



---

**eSi-EMAC**

---

---

# 1 Contents

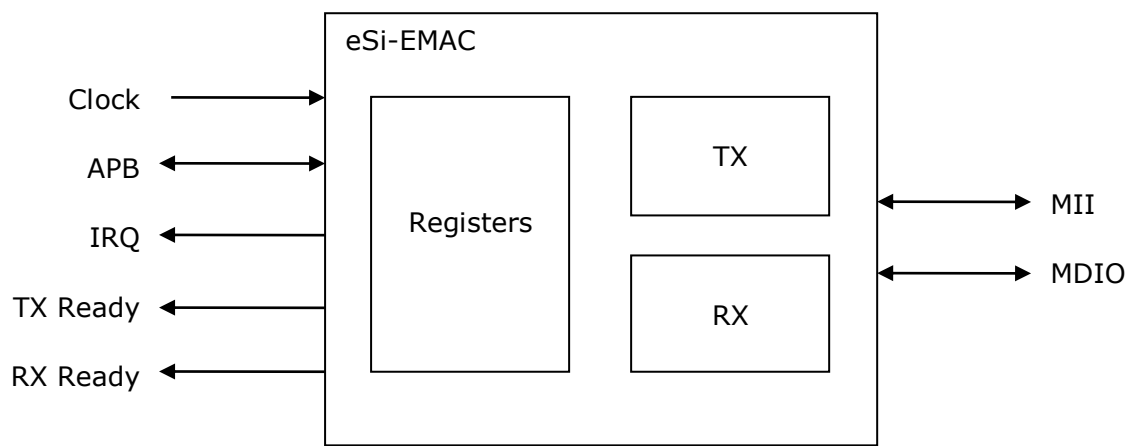
---

1	Contents	2
2	Overview	3
3	Hardware Interface	4
3.1	Timing Constraints	4
4	Software Interface	5
4.1	Register Map	5
4.2	Interrupts	8

## 2 Overview

The eSi-EMAC core implements an Ethernet MAC (Media Access Controller), providing access to 10/100Mbps Ethernet networks. It supports the following features:

- CSMA/CD protocol implementation.
- Optional FCS (Frame Check Sequence) insertion and verification.
- Optional filtering of received frames by MAC address.
- Independent transmit and receive FIFOs.
- 10/100Mbps MII (Media Independent Interface) to external PHY.
- MDIO interface to external PHY.
- AMBA 3 APB slave interface.
- DMA interface.



**Figure 1: eSi-EMAC**

### 3 Hardware Interface

<b>Module Name</b>	cpu_apb_emac
<b>HDL</b>	Verilog
<b>Technology</b>	Generic
<b>Source Files</b>	cpu_apb_emac.v, cpu_emac_tx.v, cpu_emac_rx.v, cpu_async_fifo.v

Port	Direction	Width	Description
scan_mode	Input	1	Indicates when in scan test mode and causes scan_reset_n to be muxed in to internal resets
scan_reset_n	Input	1	Reset to use when in scan_mode is asserted
pclk	Input	1	APB clock
presetn	Input	1	APB reset, active-low
paddr	Input	8	APB address
pssel	Input	1	APB slave select
penable	Input	1	APB enable
pwrite	Input	1	APB write
pwdata	Input	16	APB write data
tx_clk	Input	1	MII transmit clock
rx_clk	Input	1	MII receive clock
rx_d	Input	4	MII receive data
rx_dv	Input	1	MII receive data valid
rx_er	Input	1	MII receive error
crs	Input	1	MII carrier sense
col	Input	1	MII collision
mdio_in	Input	1	MDIO data input
cactive	Output	1	Clock active
pready	Output	1	APB ready
prdata	Output	16	APB read data
pslverr	Output	1	APB slave error
interrupt_n	Output	1	Interrupt request, active-low
tx_en	Output	1	MII transmit enable
tx_d	Output	4	MII transmit data
tx_er	Output	1	MII transmit error
tx_ready	Output	1	Indicates device can accept new data
rx_ready	Output	1	Indicates device has data to be read
mdc	Output	1	MDIO clock
mdio_out	Output	1	MDIO data output
mdio_out_enable	Output	1	MDIO data output enable

**Table 1: I/O Ports**

For complete details of the APB signals, please refer to the AMBA 3 APB Protocol v1.0 Specification available at <http://www.arm.com/products/solutions/AMBAHomePage.html>

For further details of the MII and MDIO signals, please refer to the IEEE 802.3 specification.

#### 3.1 Timing Constraints

For MII operation at 100Mbps, the tx\_clk and rx\_clk inputs should be constrained for a minimum of 25MHz. If only 10Mbps operation is required, the clock constraint should be set to 2.5MHz. pclk may be asynchronous to both tx\_clk and rx\_clk.

## 4 Software Interface

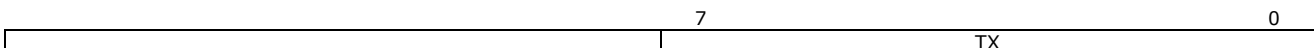
### 4.1 Register Map

Register	Address offset	Access	Description
tx_data	0x00	W	Transmit data
rx_data	0x04	R	Receive data
tx_length	0x08	W	Transmit data length
rx_length	0x0c	R	Receive data length
status	0x10	R/W	Status register
control	0x14	R/W	Control register
mac_address_0	0x20	R/W	MAC address byte 0
mac_address_1	0x24	R/W	MAC address byte 1
mac_address_2	0x28	R/W	MAC address byte 2
mac_address_3	0x2c	R/W	MAC address byte 3
mac_address_4	0x30	R/W	MAC address byte 4
mac_address_5	0x34	R/W	MAC address byte 5
mdio_address	0x40	R/W	MDIO PHY address
mdio_register	0x44	R/W	MDIO PHY register index
mdio_data	0x48	R/W	MDIO read / write data
mdio_status	0x4c	R	MDIO status register
mdio_control	0x50	W	MDIO control register
mdio_cycles_per_bit	0x54	R/W	MDIO clock divider

**Table 2: Register Map**

#### 4.1.1 Transmit Data Register

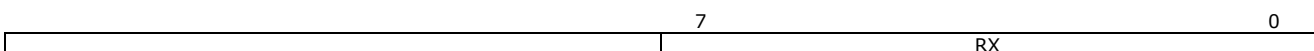
Data written to the transmit register is stored in the transmit FIFO.



**Figure 2: Format of the tx\_data register**

#### 4.1.2 Receive Data Register

Data that is received from the MII interface will be stored in the receive FIFO and can be read via the receive register.



**Figure 3: Format of the rx\_data register**

#### 4.1.3 Transmit Length Register

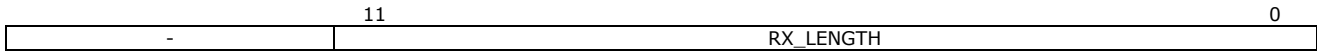
The transmit length register should be written with the number of bytes to transmit. The transmit FIFO should contain this number of bytes before this register is written. Writing the transmit length register will start transmission.



**Figure 4: Format of the tx\_length register**

#### 4.1.4 Receive Length Register

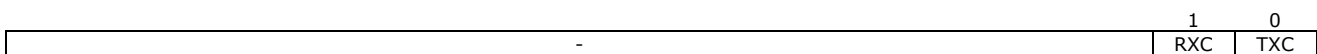
The receive length register contains the number of bytes in a received frame.



**Figure 5: Format of the rx\_length register**

#### 4.1.5 Status Register

The status register contains a selection of flags that indicate the current status of the EMAC. To clear a bit in the status register, write a 1 to it. Writing 0 will leave it unchanged.



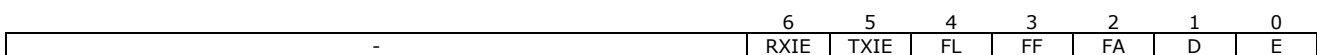
**Figure 6: Format of the status register**

Register	Values	Description
TXC	0 - Not complete 1 - Complete	Transmit complete
RXC	0 - Not complete 1 - Complete	Receive complete

**Table 3: Fields of the status register**

#### 4.1.6 Control Register

The control register contains a selection of flags that control the operation of the EMAC.



**Figure 7: Format of the control register**

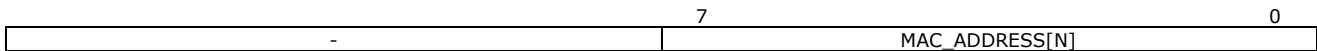
Register	Values	Description
E	0 - Disabled 1 - Enabled	Enables the EMAC. When disabled, data will not be received or transmitted
D	0 - Full duplex 1 - Half duplex	Duplex mode. When set to full duplex, the EMAC can transmit and receive simultaneously
FA	0 - Disabled 1 - Enabled	Filter received frames where the destination address is not the same as the address in the mac_address_N registers or the broadcast address
FF	0 - Disabled 1 - Enabled	Filter received frames where the FCS is invalid
FL	0 - Disabled 1 - Enabled	Filter received frames with illegal lengths
TXIE	0 - Disabled 1 - Enabled	Transmit interrupt enable

RXIE	0 - Disabled 1 - Enabled	Receive interrupt enable
------	-----------------------------	--------------------------

**Table 4: Fields of the control register**

#### 4.1.7 MAC Address Registers

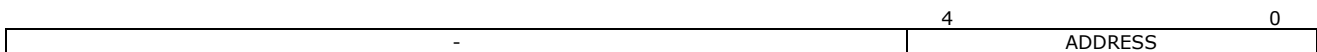
The MAC address registers are used to store the 48-bit MAC address of the EMAC. This can be used to filter received frames, so that only frames that are addressed to this EMAC or the broadcast address generate receive interrupts.



**Figure 8: Format of the mac\_address\_N registers**

#### 4.1.8 MDIO Address Register

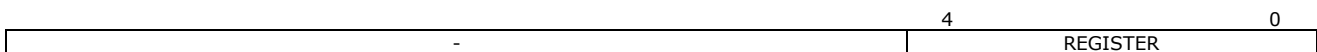
The MDIO address register specifies the address of the PHY to which the MDIO command should be addressed.



**Figure 9: Format of the mdio\_address register**

#### 4.1.9 MDIO Register Index Register

The MDIO register index register specifies the index of the PHY register which should be accessed by the MDIO command.



**Figure 10: Format of the mdio\_register register**

#### 4.1.10 MDIO Data Register

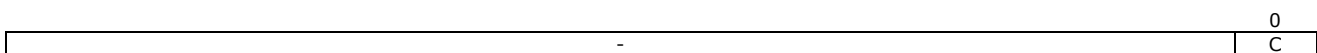
The MDIO data register holds the data to write to the PHY for a MDIO write command, or the data read from a PHY for a MDIO read command.



**Figure 11: Format of the mdio\_data register**

#### 4.1.11 MDIO Status Register

The MDIO status register indicates the status of the last MDIO command.



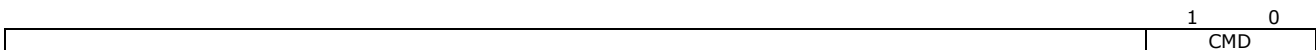
**Figure 12: Format of the mdio\_status register**

Register	Values	Description
C	0 – Not complete 1 – Complete	Whether the last MDIO command has completed

**Table 5: Fields of the `mdio_status` register**

#### 4.1.12 MDIO Control Register

The MDIO control contains a register which when written causes an MDIO command to be sent to the PHY.

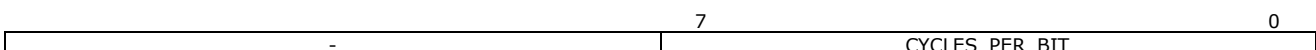

**Figure 13: Format of the `mdio_control` register**

Register	Values	Description
CMD	0 – No action 1 – Write 2 – Read	MDIO command

**Table 6: Fields of the `mdio_control` register**

#### 4.1.13 MDIO Cycles per Bit Register

The MDIO cycles per bit register specifies how many cycles of the clock, `clk`, each bit of data on `mdio` is transmitted for. The frequency of `mdc` is therefore the frequency of `clk` divided by `mdio_cycles_per_bit`.


**Figure 14: Format of the `mdio_cycles_per_bit` register**

## 4.2 Interrupts

The EMAC supports the following interrupts.

- Transmit interrupt
- Receive interrupt

The transmit interrupt will be raised when the EMAC has finished transmitting a frame and the `TXIE` flag in the `control` register is set to 1. This indicates that the EMAC is ready to transmit a new frame.

The receive interrupt will be raised when the EMAC has received a valid frame and the `RXIE` flag in the `control` register is set to 1. In this context, a valid frame is one that has passed FCS verification (if enabled), length verification (if enabled) and has not been filtered due to the destination address not being for this device (if enabled).