</b>	<b>Connect VME SYSRESET to FPGA reset</b>

### 3 FPGA Ethernet UDP protocol implementation

The XILINX Soft Tri-Mode Ethernet MAC core is used in the VME/CTRL FPGA (Spartan 6) for the Ethernet interface implementation.

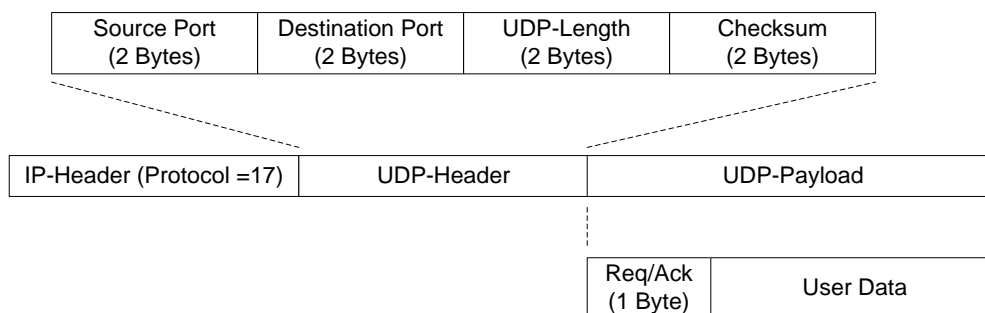
With VME FPGA Version V3316-2003 (0x33162003) and higher the fully licensed core without run time restriction is used.

#### Note 1: the Ethernet interface supports 1Gbit/sec, only!

The Ethernet interface of the SIS3316 (server, acknowledge) is based on the UDP protocol to communicate with a PC (client, request).

The PC (client) sends a “request command” (Req) and the SIS3316 (server) answers with an “acknowledge” (Ack) in case of the Requests 0x10, 0x20, 0x21, 0x30 and 0x31.

Ethernet protocol:



The data of the UDP-Payload packet is used for the communication with the SIS3316.

Eight UDP Request commands (Req) are implemented as listed with its corresponding methods in the table below:

Req	Command	Methods in sis3316_ethernet_access_class.cpp and .h
0x10	Read VME/CTRL FPGA Link interface register space	<code>int udp_register_read(..)</code>
0x11	Write VME/CTRL FPGA Link interface register space	<code>int udp_register_write(..)</code>
0x20	Read SIS3316 register space	<code>int udp_sis3316_register_read(..)</code>
0x21	Write SIS3316 register space	<code>int udp_sis3316_register_write(..)</code>
0x30	Read SIS3316 memory (FIFO) space	<code>int udp_sis3316_fifo_read(..)</code>
0x31	Write SIS3316 memory (FIFO) space	<code>int udp_sis3316_fifo_write(..)</code>
0xEE	Read Last Packet again	<code>int udp_retransmit_cmd(..)</code>
0xFF	Reset command	<code>int udp_reset_cmd(void)</code>

The Requests 0x10, 0x11, 0x20, 0x21, 0x30 and 0x31 enable the complete control/handling of the SIS3316.

**Note 2: the Ethernet UDP protocol has changed with VME FPGA Version V3316-2008 and it is not compatible with previous versions !**

Software has to be adapted (see:sis3316\_ethernet\_access\_class.cpp and .h).

The “#define VME\_FPGA\_VERSION\_IS\_0008\_OR\_HIGHER” has to be set as comment (set false) in case of VME FPGA version V\_3316-2007 and lower:

```
//#define VME_FPGA_VERSION_IS_0008_OR_HIGHER
```

The “#define VME\_FPGA\_VERSION\_IS\_0008\_OR\_HIGHER” has to be set true in case of VME FPGA version V\_3316-2008 and higher:

```
#define VME_FPGA_VERSION_IS_0008_OR_HIGHER
```

Two features are implemented with version V3316-2008 to recover from the occurrence of “lost packets”:

- add packet identifier byte in UDP protocol
- add “Read Last Packet again” command (Retry)

Lost packets are standard with UDP transfers and depend on Operating System, hardware (PC Ethernet interface, router hubs, ...) and network load. So far, we observed “lost packets” with Window PCs in networks, only and only received packets are lost. Normally, a “Read Last Packet again” command will recover the “lost packet” behaviour. The retry received packet identifier equates to the requested packet identifier.

So far, we never observed “lost packets” with Linux and MAC PCs and with a point to point connection.

The Requests 0x10, 0x11, 0x20, 0x21, 0x30 and 0x31 can be emulated by “VME calls”. This enables the use of the same application software for VME interfaces and UDP interface with the help of a virtual vme\_interface\_class (see vme\_interface\_class.h).

These “VME calls” are part of sis3316\_ethernet\_access\_class.cpp and .h, too.

```
public:
    int vme_A32D32_read (UINT addr, UINT* data);
    int vme_A32D32_write (UINT addr, UINT data);

    int vme_A32_2EVMEFIFO_read (UINT addr, UINT* data, UINT request_nof_words,
                                UINT* got_nof_words );

    ..
    ..
```

### 3.1 VME/CTRL FPGA Link interface register space access protocol

The following methods (sis3316\_ethernet\_access\_class.cpp and sis3316\_ethernet\_access\_class.h) correspond to these access protocols:

```
public:
    int udp_register_read (UINT addr, UINT* data);
    int udp_register_write (UINT addr, UINT data);
```

The “#define VME\_FPGA\_VERSION\_IS\_0008\_OR\_HIGHER” has to be set as comment (set false) in case of VME FPGA version V\_3316-2007 and lower:

```
//#define VME_FPGA_VERSION_IS_0008_OR_HIGHER
```

The “#define VME\_FPGA\_VERSION\_IS\_0008\_OR\_HIGHER” has to be set true in case of VME FPGA version V\_3316-2008 and higher:

```
#define VME_FPGA_VERSION_IS_0008_OR_HIGHER
```

#### 3.1.1 Access protocol with VME FPGA versions V3316-2007 and lower

Register Read Request command (transmit to SIS3316)

Req = 0x10	Address (4 Bytes)
---------------	----------------------

Register Read Ack (receive from SIS3316)

Ack = 0x10	Address (4 Bytes)	Data (4 Bytes)
---------------	----------------------	-------------------

Register Write Request command (transmit to SIS3316)

Req = 0x11	Address (4 Bytes)	Data (4 Bytes)
---------------	----------------------	-------------------

#### 3.1.2 Access protocol with VME FPGA versions V3316-2008 and higher

An additional byte “Packet Identifier” is added. The SIS3316 acknowledges with the requested Packet Identifier.

Register Read Request command (transmit to SIS3316)

Req = 0x10	Packet Identifier	Address (4 Bytes)
---------------	----------------------	----------------------

Register Read Ack (receive from SIS3316)

Ack = 0x10	Packet Identifier	Address (4 Bytes)	Data (4 Bytes)
---------------	----------------------	----------------------	-------------------

Register Write Request command (transmit to SIS3316)

Req = 0x11	Address (4 Bytes)	Data (4 Bytes)
---------------	----------------------	-------------------

### 3.2 SIS3316 (VME/CTRL and ADC FPGA) register space access protocol

The following methods (sis3316\_ethernet\_access\_class.cpp and sis3316\_ethernet\_access\_class.h) correspond to these access protocols:

```
public:
    int udp_sis3316_register_read ( unsigned int nof_read_registers,
                                    UINT* addr_ptr, UINT* data_ptr);
    int udp_sis3316_register_write ( unsigned int nof_read_registers,
                                    UINT* addr_ptr, UINT* data_ptr);
```

The “#define VME\_FPGA\_VERSION\_IS\_0008\_OR\_HIGHER” has to be set as comment (set false) in case of VME FPGA version V\_3316-2007 and lower:

```
//#define VME_FPGA_VERSION_IS_0008_OR_HIGHER
```

The “#define VME\_FPGA\_VERSION\_IS\_0008\_OR\_HIGHER” has to be set true in case of VME FPGA version V\_3316-2008 and higher:

```
#define VME_FPGA_VERSION_IS_0008_OR_HIGHER
```

#### 3.2.1 Access protocol with VME FPGA versions V3316-2007 and lower

SIS3316 Register Space Read Request command (transmit to SIS3316)

- Read Request of N register values (max. 64 )

Req = 0x20	N-1 (2 Bytes)	Address 0 (4 Bytes)	Address 1 (4 Bytes)	..	Address N-1 (4 Bytes)
---------------	------------------	------------------------	------------------------	----	--------------------------

SIS3316 Register Space Read Ack (receive from SIS3316)

Ack = 0x20	Status (1 Byte)	Data 0 (4 Bytes)	Data 1 (4 Bytes)	..	Data N-1 (4 Bytes)
---------------	--------------------	---------------------	---------------------	----	-----------------------

SIS3316 Register Space Write Request command (transmit to SIS3316)

- Write Request of N register values (max. 64 )

Req = 0x21	N-1 (2 Bytes)	Address 0 (4 Bytes)	Data 0 (4 Bytes)	Address 1 (4 Bytes)	Data 1 (4 Bytes)	..	Address N-1 (4 Bytes)	Data N-1 (4 Bytes)
---------------	------------------	------------------------	---------------------	------------------------	---------------------	----	--------------------------	-----------------------

SIS3316 Register Space Write Ack (receive from SIS3316)

Ack = 0x21	Status (1 Byte)
---------------	--------------------

#### Status

bit 7: 1-bit Req. Counter (toggles with each Req)  
 bit 6: 1 if Protocol error (Request command packet error )  
 bit 5: 1 if sis3316 access timeout (Fifo Empty)  
 bit 4: 1 if Ethernet interface has no Grant

bits 3-0: 0x0

### 3.2.2 Access protocol with VME FPGA versions V3316-2008 and higher

An additional byte “Packet Identifier” is added. The SIS3316 acknowledges with the requested Packet Identifier.

#### SIS3316 Register Space Read Request command (transmit to SIS3316)

- Read Request of N register values (max. 64 )

Req = 0x20	Packet Identifier	N-1 (2 Bytes)	Address 0 (4 Bytes)	Address 1 (4 Bytes)	..	Address N-1 (4 Bytes)
---------------	----------------------	------------------	------------------------	------------------------	----	--------------------------

#### SIS3316 Register Space Read Ack (receive from SIS3316)

Ack = 0x20	Packet Identifier	Status (1 Byte)	Data 0 (4 Bytes)	Data 1 (4 Bytes)	..	Data N-1 (4 Bytes)
---------------	----------------------	--------------------	---------------------	---------------------	----	-----------------------

#### SIS3316 Register Space Write Request command (transmit to SIS3316)

- Write Request of N register values (max. 64 )

Req = 0x21	Packet Identifier	N-1 (2 Bytes)	Address 0 (4 Bytes)	Data 0 (4 Bytes)	Address 1 (4 Bytes)	Data 1 (4 Bytes)	..	Address N-1 (4 Bytes)	Data N-1 (4 Bytes)
---------------	----------------------	------------------	------------------------	---------------------	------------------------	---------------------	----	--------------------------	-----------------------

#### SIS3316 Register Space Write Ack (receive from SIS3316)

Ack = 0x21	Packet Identifier	Status (1 Byte)
---------------	----------------------	--------------------

#### Status

- bit 7: 1-bit Req. Counter (toggles with each Req)
- bit 6: 1 if Protocol error (Request command packet error )
- bit 5: 1 if sis3316 access timeout (Fifo Empty)
- bit 4: 1 if Ethernet interface has no Grant
  
- bits 3-0: 0x0

### 3.3 SIS3316 memory (FIFO) space access protocol

The following methods (sis3316\_ethernet\_access\_class.cpp and sis3316\_ethernet\_access\_class.h) correspond to these access protocols:

```
public:
    int udp_sis3316_fifo_read ( unsigned int nof_read_words, UINT addr,
                                UINT* data_ptr, UINT* got_nof_words );
    int udp_sis3316_fifo_write ( unsigned int nof_write_words, UINT addr,
                                UINT* data_ptr, UINT* written_nof_words );
```

The “#define VME\_FPGA\_VERSION\_IS\_0008\_OR\_HIGHER” has to be set as comment (set false) in case of VME FPGA version V\_3316-2007 and lower:

```
//#define VME_FPGA_VERSION_IS_0008_OR_HIGHER
```

The “#define VME\_FPGA\_VERSION\_IS\_0008\_OR\_HIGHER” has to be set true in case of VME FPGA version V\_3316-2008 and higher:

```
#define VME_FPGA_VERSION_IS_0008_OR_HIGHER
```

**To optimize/increase the data transfer rate two transfer parameters are configurable:**

1. enable/disable Jumbo frames (hardware feature)
2. define the number of packets per request (software feature)

#### 1. Enable/disable Jumbo frames (hardware feature)

The default (jumbo frame option is disabled) maximal packet size of a transmitted packet from the SIS3316 to the PC is 1484/1485 bytes (version V3316-2008 and higher one byte more). This packet contains 360 32-bit words (1440 bytes) from the SIS3316 memory.

If the jumbo frame option is enabled the maximal packet size will be increased to 8236/8237 bytes. This packet contains 2048 32-bit words (8192) from the SIS3316 memory.

Methods:

```
int get_UdpSocketJumboFrameStatus(void);
int set_UdpSocketEnableJumboFrame(void);
int set_UdpSocketDisableJumboFrame(void);
```

#### 2. Define the number of packets per request (software feature)

Depend on the “number of packets per request” (max. 32) the software controls the requested length. The SIS3316 can send up to 262.144 bytes in N packets with one request. The N packets will be transmitted to the PC with a programmed gap from 256 ns to 57 us (see UDP protocol configuration register).

The udp\_sis3316\_fifo\_read can resolve a “lost packet” behavior in case of “number of packets per request” = 1. In case of “number of packets per request” > 1 the method read\_DMA\_Channel\_PreviousBankDataBuffer in sis3316\_class.cpp will recover the read of N-packets.

Observation: A firewall like Symantec can run on multiple cores and can change the order of the incoming packets. This will be detected but not reordered, yet.

Methods:

```
int set_UdpSocketReceiveNofPackagesPerRequest(unsigned int nofPacketsPerRequest);
```

SIS3316 read memory data transfer rate table:

nofPacketsPerRequest	Jumbo frame disabled	Jumbo frame enabled
1	~5.5 Mbyte/sec	~15 Mbyte/sec
5	~21 Mbyte/sec	~53 Mbyte/sec
10	~37 Mbyte/sec	~73 Mbyte/sec
20	~63 Mbyte/sec	~90 Mbyte/sec
32	~74 Mbyte/sec	~100 Mbyte/sec

See program/project sis3316\_eth\_access\_rate\_test

### 3.3.1 Access protocol with VME FPGA versions V3316-2007 and lower

SIS3316 Fifo Space Read Request command (transmit to SIS3316)

- Read Request of N 32-bit fifo values (max. 262.144 Bytes)

Req = 0x30	N-1 (2 Bytes)	Address (4 Bytes)
---------------	------------------	----------------------

SIS3316 Fifo Space Read Ack (receive from SIS3316)

Ack = 0x30	Status (1 Byte)	Data 0 (4 Bytes)	Data 1 (4 Bytes)	..	Data X-1 (4 Bytes)	packet 0
Ack = 0x30	Status (1 Byte)	Data X (4 Bytes)	Data X+1 (4 Bytes)	..	Data Y-1 (4 Bytes)	packet 1
<div style="text-align: center;"> <div style="display: inline-block; width: 10px; height: 10px; background-color: black; margin: 0 5px;"></div> <div style="display: inline-block; width: 10px; height: 10px; background-color: black; margin: 0 5px;"></div> <div style="display: inline-block; width: 10px; height: 10px; background-color: black; margin: 0 5px;"></div> </div>						
Ack = 0x30	Status (1 Byte)	Data Z (4 Bytes)	Data Z+1 (4 Bytes)	..	Data N-1 (4 Bytes)	packet n

SIS3316 Fifo Space Write Request command (transmit to SIS3316)

- Write Request of N 32-bit register values (max. 256 )

Req = 0x31	N-1 (2 Bytes)	Address (4 Bytes)	Data 0 (4 Bytes)	Data 1 (4 Bytes)	..	Data N-1 (4 Bytes)
---------------	------------------	----------------------	---------------------	---------------------	----	-----------------------

SIS3316 Fifo Space Write Ack (receive from SIS3316)

Ack = 0x31	Status (1 Byte)
---------------	--------------------

#### Status

- bit 7: 1-bit Req. Counter (toggles with each Req)
- bit 6: 1 if Protocol error (Request command packet error )
- bit 5: 1 if sis3316 access timeout (Fifo Empty)
- bit 4: 1 if Ethernet interface has no Grant
  
- bit 3-0: 4-bt Packet Counter



### 3.3.2 Access protocol with VME FPGA versions V3316-2008 and higher

An additional byte “Packet Identifier” is added. The SIS3316 acknowledges with the requested Packet Identifier.

#### SIS3316 Fifo Space Read Request command (transmit to SIS3316)

- Read Request of N 32-bit fifo values (max. 262.144 Bytes)

Req = 0x30	Packet Identifier	N-1 (2 Bytes)	Address (4 Bytes)
---------------	----------------------	------------------	----------------------

#### SIS3316 Fifo Space Read Ack (receive from SIS3316)

Ack = 0x30	Packet Identifier	Status (1 Byte)	Data 0 (4 Bytes)	Data 1 (4 Bytes)	..	Data X-1 (4 Bytes)	packet 0
Ack = 0x30	Packet Identifier	Status (1 Byte)	Data X (4 Bytes)	Data X+1 (4 Bytes)	..	Data Y-1 (4 Bytes)	packet 1
■ ■							
Ack = 0x30	Packet Identifier	Status (1 Byte)	Data Z (4 Bytes)	Data Z+1 (4 Bytes)	..	Data N-1 (4 Bytes)	packet n

#### SIS3316 Fifo Space Write Request command (transmit to SIS3316)

- Write Request of N 32-bit register values (max. 256 )

Req = 0x31	Packet Identifier	N-1 (2 Bytes)	Address (4 Bytes)	Data 0 (4 Bytes)	Data 1 (4 Bytes)	..	Data N-1 (4 Bytes)
---------------	----------------------	------------------	----------------------	---------------------	---------------------	----	-----------------------

#### SIS3316 Fifo Space Write Ack (receive from SIS3316)

Ack = 0x31	Packet Identifier	Status (1 Byte)
---------------	----------------------	--------------------

#### Status

- bit 7: 1-bit Req. Counter (toggles with each Req)
- bit 6: 1 if Protocol error (Request command packet error )
- bit 5: 1 if sis3316 access timeout (Fifo Empty)
- bit 4: 1 if Ethernet interface has no Grant
  
- bit 3-0: 4-bt Packet Counter

## 4 SIS3316 Addressing

### 4.1 Address Map Overview

The SIS3316 resources and their locations are listed in the tables below.

Offset	Request Command	Function
0x000000 – 0x00001C	0x10/0x11 (R/W)	VME/CTRL FPGA Link interface registers
0x000020 – 0x0000FC	0x20/0x21 (R/W)	VME FPGA registers
0x000400 – 0x00043C	0x21 (W only)	VME FPGA key addresses
0x001000 – 0x001FFC	0x20/0x21 (R/W)	ADC FPGA 1: ch1-ch4 registers
0x002000 – 0x002FFC	0x20/0x21 (R/W)	ADC FPGA 2: ch5-ch8 registers
0x003000 – 0x003FFC	0x20/0x21 (R/W)	ADC FPGA 3: ch9-ch12 registers
0x004000 – 0x004FFC	0x20/0x21 (R/W)	ADC FPGA 4: ch13-ch16 registers
0x100000 – 0x1FFFFC	0x30/0x31 (R/W)	ADC FPGA 1: ch1-ch4 Memory Data FIFO
0x200000 – 0x2FFFFC	0x30/0x31 (R/W)	ADC FPGA 2: ch5-ch8 Memory Data FIFO
0x300000 – 0x3FFFFC	0x30/0x31 (R/W)	ADC FPGA 3: ch9-ch12 Memory Data FIFO
0x400000 – 0x4FFFFC	0x30/0x31 (R/W)	ADC FPGA 4: ch13-ch16 Memory Data FIFO

- Note 1:**
- Read/Write access to the VME/CTRL FPGA Link interface register space is always possible.
  - Read access to the VME FPGA register space is always possible.
  - Write access to the VME FPGA register and “key address” space is only possible if the “Link” interface has grant (see Link Interface Access Arbitration control/status register).
  - Read/Write access to the ADC FPGA register and memory space is only possible if the “Link” interface has grant (see Link Interface Access Arbitration control/status register).

- Note 2:** Write access to a key address (KA) with arbitrary datum invokes the respective action

#### 4.1.1 VME/CTRL FPGA Link interface register set

Offset	Access	Function
0x00000000	W/R	Control/Status Register (J-K register)
0x00000004	R only	Module Id. and Firmware Revision register
0x00000008	R/W	UDP protocol configuration register
0x0000000C	R/W	last UDP Acknowledge Status
0x00000010	R/W	Link Interface Access Arbitration control/status register
0x00000014	R/W	Error Counters
0x00000018	R/W	Ethernet Speed test counter
0x0000001C	R/W	Hardware Version register

#### 4.1.2 SIS3316 register / memory space

Refer to the SIS3316 User manual.

## 5 VME/CTRL FPGA Link interface Register Description

The function of the individual registers is described in detail in this section.

The first line after the subsection header (in Courier font) like:

```
#define SIS3316_CONTROL_STATUS          0x0          /* read/write; D32 */
refers to the SIS3316.h header file
```

### 5.1 Control/Status Register(0x0, write/read)

```
#define SIS3316_CONTROL_STATUS          0x0          /* read/write; D32 */
```

The control register is implemented as a selective J/K register, a specific function is enabled by writing a 1 into the set/enable bit, the function is disabled by writing a 1 into the clear/disable bit (which location is 16-bit higher in the register). An undefined toggle status will result from setting both the enable and disable bits for a specific function at the same time. The same register represents the status register on read access.

Bit	write Function	read Function
31	Clear Reboot FPGAs (*)	Status Switch SW80-4 ON
30	Clear reserved 14 (*)	Status Onewire EEPROM Option value bit 1 (addr=2)
29	Clear reserved 13 (*)	Status Onewire EEPROM Option value bit 0 (addr=2)
28	Clear reserved 12 (*)	Status DHCP enable
27	Clear reserved 11 (*)	0
26	Clear reserved 10 (*)	0
25	Clear reserved 9 (*)	0
24	Clear reserved 8 (*)	0
23	Clear reserved 7 (*)	0
22	Clear Led 2 Application Mode (*)	0
21	Clear Led 1 Application Mode (*)	0
20	Clear Led U Application Mode (*)	0
19	Clear Led-Application Mode (*)	0
18	Switch off LED 2(*)	0
17	Switch off LED 1(*)	0
16	Switch off LED U (*)	0
15	Set Reboot FPGAs (**)	Status Reboot FPGAs
14	Set reserved 14	Status reserved 14
13	Set reserved 13	Status reserved 13
12	Set reserved 12	Status reserved 12
11	Set reserved 11	Status reserved 11
10	Set reserved 10	Status reserved 10
9	Set reserved 9	Status reserved 9
8	Set reserved 8	Status reserved 8
7	Set Led 2 Switch On/Off Mode	Status Led 2 Switch On/Off Mode
6	Set Led 2 Application Mode	Status Led 2 Application Mode
5	Set Led 1 Application Mode	Status Led 1 Application Mode
4	Set Led U Application Mode	Status Led U Application Mode
3		
2	Switch on LED 2 (***)	Status LED 2
1	Switch on LED 1 (***)	Status LED 1
0	Switch on LED U (***)	Status LED U (1=LED on, 0=LED off)

(\*) denotes power up default setting

(\*\*) provided that the switch SW80-7 is on (Watchdog enable) and that its own interface grant bit is set

(\*\*\*) provided that its own interface grant bit is set and the Led Application Mode is cleared

Reboot FPGAs = 1 will reboot all FPGAs provided that the switch SW80-7 is on (Watchdog enable) and that its own interface grant bit is set.

The setting of the LEDs U, 1 and 2 are defined by the Control/Status register of the granted interface (VME, Ethernet or Optical interface).

If the Link interface (Ethernet, Optical) has the permission, then this interfaces control/ status register defines the setting of the LEDs.

The LEDs reflect the setting of the the VME control/status register if the Link interface has not the permission or if the VME interface has the permission..

LED U Application Mode	LED U is On if
0	Status LED U = 1
1	Sample logic is sampling (write event into memory)

LED 1 Application Mode	LED 1 is On if
0	Status LED 1 = 1
1	Sample logic bankx is enabled

LED 2 Application Mode	LED 2 Switch On/Off Mode	LED 2 is On if
0	0	Depends on Ethernet Link Status: - Blinking with 4 Hz (every 0.25 sec) -> no Ethernet Link - Blinking with 0.8 Hz (every 1.25 sec) -> DHCP request busy
0	1	Status LED 2 = 1
1	x	Sample logic bank 2 active

## 5.2 Module Id. and Firmware Revision Register

```
#define SIS3316_MODID 0x4 /* read only; D32 */
```

This register reflects the module identification of the SIS3316 and its minor and major firmware revision levels. The major revision level will be used to distinguish between substantial design differences and experiment specific designs, while the minor revision level will be used to mark user specific adaptations.

Bit	Function	Reading
31	Module Id. Bit 15	3
30	Module Id. Bit 14	
29	Module Id. Bit 13	
28	Module Id. Bit 12	
27	Module Id. Bit 11	3
26	Module Id. Bit 10	
25	Module Id. Bit 9	
24	Module Id. Bit 8	
23	Module Id. Bit 7	1
22	Module Id. Bit 6	
21	Module Id. Bit 5	
20	Module Id. Bit 4	
19	Module Id. Bit 3	6
18	Module Id. Bit 2	
17	Module Id. Bit 1	
16	Module Id. Bit 0	
15	Major Revision Bit 7	
14	Major Revision Bit 6	
13	Major Revision Bit 5	
12	Major Revision Bit 4	
11	Major Revision Bit 3	
10	Major Revision Bit 2	
9	Major Revision Bit 1	
8	Major Revision Bit 0	
7	Minor Revision Bit 7	
6	Minor Revision Bit 6	
5	Minor Revision Bit 5	
4	Minor Revision Bit 4	
3	Minor Revision Bit 3	
2	Minor Revision Bit 2	
1	Minor Revision Bit 1	
0	Minor Revision Bit 0	

### 5.2.1 Major revision numbers

Find below a table with major revision numbers used to date

Major revision number	Application/user
0x20	n/Gamma

### 5.3 UDP protocol configuration register (0x8)

```
#define SIS3316_UDP_PROTOCOL_CONFIG      0x8          /* read/write; D32 */
```

This register is used to control the UDP data packets.

Bit	31 - 9	8	7 - 5	4	3 - 0
Function	reserved	UDP transmit Data packet format bit	reserved	UDP transmit jumbo Packet enable bit	UDP transmit packet gap bits

UDP transmit packet gap value	Gap time between UDP transmit packets
0	256 ns
1	512 ns
2	1 us
3	2 us
4	4 us
5	8 us
6	10 us
7	12 us
8	14 us
9	16 us
0xA	20 us
0xB	28 us
0xC	32 us
0xD	41 us
0xE	50 us
0xF	57 us

The power up default value reads 0x 00000000

#### 5.4 Last UDP Acknowledge Status Register

```
#define SIS3316_LAST_UDP_ACKN_STATUS 0x0C /* rd only ; D32 */
```

This register stores and shows the Ack- and Status-byte of the last and last but one transmitted UDP packet.

Bits	31-24	23-16	15-8	7-0
Function	last Ack	last Status	last but one Status	last but one Status

#### 5.5 Link Interface Access Arbitration Control Register

```
#define SIS3316_INTERFACE_ACCESS_ARBITRATION_CONTROL 0x10 /* r/w; D32 */
```

The Interface Access Arbitration Control register controls the permission to access to the internal SIS3316 space.

If no interface side has set the request bit, then VME has the permission to access the internal SIS3316 space.

Bit	write Function	read Function
31	Kill of other interface request bit command	0
30		0
29		0
28		0
27		0
26		0
25		0
24		0
23		0
22		0
21		Status of other interface grant bit
20		Status of own interface grant bit
19		0
18		0
17		Status of other interface request bit
16		Status of own interface request bit
15		0
..		0
..		0
0	Own interface request bit	Status of own interface request bit

```
// kill request and grant from vme interface (in case of use using etehrnet interface)
gl_virtual_vme_crate->vme_A32D32_write(module_base_addr +
                                         SIS3316_INTERFACE_ACCESS_ARBITRATION_CONTROL, 0x80000000);

// arbitrate
gl_virtual_vme_crate->vme_A32D32_write(module_base_addr +
                                         SIS3316_INTERFACE_ACCESS_ARBITRATION_CONTROL, 1);
```



## 5.6 Error counters

```
#define SIS3316_ETHERNET_ERROR_COUNTERS 0x14 /* rd only, D32 */
```

These counters count special detected errors and will be cleared with a UDP-Reset command.

Bits	meaning
31 - 28	Statistic Tx error counter
27 - 24	Statistic Rx error counter
23 - 16	GMII Badframe error counter
15 - 8	GMII Tx error counter
7 - 0	GMII Rx error counter

### 5.7 Ethernet Speed test counter

```
#define SIS3316_ETHERNET_SPEED_TEST_COUNTER    0x18    /* rd only, D32 */
```

This 32-bit read only counter increments every 8ns (125MHz).

Bit	31-0
Function	32-bit Ethernet Speed test counter

### 5.8 Hardware Version Register

```
#define SIS3316_HARDWARE_VERSION    0x1C    /* read only; D32 */
```

This register reflects the PCB/hardware version of the SIS3316.

Bit	Function
31	0
..	..
..	..
4	0
3	Hardware Version Bit 3
2	Hardware Version Bit 2
1	Hardware Version Bit 1
0	Hardware Version Bit 0

The register value is mapped to the PCB revision as shown below.

Hardware Version Register value	PCB revision
1	V1
2	V2/V3

## 6 Software

This chapter will guide you through the build process of the software project “sis3316\_eth\_access\_rate\_test”, which is helpful to demonstrate the use of the software and libraries and to test the “Ethernet performance”.

All project files and libraries are available in source code.

Executable files are available, also.

The software project “sis3316\_eth\_access\_rate\_test” supports the three platforms:

- Windows (WIN7) built with the development tool VisualC++ 2010 (VC10)
- Scientific Linux 6.5 built with the development tool Eclipse (Helios, 3.6.1)
- Mac OS 10.9.2 built with the development tool Eclipse (Kepler Service Release 2)

The OS is selected with “#define(s)” in the file project\_system\_define.h.

An example application “sis3316\_root\_gui” is described in the manual “SIS3316-M-Application-Software-Root-Gui-V100.pdf”

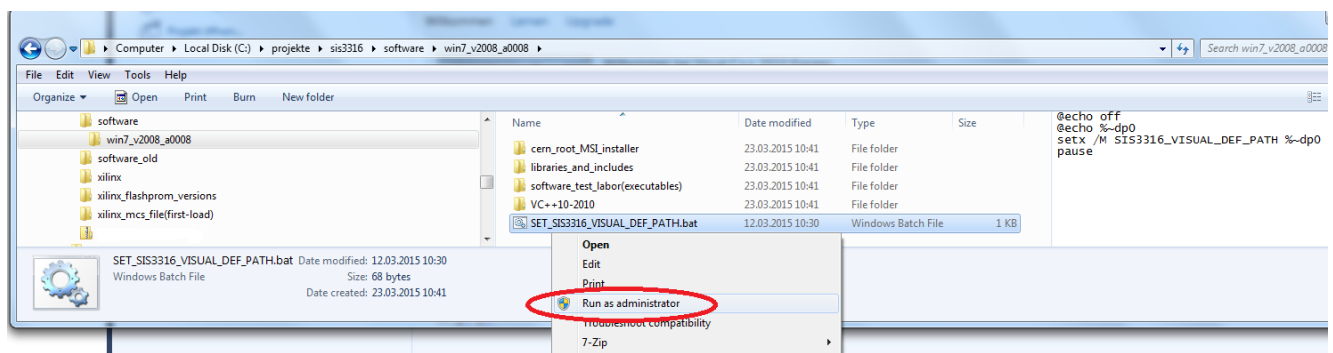
### 6.1 Build the program “sis3316\_eth\_access\_rate\_test”

The development tools **VisualC++ 2010** and **Eclipse 3.6.1** are used to build (compile and link) the executable. Valid settings for Include and Library paths are needed for the development tools.

#### 6.1.1 Windows 7 (Windows 8)

The Development Tools **VisualC++ 2010** is used to build the project.

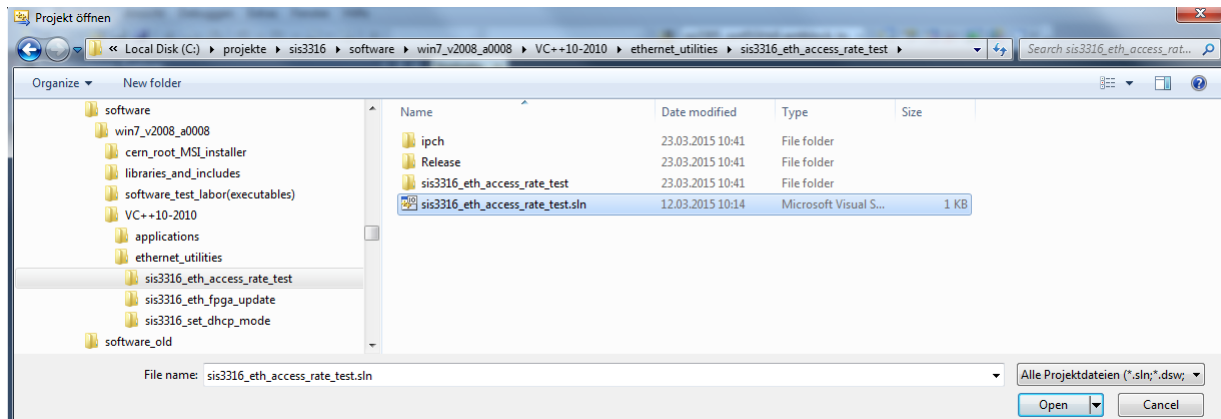
Copy the directory “sis3316/software/win7\_v2008\_a0008” from the DVD to your local disk. Execute the batch file “SET\_SIS3316\_VISUAL\_DEF\_PATH.bat” as administrator. It sets the environment variable SIS3316\_VISUAL\_DEF\_PATH with the path of the project. **VisualC++ 2010** uses this variable for the include/library path settings.



### 6.1.1.1 Open the project “sis3316\_eth\_access\_rate\_test.sln” with VisualC++ 2010.

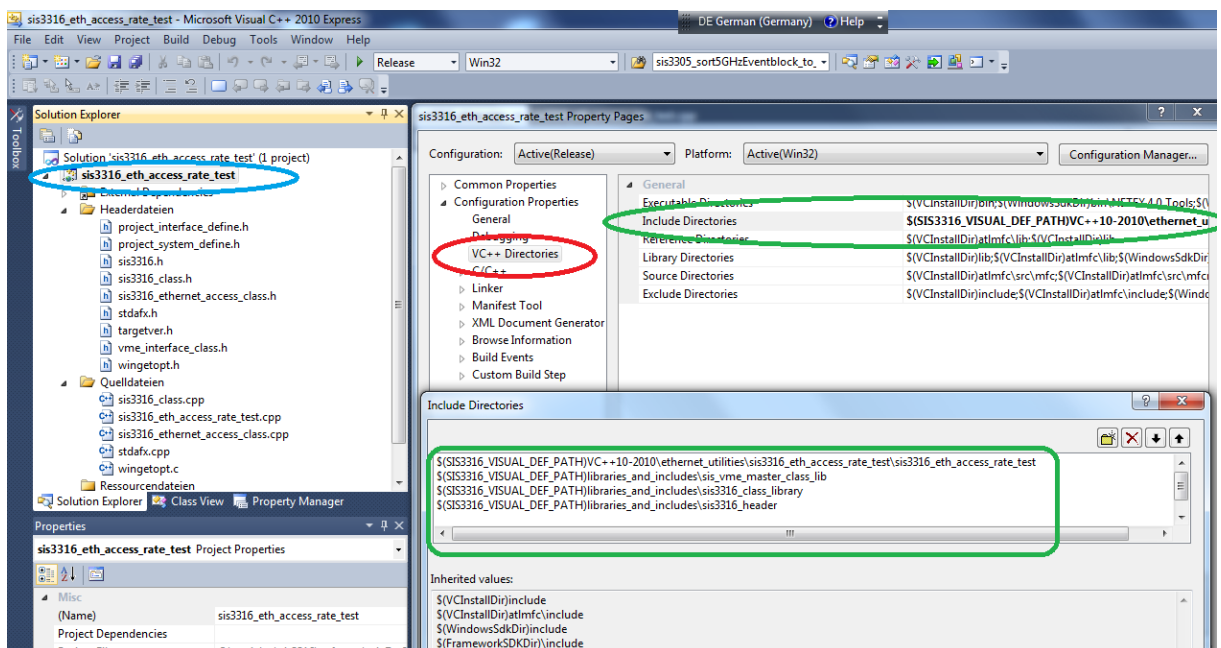
This project contains the C++ files and the property-settings to run with the communicate interface:

- sis3316-DT Ethernet/UDP interface (sis3316\_ethernet\_access\_class.cpp /h)



### 6.1.1.2 VC10 Project Properties

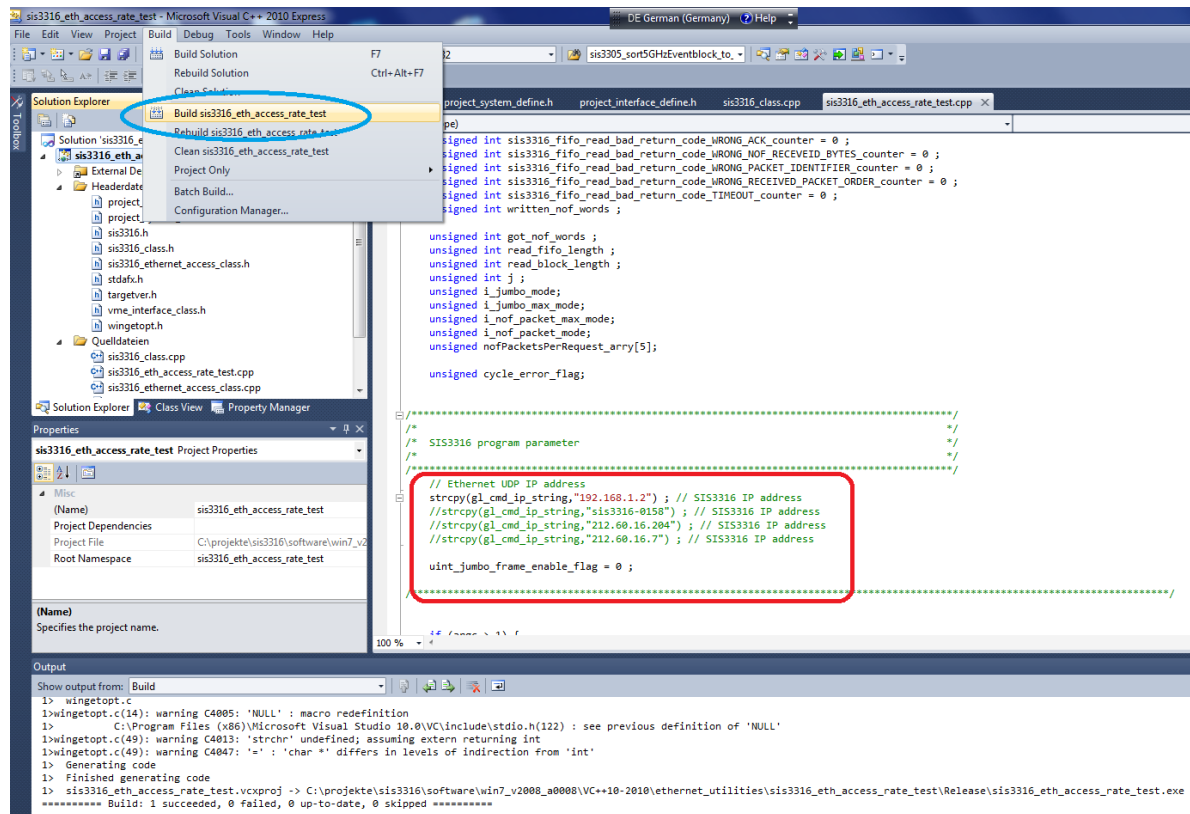
Select the project “sis3316\_eth\_access\_rate\_test” (marked in blue) and make a right click. A window pop up and click on Properties. The property-window pop up.



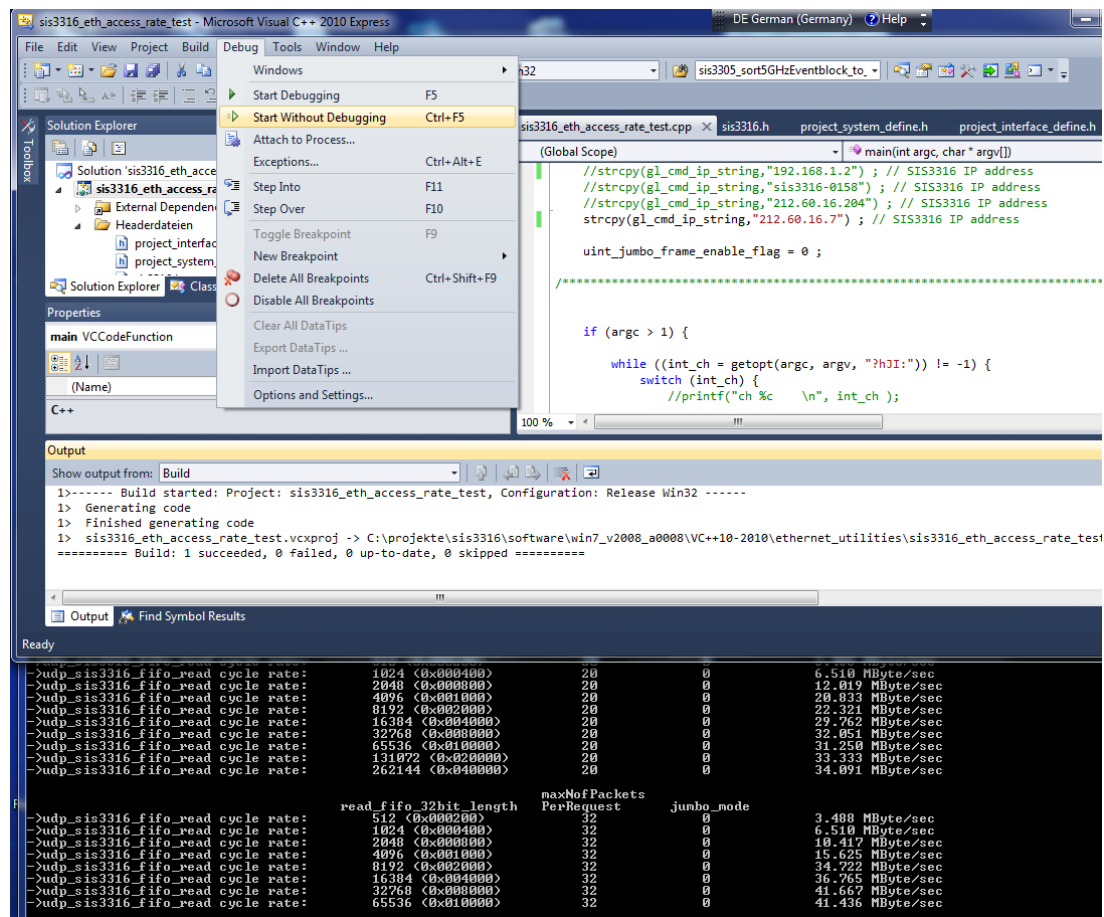
The above project contains the Communication interface files and the project settings of the valid Include path. No Library path setting is necessary.

### 6.1.1.3 Build

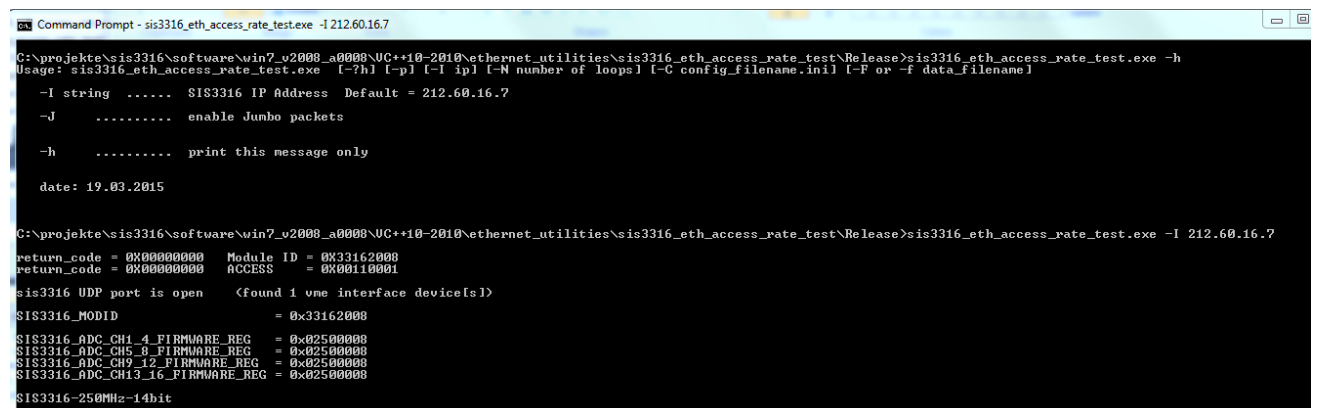
Build the program with default values (red marked).



## 6.1.1.4 Run program from Visual with default values



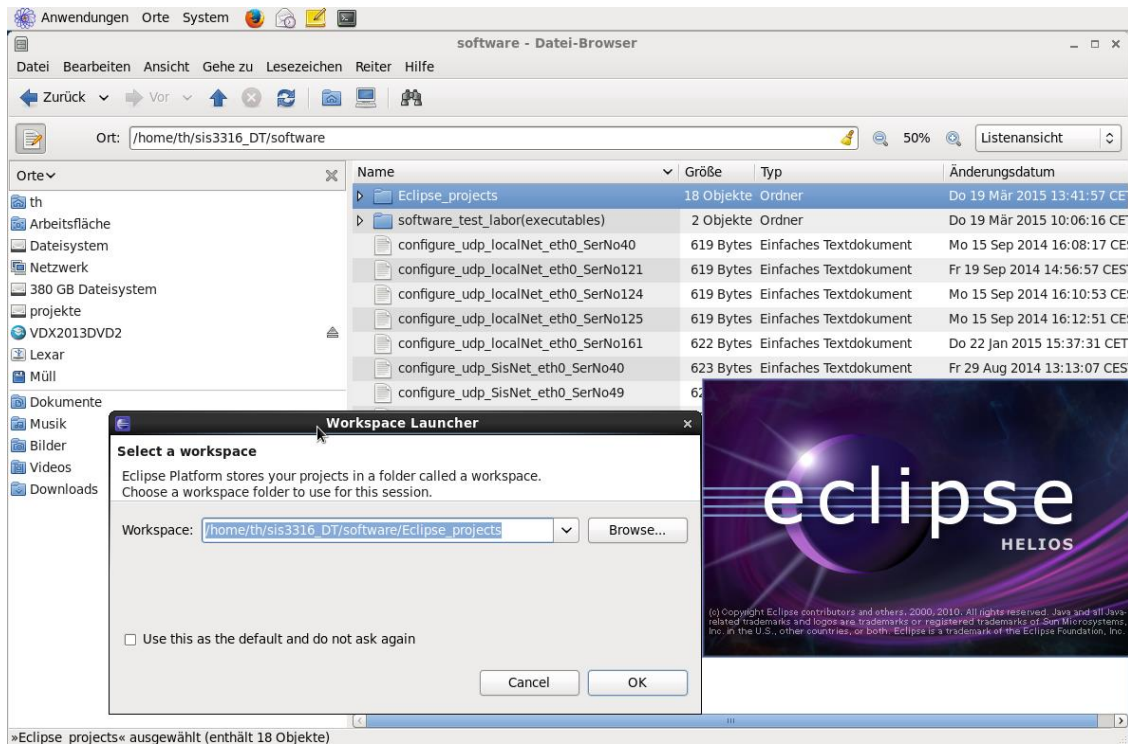
## 6.1.1.5 Run program from DOS box



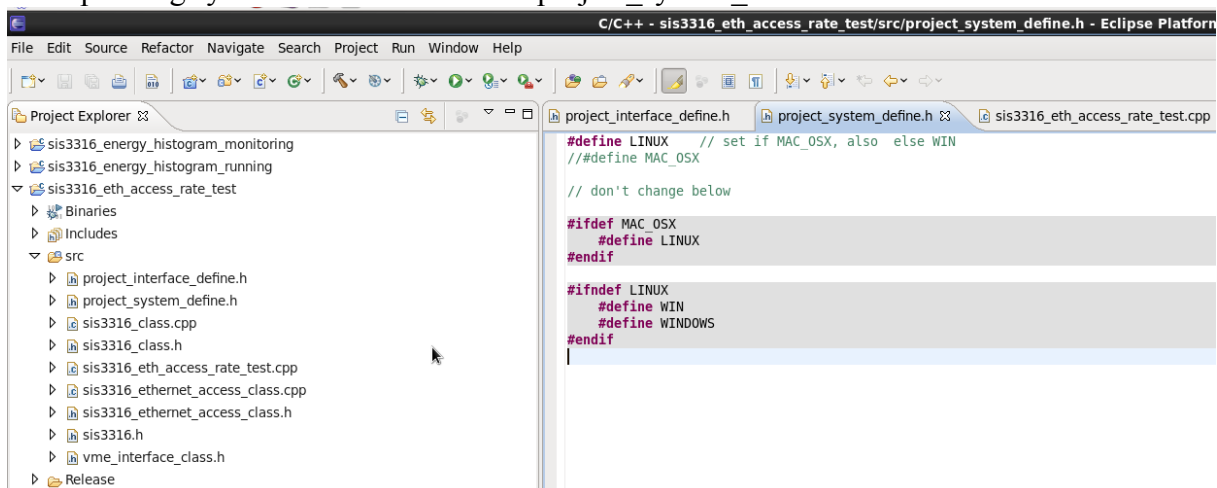
## 6.1.2 Linux (Scientific 6.6)

The development tool Eclipse (Helios, 3.6.1) is used to build the project.

Open the projects with Eclipse:

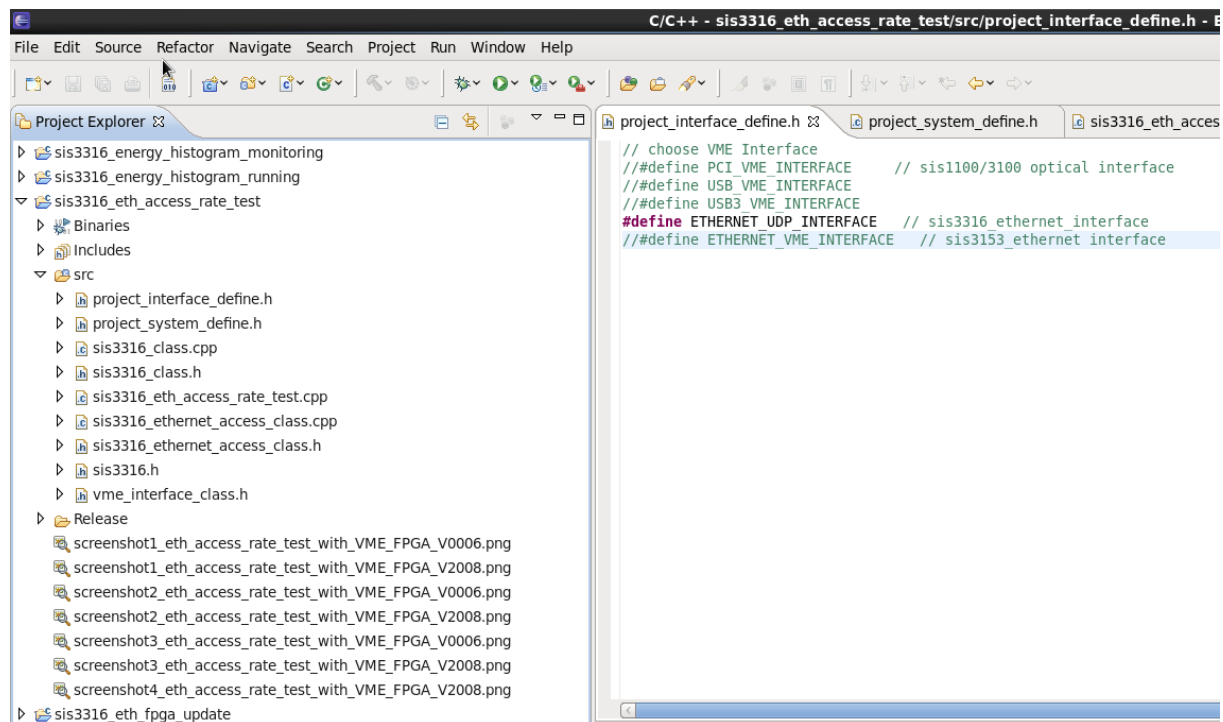


The operating system will be defined in “project\_system\_define.h”:



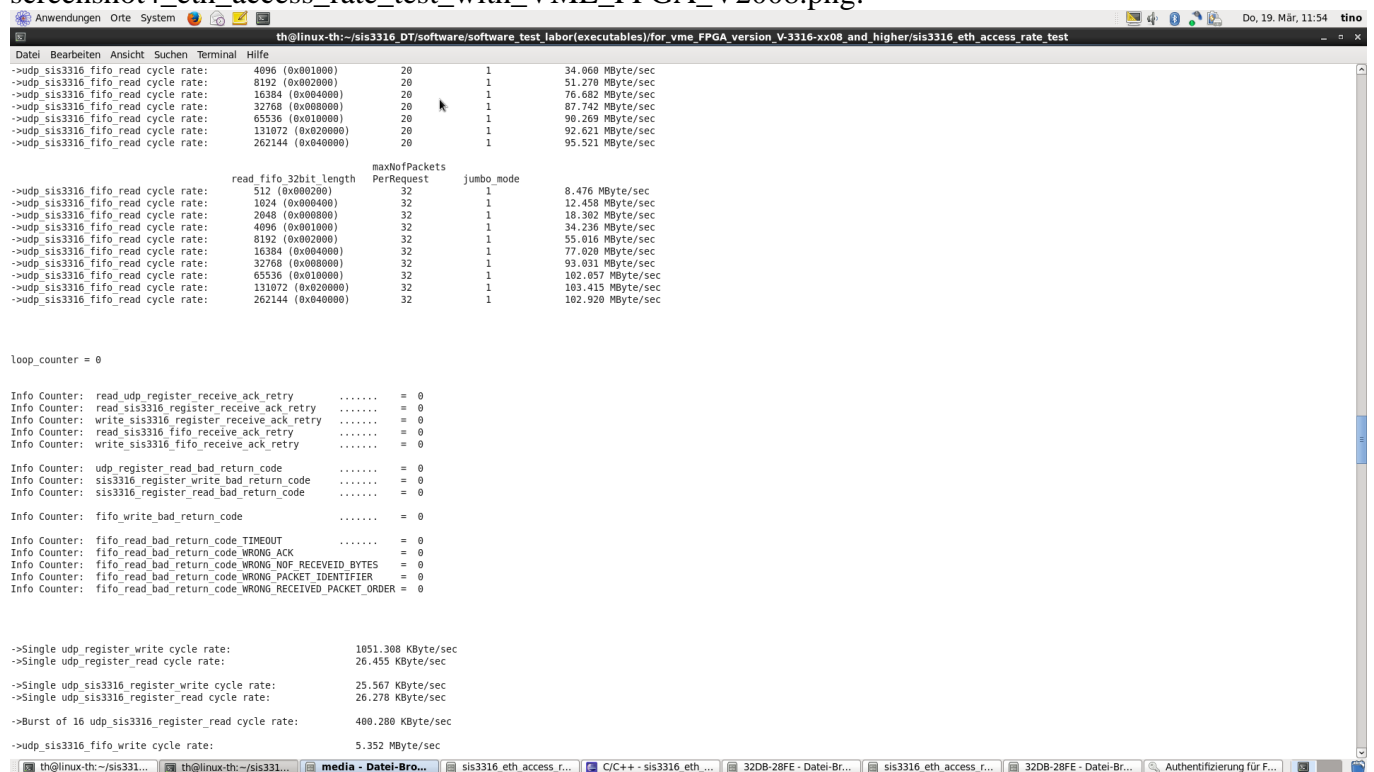


The interface will be defined in “project\_interface\_define.h”:



Screenshots of the start and execution of the program are save in the directory “sis3316\_eth\_access\_rate\_test”.

screenshot4\_eth\_access\_rate\_test\_with\_VME\_FPGA\_V2008.png:





---

### 6.1.3 MAC OS (10.9.4)

The development tool Eclipse (Kepler Service Release 2) is used to build the project.

## 6.2 Executable programs

**Note:** the Ethernet UDP protocol has changed with VME FPGA Version V3316-2008 and it is not compatible with previous versions!

On the DVD you will find two directories:

- sisdvd\_nnnn\sis3316\software\win7\_v2006\_a0008\software\_test\_labor(executables)
- sisdvd\_nnnn\sis3316\software\win7\_v2008\_a0008\software\_test\_labor(executables)

### 6.2.1 sis3316\_eth\_access\_rate\_test

```
C:\...\sis3316_eth_access_rate_test>sis3316_eth_access_rate_test.exe -h
Usage: sis3316_eth_access_rate_test.exe [-?h] [-p] [-I ip] [-N number of loops] [-C config_filename.ini] [-F or -f data_filename]

-I string ..... SIS3316 IP Address Default = 192.168.1.2
-J ..... enable Jumbo packets
-h ..... print this message only

date: 19.03.2015
```

### 6.2.2 sis3316\_eth\_fpga\_update

```
C:\projekte\s3316\software\software_test_labor(executables)\Ethernet-Interface\for_vme_FPGA_version_V-3316-xx08_and_higher\s3316_eth_fpga_update>sis3316_eth_fpga_update.exe -h
sis3316_fpga_update_udp
Usage: sis3316_eth_fpga_update.exe [-?h] [-I ip] [-V or -A] [-F filename]
-I string SIS3316 IP Address Default = 192.168.1.100
-V or -A -V->VME FPGA or -A->ADC FPGA
-F filename for example sis3316vme_top.bin or sis3316_adc_top.bin

-h Print this message

date: 18.03.2015
```

### 6.2.3 sis3316\_set\_dhcp\_mode

```
C:\projekte\s3316\software\software_test_labor(executables)\Ethernet-Interface\for_vme_FPGA_version_V-3316-xx08_and_higher\s3316_set_dhcp_mode>sis3316_set_dhcp_mode.exe -h
sis3316_set_dhcp_mode
Usage: sis3316_set_dhcp_mode.exe [-?h] [-I ip] [-C or -E or -D]
-I string SIS3316 IP Address Default = 192.168.1.100
-C Clear DHCP option value (DHCP mode depends on Switch SW80-4)
-E Enable DHCP
-D Disable DHCP

-h Print this message

date: 19.03.2015
```

## 7 Index

Address Map .....	17	PCB revision .....	25
addressing .....	17	ping .....	6, 8
ARP .....	6, 8	ping .....	5
bind .....	6	Reboot FPGAs .....	19
Build the program sis3316_eth_access_rate_test	26	register	
Computer UDP-socket connection .....	5	Control/Status .....	19
data transfer rate table .....	15	Ethernet Speed test counter .....	24, 25
DHCP .....	5	firmware revision .....	21
Ethernet .....	20	hardware version .....	25
executables .....	33	Link Interface Access Arbitration Control .....	23
FPGA Ethernet UDP protocol .....	9	module Id. ....	21
Hardware prerequisite .....	4	UDP protocol configuration .....	22
ipconfig .....	7	VME/CTRL FPGA Link interface description	19
Jumbo frames .....	14	register set	
key address .....	17	VME/CTRL FPGA Link interface .....	18
LED		Scientific Linux .....	26
Application Mode .....	20	SPF .....	4
U	19	Static ARP entry example .....	6, 8
LED 1 .....	19, 20	SW80 .....	20
LED 2 .....	19, 20	VC10 Project Properties .....	27
LED U .....	20	VME .....	20
LINUX .....	7	Windows .....	6
Optical .....	20	Windows 7 / Windows 8 .....	26, 30, 32