

10Gb/s Ethernet Platform Implementation

John Chaiyasarikul, Yumeng Xu, Shuguan Yang, Jian Zhong

May 31st, 2014

APPLICATIONS

- DATA Centers
- Finance
- Clusters







\$1.6k - \$3.2k



\$300 + Low Power



Dev Kit Unavailable



\$3.6k



Dev Kit Unavailable

Motivation/Significance

Architecture of the Design



I. HSMC - 4 lanes @ 3.125Gb/s II. XAUI/XGMII - 8 Bytes with 8bits control signals



Interfaces

III. Avalon ST - Unidirectional flow of data @ 156.25MHz.

IV. Avalon MM - A standard address-based read/write interface typical of master-slave connections.





Pin Assignment Matching

assign CONFIG0_1 = 1'b1; assign CONFIG1_1 = 1'b0; assign CONFIG0_2 = 1'b1; assign CONFIG1_2 = 1'b0; assign SS338_CLKIN = 1'b0; assign SER_BOOT = 1'b0; assign SMBSPDSEL1 = 1'b0; assign SMBSPDSEL2 = 1'b0; assign SMBWEN = 1'b1; assign GPIO0_1 = 1'b0; assign GPIO1_1 = 1'b0;	<pre>assign MDIO1 = !MDOEN1? MDO1 : 1 assign MDIN1 = MDIO1; assign MDC2 = 1'bz; assign MDIO2 = 1'bz; assign PHYRESET = KEY[1]; assign STOPMON = ~KEY[2]; {PRTAD4, PRTAD3, PRTAD2, PRTAD1, PR assign PRTAD02 = 1'b0; assign TXONOFF1 = 1'b1; assign TXONOFF2 = 1'b1; assign OPOUTLVL = 1'b0;</pre>	"bz; TAD01} = 5'b00000;		
assign GPIO0_2 = 1'b0; assign GPIO1_2 = 1'b0; assign NVMA1SEL = 1'b1;	assign OPINLVL = 1'b1; assign USER_LED_R = 8'b00001111; assign USER_LED_C = 8'b11110000;	XAUI PHY		HSMC
assign NVMPROT = 1'b0;	assign 03ck_ccb_6 = 8 011110000,	xaui_tx_seriel_data [3:0]	>>	HSMC_XAUI_TX_p0[3:0]
		phy_mgmt_clk_reset	<<	IRESET_N
		rx_ready	>>	LED[2]
		tx_ready	>>	LED[3]
		xgmii_rx_clk xgmii_tx_clk phy_mgmt_address [8:0] phy_mgmt_write	> < << 3'h000 << 0	
Configur	ation	phy_mgmt_writeata [31:0] phy_mgmt_read phy_mgmt_readdata[31:0]	<< 0 << 8'h0000000	

Fast/Transparent MAC 3 clock cycle latency rx 2 clock cycle latency tx

Convert between XGMII and Avalon ST Interfaces Calculate Checksum Check or Append Checksum

۲	/packet_generator_testbench/counter	1101	0001	0010	0011	0100	_)010)1 (0110	0111	(100	0 (1001	<u>),1010</u>	(101	1 <u>(1100</u>	X1101				
۲	/packet_generator_testbench/flag	1																	
۲	/packet_generator_testbench/fb	0																	
Þ	/packet_generator_testbench/packet	0010101010	001010101	<u>.</u> ¢01010	10100	0101010	11010	010111111	011001	01010	100101010101	0010101	01 00	000000000000000000000000000000000000000	0000010	<u>300000</u>	000000010	11100	000000
×	/packet_generator_testbench/xgmiic	11111111	000	<u>(200000)</u>	00										000)1111	1111		
÷	/packet_generator_testbench/contro	1																	
÷	/packet_generator_testbench/clk	0																	
۲	/packet_generator_testbench/b0	07)fb	01) <u>F2</u>)00	103)00	1)2f)59)2e)6d)3a),78	Vee)bc)07		
Þ	/packet_generator_testbench/b1	07)55	do 👘)93)46	_12d)01)d6)3a)63)62		73)(FF)bc)07		
÷	/packet_generator_testbench/b2	07)55	5e)c2	/41	-)80)02)53)64	73)69)5a)76	Vaa)bc)07		
÷	/packet_generator_testbench/b3	07	55	do 👘)d5		36)72)4e)68)2e)61)72)55)07			
÷	/packet_generator_testbench/b4	07)55	00)08)00	14)02	051	63)2e)64)aa)/a9)07			
Þ	/packet_generator_testbench/b5	07	55	01)00		_)c2)72)55)6f)65)bb)(4d)07			
÷	/packet_generator_testbench/b6	07)55	do 👘)45	01	_)e¢)00)45	34)6c)64	53)(5a)07			
÷	/packet_generator_testbench/b7	07)d5	(17)00	(11	<u>)00</u>	32)52	37)75)3a)dd){Fd)07			
÷	/packet_generator_testbench/empty	0	0		>												5)0
÷	/packet_generator_testbench/sop	0																	
÷	/packet_generator_testbench/eop	0																	
÷	/packet_generator_testbench/valid	0																	
÷	/packet_generator_testbench/checks	0																	
÷	/packet_generator_testbench/d0	00	-00				101)£2	100	03)00)2f)59)2e)6d)3a)78)ee)00
F	/packet_generator_testbench/d1	00	-00)93)46)2a)01)d6)3a)63)62)72)73	XEF)00
Þ	/packet_generator_testbench/d2	00	-00)5e)c2	41)80)02)53)64)73)69)5a)76)aa)00
Þ	/packet_generator_testbench/d3	00	-00)d5	D/7c)3b	.72)4e)68)2e)61	X77)72)55)00
÷	/packet_generator_testbench/d4	00	-00)08	100	(14)02)51	63)2e)64)aa 👘	Xa9)00
÷	/packet_generator_testbench/d5	00	-00				-101)00)c2	.72)55)70)6f)65)72	dd()4 <u>a</u>)00
•	/packet_generator_testbench/d6	00	-00)45)01)e0)00)45)34)6c)64)53)6a)00
•	/packet_generator_testbench/d7	00	-00				117)00	(11)00()32)52	37	(75)3a)dd	XOD	
_																			

MAC



A

MAC

1101 01011101 000	← Append zeros equal to the size of the checksum	1101 01011101 101 1101	← Append checksum to data
1101			
	Derform VOD until more bits are needed	1000	← Compute as before
1000	← Perform AOR until more bits are needed	1101	
01011	← Omit leading zeros and bring down next bit	01011	
1101		1101	
		01101	
01101	Depeat stops until nur out of hits	1101	
1101	← Repeat steps unui run out of bits		
0000 0100		0000 0110	
1101		1101	
1001		1011	
1101		1101	
01000		01101	
1101		1101	
<mark>0</mark> 101	← Remainder is the checksum	0000	← Checksum will be zero

For parallel computing refer to <u>www.cypress.com/?docID=31573</u> and easics.com

CRC-32



Connections

Problem: Data is received and transmitted with a bus width of 64 bits of data, MM interface supports only 32 bits with no control signals Solution for Control 7a7a7a7a = SOP 7b7b7b7b = EOP 7d7d7d7d = escape 7e7e7e7e = no data Dual Clock FIFO



ST/MM

No/Incorrect documentation New Board / incompatible IP cores Burnt Daughter board Dual Port cant be compiled CRC computation Timing constraints MM and HPS Unreliable Cables

Challenges/Roadblocks