RX63N Group

Zero-Copy Ethernet Driver Demonstration

R01AN1190EU0100
Rev.1.00
Jun 01, 2012

Introduction

This application note explains the features and capabilities of the Renesas RX63N zero-copy Ethernet driver. This application note assumes some experience with Ethernet and device driver usage. For more introductory material on these subjects please see the references. The content in this document is a follow-up to the topics discussed in R01AN0169EU uIP TCP/IP Protocol Stack Demonstration application note and the reader is strongly recommended to read it along with this document.

Target Devices

RX63N Group.

Contents

1. Overview

This document and accompanying software provide an update to the initial Ethernet drivers and the uIP TCP/IP stack demonstration project released in R01AN0169EU application note. Main focus of this document is the improvements to the RX63N Ethernet driver. Further, the uIP TCP/IP demonstration project is updated with "real-life" usage scenarios.

2. Renesas Ethernet Drivers

Zero-copy Ethernet drivers include circular buffers to transmit and receive Ethernet frames. The operation of Ethernet Controller Direct Memory Access Controller (E-DMAC) is controlled by descriptor lists and data buffers. This is explained in detailed in E-DMAC section of the Hardware Manual. In summary, E-DMAC provides hardware support for transferring data between the Ethernet Controller (EtherC) and buffers in RAM.

Zero-copy Ethernet driver that comes with this document has the following features:

- Interrupt driven
- Zero-copy driver
- MAC layer flow control
- Physical layer link status

These new features will be described in the following sections.

2.1 Interrupts

Zero-copy Ethernet driver makes use of the E-DMAC's frame reception and frame transmit complete interrupts. There are several other interrupt sources such as CRC error detection, multicast frame reception, and receive FIFO overflow available the end user to advance the design of the driver.

Ethernet interrupt service routine (ISR) for frame reception checks the received frame against errors and marks it for further processing by the R_Ether_Read_ZC() function. The R_Ether_Read_ZC() function reads the received data pointed by the receive descriptor. The erroneous frames are simply dropped. However, the error information is available to the R_Ether_Read_ZC() function if further processing is required by the end user.

Because the sample demonstration project does not have an RTOS, the signaling between the ISR and R_Ether_Read_ZC() function is accomplished by a global variable.

2.2 Zero-Copy Driver

A driver either copies data to/from a user application (or a communication stack) or manages data itself and transfers pointers to move data. The Ethernet driver described in this documentation uses pointers to move received data from E-DMAC to uIP TCP/IP stack. R_Ether_Read_ZC() function uses a receive descriptor and passes only the pointer to the received data to the user application. In Figure 1, the RBA (receive buffer address) field points to the received data and the zero-copy driver passes this pointer instead of copying data to another buffer used by the application.

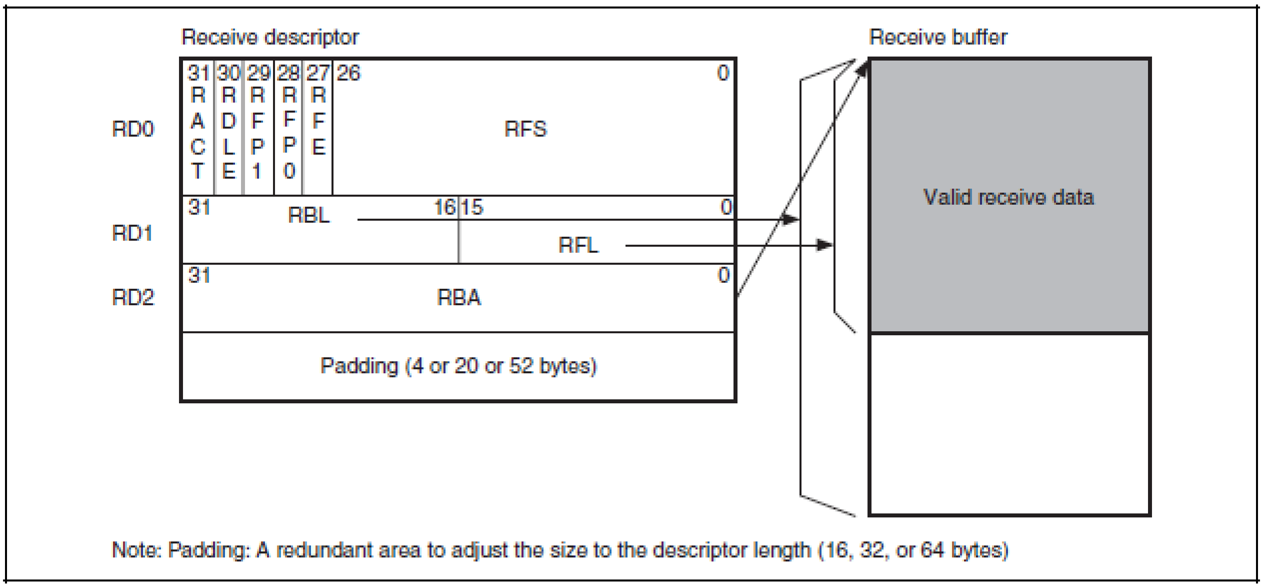

Figure 1 Receive Descriptor

For transmit, R_Ether_Write_ZC() function moves data from the user application to E-DMAC. Zero-copy Ethernet driver always makes a buffer available to uIP TCP/IP stack by passing a pointer to it. The TBA (transmit buffer address) field as shown in Figure 2 points to a memory location available from the pool of transmit buffers. The stack writes the data to be transmitted directly into this buffer.

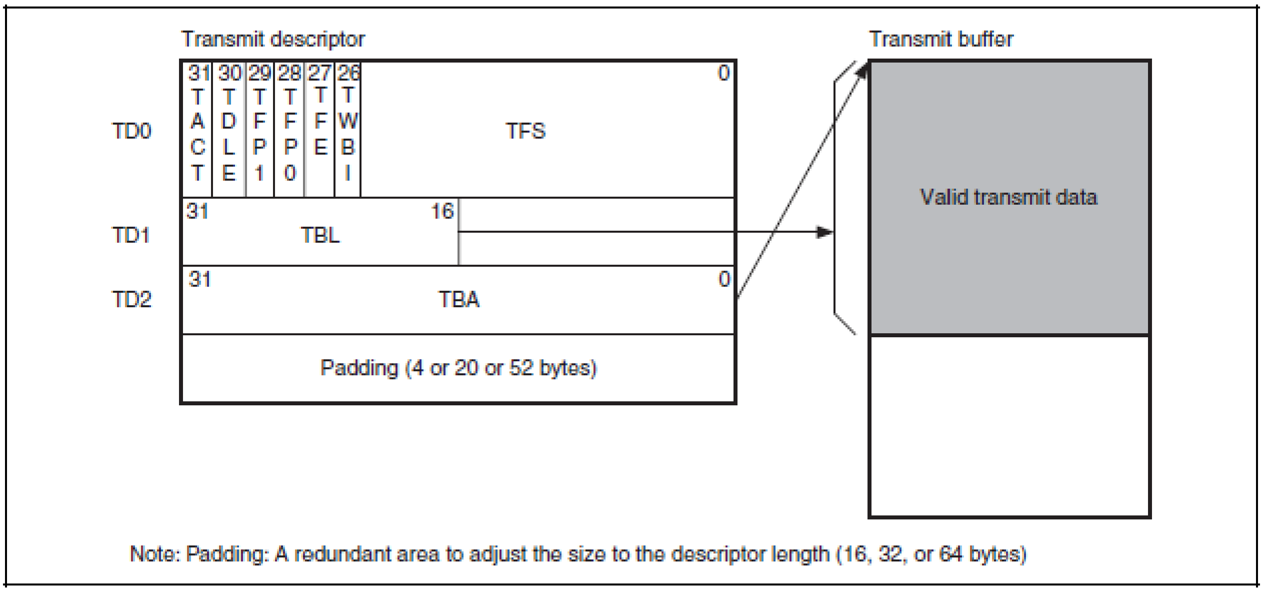

Figure 2 Transmit Descriptor

To implement zero-copy driver design, the buffer originally used by the uIP TCP/IP stack is not needed anymore. Below code snippet shows the modification to the uIP TCP/IP stack.

```
#ifndef UIP_EXTERNAL_BUFFER
u8_t uip_buf[UIP_BUFSIZE + 2];   /* Packet buffer that contains incoming packets. */
#else
u8_t *uip_buf = NULL;            /* Packet buffer pointer that points data */
#endif /* UIP_CONF_EXTERNAL_BUFFER */
```

Another modification to the original driver described in R01AN0169EU application note is that the zero-copy Ethernet driver runs only in single-frame/single-buffer mode. With 1516-byte long buffers, each Ethernet frame can be completely stored in one buffer. This removes the burden of frame segmentation and reassembly and provides an efficient mechanism to pass only buffer pointers to move data.

2.3 MAC Layer Flow Control

The driver adds MAC layer flow control when full-duxplex transfer is selected. The details of PAUSE resolution is documented in IEEE 802.3-2008 specification, Section 2, Annex 28B.

The implementation in the driver sets up the EtherC for automatic flow control and lets EtherC to receive and transmits PAUSE frames as defined in IEEE 802.3x protocol. Once PAUSE settings are complete, the operation is automatic and controlled completely by EtherC.

2.4 PHY Link Status

PHY link status is monitored by R_Ether_CheckLink_ZC() function. This function uses PHY management interface to read PHY basic status register and reports either the link is up or down.

The same information can be made available to RX family MCUs via ET_LINKSTA pin. However, this pin on the Renesas Starter Kit (RSK) is used for another purpose by default. Therefore, R_Ether_CheckLink_ZC() function is designed to get the link status through the PHY management interface.

3. Driver Details

This section covers how to use the zero-copy Ethernet driver in your application. The driver installation, configuration, its limitations, and usage details are described in the following sections.

3.1 Installing the Driver

Copy the driver files into your application. The files to copy are: phy.c, phy.h, r_ether.c, r_ether.h

Define memory sections for B_RX_DESC, B_TX_DESC, and B_ETHERNET_BUFFERS. On HEW, this is found on menu Build >> RX Standard Toolchain >> Link/Library >> Category Section. The demonstration project uses the address of 0x00001000 for this setting.

3.2 Driver Configuration

The configuration parameters for the driver and demonstration project are stored in 64 bytes of the MCU flash memory at 0xFFFFFF00. This memory section is called C_FLASH_CONFIG_PARAMS_1. Since these configurations are user specific, the section is defined in user-app.c file. The default MAC, IP addresses and netmask are defined here.

Driver specific configurations are in r_ether.c file. Ethernet buffer size, the number of receive and transmit buffers, and their sections are defined in this file.

3.3 Driver Limitations

The driver is built and tested on RSK+RX63N board using the Renesas RX Family C/C++ compiler and HEW IDE.

The driver configures a single-frame/single-buffer which requires that the buffer memory is contiguous.

There is no support for jumbo frames.

There is no support for manual PAUSE frames.

Automatic PAUSE frames are configured for a hardcoded FIFO level.

PAUSE is implemented strictly per the IEEE 802.3 standard. Inherent to this strict definition is the use of auto-negotiate. The use in legacy systems that do not perform auto-negotiate will not work with the driver, as coded, and changes to the code to force the speed, duplex and/or PAUSE configuration are not recommended.

ISR code is generic. If specific performance improvements are needed, they are best added by a developer who is knowledgeable about the synchronization of driver buffers to their application.

3.4 Using the Driver

The driver must be initialized in a specific order to ensure proper operation.

1. The Ethernet peripheral must be enabled (or, as it is sometimes referred to, taken out of standby or “module stop state is canceled”). The SYSTEM.MSTPCRB.BIT.MSTPB15 must be set to 0. This is not done by the driver functions. The RX Project Generator hwsetup.c file is one choice for performing this operation.
2. Call R_Ether_Open_ZC(). This configures the MAC address and initializes the Ethernet peripherals (EtherC and E_DMAC).
3. The function R_Ether_WaitLink_ZC() must be called at least once after R_Ether_Open_ZC(). The link partner must be capable to auto-negotiate. When a link is negotiated the configuration of the receiver and the transmitter is determined, including the configuration of duplex, speed and flow control using PAUSE frames. The Ethernet hardware will not receive or transmit Ethernet frames if this function is not called.
4. The function R_Ether_CheckLink_ZC() call may be performed to make sure if physical layer link has established. If the Ethernet cable is not plugged in “Link down” is displayed on the LCD. The same message is displayed at least once after power up since the Ethernet peripheral is reset by the driver.
5. Additional calls to R_Ether_CheckLink_ZC() are optional to determine if the link has been lost or reestablished.
6. Receive and transmit Ethernet data with calls to R_Ether_Read_ZC() or R_Ether_Write_ZC(). Either raw Ethernet frames (those without a TCP/IP stack) or Ethernet frames from a TCP/IP stack can be processed.
7. When there is no need for further Ethernet communications call R_Ethernet_Close_ZC(). This function will disable the Ethernet receiver and transmitter.

4. Updates to uIP TCP/IP Demonstration Project

The uIP TCP/IP demonstration project behaves similar as in R01AN0169EU application note. RSK board either receives its IP address from a DHCP server or uses its default setting of 192.168.1.10. Make sure Ethernet cables are connected and devices are powered up if “Link down” is displayed continuously 10 seconds after power on reset. Some of the possible LCD settings are shown below.

“Link down” is displayed when the Ethernet cable is disconnected from the RSK board. When cable is connected back, DHCP client tries to get a new IP address from a DHCP server. If a DHCP server is not found, the demonstration project uses it default IP address.

Web server demonstration is similar to the original release in R01AN0169EU application note. One of the notable changes is to reduce the Web page size of RSK custom page. The size of the image on this page is reduced so that the demonstration project can still be built with an evaluation version of the Renesas RX Family C/C++ compiler. This Web page is shown in Figure 3.

All other pages can be accessed by links provided in the top banner. The file statistics page shows the number of times a specific page is accessed. The network statistics page displays the number of IP, ICMP, and TCP packet reception and transmission information. The network connections page shows the current status of established TCP connections within the uIP stack. These pages are dynamic and recreated every time when they are accessed.

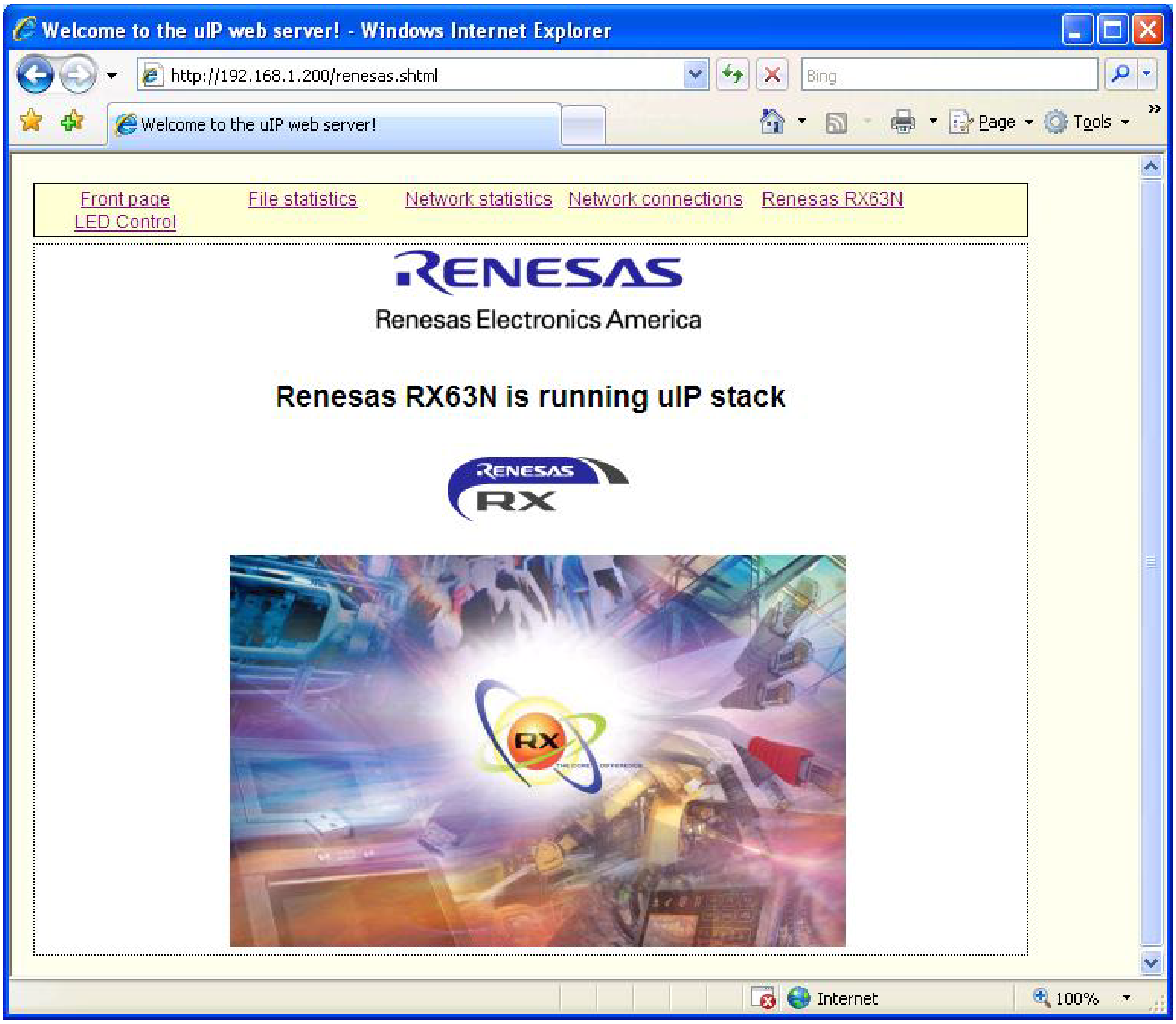

Figure 3 RSK Custom Page

5. Renesas Ethernet Driver APIs

The driver APIs that comes with this application note have _ZC suffix to differentiate them from the ones in R01AN0169EU uIP TCP/IP Protocol Stack Demonstration application note. The API functions are very similar except some of the parameters used. However, zero-copy APIs should not be used with the original ones and each set of driver APIs should be used as a unique group of functions.

5.1 Renesas Unique Functions

The following functions serve as the zero-copy driver interface to the uIP TCP/IP stack.

R_Ether_Open_ZC

R_Ether_Close_ZC

R_Ether_Read_ZC

R_Ether_Write_ZC

R_Ether_CheckLink_ZC

R_Ether_WaitLink_ZC

R_Ether_Open_ZC

The R_Ether_Open_ZC function performs EtherC and E-DMAC peripheral, physical device, and transmit and receive data buffer initializations. The EtherC and E-DMAC are powered up separately as part of power on reset initialization.

Format

```
int32_t R_Ether_Open_ZC(uint32_t ch, uint8_t mac_addr[], void **buf);
```

Parameters

ch

Specifies the EtherC channel number.

mac_addr

Specifies the MAC address of EtherC.

buf

Points to the buffer pointer used by the stack.

Return Values

R_ETHER_OK(0)
R_ETHER_ERROR(-1)

Properties

Prototyped in file r_ether.h
Implemented in file r_ether.c

Description

The R_Ether_Open_ZC function initializes the EtherC and E-DMAC subsystems. E-DMAC descriptors and buffers are setup for the initial use. The MAC address is used to initialize MAC address registers in EtherC.

The pointer to buffer pointer is initialized with the first available transmit buffer. This provides a data buffer to the stack for transmitting data.

By default, the physical device is configured to auto-negotiate mode.

If there is only one Ethernet channel, then channel number is set to zero. If there are two Ethernet channels, then 0 and 1 are used for channel number.

R_Ether_Close_ZC

The R_Ether_Close_ZC function disables transmit and receive functionality of EtherC peripheral. This function does not power down EtherC and E-DMAC peripherals.

Format

```
int32_t R_Ether_Close_ZC(uint32_t ch);
```

Parameters

ch

Specifies the EtherC channel number.

Return Values

R_ETHER_OK(0)
R_ETHER_ERROR(-1)

Properties

Prototyped in file r_ether.h
Implemented in file r_ether.c

Description

The R_Ether_Close_ZC function disables transmit and receive functionality of the EtherC peripheral

If there is only one Ethernet channel, then channel number is set to zero. If there are two Ethernet channels, then 0 and 1 are used for channel number.

R_Ether_Read_ZC

The R_Ether_Read_ZC function receives data into application receive buffer.

Format

```
int32_t R_Ether_Read_ZC(uint32_t ch, void **buf);
```

Parameters

ch

Specifies the EtherC channel number.

buf

Points to the buffer pointer used by the stack.

Return Values

Returns the number of bytes received. A zero value indicates no data is received.

Properties

Prototyped in file r_ether.h
Implemented in file r_ether.c

Description

The driver's buffer pointer of the read data is returned in the parameter *buf*. Returning the pointer allows the operation to be performed with zero-copy. Return value shows the number of received bytes. If there is no data is available at the time of the call, a zero value is returned.

The E-DMAC hardware operates independent of the R_Ether_Read_ZC function and reads data off the Ethernet link into a buffer pointed by the E-DMAC receive descriptor. It updates the status of the receive descriptor as new data is processed. The buffer pointed to by the E-DMAC receive descriptor is statically allocated by the driver.

If there is only one Ethernet channel, then channel number is set to zero. If there are two Ethernet channels, then 0 and 1 are used for channel number.

R_Ether_Write_ZC

The R_Ether_Write_ZC function transmits data from application transmit buffer.

Format

```
int32_t R_Ether_Write_ZC(uint32_t ch, void **buf, uint32_t len);
```

Parameters

ch

Specifies the EtherC channel number.

buf

Points to the buffer pointer used by the stack.

len

Ethernet frame length.

Return Values

R_ETHER_OK(0)
R_ETHER_ERROR(-1)

Properties

Prototyped in file r_ether.h
Implemented in file r_ether.c

Description

The R_Ether_Write_ZC function moves transmit data to a buffer pointed by the transmit E-DMAC descriptor. It updates the status of the transmit descriptor as new data is processed. Data written is transmitted by EtherC.

For an application (or stack) that uses a single buffer for transmitting and receiving data (e.g. uIP TCP/IP stack), _R_Ether_Swap_Buffers() function must be called within this API.

If there is only one Ethernet channel, then channel number is set to zero. If there are two Ethernet channels, then 0 and 1 are used for channel number.

R_Ether_CheckLink_ZC

The R_Ether_CheckLink_ZC function checks the status the physical Ethernet link using PHY management interface. Ethernet link is present when the cable is connected to a peer device that has properly initialized its PHY.

Format

```
int32_t R_Ether_CheckLink_ZC(uint32_t ch);
```

Parameters

ch

Specifies the EtherC channel number.

Return Values

R_ETHER_OK(0)
R_ETHER_ERROR(-1)

Properties

Prototyped in file r_ether.h
Implemented in file r_ether.c

Description

The R_Ether_CheckLink_ZC function uses PHY management interface to determine the status of the Ethernet link. This information is read from the basic status register of the PHY device. The value R_ETHER_OK is returned when a link is present otherwise the value R_ETHER_ERROR is returned.

If there is only one Ethernet channel, then channel number is set to zero. If there are two Ethernet channels, then 0 and 1 are used for channel number.

R_Ether_WaitLink_ZC

The R_Ether_WaitLink_ZC function determines that the partner PHY is operating on the link through the auto-negotiation transactions. The link abilities are handled to determine duplex, speed and flow control (PAUSE frames).

Format

```
int32_t R_Ether_WaitLink_ZC(uint32_t ch);
```

Parameters

ch

Specifies the EtherC channel number.

Return Values

R_ETHER_OK(0)
R_ETHER_ERROR(-1)

Properties

Prototyped in file r_ether.h
Implemented in file r_ether.c

Description

The R_Ether_WaitLink_ZC function performs link partner communications to determine that the link is good between the local circuitry and the remote circuitry. The local link has capabilities that are transmitted to the remote link as part of the auto-negotiation transactions. The remote link capabilities are transmitted to the local link. The capabilities of the partner and local link are then evaluated to determine the correct configuration for duplex, link speed and flow control. The final action of the function is to enable the transmitter and receiver.

This function must be successfully called to establish a working link.

If there is only one Ethernet channel, then channel number is set to zero. If there are two Ethernet channels, then 0 and 1 are used for channel number.

References

1. RX63N Group User's Manual: Hardware. Renesas 32-Bit MCU, RX Family / RX600 Series, Rev.1.00, Mar 2012.
2. uIP TCP/IP Protocol Stack Demonstration, Renesas Application Note, R01AN0169EU.
3. The uIP Embedded TCP/IP Stack, The uIP 1.0 Reference Manual, June 2006, Adam Dunkels, Swedish Institute of Computer Science
4. IEEE 802.3 Ethernet, IEEE Standards Association
5. HTTP – Hypertext Transfer Protocol. World Wide Web Consortium
6. RFC 2131 Dynamic Host Configuration Protocol, IETF
7. RFC 2132 DHCP Options and BOOTP Vendor Extensions, IETF
8. IEEE 802.3 Section 2, 802.3-2008_section2.pdf

Website and Support

Renesas Electronics Website
http://www.renesas.com/

Inquiries
http://www.renesas.com/inquiry

Revision Record

Rev.	Date	Description	
		Page	Summary
1.00	Jun.01.12	—	First edition issued

General Precautions in the Handling of MPU/MCU Products

The following usage notes are applicable to all MPU/MCU products from Renesas. For detailed usage notes on the products covered by this document, refer to the relevant sections of the document as well as any technical updates that have been issued for the products.

1. Handling of Unused Pins

   Handle unused pins in accord with the directions given under Handling of Unused Pins in the manual.

   — The input pins of CMOS products are generally in the high-impedance state. In operation with an unused pin in the open-circuit state, extra electromagnetic noise is induced in the vicinity of LSI, an associated shoot-through current flows internally, and malfunctions occur due to the false recognition of the pin state as an input signal become possible. Unused pins should be handled as described under Handling of Unused Pins in the manual.

2. Processing at Power-on

   The state of the product is undefined at the moment when power is supplied.

   — The states of internal circuits in the LSI are indeterminate and the states of register settings and pins are undefined at the moment when power is supplied.
   In a finished product where the reset signal is applied to the external reset pin, the states of pins are not guaranteed from the moment when power is supplied until the reset process is completed.
   In a similar way, the states of pins in a product that is reset by an on-chip power-on reset function are not guaranteed from the moment when power is supplied until the power reaches the level at which resetting has been specified.

3. Prohibition of Access to Reserved Addresses

   Access to reserved addresses is prohibited.

   — The reserved addresses are provided for the possible future expansion of functions. Do not access these addresses; the correct operation of LSI is not guaranteed if they are accessed.

4. Clock Signals

   After applying a reset, only release the reset line after the operating clock signal has become stable. When switching the clock signal during program execution, wait until the target clock signal has stabilized.

   — When the clock signal is generated with an external resonator (or from an external oscillator) during a reset, ensure that the reset line is only released after full stabilization of the clock signal. Moreover, when switching to a clock signal produced with an external resonator (or by an external oscillator) while program execution is in progress, wait until the target clock signal is stable.

5. Differences between Products

   Before changing from one product to another, i.e. to a product with a different part number, confirm that the change will not lead to problems.

   — The characteristics of an MPU or MCU in the same group but having a different part number may differ in terms of the internal memory capacity, layout pattern, and other factors, which can affect the ranges of electrical characteristics, such as characteristic values, operating margins, immunity to noise, and amount of radiated noise. When changing to a product with a different part number, implement a system-evaluation test for the given product.

SALES OFFICES Renesas Electronics Corporation http://www.renesas.com

Refer to "http://www.renesas.com/" for the latest and detailed information.

Renesas Electronics America Inc.
2880 Scott Boulevard Santa Clara, CA 95050-2554, U.S.A.
Tel: +1-408-588-6000, Fax: +1-408-588-6130

Renesas Electronics Canada Limited
1101 Nicholson Road, Newmarket, Ontario L3Y 9C3, Canada
Tel: +1-905-898-5441, Fax: +1-905-898-3220

Renesas Electronics Europe Limited
Dukes Meadow, Millboard Road, Bourne End, Buckinghamshire, SL8 5FH, U.K
Tel: +44-1628-585-100, Fax: +44-1628-585-900

Renesas Electronics Europe GmbH
Arcadiastrasse 10, 40472 Düsseldorf, Germany
Tel: +49-211-65030, Fax: +49-211-6503-1327

Renesas Electronics (China) Co., Ltd.
7th Floor, Quantum Plaza, No.27 ZhiChunLu Haidian District, Beijing 100083, P.R.China
Tel: +86-10-8235-1155, Fax: +86-10-8235-7679

Renesas Electronics (Shanghai) Co., Ltd.
Unit 204, 205, AZIA Center, No.1233 Lujiazui Ring Rd., Pudong District, Shanghai 200120, China
Tel: +86-21-5877-1818, Fax: +86-21-6887-7858 / -7898

Renesas Electronics Hong Kong Limited
Unit 1601-1613, 16/F., Tower 2, Grand Century Place, 193 Prince Edward Road West, Mongkok, Kowloon, Hong Kong
Tel: +852-2886-9318, Fax: +852 2886-9022/9044

Renesas Electronics Taiwan Co., Ltd.
13F, No. 363, Fu Shing North Road, Taipei, Taiwan
Tel: +886-2-8175-9600, Fax: +886 2-8175-9670

Renesas Electronics Singapore Pte. Ltd.
1 harbourFront Avenue, #06-10, keppel Bay Tower, Singapore 098632
Tel: +65-6213-0200, Fax: +65-6278-8001

Renesas Electronics Malaysia Sdn.Bhd.
Unit 906, Block B, Menara Amcorp, Amcorp Trade Centre, No. 18, Jln Persiaran Barat, 46050 Petaling Jaya, Selangor Darul Ehsan, Malaysia
Tel: +60-3-7955-9390, Fax: +60-3-7955-9510

Renesas Electronics Korea Co., Ltd.
11F., Samik Lavied' or Bldg., 720-2 Yeoksam-Dong, Kangnam-Ku, Seoul 135-080, Korea
Tel: +82-2-558-3737, Fax: +82-2-558-5141

Colophon 1.1