S i T C P Manual

Version 1.4

12th Nov. 2012

Tomohisa Uchida

Electronics system group, IPNS, KEK

1. History

Date of Modifications	Contents
2011/01/18	Enacted Version 1.0
2011/01/19	Version 1.1 Correction of errata
2011/04/14	Added detailed descriptions about internal register.
2012/10/24	Added supplementary descriptions to "9.SiTCP Internal Register/Detailed Procedures"
2012/11/12	Corrected the errata on User I/F Signal Input/Output (TCP_CLOSE_REQ, TCP_CLOSE_ACK)

2. Contents

This document describes the introduction of the Network Processor (SiTCP) and the outline of how to use it. For concrete implementation methods using the SiTCP Library, see the separate document "SiTCP Library" after reading this document.

3. What is SiTCP?

SiTCP is the technology to connect a physical experiment frontend to PC via Ethernet.

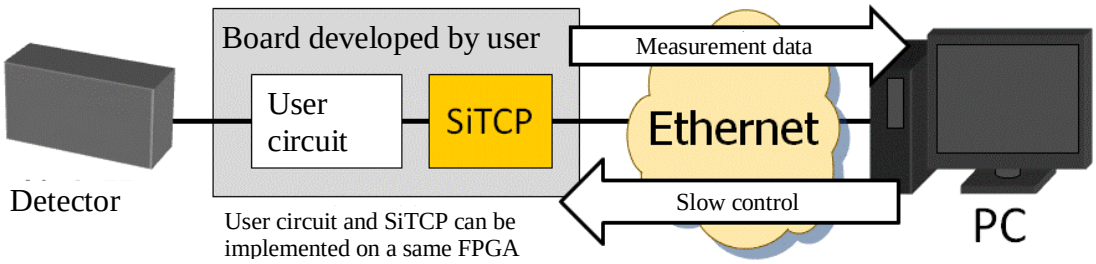


Fig.1 An example of SiTCP standard use

In the experimental electronics development, we often develop a read-out circuit close to the detector. This is because there is no generic instrument available that can be used in such near-detector parts. The data from the detector needs to be transferred to PC by any means. SiTCP can use the Ethernet for such data transfer.

The measurement data is transferred to the PC with SiTCP by writing the data in SiTCP along the procedure to write in a synchronous FIFO. SiTCP can be described as the technology to provide a data pipe between the user circuit and the PC. The data written in SiTCP will appear on the PC.

The slow control from the PC to the user circuit is a remote bus control. SiTCP has a bus operable on a simple protocol for the user circuit control, like the VME bus. Sending the UDP packet to SiTCP with its packet format pre-determined by the PC generates a bus signal to control the user circuit. Since SiTCP supports the read/write function, the slow control can be implemented by adding any simple circuit to the user circuit.

4. Advantages in Ethernet connection

The advantages in using Ethernet include the followings.

- You do not need to create the device driver on PC side.
  - In many cases, it is implemented as standard in your OS.

- You can write programs only with standard functions.
  - Socket programming
  - Many OSs support socket functions.
- You can build a virtual and flexible system structure.
  - Connected to HUB, the structure is configurable only on the software.
- Long distance communication is possible.
- Ethernet particularly has the following advantages.
  - High connectability between different technology generations (compatibility)
  - Technological evolution corresponding to any technological development (Latest is 40G/100Gbps)
  - Low cost and wide varieties of the commercially available products
  - Multiple media (Optical, UDP etc.)

5. Features

SiTCP features the followings.

- Hardware TCP/IP/Ethernet communication
  - 10Mbps to 1Gbps Ethernet
  - High speed communication stable at the upper limit of TCP
  - Slow control function (Using UDP)
- Easy implementation
  - Small circuit scale
  - Provided as FPGA library (Xilinx)
  - Simple user I/F easy to handle

6. Standard implementation example

SiTCP provided as Xilinx library is usually implemented to FPGA. Below shows the examples for which the standard twisted cable (UTP) is adopted for the Ethernet port as well as with optical cables.

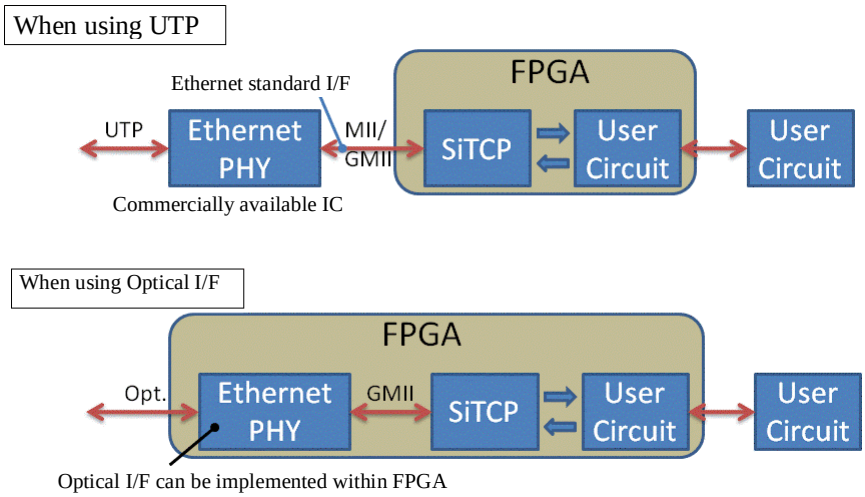


Fig.2 Standard implementation example

An example with a standard twisted cable is shown in the upper figure and with an optical cable in the lower.

When using the twisted cable, I/F chip for the Ethernet connection is required. Since SiTCP adopts a standard I/F, MII/GMII, you can use any standard chips available from the manufacturers. Use the proven and recommended devices if there are no particular problems, in using SiTCP library. When using the devices other than the recommendations, you need to fabricate a new register initialization circuit for your PHY devices.

The parts and user I/F required to operate SiTCP are shown in the figure below.

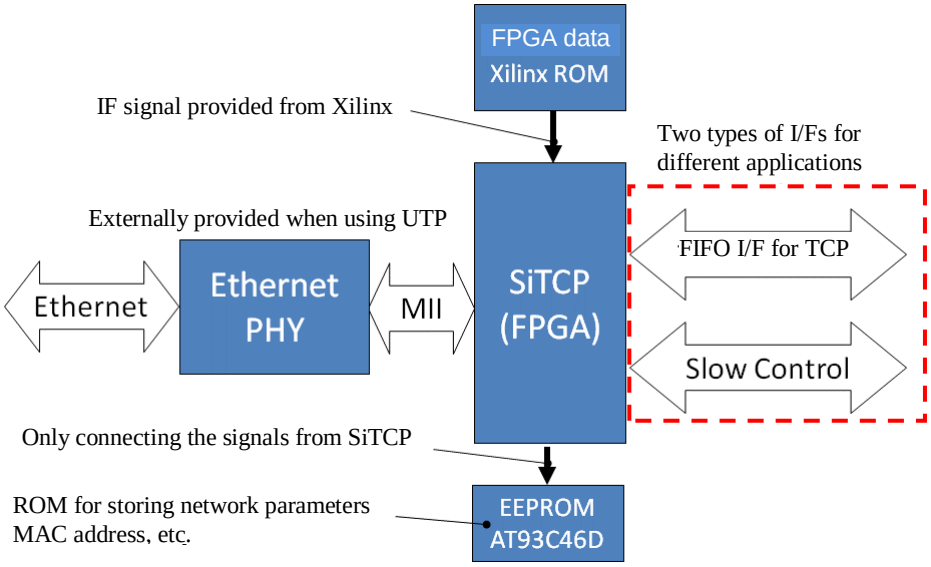


Fig.3 Required parts and user I/F

The following four parts are required to operate SiTCP.

1. FPGA from Xilinx: The FPGA to implement SiTCP
2. ROM for FPGA: The ROM to store FPGA data
3. EEPROM : The ROM to store the network parameters such as MAC address etc.
4. Ethernet PHY device: required when UTP adopted

Item 1 and 2 are required to implement SiTCP. Choose FPGA capacity with some allowance. You can find SiTCP size by synthesizing the library. Estimate and choose the FPGA capacity by pre-synthesizing the library.

Item 3 is the ROM to store the network parameters for SiTCP. Here, many parameters such as IP address, MAC address, TCP operational parameter are stored. SiTCP sets the parameters from this ROM to the read-out circuit only once after its launch. It also has the mode to read default values, i.e. standard values, compulsorily for directly after the manufacturing, or the case with abnormal values written in. This mode is normally used to re-write the contents of EEPROM or assess libraries.

Item 4 is pre-requisite when UTP used. Since SiTCP adopts a standard I/F, MII/GMII, you can use any standard chips available from the manufacturers. Use the proven and recommended devices when there are no particular problems, in the SiTCP library. When using devices other than the recommendations, you need to fabricate a new register initialization circuit for PHY devices.

Since the products incorporated with the PHY device for optical I/F are available, the external PHY devices are not always required when using optical I/F.

There are two types of user I/F to connect SiTCP and the user circuit. These are TCP FIFO I/F used for data transfer and the slow control I/F used for register control etc. SiTCP adopts simple interfaces since it has been developed assuming the users are non-expert for circuit development. Since TCP FIFO I/F for data transfer is the I/F similar to the synchronous FIFO, the data written in SiTCP sequentially is transferred to the PC automatically. The slow control I/F is the control I/F adopting a simple bus access protocol that consists of address signal, data signal, read/write signal and etc. Since this is a synchronous circuit, you can design it more easily than the standard bus systems such as VME bus etc. For detailed descriptions about the user I/F, refer to the later sections.

7. I/F Signal

Each of the following I/F signals for SiTCP library are described here.

- System I/F
- MII / GMII
- User I/F
- Network parameter setting I/F

System I/F

The following system I/Fs are associated with the entire SiTCP operation.

Signal name	**I/O**	Description
CLK	I	The system clock over 130MHz is recommended. Over 15MHz is recommended to operate Ethernet below 100Mbps.
RST	I	System reset
TIM_1US	I	1us Periodic pulse (*1)
TIM_1MS	I	1ms Periodic pulse (*1)
TIM_1S	I	1 sec Periodic pulse (*1)
TIM_1MIN	I	1 min Periodic pulse (*1)
EEPROM_CS	O	Connected to CS terminal of AT93C46D
EEPROM_SK	O	Connected to SK terminal of AT93C46D
EEPROM_DI	O	Connected to DI terminal of AT93C46D
EEPROM_DO	I	Connected to DO terminal of AT93C46D
SiTCP_RST	O	SiTCP internal reset output (*2)

All the signals are synchronous signals of the system clock CLK

(*1) The periodic signal with its system clock 1 pulse width H. All the TIM* signals must be synchronized in their phases.

(*2) The reset signal output when SiTCP is in reset state. Usable as a reset of the external circuit. Since, inside SiTCP, the system reset RST is extended along the internal circuit, it keeps reset for a certain period of time after the system reset is released.

MII/GMII

There are some notices concerning MII/GMII interface connection to SiTCP.

MII is the standard used in Ethernet up to 100Mbps. GMII is the standard used in Gigabit Ethernet. These two standards are quite similar but have some differences. Since they have much in common, the PHY signal is compliant for both of them. It needs to be switched for MII or GMII according to the Ethernet speed.

GigaSiTCP can be used exclusively for MII.

I/F Signal

The signals associated with MII/GMII are shown below.

Signal name	**I/O**	Description
MODE_GMII	I	Set this signal to 1 when using GMII-compliant PHY or 0 with PHY for MII only.
GMII_1000M	I	Generated from SPEED LED of PHY device (Notice 1)
GMII_RSTn	O	Connected to the reset of PHY device.
GTX_CLK	O	This is not a SiTCP signal but needed for GMII connection. Connected to GTX_CLK of PHY device. Generating 125MHz inside FPGA (Notice 2)
GMII_TX_CLK	I	Input TX_CLK of PHY devices on MII operation or 125MHz same as GTX_CLK generated in FPGA on GMII operation.
GMII_TX_EN	O	Connect to TX_EN of PHY device.
GMII_TXD	O	Connect to TXD of PHY device.
GMII_TX_ER	O	Connect to TX_ER of PHY device.
GMII_RX_CLK	I	Connect to RX_CLK of PHY device.
GMII_RX_DV	I	Connect to RX_DV of PHY device.
GMII_RXD	I	Connect to RX_D of PHY device.
GMII_RX_ER	I	Connect to RX_ER of PHY device.
GMII_CRS	I	Connect to RX_CRS of PHY device.
GMII_COL	I	Connect to RX_COL of PHY device.
GMII_MDC	O	Connect to MDC of PHY device.

GMII_MDIO_IN	I	Connect to MDIO of PHY device.
GMII_MDIO_OUT	O	Connect to MDIO of PHY device via 3 state buffer.
GMII_MDIO_OE	O	3 state buffer control signal for GMII_MDIO_OUT

(Notice 1) The signal to switch GMII/MII operation, generated from LED that displays the PHY output speed. 1 for Gigabit Ethernet (GMII) and 0 for 100Mbps Ethernet (MII). Notice that the logic may be reversed when using LED.

(Notice 2) The transmission clock for Gigabit Ethernet. FPGA outputs 125MHz towards PHY. Although PHY outputs the transmission clock when using MII, FPGA must output it with GMII.

Connection for GIGABIT ETHERNET compliant PHY devices

The connections between SiTCP and MII/GMII when using Gigabit Ethernet compliant PHY devices are shown in the figure below.

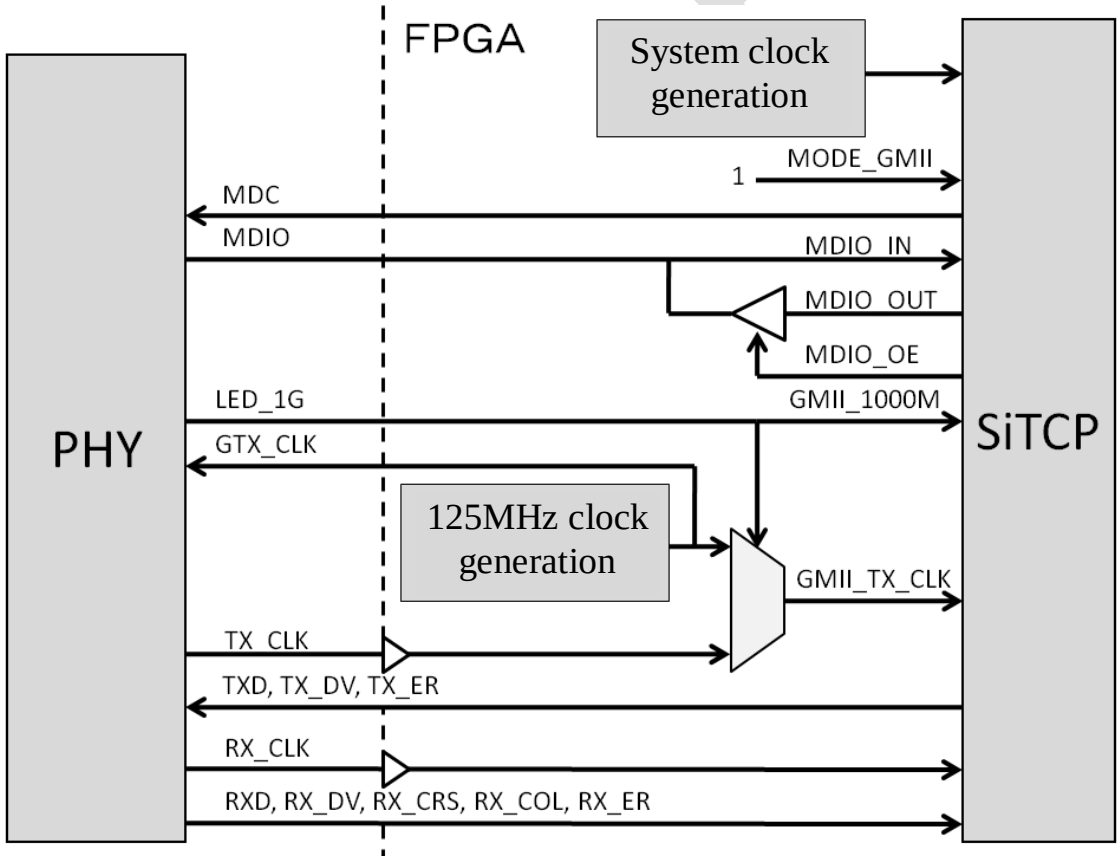


Fig.4 Connections between SiTCP and MII/GMII (Gigabit Ethernet compliant)

Some notices are listed below.

- Connect TX_CLK and RX_CLK to the global clock input pins of FPGA.
- Connect MDIO via 3 state buffer.
- Use primitive BUFGMUX or any equivalent primitives to switch GMII_TX_CLK.

Connection for GIGABIT ETHERNET non-compliant PHY devices

The connections between SiTCP and MII when using Gigabit Ethernet non-compliant PHY devices are shown in the figure below.

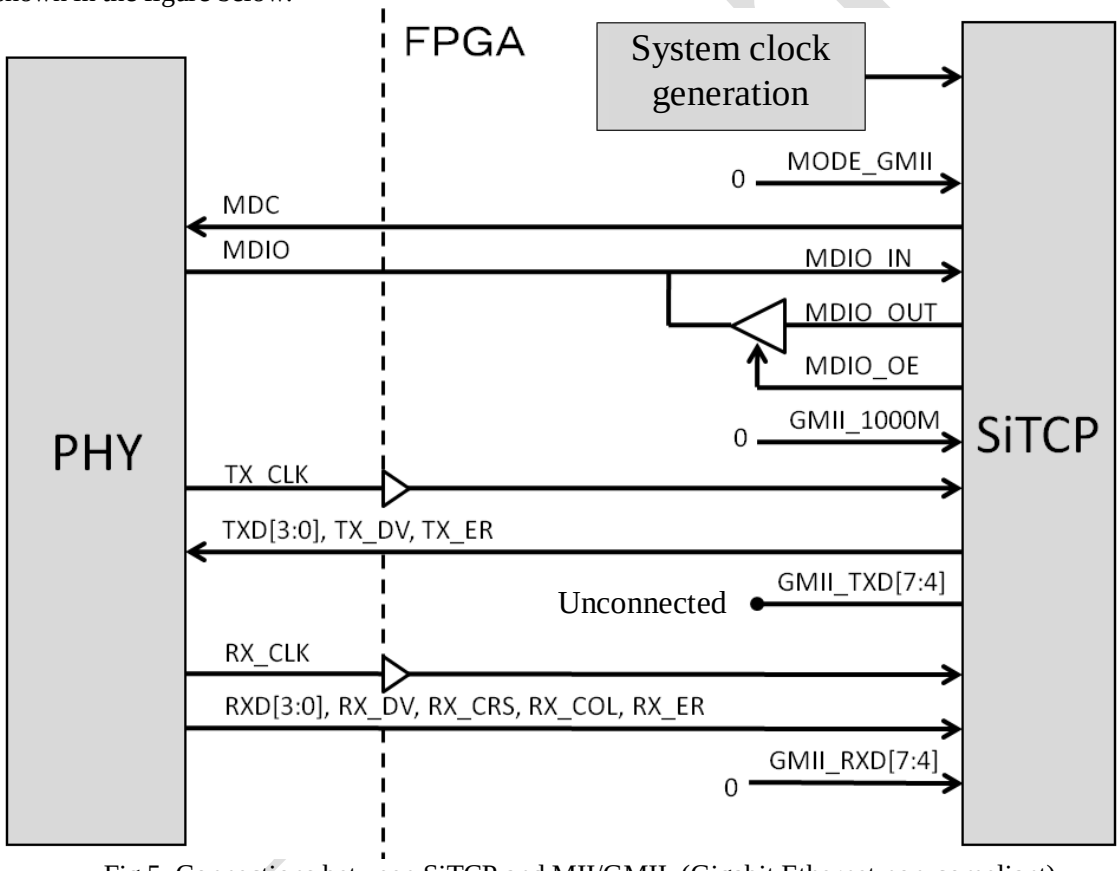


Fig.5 Connections between SiTCP and MII/GMII (Gigabit Ethernet non-compliant)

Some notices are listed below.

- Connect TX_CLK and RX_CLK to the global clock input pins of FPGA.
- Connect MDIO via 3 state buffer.

- Input 0 to GMII_1000M
- Leave GMII_TXD[7:4] unconnected since it is not used.
- Input 0 to GMII_RXD[7:4] since it is not used.

User I/F

SiTCP has the following two user I/Fs.

- TCP I/F (for data transfer)
- Slow control I/F (for control of register access etc.)

TCP I/F

TCP I/F has the following features.

- Used for data transfer with PC
- High speed transfer is possible.
- All the signals are synchronized to the system clock.
- All the signals are in positive logic.
- I/F similar to the synchronous FIFO
- Transmission/reception independent
- It transitions to a usable state when the TCP connection is established.

TCP I/F can transmit or receive signals independently. FIFO memory needs to be inserted between SiTCP and the user circuit when using the reception function. The TCP reception function is not needed when the data transfer from PC to the user circuit is not required. However, when the transfer is required and the slow control function described in the next chapter does well for it, use the slow control function instead of TCP reception function. The reasons for not using the reception function include difficulty in simultaneous control of transmission/reception processed at a high speed using socket programming as well as the need for FIFO memory addition.

I/F Signals

I/F signals are shown in below. I/Os are the input/output viewed from SiTCP. The signal given to SiTCP is I and the one output by SiTCP is O.

Signal name	**I/O**	Description
TCP_OPEN_ACK	O	Indicates the communicable state
TCP_CLOSE_REQ	O	Communication-termination response
TCP_CLOSE_ACK	I	Indicates the reception of communication-termination demand from PC
TCP_TX_DATA[7:0]	I	Transmission byte data
TCP_TX_WR	I	Transmission data write enable.
TCP_TX_FULL	O	Transmission buffer full flag.
TCP_RX_DATA[7:0]	O	Reception data
TCP_RX_WR	O	Reception data valid
TCP_RX_WC[15:0]	I	The number of data bytes stored in reception buffer.

Communication state control

In TCP communication, both sender and receiver ensure the data transfer by notifying the data transfer status to each other.

First, each counterpart is checked if communicable. After the check (establishment of TCP connection), the data transfer is started. The establishment of TCP connection is equivalent to the normal termination of the ‘connect’ function in the socket programming.

A communication-stop demand is issued to the counterpart when the communication needs to be terminated. The communication is terminated on the reception of the communication-termination response from the counterpart.

As above, the TCP communication must be done while the TCP connection is established. Also, on the reception of communication-termination demand from the PC, the data transfer must be quickly terminated and the communication-termination response must be transmitted. SiTCP has the control signals for the communication according to the above TCP provisions. The time-chart for the control signals from the TCP connection establishment to its termination, is shown below.

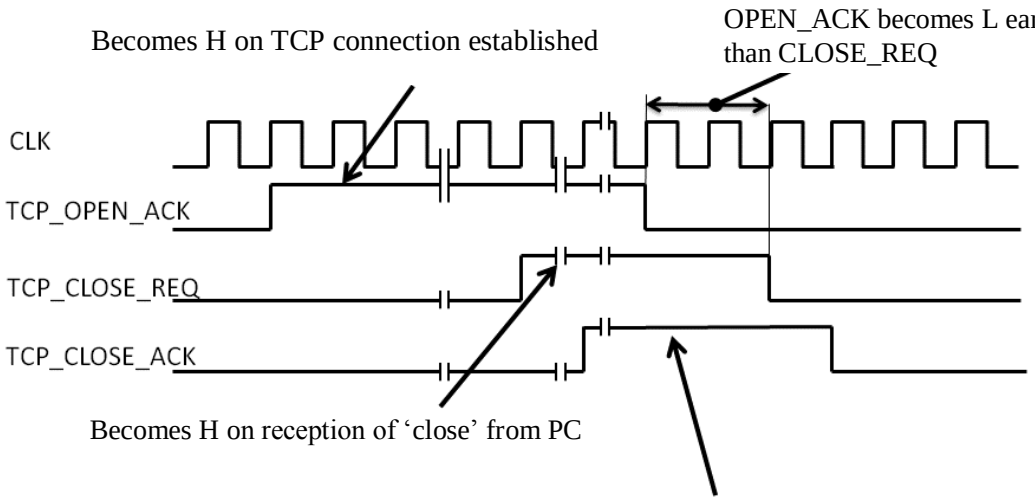


Fig.4 Time-chart for TCP control signals

TCP communication is permitted while TCP_OPEN_ACK is set to H and TCP_CLOSE_REQ set to L. After TCP_CLOSE_REQ is set to H, the user circuit must finish its data-writing up to any terminatable point and then set TCP_CLOSE_ACK to H in order to issue a termination-response. However, since the PC has stored the received data already up to when it receives TCP_CLOSE_REQ, the data transmitted thereafter will be discarded by the PC. Therefore, it works well without any problems if the user circuit directly connects TCP_CLOSE_REQ and TCP_CLOSE_ACK, and terminates the transmission immediately when TCP_CLOSE_REQ=H. Retain TCP_CLOSE_ACK to H while TCP_CLOSE_REQ is H.

As above, TCP_OPEN_ACK signal is equivalent to the data receivable state of PC. TCP_OPEN_ACK can be used as a negative logic reset signal of the user circuit when the PC is not receivable and the user circuit is preferred unoperated. TCP_OPEN_ACK becomes L earlier than TCP_CLOSE_REQ. So, when using TCP_OPEN_ACK as a reset, design it very carefully so that TCP_CLOSE_ACK does not become L on the reset while TCP_CLOSE_REQ is H.

Data Transmission

The data can be transmitted only in the communicable state where the TCP connection has been established. When writing in the transmission data, notice that the writing operation needs to be stopped when the transmission buffer becomes full. The time-chart for the transmission data write-in is shown below.

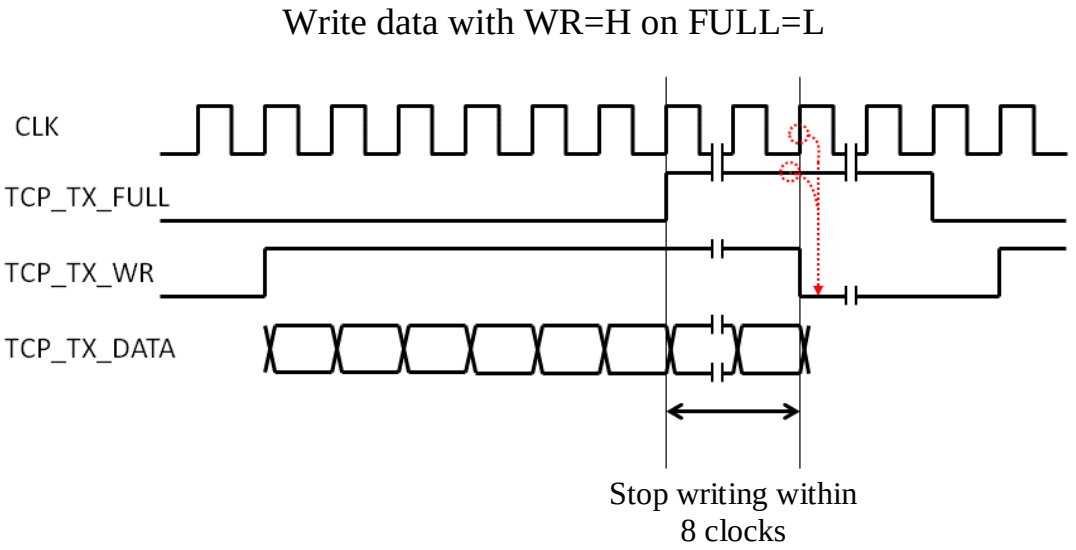


Fig.5 Time-chart for transmission data write-in

This is the same interface as that for the synchronous FIFO memory.

TCP_TX_WR and TCP_TX_DATA[7:0] are used to write-in the transmission data. Putting the transmission data in TCP_TX_DATA with a synchronization to the system clock and setting TCP_TX_WR=H, will make the transmission. Since the circuit is synchronous, every leading edge of CLK is regarded as a byte data.

Now, it is in the state where the TCP connection has been established, and possible to write in the data on TCP_TX_FULL=L. In the course of the transmission, whenever TCP_TX_FULL=H, stops the transmission operation within 8 clocks and set TCP_TX_WR=L. The transmission operation can be resumed when TCP_TX_FULL=L. Repeat the above while the TCP connection is established.

Data Reception

When using the TCP reception function, place a reception buffer FIFO between the user circuit and SiTCP. The TCP data reception signal of SiTCP is set connectable directly to the FIFO generated by Xilinx Core Generator.

When generating FIFO memory using Core Generator, configure it as follows.

- Write-in data bit width = 8bit

- Word count option enabled, Word count bit width maximized

The time-chart for the data reception is shown below.

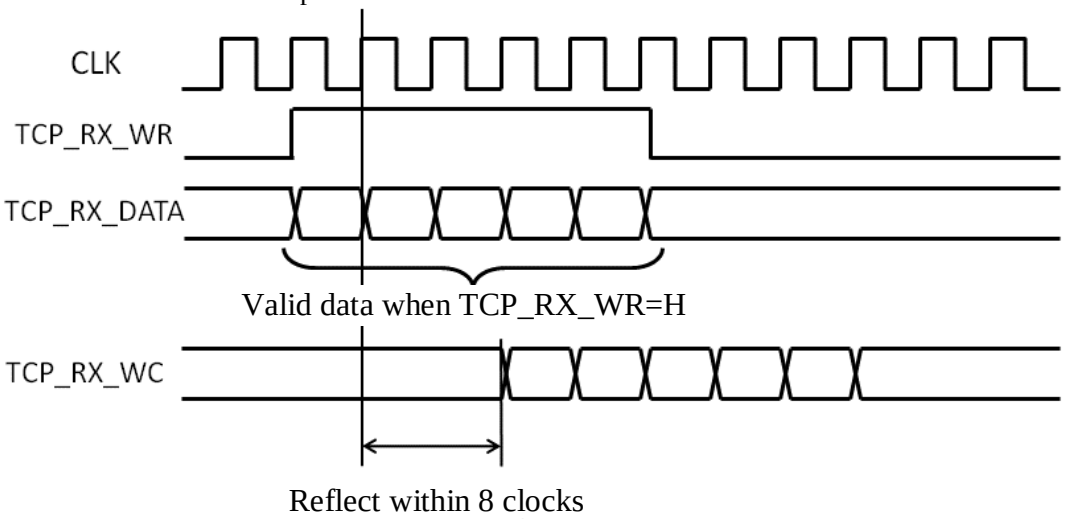


Fig.6 Data reception time-chart

When SiTCP receives the data, it puts it to TCP_RX_DATA[7:0] in the received order and sets TCP_RX_WR=H to notify the reception of the valid data.
Connect TCP_RX_DATA[7:0] to the write-in data terminal and TCP_RX_WR to the write-in enable terminal, of the synchronous FIFO, respectively.

Since the receiver must always notify to the sender how many bytes are receivable, notify using TCP_RX_WC[15:0] how many bytes of data are stored in the reception buffer, within 8 clocks after the data reception. Connect TCP_RX_WC to Word count terminal of the synchronous FIFO from its lower bit. Set 1 to all the unused bits. For example, when the capacity of reception FIFO is 2Kbyte, Word count terminal is 11bit. So, connect TCP_RX_WC[10:0]=Word count[10:0] and set 1 to all the bits of TCP_RX_WC[15:11]. In Verilog HDL, this is coded as "TCP_RX_WC[15:11] = 5'b11111;".

Refer to the next section when not using the TCP reception function.

When not using TCP reception function

Set 0 to TCP_RX_WC[15:0] when not using the TCP reception function. Setting 0 is recommended although it works with the other value setting.

Slow control I/F

This I/F is controlled with UDP packets. Here, we describe its operation and how to use it from the view of the user circuit. The control on the PC side will be described in the next chapter.

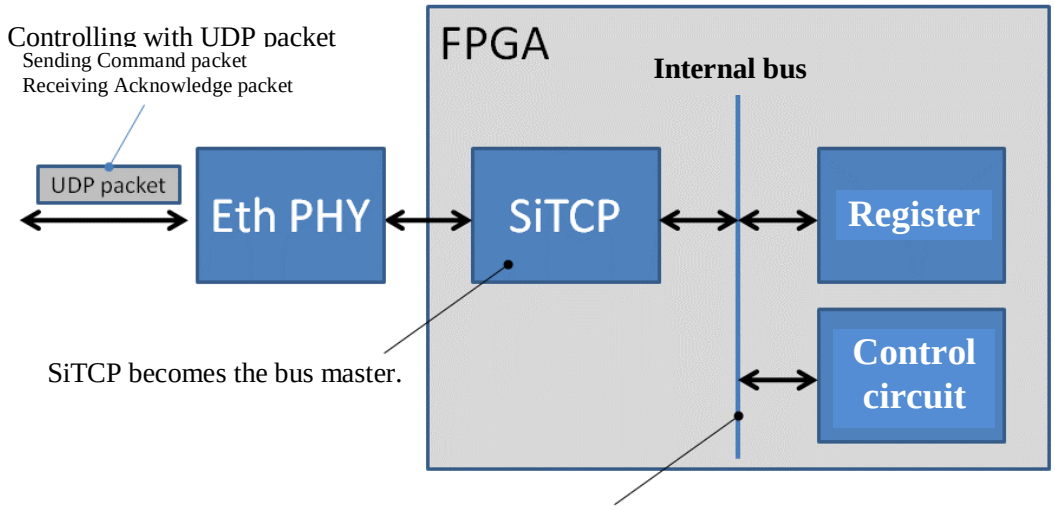


Fig.7 Schematics of slow control

The schematics of the slow control is shown in the above figure. The UDP packet is transmitted from PC to SiTCP in order to control the internal bus for which SiTCP works as its bus master. The internal bus consists of the address, data and control signal, and adopts a simple bus access protocol. When the PC writes data into the user circuit, it stores the address and write-data in the UDP packet to be transferred to SiTCP, and then the bus master in SiTCP operates to write data into the user circuit via the internal bus. When the PC reads data, it designates the address to make SiTCP send the data read in from the user circuit, back to the PC.

Notice Concerning Address Space

RBCP addresses from 0xFFFF0000 to 0xFFFFFFFF are reserved for the SiTCP internal register and cannot be used. Use the address other than the above mentioned.

Control Signals

Signal name	**I/O**	Description
CLK	I	System clock Same signal as CLK of TCP IF
RBCT_ACT	O	Indicates the bus operating.

RBCT_ADDR[31:0]	O	Address in access
RBCT_WE	O	Write enable
RBCT_WD[7:0]	O	Write data
RBCT_RE	O	Read enable
RBCT_RD[7:0]	I	Read data
RBCT_ACK	I	Access response

Write (From PC to User circuit)

When writing data from the PC to the user circuit, SiTCP sets the access address and write-data to RBCP_ADDR while setting RBCP_WE=H to notify that the operation is write-in.

When the user circuit receives the corresponding address, it must carry out a write-in operation and after finishing the operation, respond by setting RBCP_ACK to H. In the cases including an access to the address having no corresponding circuit, RBCP_ACK remains L although SiTCP judges the waiting time beyond approx. 100msec as an error and terminates the bus cycle. A packet to notify the error will be sent to the PC.

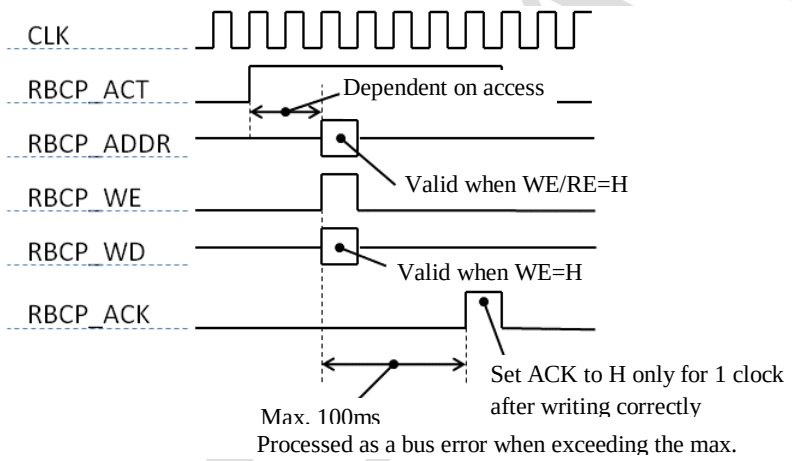


Fig.8 Data write-in by slow control

The time-chart for data write-in is shown in the above figure.

Read-out (From User circuit to PC)

When the PC reads data from the user circuit, SiTCP sets the access address to RBCP_ADDR while setting RBCP_RE=H to notify the user circuit that the operation is read-out.

The user circuit must set the data in corresponding address to RBCP_RD[7:0] while it responds by setting RBCP_ACK to H only for 1 clock. Also in read-out, SiTCP judges the ACK response time beyond approx. 100msec as an error, terminates the bus cycle and sends the packet to notify the error to the PC.

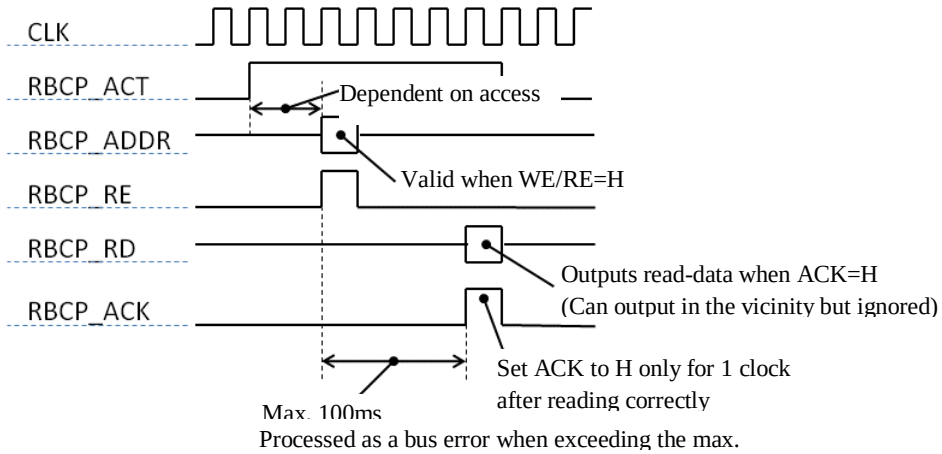


Fig.9 Data-read by slow control

The time-chart for the data-read is shown in the above figure.

Network parameters settings I/F

Although the network parameters such as IP address etc. are operated using the values stored in EEPROM, it can be used by the external input signals including DIP SW. Also, the I/F includes the terminal to read the default values compulsorily for the circuit operation, on re-writing EEPROM or any abnormality.

Signal name	I/O	Description
FORCE_DEFAULTn (*1)	I	0: Compulsorily uses default value 1: Uses EEPROM value
IP_ADDR_IN (*2)	I	IP address input terminal
IP_ADDR_DEFAULT	O	IP address value set to the register
TCP_MAIN_PORT_IN (*2)	I	TCP port number input terminal (main port)

TCP_MAIN_PORT_DEFAULT	O	TCP port number set to register (main port)
TCP_SUB_PORT_IN (*2)	I	TCP port number input terminal (sub port)
TCP_SUB_PORT_DEFAULT	O	TCP port number set to register (sub port)
RBCP_PORT_IN (*2)	I	RBCP port number input terminal
RBCP_PORT_DEFAULT	O	RBCP port number set to register
PHY_ADDR (*3)	I	Sets MIF address of PHY device

(*1) Normally, it connects DIP_SW, jumper pins or etc.

(*2) Normally, connect this to the same signal name having _DEFAULT at its end. When setting the network parameters configurable by the external signal (e.g. DIP SW, etc.), refer to "How to set network parameters by external signals including DIP SW" in the next section.

(*3) Read the data sheet of your PHY device for this setting. In many cases, this is determined according to the state of PHY device terminal on a power-on. How to set the address of the PHY device is different for each device.

How to set network parameters by external signals including DIP SW

The following network parameters can be set by the external signals, using DIP SW etc.

- IP address
- TCP main port number
- TCP sub port number
- RBCP port number

Below, we explain about IP address. For the other values, the same explanation is applied.

SiTCP always uses the value input in IP_ADDR_IN as IP address to operate. It does not depend on the value in FORCE_DEFAULTn.

When setting the IP address by DIP SW, connect DIP SW to IP_ADDR_IN.

The register value corresponding to the IP address is output to IP_ADDR_DEFAULT. When FORCE_DEFAULTn =0, a default value is output. With FORCE_DEFAULTn=1, the value read out from EEPROM is set. Some bits of this IP_ADDR_DEFAULT and the DIP SW can be combined and input to IP_ADDR_IN. For example, the register value and DIP SW value can be set to its upper and lower bits, respectively.

8. RBCP

Here, we describe a control protocol for slow control, Remote Bus Control Protocol (RBCP).

RBCP is the protocol to provide a remote control over the bus for which SiTCP becomes its bus master. RBCP packet encapsulated in the UDP packet is used.

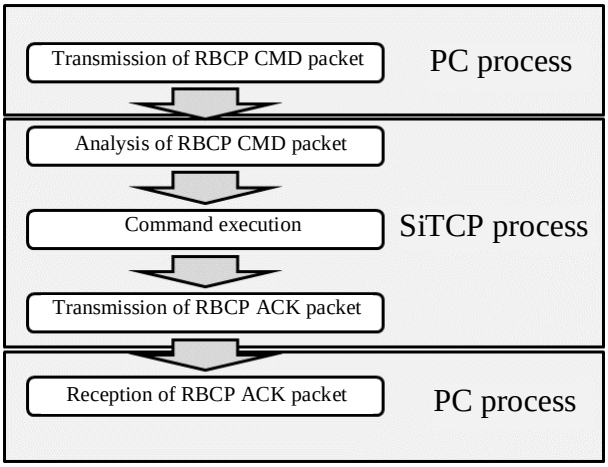


Fig.10 RBCP process flow

The process flow is shown in the above figure. First, the command packet generated according to the process contents is sent from the PC to SiTCP. The SiTCP waiting port at this moment is the value stored in EEPROM or the default value, 4660. The command packet is processed by SiTCP and an internal bus access is executed. When the bus access terminates, the ACK packet is sent back to the PC as a response. The PC can recognize that the access is terminated on the reception of the ACK packet.

Since RBCP packet is transferred using UDP, it can be lost within the network. So, the PC must move on to its next process after receiving the ACK packet and confirming the access termination. Ensure the access termination by having the PC set a timer while waiting for the ACK packet and, on its time-out, re-send the command packet.

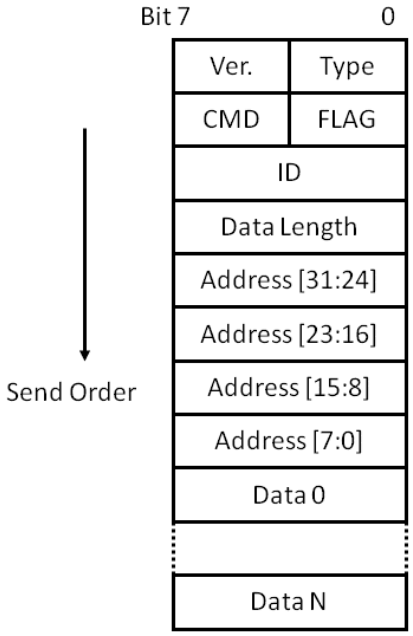
The time-out duration should be set to the value over 200msc, considering that the bus time-out of SiTCP is set to 100ms. Confirm the time-out duration by an actual operation since it depends on the PC's processing time and the transfer-time within the network. If slower operation is allowed, set the time-out to approx. 1sec. Speaking from our practical use experiences, it is rare to lose the packet. In many cases, the ACK packet is returned normally.

For the concrete implementation of RBCP, see the attached sample program "RBCP.zip". Its operation has been confirmed with Cygwin and Linux.

RBCP Packet Format

Here, we encapsulate the RBCP packet with UDP.

In your socket programming, transmit the RBCP packets shown below by the 'send' function.



CMD bit	Name	Description
3	Access	Bus Access
2	R/W	0: Write, 1:Read
1	Reserve	Always 0
0	Reserve	Always 0

FLAG bit	Name	Description
3	REQ/ACK	0: Request, 1: Acknowledge
2	Reserve	Always 0
1	Reserve	Always 0
0	Error	0: Normal, 1: Bus Error

Fig. 11 RBCP Packet format

The packet format is shown in the above figure. The meaning of every field is explained below.

VER

This is the version of the RBCP Packet. Currently, only 15(0xF) is accepted for it. Always, set 15 to it. Packets with other values will be discarded.

TYPE

This is 'Packet type'. Currently, only 15 (0xF) is accepted. Always, set 15 to it. Packets with other values will be discarded.

CMD

This is the code indicating the access type. The following two values are currently used.

- 0xc: Read
- 0x8: Write

Refer to 'CMD bit' listed on the right side in the figure.

FLAG

This indicates the packet type and execution result. It is the field valid only for the ACK packet. For the command packet, always set bit3=0. With a bus error to occur, the ACK packet with its bit0 set to 1 will be returned.

Also, refer to 'Flag bit' listed on the right side of the figure.

ID

This is the packet ID. It is used for an execution confirmation.

The ACK packet has its ID same as that of the corresponding command packet. Set an arbitrary value to this field since SiTCP does not confirm nor operate it.

Changing this ID every time allows a judgment if the packet is the ACK packet corresponding to the command packet.

DATA LENGTH

For the command packet: The Read/Write-data length is set to this field in bytes. Since setting this to 0 will not work, set any length other than 0.

For the ACK packet: It is the length of the data executed actually and normally. On an error interruption, the number of data normally executed is stored here. With an error occurrence, this value tells in which address the error occurred.

Since this is an 8bit length field, the byte length accessible at a time is limited to 255 bytes. When accessing over 255bytes, divide it into the multiple accesses for which the data length of an access is no more than 255.

ADDRESS

This is the address of the bus to access. The data of their length stored in 'Length' field is accessed from their starting address stored in this field.

DATA

This means differently for the command packet or the ACK packet.

Command packet

- On write: Store the data of the byte length equivalent to 'Data length'. If the data shorter than the value in the 'Data length' is transmitted, the data remaining in the internal memory will be written in and the write-value becomes undefined.
- On read: Do not use this field.

ACK Packet

- On write: The data actually written is stored. Here, the data are not the ones read from the equivalent address. These are the data used on the write. For example, when writing data in the read-only register, the write-operation is not actually valid but the written data will be stored in this field.
- On read: The data read-out is stored here.

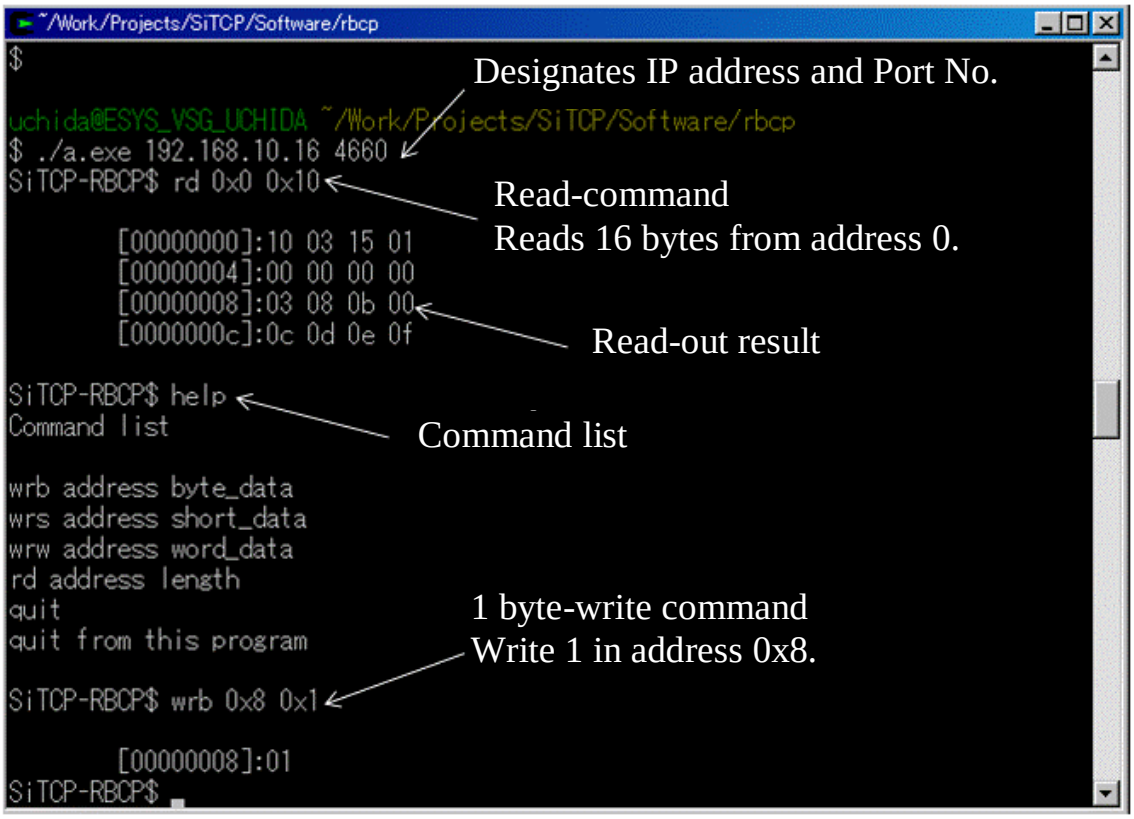
RBCP sample program

For an implementation example of the RBCP program, a sample program used for debugging is distributed as an attachment "RBCP_sample.zip".

Compile 'rbcp.c' with 'gcc' or alike for your use. Its operation has been confirmed on Linux. For the other OSs, you need to modify any necessary parts.

Here, a compiled executable file “rbcpWin.exe” for Windows is stored. Execute it from the command prompt of Windows.

The start screen of the sample program is shown below.



This may display differently for the latest version.

Fig.4 Sample program start screen

It is executed as follows.

Executable file name [IP address] [Port No.]

The figure shows the case with its IP address 192.168.10.16 and port No. 4660. On its start, the prompt changes to ‘SiTCP-RBCP$’. The input number is in decimal or hexadecimal expressions. When using the hexadecimal expression, add ‘0x’ to the head of the number. Each command is described below.

- wrb [address] [write-data]
  - 1 byte write
  - Writes the write-data to the designated address.

- wrs [address] [write-data]
  - 2-byte write, big-endian
- wrw [address] [write-data]
  - 4-byte write, big-endian
- rd [address] [length]
  - Reads and displays the data of the byte length [length] from start address [address].
- help
  - A list of the available commands
- quit
  - Terminates the program

Test example

- Read of DIP SW value
  - Read from the address 0 for 16 bytes.
  - Re-read the value by manipulating the DIP SW.
    - The value on the address 0x8 changes according to the DIP SW value.
- LED light on/off
  - Write '0xFF' to the address '0x7'.
    - The seven LEDs will light on.
  - Write '0x0' to the address '0x7'.
    - The seven LEDs will light off.

9. SIT CP Internal Register

SiTCP has its internal registers for storing the network parameters and controlling the PHYs. They are accessible using RBCP.

Address space map

RBCP address	Description
0xFFFF_0000 – 0xFFFF_FBFF	Reserved
0xFFFF_FC00 – 0xFFFF_FCFF	EEPROM
0xFFFF_FD00 – 0xFFFF_FDFF	Reserved
0xFFFFFE00 – 0xFFFF_FEFF	Ethernet PHY MIF I/F
0xFFFF_FF00 – 0xFFFF_FFFF	Internal register, SiTCP control register

Do not access 'Reserved' spaces.

Internal register map

Access them in big-endian.

The addresses in the table are shown in the differentiation from the base address '0xFFFFFF00'.

Address	**R/W**		Standard value
+0x00-03	R	SiTCP synthesis date-time: YY MM DD NN (binary-coded decimal expression) Y=Last two figures of A.D., M=Month, D=Date, N=Number of synthesis on the same date.	
+0x04-0F	R	ID: A specific number is stored here.	
+0x10	R/W	SiTCP and TCP control [7] SiTCP reset (1: Reset / 0: Normal) [6-3] Reserve (Should be set zero) [2] Keep alive packet (1: ON / 0:OFF) [1] Fast retrains. (1: ON / 0:OFF) [0] Nagle buffering (1: ON / 0:OFF)	0x01

+0x11	R/W	Reserved	
+0x12-17	R/W	MAC address	
+0x18-1B	R/W	IP address	
+0x1C-1D	R/W	TCP Port No. (main port)	0x18
+0x1E-1F	R/W	TCP Port No. (alternative port)	0x17
+0x20-21	R/W	TCP MSS (the number of bytes)	0x05B4
+0x22-23	R/W	RBCP Port No.	0x1234
+0x24-25	R/W	TCP keep-alive-packet transmission timer (with data in transmission buffer) (msec unit)	0x03E8
+0x26-27	R/W	TCP keep-alive-packet transmission timer (with transmission buffer empty) (msec unit)	0xEA60
+0x28-29	R/W	Time-out duration when TCP open (msec)	0x1388
+0x2A-2B	R/W	Time-out duration when TCP close (256msec unit)	0x2BF2
+0x2C-2D	R/W	Waiting-time between TCP connections (msec)	0x01F4
+0x2E-2F	R/W	Time-out duration of TCP re-transmission time (msec)	0x01F4
+0x30-3F	R/W	Reserved	0x0
+0x40-FF		Access-forbidden area	

(Notice) Work very carefully not to write any value in the MAC address and the access-forbidden area, otherwise the system will corrupt and not operate normally.

Re-write of EEPROM

We describe below how to change the network parameters with an example of changing IP address.

To change the IP address, the SiTCP parameters stored in EEPROM need to be changed. Work very carefully not to write any improper parameters to EEPROM, otherwise SiTCP will corrupt leading to the abnormal operation and communication.

The memory map of the parameters stored in EEPROM is the one in the previous section “Internal register map” but with its base address substituted with ‘0xFFFFFC00’ (excluding from +0x00 to +0x0F). For example, the IP address is stored in ‘0xFFFFFC18-1B’.

Work procedure outline

Start SiTCP in its default-value compulsory setting mode to make it communicable with the following fixed address.

- IP address: 192.168.10.16
- RBCP Port No.: 4660

This mode operates with always the same parameters regardless of the EEPROM contents.

Then, set the mode for re-writing, by RBCP, the IP address stored in EEPROM back to normal, and re-boot the system.

Detailed procedures

1. Start in the default mode
   A) Power off
   B) Switch to the default-value compulsory setting mode (FORCE_DEFAULTn=0). Normally, this is often done with a jumper or DIPSW.
   C) Power on.
   D) Confirm SiTCP operating with Ping command.(*1)
2. Re-write EEPROM
   A) Write '0x00' of a byte to '0xFFFFFCFF' using RBCP.
   B) Write the IP address to '0xFFFFFC18-1B'.
   (1) '0xFFFFFC18' is equivalent to the upper bytes which are the bit31-24 of the IP address.
   C) Read '0xFFFFFC18-1B' by RBCP to confirm the set value.
   D) Power off.
3. Operation check
   A) Set the operation mode to the normal mode (FORCE_DEFAULTn=1).
   B) Power on.
   C) After the FPGA download is finished, confirm that the IP address has been changed to the desired one, using Ping command.
4. Terminate.

(*1) The IP address is determined with the value input in 'IP_ADDR_IN' of the library. It does not depend on the value of 'FORCE_DEFAULTn'. When using SiTCP library with a wrapper, this depends on the signal connections within the wrapper. Study the value from the signal connections of the wrapper or ask the distributor of the SiTCP library.

(Notice) When changing the IP address, it may be necessary to change the network settings on PC side.

10.Appendix

100Mbps Ethernet reference circuit diagram (mii.pdf)

Gigabit Ethernet reference circuit diagram (gmii.pdf)

RBCP control program for debugging (RBCP.zip)

11.References

1. SiTCP web site http://e-sys.kek.jp/tech/sitcp/
2. BBT web site http://www.bbtech.co.jp/
3. A paper on SiTCP : T. Uchida, “Hardware-Based TCP Processor for Gigabit Ethernet,” http://hdl.handle.net/2261/15490