17 Emac

17.1 Emac introduction

The EMAC module is a 10/100Mbps Ethernet MAC (Ethernet Media Access Controller) compatible with IEEE 802.3. It includes status and control register group, transceiver module, transceiver buffer descriptor group, host interface, MDIO, physical layer chip (PHY) interface.

The status and control register group contains the status bits and control bits of the EMAC, which is the interface with the user program, and is responsible for controlling data receiving and sending and reporting the status.

The transceiver module is responsible for obtaining the data frame from the designated memory location according to the control word in the transceiver descriptor, adding the preamble, CRC, and expanding the short frame before sending it out through the PHY; Or receive data from the PHY, and put the data into the designated memory according to the transmit and receive buffer descriptor. Configure related event flags after sending and receiving. If the event interrupt is enabled, the interrupt request will be sent to the host for processing.

The MDIO and MII/RMII interfaces are responsible for communicating with the PHY, including reading and writing PHY registers and sending and receiving data packets.

17.2 Emac main features

- Compatible with the MAC layer functions defined by IEEE 802.3
- Support MII and RMII interface PHY defined by IEEE 802.3
- Interaction with PHY through MDIO
- Support 10Mbps and 100Mbps Ethernet
- Support half duplex and full duplex
- In full duplex mode, support automatic flow control and control frame generation
- Support collision detection and retransmission in half-duplex mode

- Support CRC generation and verification
- Data frame preamble generation and removal
- When sending, automatically expand short data frames
- Detect too long or too short data frame (length limit)
- Can transmit long data frames (> standard Ethernet frame length)
- Automatically discard packets that exceed the number of retransmissions or the frame gap is too small
- Broadcast packet filtering
- Internal RAM for storing up to 128 BD (Buffer Descriptor)
- Various event flags sent/received
- Generate a corresponding interrupt when an event occurs

17.3 Emac function description

The composition of the EMAC module is shown in the figure below.

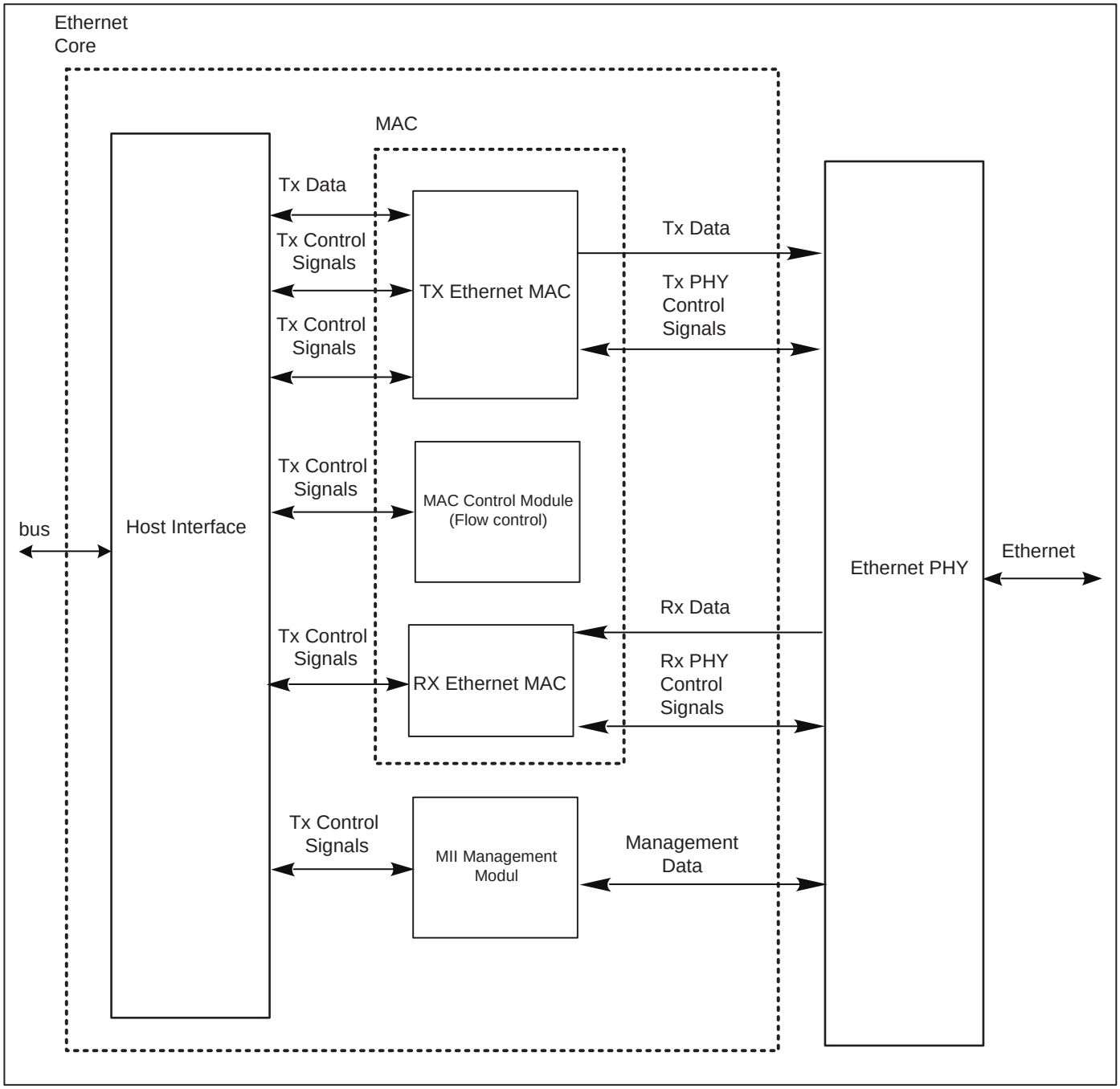


Fig. 17.1: EMAC architecture

The module’s control register can read and write the PHY register through MDIO to realize configuration, select mode (half/full duplex), initiate negotiation and other operations.

The receiving module filters and checks the received data frame: whether there is a legal preamble, FCS, length, etc. And according to the descriptor, the data frame is stored in the designated buffer address.

The sending module obtains data from the memory according to the data buffer descriptor, adds preamble, FCS, pad, etc., and then sends the data according to the CSMA/CD protocol.

If CRS is detected, the retry will be delayed.

The send and receive buffer descriptor group is connected to the external RAM, which is used to store the Ethernet data frames sent and received. Each descriptor contains the corresponding control status word and the corresponding buffer memory address. There are 128 groups of descriptors, which can be flexibly allocated for sending or receiving.

17.4 Emac clock

The EMAC module needs a clock for synchronous transmission and reception (at 100Mbps, 25MHz (MII) or 50MHz (RMII); at 10Mbps, 2.5MHz). This clock must be synchronized between EMAC and PHY.

17.5 Send and receive buffer descriptor (BD, Buffer Descriptor)

The transceiver buffer descriptor is used to provide the association between the EAMC and the data frame buffer address information, to control the transceiver data frame, and to provide the transceiver status prompt.

Each descriptor is composed of two consecutive words (32 bits). The low address word provides the length, control and status bits of the data frame contained in the buffer; the high address word is a memory pointer.

Specific BD description can refer to the register description chapter.

Note that: For BD, you need to write in word.

The EMAC module supports 128 BDs, which are shared by the sending/receiving logic and can be freely combined. But the sending BD always occupies the previous continuous area (the number is specified by the TXBDNUM field in the MAC_TX_BD_NUM register).

EMAC processes the sending/receiving BD in the order of BD, until it encounters the BD marked as WR, it wraps around to send/receive the respective first BD.

17.6 PHY interaction

The PHY interaction register group provides the commands and data communication methods needed for PHY interaction. EMAC controls the working mode of PHY through MDIO and ensures that the two match (rate, full/half duplex).

The data packet interacts between EMAC and PHY through the MII/RMII interface, and can be selected by RMII_EN in the EMAC mode register (EMAC_MODE). When this bit is 1, select RMII mode, otherwise it is MII mode.

Both MII and RMII modes support the 10Mbps and 100Mbps transmission rates specified in the IEEE 802.3u standard.

The transmission signal description of MII and RMII is shown in the table below.

Table 17.1: Transmission signal

Name	MII	RMII
EXTCK_EREFCK	ETXCK: send clock signal	EREFCK: reference clock
ECRS	ECRS: carrier detection	-
ECOL	ECOL: collision detection	-
ERXDV	ERXDV: data valid	ECRSDV: Carrier detect/data valid
ERX0-ERX3	ERX0-ERX3: 4-bit receive data	ERX0-ERX1: 2-bit receive data
ERXER	ERXER: Receive error indication	ERXER: Receive error indication
ERXCK	ERXCK: Receive clock signal	-
ETXEN	ETXEN: transmit enable	ETXEN: transmit enable
ETX0-ETX3	ETX0-ETX3: 4-bit transmit data	ETX0-ETX1: 2-bit transmit data
ETXER	ETXER: Send error indication	-
EMDC	MDIO Clock	MDIO Clock
EMDIO	MDIO Data Input Output	MDIO Data Input Output

The RMII interface has fewer pins, and a 2-bit data line is used for receiving and sending. At a rate of 100Mbps, a 50MHz reference clock is required.

17.7 Programming process

17.7.1 PHY initialization

- According to the PHY type, set the RMII_EN bit in the EMAC_MODE register to select the appropriate connection method
- Set the MAC address of EMAC to EMAC_MAC_ADDR0 and EMAC_MAC_ADDR1
- Set the appropriate clock for the MDIO part by programming the field CLKDIV in the EMAC_MIIMODE register
- Set the corresponding PHY address to the FIAD field of the register EMAC_MIIADDRESS
- According to the PHY manual, send commands through the EMAC_MIICOMMAND and EMAC_MIITX_DATA registers
- The data read from the PHY will be stored in the EMAC_MIIRX_DATA register
- The status of interaction with PHY commands can be queried through the EMAC_MIISTATUS register

After the basic interaction is completed, the PHY should enter the auto-negotiation state. After the negotiation is completed, program the mode to the FULLD bit in the EMAC_MODE register according to the negotiation result.

17.7.2 Send data frame

- Configure bit fields such as data frame format and interval in the EMAC_MODE register
- By configuring the TXBDNUM field in the EMAC_TX_BD_NUM register to specify the number of BDs used for transmission, the remaining BDs are RX BDs
- Prepare the data frame to be sent in the memory
- Fill in the address of the data frame into the data pointer field (word 1) corresponding to the sent BD
- Clear the status flag in the control and status field (wrod 0) corresponding to the sent BD, and set the control field (CRC enable, PAD enable, interrupt enable, etc.)
- Write the length of the data frame and set the RD field to inform EMAC that this BD data needs to be sent; if necessary, set the IRQ bit to enable interrupts
- In particular, if it is the last BD sent, the WR bit needs to be set, and EMAC will "wrap around" to the first sent BD for processing after processing this BD
- If there are multiple BDs to be sent, repeat the steps of setting BD to fill all sending BDs
- If you need to enable the transmit interrupt, you also need to configure the TX related bits in the EMAC_INT_MASK register
- Configure the TXEN bit in the EMAC_MODE register to enable transmission
- If the interrupt is enabled, in the interrupt sent, the current BD can be obtained through the TXBDNUM field in the EMAC_TX_BD_NUM register
- Perform corresponding processing according to the current BD status word
- For the BD whose data has been sent, the RD bit in the control field will be cleared by hardware and will not be sent again; after filling in new data, set RD, and this BD can be used for sending again

17.7.3 Receive data frame

- Configure bit fields such as data frame format and interval in the EMAC_MODE register
- By configuring the TXBDNUM field in the EMAC_TX_BD_NUM register to specify the number of BDs used for transmission, the remaining BDs are RX BDs
- The area in the memory that is ready to receive data
- Fill in the address of the data frame into the data pointer field (word 1) corresponding to the received BD
- Clear the status flag in the control and status field (wrod 0) corresponding to the sending BD, and set the control field (interrupt enable, etc.)
- Write the receivable data frame length and set the E bit field to inform EMAC that the BD is idle and can be used for data reception; if necessary, set the IRQ bit to enable interrupts

- In particular, if it is the last valid receiving BD, the WR bit needs to be set, and EMAC will “wrap around” to the first receiving BD for processing after processing this BD
- If there are multiple BDs available to receive data, repeat the steps of setting BD to fill all BDs
- If you need to enable the receive interrupt, you also need to configure the RX related bits in the EMAC_INT_MASK register
- Configure the RXEN bit in the EMAC_MODE register to enable reception
- If the interrupt is enabled, in the received interrupt, the current BD can be obtained through the RXBDNUM field in the EMAC_TX_BD_NUM register
- Perform corresponding processing according to the current BD status word
- For the received BD, the E bit in the control field will be cleared by hardware and will not be used for receiving again; the data needs to be taken away, and E is set, this BD can be used for receiving again

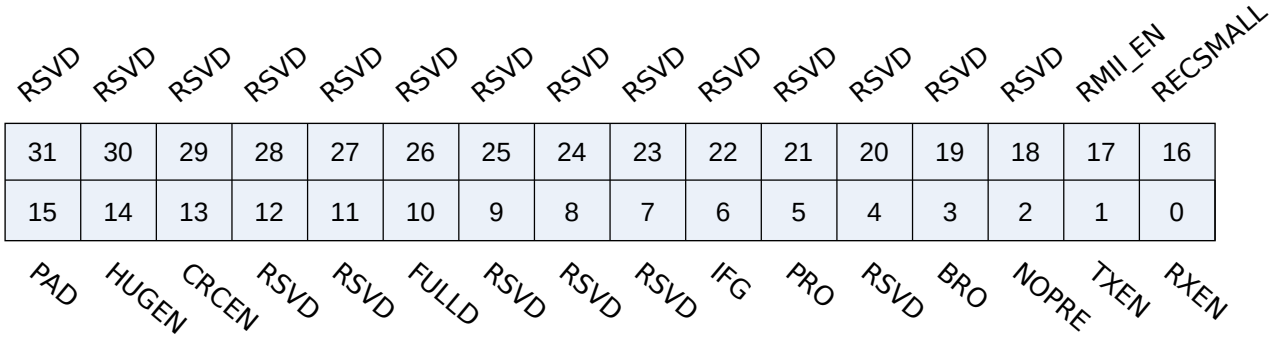
17.8 Register description

Name	Description
MODE	EMAC configuration
INT_SOURCE	EMAC transmit control
INT_MASK	EMAC interrupt mask
IPGT	Inter packet gap
PACKETLEN	Frame length
COLLCONFIG	Collision configuration
TX_BD_NUM	TX buffer descriptors number
MIIMODE	Management Data configuration
MIICOMMAND	Trigger command
MIIADDRESS	Register address
MIITX_DATA	Control data to be written to PHY
MIIRX_DATA	Received data from PHY
MIISTATUS	MIIM I/F status
MAC_ADDR0	Ethernet MAC address0
MAC_ADDR1	Ethernet MAC address1
HASH0_ADDR	Lower 32-bit of HASH register

Name	Description
HASH1_ADDR	Upper 32-bit of HASH register
TXCTRL	TX control

17.8.1 MODE

Address：0x4000d000

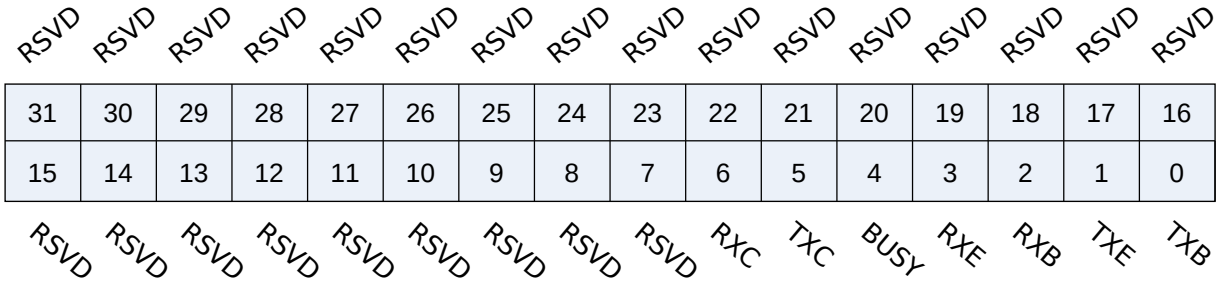


Bits	Name	Type	Reset	Description
31:18	RSVD			
17	RMII_EN	r/w	1'b0	RMII mode enable 0: MII PHY I/F is used 1: RMII PHY I/F is used
16	RECSMALL	r/w	1'b0	Receive small frame enable 0: Frames smaller than MINFL are ignored. 1: Frames smaller than MINFL are accepted.
15	PAD	r/w	1'b1	Padding enable 0: Do not add pads to frames shorter than MINFL. 1: Add pads to short frames, until the length equals MINFL.
14	HUGEN	r/w	1'b0	Huge frames enable 0: The maximum frame length is MAXFL. All additional bytes are dropped. 1: Frame size is not limited by MAXFL and can be up to 64K bytes.
13	CRCEN	r/w	1'b1	CRC Enable 0: TX MAC does not append CRC field. 1: TX MAC will append CRC field to every frame.
12:11	RSVD			

Bits	Name	Type	Reset	Description
10	FULLD	r/w	1'b0	Full duplex 0: Half duplex mode. 1: Full duplex mode.
9:7	RSVD			
6	IFG	r/w	1'b0	Inter frame gap check 0: IFG is verified before each frame be received. 1: All frames are received regardless to IFG requirement.
5	PRO	r/w	1'b0	Promiscuous mode enable 0: The destination address is checked before receiving. 1: All frames received regardless of the address.
4	RSVD			
3	BRO	r/w	1'b1	Broadcast address enable 0: Reject all frames containing the broadcast address unless the PRO bit is asserted. 1: Receive all frames containing broadcast address.
2	NOPRE	r/w	1'b0	No preamble mode 0: 7-byte preamble will be sent. 1: No preamble will be sent.
1	TXEN	r/w	1'b0	Transmit enable 0: Transmitter is disabled. 1: Transmitter is enabled. If TX_BD_NUM equals 0x0 (zero buffer descriptors are used), then the transmitter is disabled regardless of TXEN.
0	RXEN	r/w	1'b0	Receiver enable 0: Receiver is disabled. 1: Receiver is enabled. If TX_BD_NUM equals 0x80 (all buffer descriptors are used for TX), then the receiver is disabled regardless of RXEN.

17.8.2 INT_SOURCE

Address: 0x4000d004

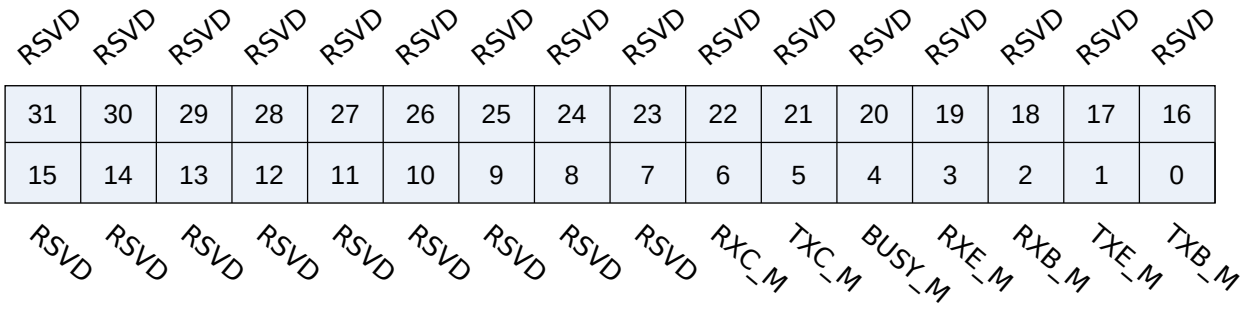


Bits	Name	Type	Reset	Description
31:7	RSVD			
6	RXC	r/w	1'b0	Receive control frame This bit indicates that the control frame was received. It is cleared by writing 1 to it. Bit RXFLOW in the CTRLMODE register must be set to 1 in order to get the RXC bit set.
5	TXC	r/w	1'b0	Transmit control frame This bit indicates that a control frame was transmitted. It is cleared by writing 1 to it. Bit TXFLOW in the CTRLMODE register must be set to 1 in order to get the TXC bit set.
4	BUSY	r/w	1'b0	Busy This bit indicates that RX packet is being received and there is no empty buffer descriptor to use. It iscleared by writing 1 to it. This bit appears regardless to the IRQ bits in the Receive Buffer Descriptor.
3	RXE	r/w	1'b0	Receive error This bit indicates that an error occurred while receiving data (overrun, receiver error, dribble nibble, too long, >64K, CRC error, bus error or late collision. It is cleared by writing 1 to it. This bit appears only when IRQ bit is set in the Receive Buffer Descriptor.
2	RXB	r/w	1'b0	Receive frame This bit indicates that a frame was received. It is cleared by writing 1 to it. This bit appears only when IRQ bit is set in the Receive Buffer Descriptor.

Bits	Name	Type	Reset	Description
1	TXE	r/w	1'b0	Transmit error This bit indicates that a buffer was not transmitted due to a transmit error (underrun, retransmission limit, late collision, bus error or defer time-out). It is cleared by writing 1 to it. This bit appears only when IRQ bit is set in the Transmit Buffer Descriptor.
0	TXB	r/w	1'b0	Transmit buffer This bit indicates that a buffer has been transmitted. It is cleared by writing 1 to it. This bit appears only when IRQ bit is set in the Transmit Buffer Descriptor.

17.8.3 INT_MASK

Address: 0x4000d008

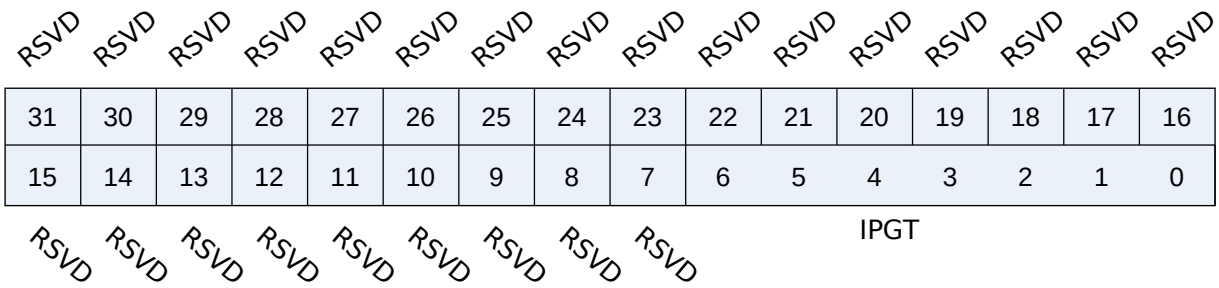


Bits	Name	Type	Reset	Description
31:7	RSVD			
6	RXC_M	r/w	1'b1	Receive control frame mask ENABLE 0: Interrupt is un-masked 1: Interrupt is masked
5	TXC_M	r/w	1'b1	Transmit control frame mask ENABLE 0: Interrupt is un-masked 1: Interrupt is masked
4	BUSY_M	r/w	1'b1	Busy mask ENABLE 0: Interrupt is un-masked 1: Interrupt is masked

Bits	Name	Type	Reset	Description
3	RXE_M	r/w	1'b1	Receive error mask ENABLE 0: Interrupt is un-masked 1: Interrupt is masked
2	RXB_M	r/w	1'b1	Receive frame mask ENABLE 0: Interrupt is un-masked 1: Interrupt is masked
1	TXE_M	r/w	1'b1	Transmit error mask ENABLE 0: Interrupt is un-masked 1: Interrupt is masked
0	TXB_M	r/w	1'b1	Transmit buffer mask ENABLE 0: Interrupt is un-masked 1: Interrupt is masked

17.8.4 IPGT

Address: 0x4000d00c



Bits	Name	Type	Reset	Description
31:7	RSVD			
6:0	IPGT	r/w	7'h18	Inter packet gap The recommended value is 0x18 (24 clock cycles), which equals 9.6 us for 10 Mbps and 0.96 us for 100 Mbps mode

17.8.5 PACKETLEN

Address: 0x4000d018

MINFL

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

MAXFL

Bits	Name	Type	Reset	Description
31:16	MINFL	r/w	16'h40	Minimum frame length The minimum Ethernet packet is 64 bytes long (0x40). To receive small packets, assert the RECSMALL bit or change the MINFL value. To transmit small packets, assert the PAD bit or change the MINFL value.
15:0	MAXFL	r/w	16'h600	Maximum frame length The maximum Ethernet packet is 1518 bytes long. To support this and to have some additional space for tags, a default maximum packet length equals to 1536 bytes (0x600). For bigger packets, you can assert the HUGEN bit or increase the value of MAXFL field.

17.8.6 COLLCONFIG

Address: 0x4000d01c

RSVD RSVD RSVD RSVD RSVD RSVD RSVD RSVD RSVD RSVD RSVD RSVD MAXRET

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

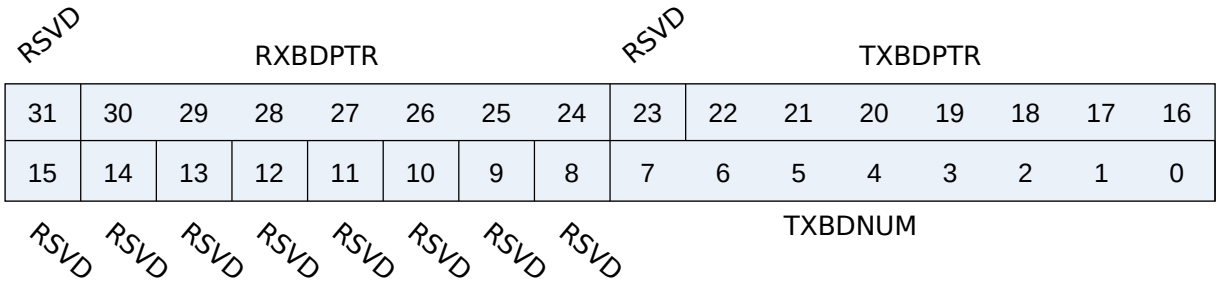
RSVD RSVD RSVD RSVD RSVD RSVD RSVD RSVD RSVD RSVD COLLVALID

Bits	Name	Type	Reset	Description
31:20	RSVD			

Bits	Name	Type	Reset	Description
19:16	MAXRET	r/w	4'hF	Maximum retry This field specifies the maximum number of consequential retransmission attempts after the collision is detected. When the maximum number has been reached, the TX MAC reports an error and stops transmitting the current packet. According to the Ethernet standard, the MAXRET default value is set to 0xf (15).
15:6	RSVD			
5:0	COLLVALID	r/w	6'h3F	Collision valid This field specifies a collision time window. A collision that occurs later than the time window is reported as a "Late Collisions" and transmission of the current packet is aborted.

17.8.7 TX_BD_NUM

Address: 0x4000d020

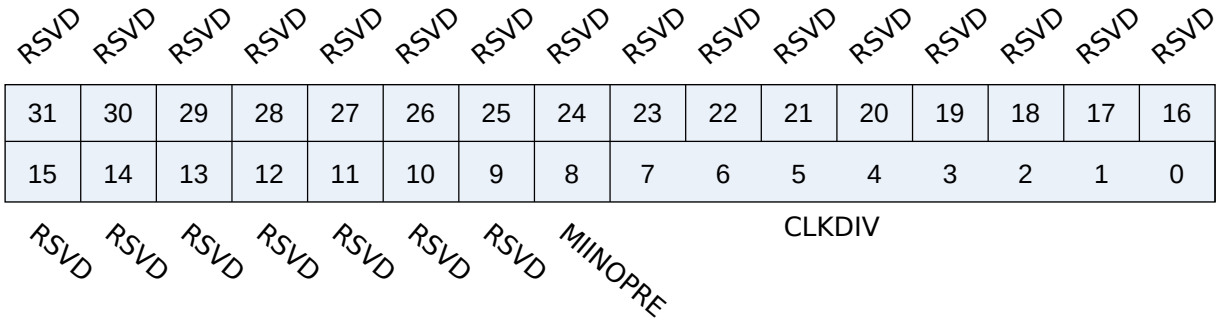


Bits	Name	Type	Reset	Description
31	RSVD			
30:24	RXBDPTR	r	7'h0	RX buffer descriptors (BD) pointer, pointing at the RXBD currently being used
23	RSVD			
22:16	TXBDPTR	r	7'h0	TX buffer descriptors (BD) pointer, pointing at the TXBD currently being used
15:8	RSVD			

Bits	Name	Type	Reset	Description
7:0	TXBDNUM	r/w	8'h40	TX buffer descriptors (BD) number Number of TX BD. TX and RX share 128 (0x80) descriptors, so the number of RX BD equals 0x80 - TXBDNUM. The maximum number of TXBDNUM is 0x80. Values greater then 0x80 cannot be written into this register.

17.8.8 MIIMODE

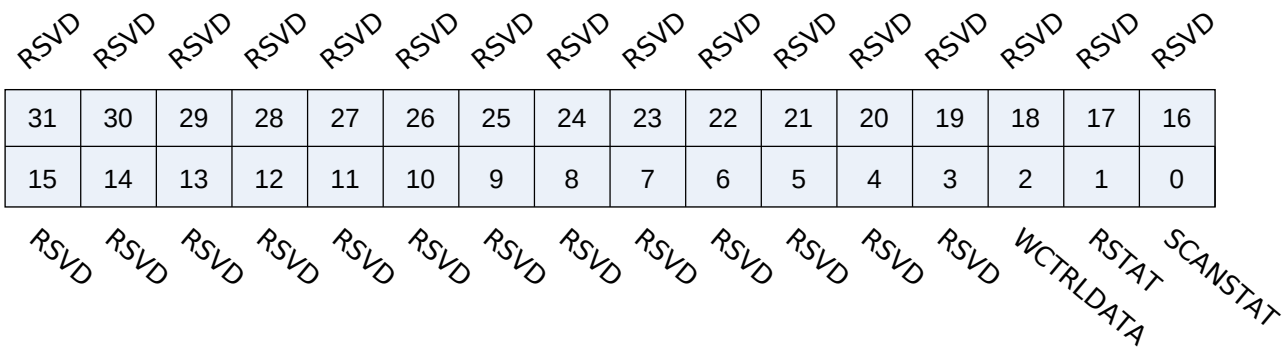
Address: 0x4000d028



Bits	Name	Type	Reset	Description
31:9	RSVD			
8	MIINOPRE	r/w	1'b0	No preamble for Management Data (MD) 0: 32-bit preamble will be sent. 1: No preamble will be sent.
7:0	CLKDIV	r/w	8'h64	Clock divider for Management Data Clock (MDC) The source clock is bus clock and can be divided by any even number.

17.8.9 MIICOMMAND

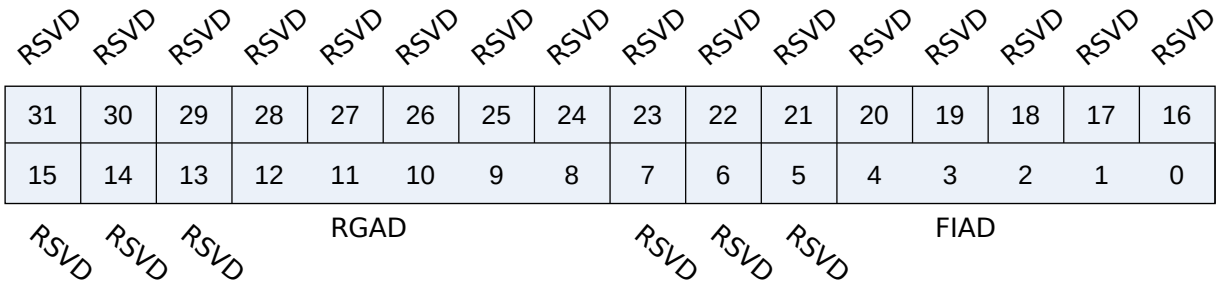
Address: 0x4000d02c



Bits	Name	Type	Reset	Description
31:3	RSVD			
2	WCTRLDATA	r/w	1'b0	Write control data, setting this bit to 1 will trigger the command (auto cleared) Note: [2]/[1]/[0] cannot be asserted at the same time, execute one command at a time
1	RSTAT	r/w	1'b0	Read status, setting this bit to 1 will trigger the command (auto cleared) Note: [2]/[1]/[0] cannot be asserted at the same time, execute one command at a time
0	SCANSTAT	r/w	1'b0	Scan status, setting this bit to 1 will trigger the command (auto cleared) Note: [2]/[1]/[0] cannot be asserted at the same time, execute one command at a time

17.8.10 MIIADDRESS

Address：0x4000d030



Bits	Name	Type	Reset	Description
31:13	RSVD			
12:8	RGAD	r/w	5'h0	Register Address
7:5	RSVD			
4:0	FIAD	r/w	5'h0	PHY Address

17.8.11 MIITX_DATA

Address: 0x4000d034

RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CTRLDATA															

Bits	Name	Type	Reset	Description
31:16	RSVD			
15:0	CTRLDATA	r/w	16'h0	Control Data to be written to PHY

17.8.12 MIIRX_DATA

Address: 0x4000d038

RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PRSD															

Bits	Name	Type	Reset	Description
31:16	RSVD			
15:0	PRSD	r	16'h0	Received Data from PHY

17.8.13 MIISTATUS

Address: 0x4000d03c

RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	MIIM_BUSY	MIIM_LINKFAIL

Bits	Name	Type	Reset	Description
31:2	RSVD			
1	MIIM_BUSY	r	1'b0	MIIM I/F busy signal 0: The MIIM I/F is ready. 1: The MIIM I/F is busy.
0	MIIM_LINKFAIL	r	1'b0	MIIM I/F link fail signal

17.8.14 MAC_ADDR0

Address: 0x4000d040

MAC_B2								MAC_B3							
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MAC_B4								MAC_B5							

Bits	Name	Type	Reset	Description
31:24	MAC_B2	r/w	8'd0	Ethernet MAC address byte 2
23:16	MAC_B3	r/w	8'd0	Ethernet MAC address byte 3
15:8	MAC_B4	r/w	8'd0	Ethernet MAC address byte 4
7:0	MAC_B5	r/w	8'd0	Ethernet MAC address byte 5

17.8.15 MAC_ADDR1

Address: 0x4000d044

RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MAC_B0								MAC_B1							

Bits	Name	Type	Reset	Description
31:16	RSVD			
15:8	MAC_B0	r/w	8'd0	Ethernet MAC address byte 0

Bits	Name	Type	Reset	Description
7:0	MAC_B1	r/w	8'd0	Ethernet MAC address byte 1

17.8.16 HASH0_ADDR

Address: 0x4000d048

HASH0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

HASH0

Bits	Name	Type	Reset	Description
31:0	HASH0	r/w	32'h0	Lower 32-bit of HASH register

17.8.17 HASH1_ADDR

Address: 0x4000d04c

HASH1

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

HASH1

Bits	Name	Type	Reset	Description
31:0	HASH1	r/w	32'h0	Upper 32-bit of HASH register

17.8.18 TXCTRL

Address: 0x4000d050

RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	TXPAUSERQ
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

TXPAUSETV