mikroprojekt

KONDOR AX

Advanced System Development Board

BASIC DEMOS REFERENCE DESIGN GUIDE

1.10.2015.
UM0029
Rev. 1.1

Table of contents

Revision History

Revision	Date	Author	Modification
1.0	21.09.2015.	SH	Initial
1.1	29.09.2015	SH	Improvements after feedback

Related Documents

ID	Code	Description
1	UM0026	KONDOR AX – User Manual
2	UM0027	KONDOR AX – Linux BSP Build Setup Guide
3	UM0028	KONDOR AX – Basic Demos Guide

1 Introduction

This document contains information about basic demos for the KONDOR AX - Advanced System Development Board, demonstrating communication with the i.MX6 processor and the use of various communication interfaces.

This document and accompanying source code will help you to integrate PCIe, EIM, LPDDR3, SGMII, RMII, FMC, LVDS, UART and I²C interfaces in your designs.

All demos in this document run in **Diamond 3.5.1.**

The demo designs described in this document include:

- *PCI Express Memory Demo*
- *PCI Express SFP Gigabit Ethernet Demo*
- *PCI Express Fast Ethernet Demo*
- *EIM Fast Ethernet Demo*
- *EIM FMC Demo*
- *EIM LVDS Demo*
- *UART I²C Demo*

Each demo presents a different interface combination. Recurring interfaces are only described on their first occurrence to avoid unnecessary repetition.

PCIe demos include a Mico32 processor that initializes the onboard Si5338 PLL chip over I²C. That module (mico32_clock) is supplied in netlist (.ngo) format. Chapter 9 documents the project for generating that netlist.

This document focuses on the FPGA part of the demo. For more information about the hardware requirements and demo running instructions consult the ECP5COM Basic Demos document. For pinouts and other board specific information consult the ECP5COM Eval Board User Manual. For details about the demo software consult the ECP5COM Software User Guide.

1.1 Directory structure

The demo designs are organized in several folders, as shown in Figure 1.

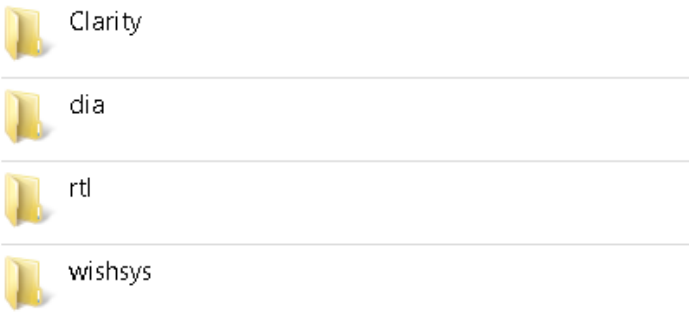

Figure 1: Demo directory structure

The 'Clarity' folder contains IP Cores instantiated in Clarity. Some IP Cores are located in the rtl/ip folder.

The 'dia' folder contains the Diamond project files (.ldf, .lpf), implementation files and other related files.

The 'rtl' folder contains the Verilog source code files.

The 'wishsys' folder contains Lattice Mico System files for the Wishbone interconnect.

2 RD0012 – PCI Express Memory Demo

2.1 Overview

The name of the design for this demo is RD0012_ecp5com_pcie_mem.

This demo demonstrates how software running on the i.MX6 ARM processor can use the PCIe x1 link to accesses LPDDR3 memory and LEDs controlled by the ECP5 FPGA. This demo also serves as the basis for other PCIe demos in this document. Figure 2 shows the block diagram for this design.

The PCIe Endpoint IP and attached logic receive Transport Layer Packets (TLPs) which the i.MX6 sends over the PCIe link. The Wishbone Transaction Level Completer (WB TLC) module parses incoming write and read TLPs and generates completion TLPs containing read data. WB TLC is also the master on the narrow Wishbone bus which has 32-bit addresses and 16-bit data.

The WB 16 to 32 Converter converts the narrow 16-bit Wishbone bus to the wide 32-bit Wishbone bus of the Mico System (wishsys). The WB TLC module acts as another master on the wide Wishbone bus. Slaves attached to the Wisbone bus are the LPDDR3 memory controller and a GPIO.

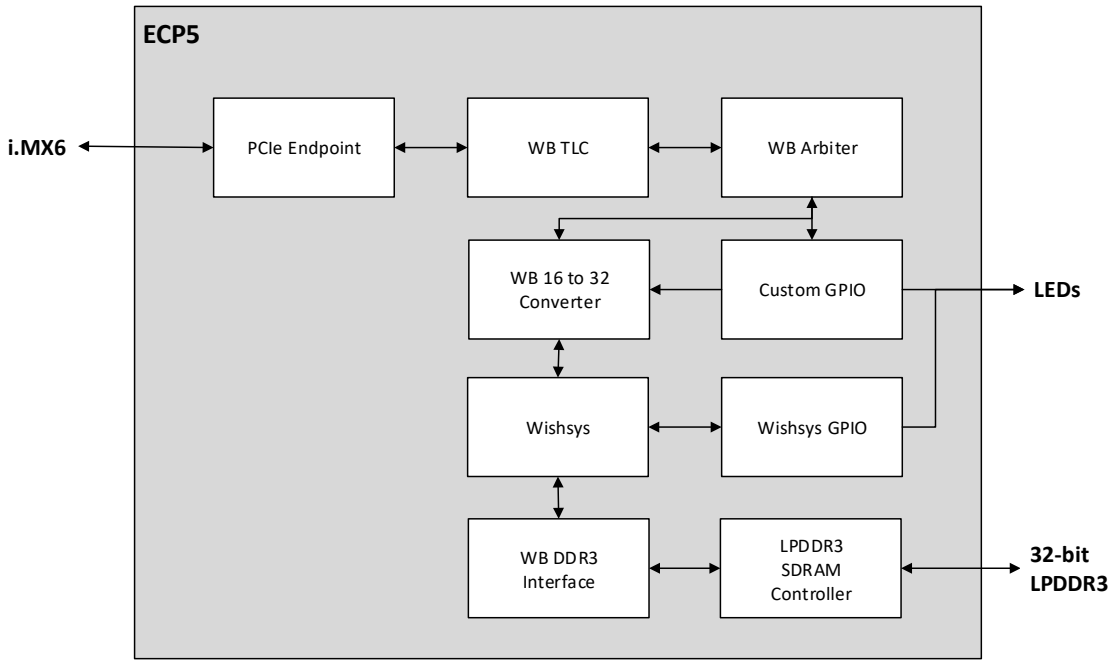

Figure 2: PCI Express Memory Demo block diagram

2.2 Clocking

The differential reference clock for PCIe cores is 100 MHz. It is generated by the onboard PLL chip which needs to be configured over I²C. That is why all PCIe designs include one Lattice Mico32 which serves only to configure the PLL chip.

All PCIe-related logic and the Wishbone bus run on a 125MHz clock supplied by the PCIe Endpoint IP Core.

The LPDDR3 SDRAM Controller uses its dedicated onboard 100MHz reference clock (clk_in) to generate a 300 MHz memory clock, and a 150 MHz system clock (sclk). The Wishbone DDR3 interface module runs on this 150 MHz system clock.

The reset signal (rstn) is generated by a 20-bit wide counter which is initialized on FPGA configuration, and runs on a 38.8 MHz clock generated by an internal oscillator.

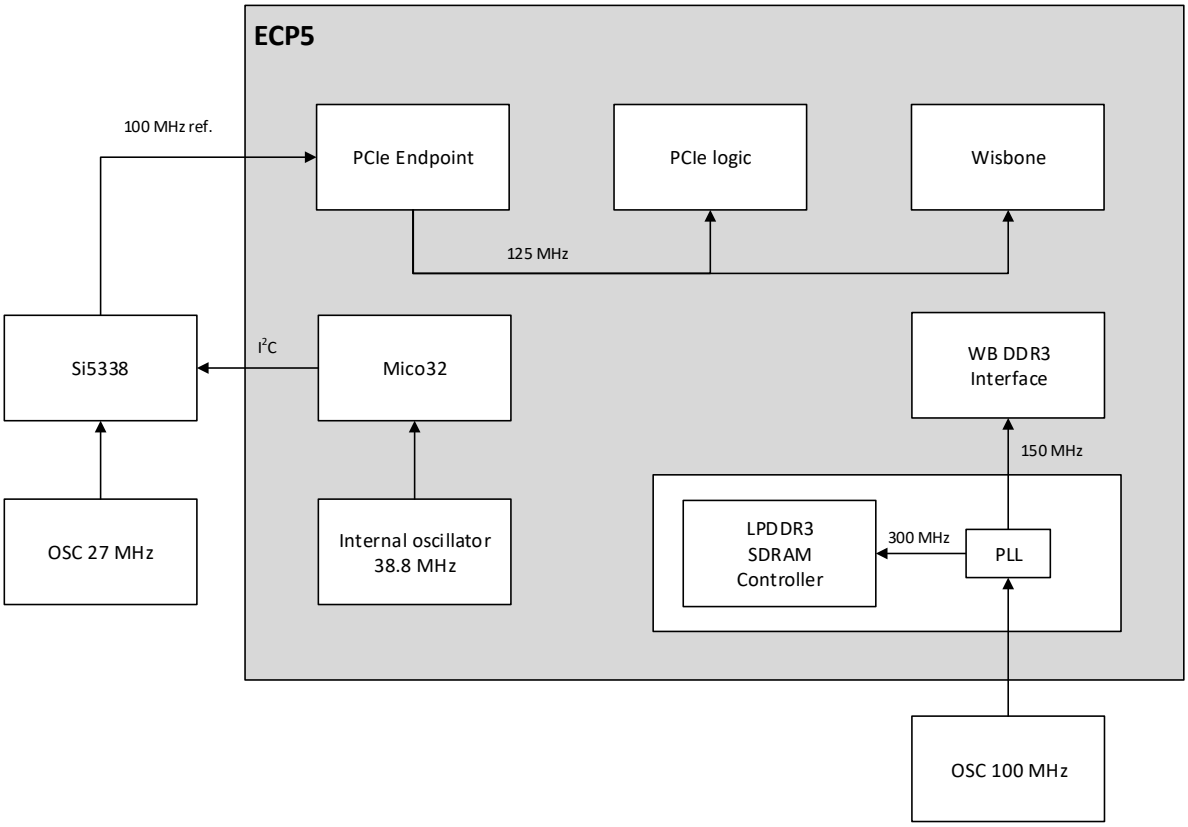

Figure 3 PCI Express Memory Demo clocking scheme

2.3 Peripheral addressing

The ECP5 is configured as a PCIe peripheral with two BARs (Base Address Registers), BAR0 and BAR1 which map it to two 256 KB regions in the ARM's address space. In this demo both regions (BARs) map to the same PCIe address space. The WB TLC using the WB Arbiter can access either the GPIO (mapped to

address 0x10000) or the wide Wishbone bus (mapped to 0x0 – 0xFFFF). The mentioned addresses are relative to the beginning of the BAR.

Since the BARs are only 256 KB in size, this means that after address translation we are left with only 18-bit addresses on the Wishbone bus. Table 1 shows the address mapping of peripherals on the narrow and wide Wishbone bus. It is obvious that 18 bits of address are not enough to access those peripherals. The solution to this problem is to have the WB 16 to 32 Converter use a scratch register for higher 16 bits of the address. The scratch register is accessed using the Custom GPIO module (wbs_gpio.v).

The scratch register defines which 64KB memory bank of the wide Wishbone bus address space will be accessed when the 0x0 – 0xFFFF range is addressed on the narrow Wishbone bus.

Table 1 PCI Express Memory Demo address mappings

Narrow bus address	Wide bus address	Peripheral	Note
0x00000 – 0x0FFFF		Wide bus mapped	
0x10004		GPIO/Scratchpad register	used for higher 16 bits of address
0x10008		GPIO/LEDs	bits 8 & 9 are mapped to LED8 & LED9
	0x00000000 – 0x0FFFFFFF	LPDDR3	size 256 MB
	0x10000000 – 0x10003FFF	EBR	size 32 KB
	0x80000000 - 0x8000000F	Wishsys GPIO	writing 0xFF to 0x80000000 lights up LED0 – LED7

LED10 and LED11 are controlled by the Mico processor that configures the Si5338 PLL chip, and indicate that the PLL has been successfully configured.

2.4 PCIe x1 interface

The PCIe x1 interface consists of three layers: The Transport, Adaptation and Application Layers.

The Transport Layer takes care of moving Transport Layer Packets across the PCIe link. The functionality of the Transport Layer is handled by the PCIe Endpoint IP Core (pcie_core module).

The Adaptation Layer unpacks PCIe TLPs and presents them to the Application Layer. In other words it translates PCIe transactions into Application Layer transactions (in this case Wishbone bus transactions). The PCIe specification mandates that all transactions that need a response need to receive one regardless of whether that type of transaction is supported or not. The UR_gen module generates responses for unsupported requests.

The wb_tlc module is the Transaction Layer Completer that converts PCIe transactions into Wishbone transactions. The ip_tx_arbiter module arbitrates which of the UR_gen and wb_tlc modules will transmit data to the Transport Layer.

For more detail about the PCIe x1 interface, please refer to [UG15].

2.5 LPDDR3 interface

The LPDDR3 interface consists of the WB DDR3 interface module, which translates Wishbone accesses to memory access commands, and the LPDDR3 SDRAM Controller IP Core from Lattice. The memory is 32-bit wide and is accessed with a 300 MHz DDR clock. This means that on each rising edge of the 150 MHz LPDDR system clock, up to 128 bits of data are available.

For more information about the LPDDR3 SDRAM Controller IP Core and DDR timing, refer to [IPUG110] and [TN1265].

2.6 IP versions

Table 2 PCI Express SFP Gigabit Ethernet Demo IP Core versions

IP Name	IP Version
LPDDR3 SDRAM Controller	1.0
PCI Express Endpoint Core	6.1
PLL	5.8
EXTREF	1.1

3 RD0013 – PCI Express SFP Gigabit Ethernet Demo

3.1 Overview

The name of the design for this demo is RD0013_ecp5com_pcie_sgmii.

This demo demonstrates how software running on the i.MX6 ARM processor can use the PCIe x1 link to access SFP Gigabit Ethernet modules, LPDDR3 memory and LEDs controlled by the ECP5 FPGA. Figure 4 shows the block diagram for this design.

The PCIe Endpoint IP and attached logic receive Transport Layer Packets (TLPs) which the i.MX6 sends over the PCIe link. The Wishbone Transaction Level Completer (WB TLC) module parses incoming write and read TLPs and generates response completion TLPs containing read data. WB TLC is also the master on the narrow Wishbone bus which has 32-bit addresses and 16-bit data.

The WB 16 to 32 Converter converts the narrow 16-bitWishbone bus to the wide 32-bit Wishbone bus of the Mico System (wishsys). The WB TLC module acts as another master on the wide Wishbone bus. Slaves attached to the Wisbone bus are the Tri-Speed Ethernet MAC, LPDDR3 memory controller and a GPIO.

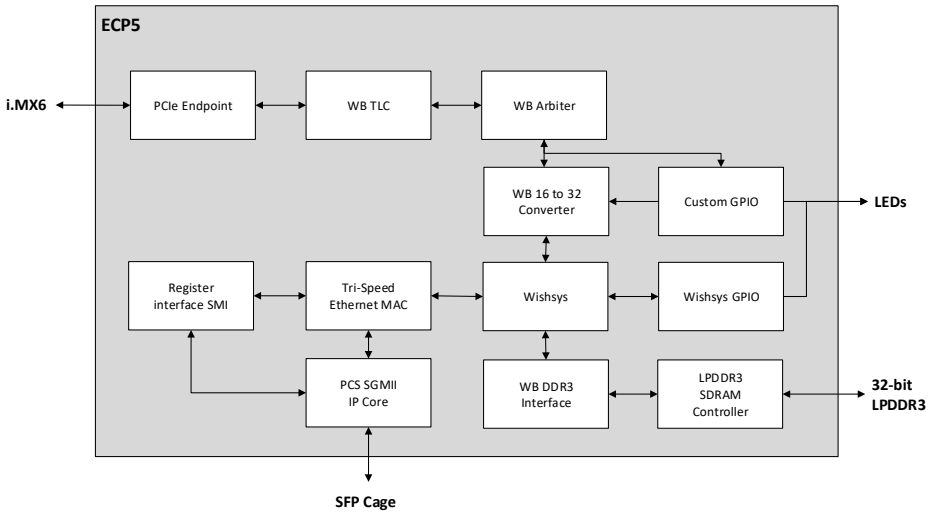

Figure 4 PCI Express SFP Gigabit Ethernet Demo block diagram

3.2 Clocking

The differential reference clock for PCIe cores is 100 MHz. It is generated by the onboard PLL chip which needs to be configured over I²C. That's why all PCIe designs include a Lattice Mico32 which serves only to configure the PLL chip.

All PCIe-related logic runs on a 125MHz clock supplied by the PCIe Endpoint IP Core. The Tri-Speed MAC also runs on this 125MHz clock, and internally regenerates 125 MHz transmit and receive clocks for the PCS SGMII Core.

The LPDDR3 SDRAM Controller uses its dedicated onboard 100MHz reference clock (clk_in) to generate a 300 MHz clock for memory access, and a 150 MHz system clock (sclk). The Wishbone DDR3 interface module runs on this 150 MHz system clock.

The Wishbone bus runs on a 75 MHz clock generated with a PLL from the 150 MHz system clock generated by the LPDDR3 SDRAM Controller.

The reset signal (rstn) is generated by a 20-bit wide counter which is initialized on FPGA configuration, and runs on a 77.5 MHz clock generated by an internal oscillator.

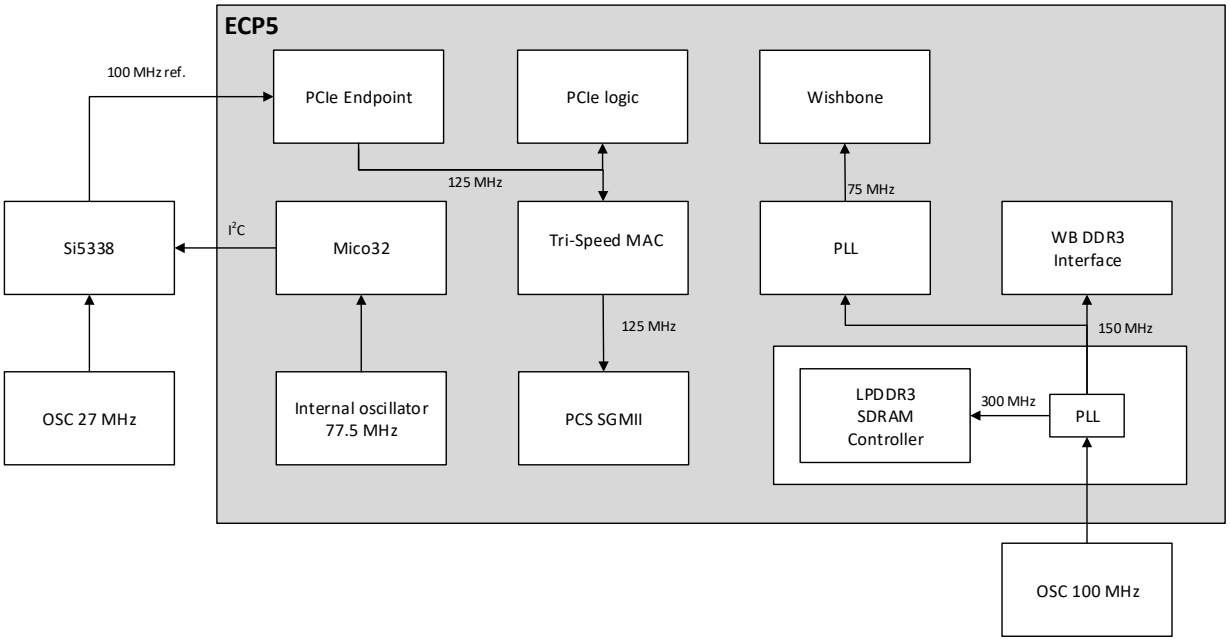

Figure 5 PCI Express SFP Gigabit Ethernet Demo clocking scheme

3.3 Peripheral addressing

The ECP5 is configured as a PCIe peripheral with two BARs (Base Address Registers), BAR0 and BAR1 which map it to two 256 KB regions in the ARM's address space. In this demo both regions (BARs) map to the same PCIe address space. The WB TLC using the WB Arbiter can access either the GPIO (mapped to address 0x10000) or the wide Wishbone bus (mapped to 0x0 – 0xFFFF). The mentioned addresses are relative to the beginning of the BAR.

Since the BARs are only 256 KB in size, this means that after address translation we are left with only 18-bit addresses on the Wishbone bus. Table 1 shows the address mapping of peripherals on the narrow and wide Wishbone bus. It is obvious that 18 bits of address are not enough to access those peripherals. The solution to this problem is to have the WB 16 to 32 Converter use a scratch register for higher 16 bits of the address. The scratch register is accessed using the Custom GPIO module (wbs_gpio.v).

The scratch register defines which 64KB memory bank of the wide Wishbone bus address space will be accessed when the 0x0 – 0xFFFF range is addressed on the narrow Wishbone bus.

Table 3 PCI Express SFP Gigabit Ethernet Demo address mappings

Narrow bus address	Wide bus address	Peripheral	Note
0x00000 – 0x0FFFF		Wide bus mapped	
0x10004		GPIO/Scratchpad register	used for higher 16 bits of address
0x10008		GPIO/LEDs	bits 8 & 9 are mapped to LED8 & LED9
	0x00000000 – 0x0FFFFFFF	LPDDR3	size 256 MB
	0x10000000 – 0x10003FFF	EBR	size 32 KB
	0xD0000000 – 0xDFFFFFFF	Tri-Speed Ethernet MAC	
	0x80000000 - 0x8000000F	Wishsys GPIO	writing 0xFF to 0x80000000 lights up LED0 – LED7
	0x90000000 - 0x9000000F	UART	USB UART on connector U33

3.4 Gigabit Ethernet interface

The Gigabit Ethernet interface consists of the:

- *Tri-Speed Ethernet MAC top model (folder: rtl/ts_mac) – contains the top module of Gigabit Ethernet MAC, including a Wishbone wrapper. It handles the MAC layer of the Ethernet protocol.*
- *GbE/SGMII PCS IP Core – packs data generated by the MAC module and transmits it using PCS which generates the final serial differential signal. The receive side works similarly.*
- *Register interface SMI – configures the SGMII PCS IP Core. It communicates with the MAC module over the MDIO interface.*

The SGMII signals (transmit and receive differential pairs) are routed off-chip to one of the SFP cages.

3.5 IP versions

Table 4 PCI Express SFP Gigabit Ethernet Demo IP Core versions

IP Name	IP Version
Tri-Speed Ethernet MAC	3.7esr
SGMII/Gb Ethernet PCS	4.1
LPDDR3 SDRAM Controller	1.0
PCI Express Endpoint Core	6.3
EXTREF	1.1
PLL	5.8

4 RD0014 – PCI Express Fast Ethernet Demo

4.1 Overview

The name of the design for this demo is RD0014_ecp5com_pcie_rmii.

This demo demonstrates how software running on the i.MX6 ARM processor can use the PCIe x1 link to use a Fast Ethernet interface chip, LPDDR3 memory and LEDs controlled by the ECP5 FPGA. Figure 6 shows the block diagram for this design.

The PCIe Endpoint IP and attached logic receive Transport Layer Packets (TLPs) which the i.MX6 sends over the PCIe link. The Wishbone Transaction Level Completer (WB TLC) module parses incoming write and read TLPs and generates response completion TLPs containing read data. WB TLC is also the master on the narrow Wishbone bus which has 32-bit addresses and 16-bit data.

The WB 16 to 32 Converter converts the narrow 16-bit Wishbone bus to the wide 32-bit Wishbone bus of the Mico System (wishsys). The WB TLC module acts as another master on the wide Wishbone bus. Slaves attached to the Wisbone bus are the Tri-Speed Ethernet MAC, LPDDR3 memory controller and a GPIO.

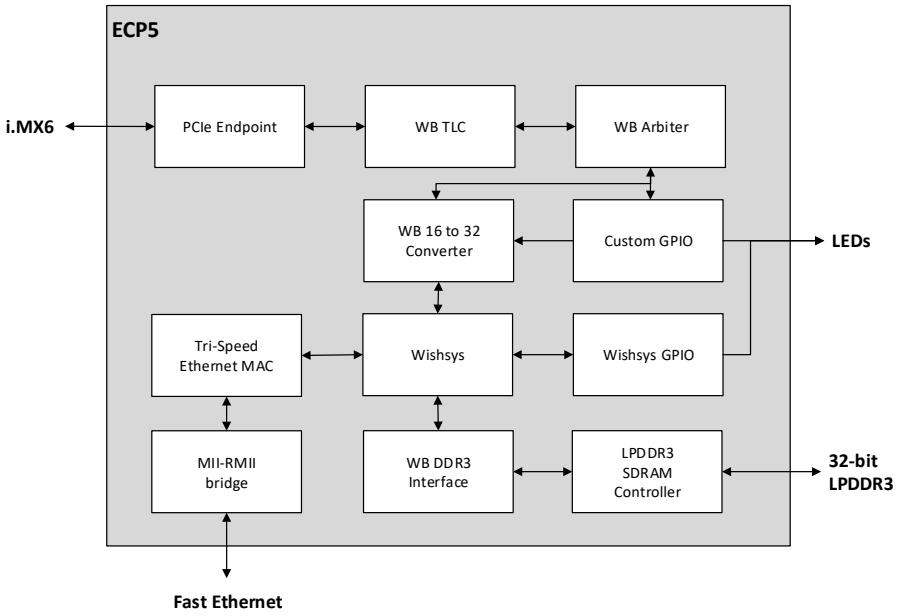

Figure 6 PCI Express Fast Ethernet Demo block diagram

4.2 Clocking

The differential reference clock for PCIe cores is 100 MHz. It is generated by the onboard PLL chip which needs to be configured over I²C. That's why all PCIe designs include a Lattice Mico32 which serves only to configure the PLL chip.

All PCIe-related logic runs on a 125MHz clock supplied by the PCIe Endpoint IP Core.

The LPDDR3 SDRAM Controller uses its dedicated onboard 100MHz reference clock (clk_in) to generate a 300 MHz clock for memory access, and a 150 MHz system clock (sclk). The Wishbone DDR3 interface module runs on this 150 MHz system clock.

The RMII interface requires a 50 MHz clock (rmii_clk50_i) and 25 MHz clock generated from the 150MHz LPDDR system clock.

The Wishbone bus runs on a 75 MHz clock generated with a PLL from the 150 MHz system clock generated by the LPDDR3 SDRAM Controller.

The reset signal (rstn) is generated by a 20-bit wide counter which is initialized on FPGA configuration, and runs on a 77.5 MHz clock generated by an internal oscillator.

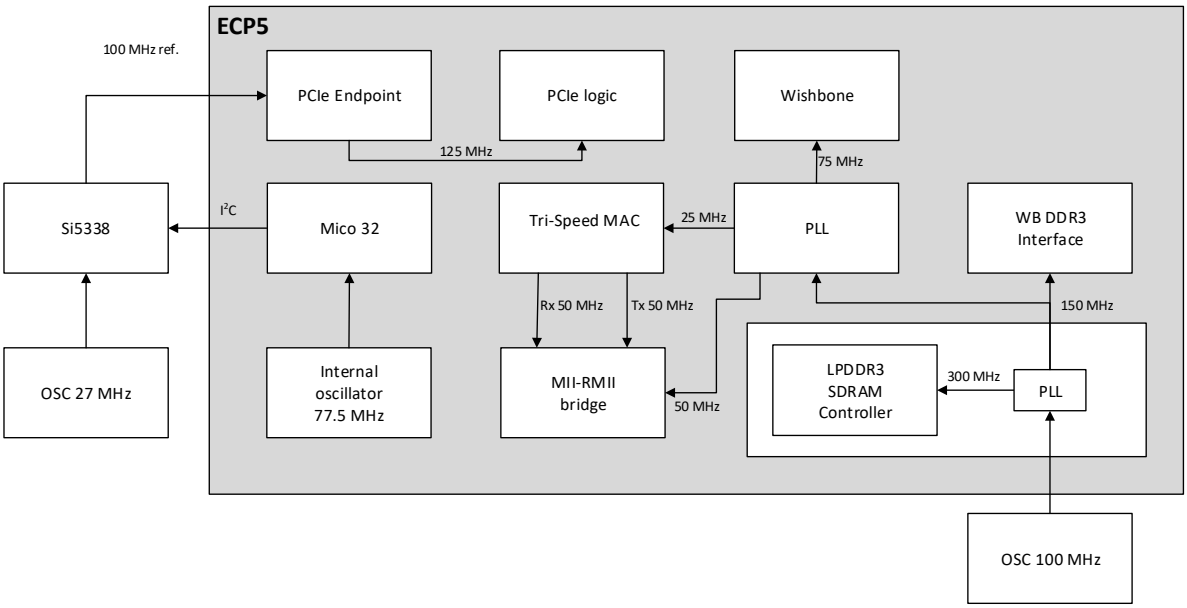

Figure 7 PCI Express Fast Ethernet Demo clocking scheme

4.3 Peripheral addressing

The ECP5 is configured as a PCIe peripheral with two BARs (Base Address Registers), BAR0 and BAR1 which map it to two 256 KB regions in the ARM's address space. In this demo both regions (BARs) map to the same PCIe address space. The WB TLC using the WB Arbiter can access either the GPIO (mapped to address 0x10000) or the wide Wishbone bus (mapped to 0x0 – 0xFFFF). The mentioned addresses are relative to the beginning of the BAR.

Since the BARs are only 256 KB in size, this means that after address translation we are left with only 18-bit addresses on the Wishbone bus. Table 1 shows the address mapping of peripherals on the narrow and wide Wishbone bus. It is obvious that 18 bits of address are not enough to access those peripherals. The solution to this problem is to have the WB 16 to 32 Converter use a scratch register for higher 16 bits of the address. The scratch register is accessed using the Custom GPIO module (wbs_gpio.v).

The scratch register defines which 64KB memory bank of the wide Wishbone bus address space will be accessed when the 0x0 – 0xFFFF range is addressed on the narrow Wishbone bus.

Table 5 PCI Express Fast Ethernet Demo address mappings

Narrow bus address	Wide bus address	Peripheral	Note
0x00000 – 0x0FFFF		Wide bus mapped	
0x10004		GPIO/Scratchpad register	used for higher 16 bits of address
0x10008		GPIO/LEDs	bits 8 & 9 are mapped to LED8 & LED9
	0x00000000 – 0x0FFFFFFF	LPDDR3	size 256 MB
	0x10000000 – 0x10003FFF	EBR	size 32 KB
	0xD0000000 – 0xDFFFFFFF	Tri-Speed Ethernet MAC	
	0x80000000 - 0x8000000F	Wishsys GPIO	writing 0xFF to 0x80000000 lights up LED0 – LED7
	0x90000000 - 0x9000000F	UART	USB UART on connector U33

4.4 Fast Ethernet interface

The Fast Ethernet interface consists of the:

- *Tri-Speed Ethernet MAC top model (folder: rtl/ts_mac) – contains the top module of Gigabit Ethernet MAC, including a Wishbone wrapper. It handles the MAC layer of the Ethernet protocol.*
  It also configures the Ethernet PHY chip over the MDIO interface.
- *MII-RMII bridge (mii_rmii_bridge.v) – bridges MII to the RMII protocol for communicating with the Fast Ethernet chip. This drives the RJ-45 connector that is closest to the FPGA.*

4.5 IP versions

Table 6 PCI Express Fast Ethernet Demo IP Core versions

IP Name	IP Version
Tri-Speed Ethernet MAC	3.7esr
LPDDR3 SDRAM Controller	1.0
PCI Express Endpoint Core	6.3
EXTREF	1.1
PLL	5.8

5 RD0015 – EIM Fast Ethernet Demo

5.1 Overview

The name of the design for this demo is RD0015_ecp5com_eim_rmii.

This demo shows how the i.MX6 ARM processor can use the EIM interface to communicate with interfaces connected to the FPGA. In this example the interfaces are the Fast Ethernet interface chip and LPDDR3 memory. The top-level of this design is simpler than the previous PCIe-based designs. This demo is also the template for the following EIM-based demos.

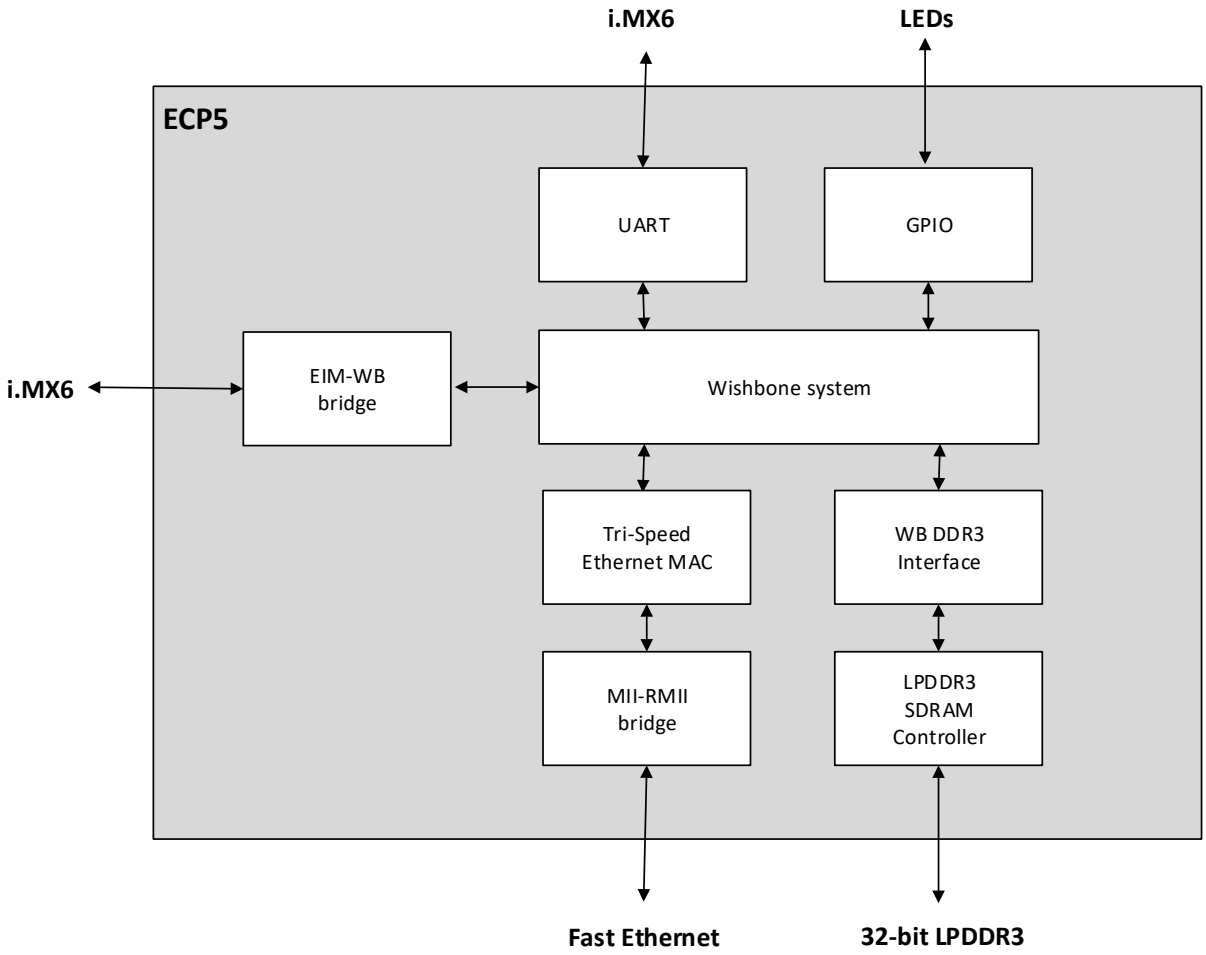

Figure 8 EIM Fast Ethernet Demo block diagram

Figure 8 shows the block diagram for this demo.

The EIM-WB bridge translates the EIM signals coming from the CPU to requests on the internal Wishbone bus. The internal Wishbone bus interconnect is generated using the Mico System builder. The bus uses 32-bit addresses and 32-bit data. Figure 10 shows the address mappings of various peripherals connected to the Wishbone bus.

5.2 Clocking

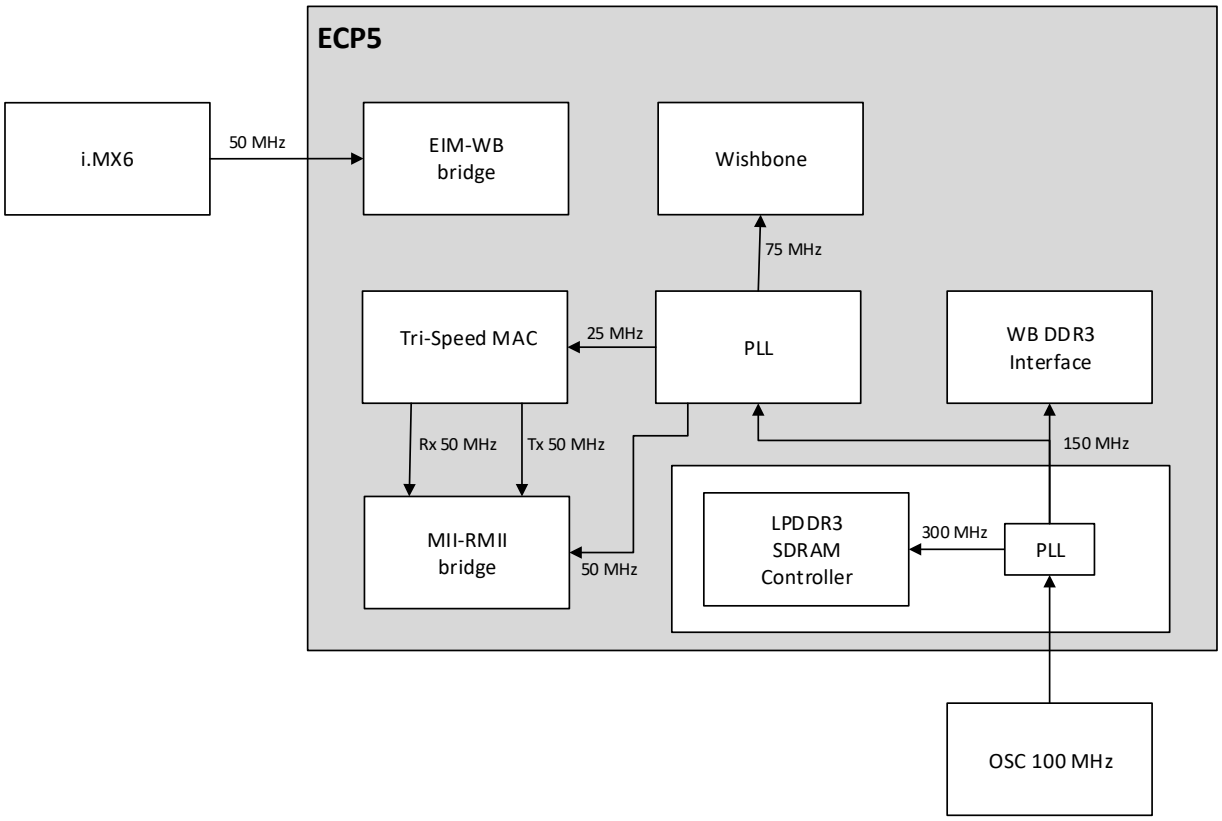

Figure 9 EIM Fast Ethernet Demo clocking scheme

The LPDDR3 SDRAM Controller uses its dedicated onboard 100MHz reference clock (clk_in) to generate a 300 MHz clock for memory access, and a 150 MHz system clock (sclk). The Wishbone DDR3 interface module runs on this 150 MHz system clock.

This 150 MHz system clock (sclk) is used as the input clock for a PLL which generates 125 MHz, 50 MHz, 25 MHz and 75 MHz clocks. The 125 MHz clock is

actually not used. The 75 MHz clock clocks the Wishbone bus. The PLL lock signal from this PLL is used as the reset for the Wishbone bus. The 50 and 25 MHz clocks are used for the RMII interface.

The reset signal (rstn) is generated by a 20-bit wide counter which is initialized on FPGA configuration, and runs on a 77.5 MHz clock generated by an internal oscillator.

5.3 Peripheral addressing

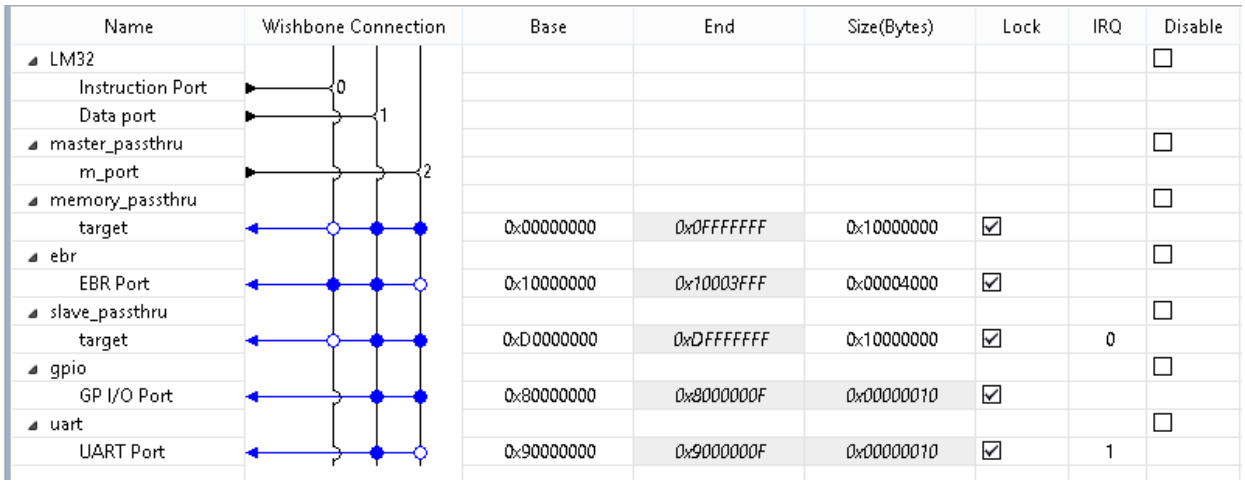

Figure 10 EIM Fast Ethernet Demo Wishbone configuration snapshot

Table 7 EIM Fast Ethernet Demo address mappings

Wishbone address	Peripheral	Note
0x00000000 – 0x0FFFFFFF	LPDDR3	size 256 MB
0x10000000 – 0x10003FFF	EBR	size 32 KB
0xD0000000 – 0xDFFFFFFF	Tri-Speed Ethernet MAC	
0x80000000 - 0x8000000F	Wishsys GPIO	writing 0xFF to 0x80000000 lights up LED0 – LED7
0x90000000 - 0x9000000F	UART	USB UART on connector U33

5.4 EIM interface

EIM (External Interface Module) is a general purpose parallel interface that enables embedded CPUs to control different types of external peripherals and memories.

The details of the EIM interface are described in chapter 22 of the i.MX6 reference manual. This EIM interface is configured to work in Synchronous write/read none multiplexed Operation mode. The transfers are clocked by a 50 MHz ARM-supplied clock (eim_clk). The EIM-WB bridge (iq_bridge_eim2wish.v) translates EIM into Wishbone requests.

5.5 IP versions

Table 8 EIM Fast Ethernet Demo IP Core versions

IP Name	IP Version
Tri-Speed Ethernet MAC	3.7esr
LPDDR3 SDRAM Controller	1.0
PLL	5.8

6 RD0016 – EIM FMC Demo

6.1 Overview

The name of the design for this demo is RD0016_ecp5com_eim_fmc.

This demo shows how the i.MX6 ARM processor can use the EIM interface to communicate with interfaces connected to the FPGA. In this example the interfaces are the SFP cage on a FMC card and LPDDR3 memory.

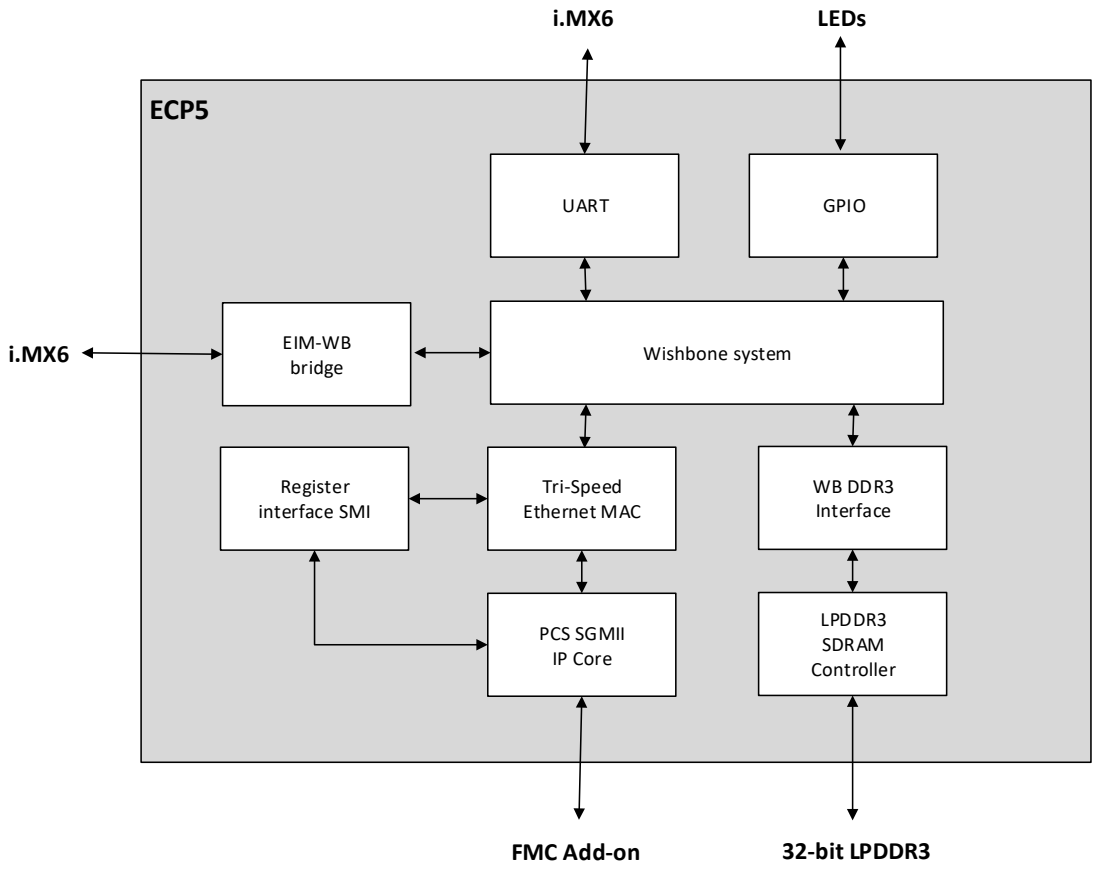

Figure 11 EIM FMC Demo block diagram

Figure 11 shows the block diagram for this demo.

The EIM-WB bridge translates the EIM signals coming from the CPU to requests on the internal Wishbone bus. The internal Wishbone bus interconnect is generated using the Mico System builder. The bus uses 32-bit addresses and 32-bit data.

6.2 Clocking

The LPDDR3 SDRAM Controller uses its dedicated onboard 100MHz reference clock (clk_in) to generate a 300 MHz clock for memory access, and a 150 MHz system clock (sclk). The Wishbone DDR3 interface module runs on this 150 MHz system clock.

This 150 MHz system clock (sclk) is used as the input clock for a PLL which generates 125 MHz, 50 MHz, 25 MHz and 75 MHz clocks. The 125 MHz clock is used for the SGMII interface. The 75 MHz clock clocks the Wishbone bus.

The reset signal (rstn) is generated by a 20-bit wide counter which is initialized on FPGA configuration, and runs on a 77.5 MHz clock generated by an internal oscillator.

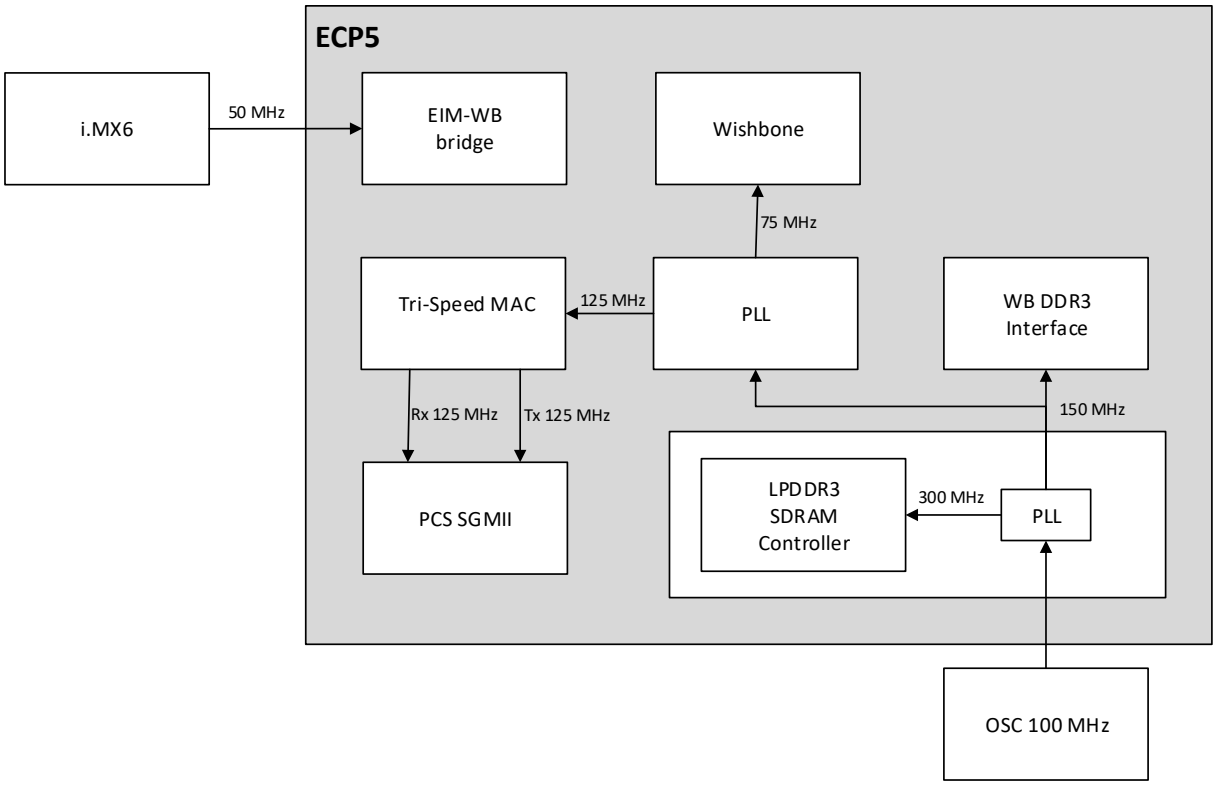

Figure 12 EIM FMC Demo clocking scheme

6.3 Peripheral addressing

Table 9 EIM FMC Demo address mappings

Wishbone address	Peripheral	Note
0x00000000 – 0x0FFFFFFF	LPDDR3	size 256 MB
0x10000000 – 0x10003FFF	EBR	size 32 KB
0xD0000000 – 0xDFFFFFFF	Tri-Speed Ethernet MAC	
0x80000000 - 0x8000000F	Wishsys GPIO	writing 0xFF to 0x80000000 lights up LED0 – LED7
0x90000000 - 0x9000000F	UART	USB UART on connector U33

6.4 FMC Gigabit Ethernet interface

The Gigabit Ethernet interface consists of the:

- ***Tri-Speed Ethernet MAC top model (folder: rtl/ts_mac)*** *– contains the top module of Gigabit Ethernet MAC, including a Wishbone wrapper. It handles the MAC layer of the Ethernet protocol.*
- ***GbE/SGMII PCS IP Core*** *– packs data generated by the MAC module and transmits it using PCS which generates the final serial differential signal. The receive side works similarly.*
- ***Register interface SMI*** *– configures the SGMII PCS IP Core. It communicates with the MAC module over the MDIO interface.*

The PCS SGMII Core is configured to use the SERDES channel (DCU0_CH1) routed to the FMC connector.

6.5 IP versions

Table 10 EIM FMC Demo IP Core versions

IP Name	IP Version
Tri-Speed Ethernet MAC	3.7esr
LPDDR3 SDRAM Controller	1.0
PLL	5.8

7 RD0017 – EIM LVDS Demo

7.1 Overview

The name of the design for this demo is RD0017_ecp5com_lvds.

This design demonstrates the LVDS capabilities of the ECP5 FPGA. The i.MX6 CPU sends read/write data over EIM to the FPGA, where the LVDS interface can be accessed over Wishbone-accessible registers. This demo tests LVDS loopback. The LVDS lines need to be manually soldered to achieve loopback.

Figure 13 shows the block diagram for this demo.

The EIM-WB bridge translates the EIM signals coming from the CPU to requests on the internal Wishbone bus. The internal Wishbone bus interconnect is generated using the Mico System builder. The bus uses 32-bit addresses and 32-bit data. Table 11 shows the address mappings of various peripherals connected to the Wishbone bus.

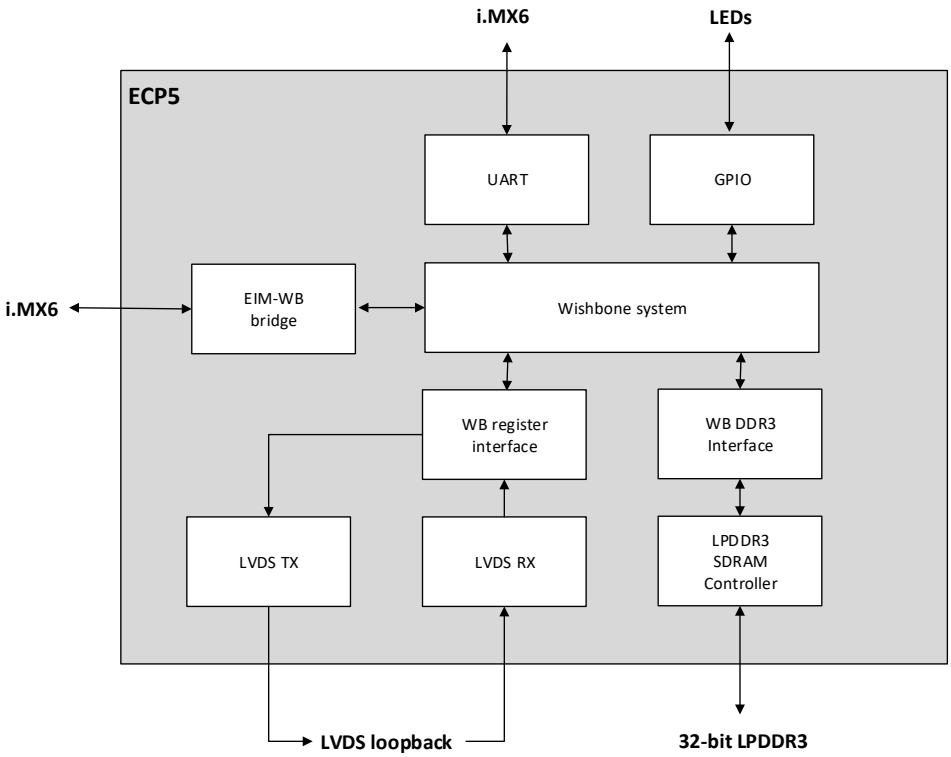

Figure 13 EIM LVDS Demo block diagram

7.2 Clocking

The LPDDR3 SDRAM Controller uses its dedicated onboard 100MHz reference clock (clk_in) to generate a 300 MHz clock for memory access, and a 150 MHz system clock (sclk). The Wishbone DDR3 interface module runs on this 150 MHz system clock.

This 150 MHz system clock (sclk) is used as the input clock for a PLL which generates 175 MHz, 100 MHz and 50 MHz clocks. The 100 MHz clock clocks the Wishbone bus. The 175 MHz clock is used for the LVDS interface.

The reset signal (rstn) is generated by a 20-bit wide counter which is initialized on FPGA configuration, and runs on a 77.5 MHz clock generated by an internal oscillator.

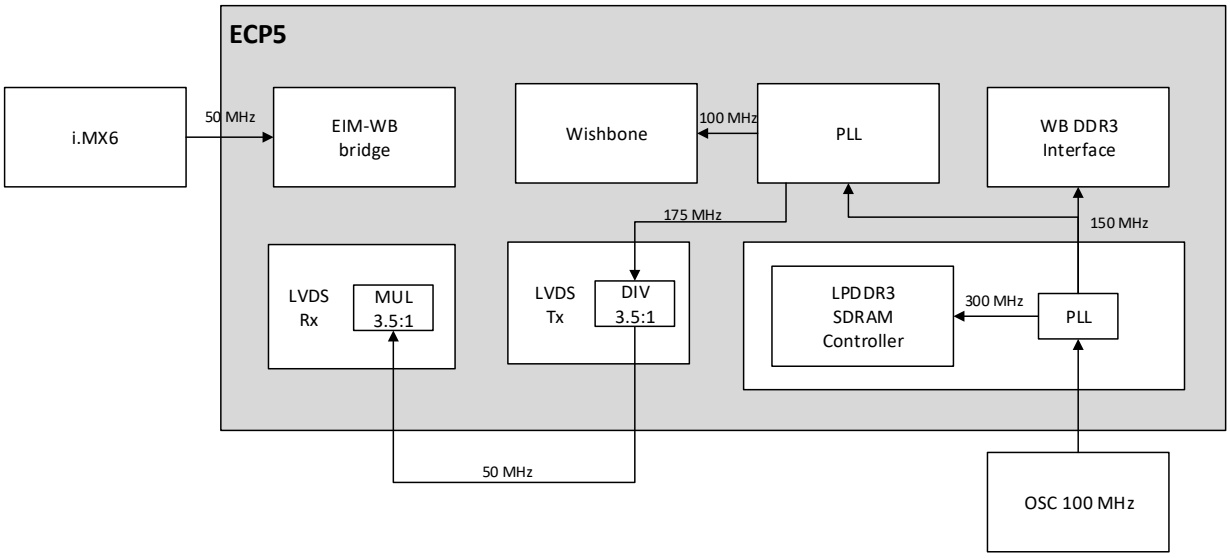

Figure 14 EIM LVDS Demo clocking scheme

7.3 Peripheral addressing

Table 11 EIM LVDS Demo address mappings

Wishbone address	Peripheral	Note
0x00000000 – 0x0FFFFFFF	LPDDR3	size 256 MB
0x10000000 – 0x100007FF	EBR	size 32 KB
0xD0000000 – 0xDFFFFFFF	LVDS register interface	TX register is on address 0xD0000000, RX register is on address 0xD0000100
0x80000000 - 0x8000000F	Wishsys GPIO	writing 0xFF to 0x80000000 lights up LED0 – LED7

7.4 LVDS interface

The LVDS interface consists of two GDDR_7:1 IP cores (one for the receive side, and one for the transmit side), which map 21 bits of data to 3 differential channels and vice-verse. The input to the LVDS transmit module is a register that can be written to from the Wishbone bus. The output of the LVDS receive module is routed to a register that can be read from the Wishbone bus. The Wishbone register interface (wish_regs_itf.v) accomplishes the reading and writing to these registers.

The LVDS transmit module starts transmitting on PLL lock. The data is looped back to the LVDS receive module, and written to a register that is readable from the Wishbone bus. The receive side is clocked with the 50 MHz clock that is sent off chip with LVDS data.

7.5 IP versions

Table 12 EIM LVDS Demo IP Core versions

IP Name	IP Version
GDDR_7:1	6.0
Tri-Speed Ethernet MAC	3.7esr
LPDDR3 SDRAM Controller	1.0
PLL	5.8

8 RD0018 – UART I²C Demo

8.1 Overview

The name of the design for this demo is RD0018_ecp5com_uart_i2c.

This design demonstrates UART and I²C communication capabilities between the i.MX6 CPU and ECP5 FPGA.

The UART interface is instantiated as part of Wishbone System, and the Mico processor is programmed to echo received data.

The I²C RAM slave used is a proprietary Mikroprojekt IP supplied in netlist format.

The design is clocked by the onboard 100 MHz oscillator. The active-low reset signal (rstn) is generated by a 20-bit counter running at 100MHz.

8.2 Peripheral addressing

Table 13: UART i2C Demo - Peripheral addressing

Wishbone address	Peripheral	Note
0x10000000 – 0x10003FFF	EBR	size 32 KB
0x80000000 - 0x8000000F	Wishsys GPIO	writing 0xFF to 0x80000000 lights up LED0 – LED7
0x90000000 - 0x9000000F	UART	USB UART on connector U33

9 PLL Configuration

The name of the design for this demo is RD00019_ecp5com_si5338.

This design demonstrates how to generate the mico32_clock.ngo used in PCIe demos. This Mico 32 processor is generated with a separate project because of tool limitations that make it impossible to instantiate multiple Mico 32 processors in a single design. Figure 15 gives the Wishbone system configuration and addresses.

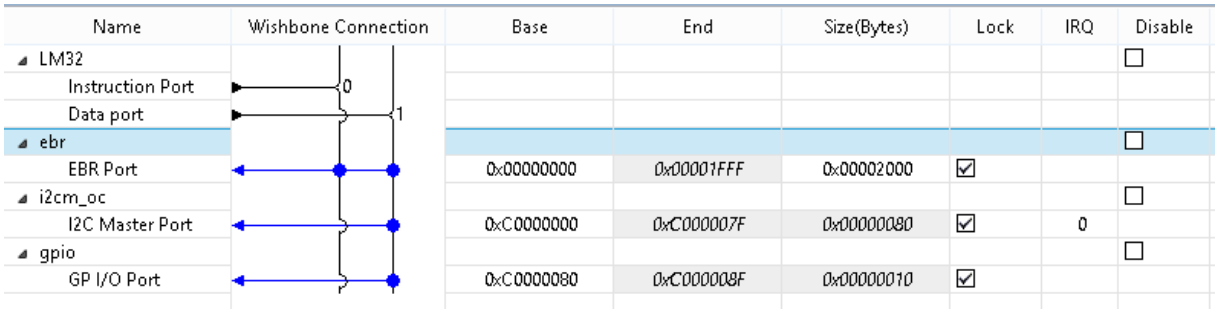

Figure 15 PLL Configuration Wishbone configuration snapshot

The I²C master is used for PLL configuration. The example Si5338 configuration program for PCIe is in the mico32_sw/pcie_test folder.

The GPIO is used for LED indication that the PLL has been successfully configured.

In the strategy for the Diamond project the Disable I/O insertion option needs to be set to True, so that the top-level module of this project can be inserted into other designs (`mico32/soc/mico32.v`).

To generate the desired netlists it is enough to only run the synthesis and translate processes in Diamond.

10 References

Lattice Semiconductors:

- *DS1044 ECP5 familiy datasheet*
- *TN1261 ECP5 SERDES/PCS Usage guide*
- *TN1263 ECP5 sysCLOCK PLL/DLL Design and Usage Guide*
- *TN1265 ECP5 High-Speed I/O Interface*
- *Lattice Diamond Tutorial 3.5*
- *Diamond 3.5 User Guide*
- *LatticeMico32 Tutorial*
- *LatticeMico32 Hardware Developer User Guide*
- *Clarity Designer User Manual*
- *FPGA Libraries Reference Guide*
- *IPUG112 PCI Express x1/x2/x4 Endpoint IP Core User's Guide*
- *UG15 PCI Express Basic Demo Verilog Source Code User's Guide*
- *TN1271 PCIe Sample Demo Debugging and Packet Analysis Guide*
- *IPUG110 LPDDR3 SDRAM Controller IP Core User's Guide*
- *IPUG51 Tri-Speed Ethernet MAC IP User Guide*
- *LatticeMico32 Triple Speed Ethernet MAC*
- *IPUG60 SGMII and Gb Ethernet PCS IP Core User Guide*
- *UG47 LatticeMico32 Tri-Speed Ethernet MAC Gigabit Demo for the LatticeECP3 Versa Evaluation Board User's Guide*
- *TN1265 ECP5 High-Speed I/O Interface*
- *RD1042 WISHBONE UART*
- *RD1054 I2C Slave/Peripheral*
- *RD1140 I2C Slave Controller - Documentation*

Freescale:

- *IMX6SDLRM i.MX 6Solo/6DualLite Applications Processor Reference Manual*

PCI-SIG:

- *PCI Express Base Specification Revision 3.0*

11 Ordering Information

Please contact us via email **contact@mikroprojekt.hr** about item availability and ordering details.

12 Technical Support Assistance

Basic technical product support is free of charge and available via e-mail to all Mikroprojekt customers, whether they are evaluating or have purchased a Mikroprojekt product.

Basic technical support can be requested by sending an e-mail to **support@mikroprojekt.hr**

Our engineers will reply to your request within 2 working days.

Additionally, Mikroprojekt offers to its customers a premium support package, allowing them to be directly supported by Mikroprojekt engineers. The premium support package consists of 10 hours of live, online support, including:

- Phone Support
- Instant Messaging Support
- TeamViewer VNC Interventions.

Furthermore, the customers who purchase our premium support package benefit from direct design assistance of our engineers.

Contact **sales@mikroprojekt.hr** to order the premium support package.

Terms of use

The reproduction, transmission or use of this document or its contents is not permitted without express written authority. Offenders shall be liable for damages.

All rights, including rights created by patent grant or registration of a utility model or design, are reserved.

Technical data is subject to change at any time.

Copyright © 2016 Mikroprojekt d.o.o. All Rights Reserved.

Contact info