Application Note

Register Interface Control

(LMAC2 Core)

LeWiz Communications, Inc.

"The Wizard of Internet Communications"

May 17, 2021
Revision 1.00

PO Box 9276
San Jose, CA 95157
www.lewiz.com
Email: support@lewiz.com

LeWiz can be contacted at: support@lewiz.com
or address: PO Box 9276, San Jose, CA 95157-9276
www.lewiz.com
Author: LeWiz Communications, Inc.

Change Log

Version	Significant Changes
1.00	Release version

Table of Contents

NOTE: This document is intended for SW/HW users using LeWiz MAC IP.

1 Introduction

Inside each LMAC core are registers for configuring the core and keeping track of the status or statistics of the information transferred by the core. This application note discusses the details of interface for accessing the registers.

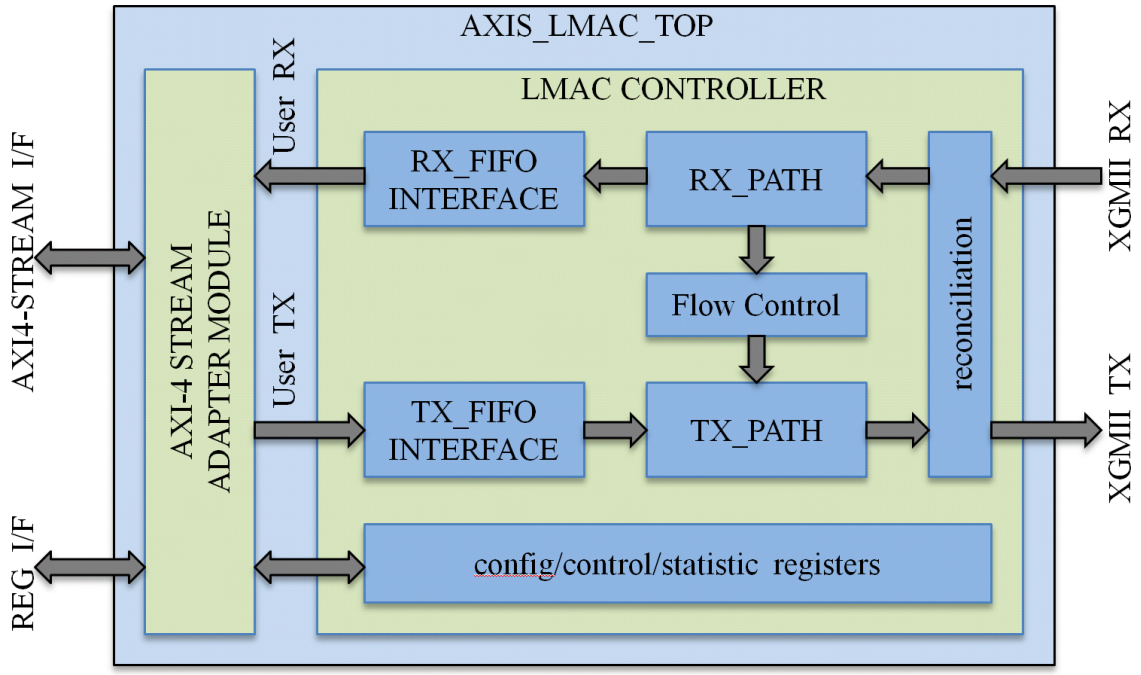

Figure 1.1: LMAC Core2 block diagram (shown with XGMII interface)

Figure 1.1 shows the block diagram of the LMAC Core2. This core supports speed modes: 10G/5G/2.5G/1G. The config/control/statistic registers are present inside the LMAC_CONTROLLER that can be accessed through the 'REG I/F' at the top level module (AXIS_LMAC_TOP).

To keep the core design simple and more accessible by the designers, information that are used to configure the core such as MAC address, speed control and others are brought out as signals to the top level of the core. User can tie these signals to the configuration applicable to the user's application or use user's own registers to control those signals. This way it is possible for the system to power up and work without requiring extensive software configuration.

The registers in the LMAC core are mainly for keeping track of conditions and statistics of the transfers.

2 Overview

The register interface consists of four important signals mentioned in the table below:

Signal	Direction	Description
host_addr_reg	Input 16-bits	16 bit host byte address bus for selecting a memory mapped register. Valid 1 clk before register read start signal. Address must be 64-bit aligned for most cases.
reg_rd_start	Input 1-bit	Pulse. Register read start. 1 = Start the read for a register in the LMAC core 0 = idle.
reg_rd_done_out	Output 1-bit	Pulse. Indicating the register read is done and its data is available on the MAC_REGDOUT bus. 1 = indicating the data is available for the read to register 0 = data not available.
FMAC_REGDOUT	Output 32-bits	Data returned from LMAC register. Read is 32 bit at a time.

Table 2.1: Register Interface Signals

This register interface is mainly used to read the internal registers of the LMAC core. These are status and statistic information collected by the hardware.

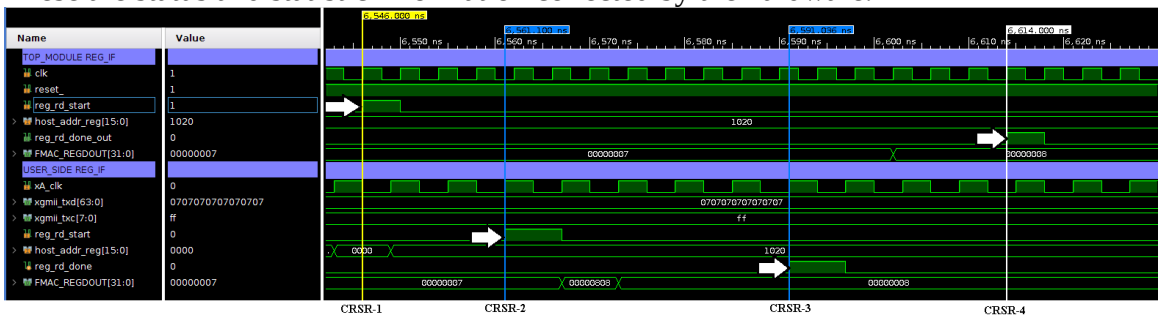

Figure 2.1: Example Register Interface Access

Figure 2.1 above shows an example of accessing the registers through the register interface. It also shows the 'reg_rd_start' delayed on the User side (bottom waves in the figure) due to synchronization. And 'reg_rd_done_out' and data come out to the TOP_MODULE after synchronization (see the waves at the top of the figures). The 'host_addr' here is 16'h1020 driven into the core with reg_rd_start pulse (marker CRSR-1 in Fig. 2.1) to read the contents of the 'FMAC_TX_PKT_CNT' register that holds the number of transmitted packets.

The host address corresponds to a memory mapped register to be read. Once the read request is made, the LMAC will output the contents of the register using 'FMAC_REGDOUT' signal and 'reg_rd_done_out' pulse (CRSR-4 in Fig. 2.1). The 'FMAC_REGDOUT' will hold the register contents that were requested. The number of

transmitted packets for the simulation were 8, so the 'FMAC_REGDOUT' shows 32'h8 with the 'reg_rd_done_out' pulse. The top part of the figure shows the signals from the top-level module (AXIS_LMAC_TOP). The bottom part shows the signals on the internal user side (LMAC_CONTROLLER). The AXI-4 Stream side can be at a different clock frequency as compared to the user side so the core contained synchronization internally. As both the sides are asynchronous (operating at different clock frequencies), the signal 'reg_rd_done_out' comes 5-clocks after the 'reg_rd_start' signal for the user side (delay between CRSR-2 and CRSR-3 in Fig. 2.1) and the delay varies for the top-level module (delay between CRSR-1 and CRSR-4 in Fig. 2.1) depending on the clock frequency that it is running on.

3. Available registers

The available registers are shown in the table below. Addr_offset is the same as the host_addr bus above. Each register has a name and can be accessed through a specific address offset.

Most of these are counters of specific condition captured by the LMAC core. The core implemented extensive statistic capturing which is useful for tracking network statistics.

Addr Offset	**Reg Name**	**Description**
h'0_1000	Reserved	(Reserved - user defined space)
h'0_1008	Reserved	
h'0_1010	Reserved	
h'0_1018	Reserved	
h'0_1020	FMAC_TX_PKT_CNT	Number of transmitted packets. (Each register is 32 bit)
h'0_1028	FMAC_RX_PKT_CNT_LO	Number of received packets (bit 31:0 of 64 bit counter)
h'0_102C	FMAC_RX_PKT_CNT_HI	Number of received packets (bit 63:32 of 64 bit counter)
h'0_1030	FMAC_TX_BYTE_CNT	Number of bytes transmitted
h'0_1038	FMAC_RX_BYTE_CNT_LO	Number of bytes received (low part of 64 bit count)
h'0_103C	FMAC_RX_BYTE_CNT_HI	Number of bytes received (high part of 64 bit count)
h'0_1040	FMAC_RX_UNDERSIZE_PKT_CNT	Number of undersize packets received (<64 byte packet)
h'0_1048	FMAC_RX_CRC32_ERR_CNT	Number of CRC error packets encountered
h'0_1050	FMAC_RX_DCNT_OVERRUN	Number of packets overrun the RxFIFO and dropped
h'0_1058	FMAC_RX_DCNT_LINK_ERR	Number of packets received encountered link error
h'0_1060	FMAC_RX_PKT_CNT_OVERSIZE	Number of packets received but

		over the MAX packet size
h'0_1068	FMAC_PHY_STAT	Internal PHY/Ethernet Link status and information
h'0_1070	Reserved	
h'0_1078	FMAC_RX_PKT_CNT_JABBER	Number of jabber packets
h'0_1080	FMAC_RX_PKT_CNT_FRAGMENT	Number of fragmented packets
h'0_1088	FMAC_RX_RAW_FRAME_CNT	Number of raw Ethernet frames received
h'0_1090	FMAC_RX_BAD_FRAME_CNT	Number of bad Ethernet frames received
h'0_1800	FMAC_RX_PKT_CNT64_LO	Number of packets with size ≤ 64 bytes (low 32 bit count)
h'0_1804	FMAC_RX_PKT_CNT64_HI	Number of packets with size ≤ 64 bytes (high 32 bit count)
h'0_1808	FMAC_RX_PKT_CNT127_LO	Number of packets with size ≤ 127 bytes (low 32 bit count)
h'0_180C	FMAC_RX_PKT_CNT127_HI	Number of packets with size ≤ 127 bytes (high 32 bit count)
h'0_1810	FMAC_RX_PKT_CNT255_LO	Number of packets with size ≤ 255 bytes (low 32 bit count)
h'0_1814	FMAC_RX_PKT_CNT255_HI	Number of packets with size ≤ 255 bytes (high 32 bit count)
h'0_1818	FMAC_RX_PKT_CNT511_LO	Number of packets with size ≤ 511 bytes (low 32 bit count)
h'0_181C	FMAC_RX_PKT_CNT511_HI	Number of packets with size ≤ 511 bytes (high 32 bit count)
h'0_1820	FMAC_RX_PKT_CNT1023_LO	Number of packets with size ≤ 1023 bytes (low 32 bit count)
h'0_1824	FMAC_RX_PKT_CNT1023_HI	Number of packets with size ≤ 1023 bytes (high 32 bit count)
h'0_1828	FMAC_RX_PKT_CNT1518_LO	Number of packets with size ≤ 1518 bytes (low 32 bit count)
h'0_182C	FMAC_RX_PKT_CNT1518_HI	Number of packets with size ≤ 1518 bytes (high 32 bit count)
h'0_1830	FMAC_RX_PKT_CNT2047_LO	Number of packets with size ≤ 2047 bytes (low 32 bit count)
h'0_1834	FMAC_RX_PKT_CNT2047_HI	Number of packets with size ≤ 2047 bytes (high 32 bit count)
h'0_1838	FMAC_RX_PKT_CNT4095_LO	Number of packets with size ≤ 4095 bytes (low 32 bit count)
h'0_183C	FMAC_RX_PKT_CNT4095_HI	Number of packets with size ≤ 4095 bytes (high 32 bit count)
h'0_1840	FMAC_RX_PKT_CNT8191_LO	Number of packets with size ≤ 8191 bytes (low 32 bit count)
h'0_1844	FMAC_RX_PKT_CNT8191_HI	Number of packets with size ≤ 8191 bytes (high 32 bit count)

h'0_1848	FMAC_RX_PKT_CNT9018_LO	Number of packets with size ≤ 9018 bytes (low 32 bit count)
h'0_184C	FMAC_RX_PKT_CNT9018_HI	Number of packets with size ≤ 9018 bytes (high 32 bit count)
h'0_1850	FMAC_RX_PKT_CNT9022_LO	Number of packets with size ≤ 9022 bytes (low 32 bit count)
h'0_1854	FMAC_RX_PKT_CNT9022_HI	Number of packets with size ≤ 9022 bytes (high 32 bit count)
h'0_1858	FMAC_RX_PKT_CNT9199_LO	Number of packets with size ≤ 9199 bytes (low 32 bit count)
h'0_185C	FMAC_RX_PKT_CNT9199_HI	Number of packets with size ≤ 9199 bytes (high 32 bit count)

Table 3.1: Available Registers for LMAC

Each register is 32-bit width to be compatible with both 32 and 64-bit processors. 64-bit registers are split into 2 registers and the register name indicated the "lo" and "hi" 32-bit.

The 'FMAC_PHY_STAT' register is described below in Section 3.1.

3.1 FMAC_PHY_STAT Register

FMAC_PHY_STAT[0] = if 1, indicating the Ethernet link is up
FMAC_PHY_STAT[1] = Reserved
FMAC_PHY_STAT[2] = if 1, indicating the RX of the SerDes channel is aligned

Other bits are reserved.