MOST Network Interface Controller v1.4

DS543 September 19, 2008 **Product Specification**

Introduction

The LogiCORE™ Media Oriented Systems Transport Network Interface Controller (MOST® NIC) core is a complete controller designed to the *MOST Specification revision 2.4*. When combined with the Xilinx Automotive (XA) solution and embedded processing, the MOST NIC core lets designers leverage the MOST open standard network by providing a higher level of customization in a scalable, flexible design solution.

Features

- Operates in both master and slave modes
- Full bandwidth sustained transfers (24 Mbps)
- Supports full control channel bandwidth utilizing two transmit and receive buffers
- Synchronous, asynchronous, and control channels with 4-60 bytes of synchronous data per frame
- Flexible user interfaces
  - 32-bit OPB interface allows use in Xilinx EDK
  - LocalLink-based streaming port for real-time data access
- Physical to logical channel mapping performed in hardware with 16 transmit and 16 receive logical channels available to the user
- Parameterizable
  - Provision to select master/slave
  - Word count triggers for egress/ingress channel buffers
- Full support of error and status notification
- Internal loopback mode for diagnostic applications
- Support for Ring Break Diagnosis (RBD) test
- MicroBlaze™, PowerPC® driver, and network services from MOCEAN® Laboratories, AB (MOCEAN)
- Low-cost recovery PLL available from Integrated Device Technology, Inc. (IDT®)
- Available through the Xilinx CORE Generator™

LogiCORE Facts				
Core Specifics				
Supported Device Family	Spartan®-3/XA, Spartan-3E/XA, Spartan-3A, Spartan-3AN, Spartan-3A DSP, Virtex®-II Pro, Virtex-4			
Resources Used	I/O	LUTs	FFs	Block RAMs
	7	4450-4590	2620-2670	6
OPB Performance	At least 75 MHz in all supported devices			
Ring Performance	44.1 kHz and 48 kHz frame rates supported			
Provided with Core				
Documentation	Product Specification Getting Started Guide User Guide Release Notes			
Design File Formats	VHDL and Verilog®			
Constraints File	UCF			
Verification	VDHL and Verilog Test Bench			
Instantiation Template	VHDL and Verilog			
Design Tool Requirements				
Xilinx Implementation Tools	ISE® 10.1			
Simulation	Mentor Graphics® ModelSim® v6.3c Cadence® IUS 6.1			
Synthesis	XST			
Support				
Provided by Xilinx, Inc. @ www.xilinx.com/support				

About the MOST NIC

MOST, a growing standard for automotive multimedia networks, is a low-cost, fiber-optic based network providing integration for passenger infotainment networks. The Xilinx MOST NIC core, in conjunction with the Xilinx automotive (XA) solution and embedded processing allows designers to leverage the MOST open standard network by providing a higher level of customization in a scalable and flexible design solution. The MOST NIC core is fully validated using independent system testing.

The MOST NIC core provides unique support for real-time access to the synchronous portion of the MOST frame through the LocalLink interface (a Xilinx standardized user interface), allowing designs to take advantage of powerful parallel-processing capabilities of the Xilinx FPGA family. The core is delivered with a 32-bit OPB bus suitable for both standalone and embedded applications. A complete hardware and software solution is realized when used in conjunction with a MicroBlaze soft processor or hard PowerPC solution, drivers, and MOCEAN Network Services.

Feature Summary

The following summary identifies some of the key features of the MOST NIC core.

Master/Slave Configuration

The MOST NIC core can be configured for master/slave or slave-only mode operation. The master/slave core can be used either as a master or as a slave depending on the user application. When only slave functionality is required, the unused logic is automatically removed for optimal resource utilization in the slave-only core.

- **Master/Slave**. When configured for master/slave mode, the MOST NIC core supports full-ring master capabilities including clock generation and control message initiation. All slave operations are also fully supported. For clock generation, an external clock source is required.
- **Slave only**. Slave only is typically used for the majority of MOST controllers. The slave mode supports the transmission and reception of all MOST data types (synchronous, asynchronous, and control). The clock is recovered from the incoming MOST stream. An external PLL is required for clock recovery and data alignment.

Full Synchronous Channel

- The MOST frame data is received on a timeslot (or physical channel) basis. The MOST NIC supports a range of 4-60 timeslots of synchronous data. The Synchronous Boundary Descriptor (SBD) sets the boundary between the synchronous and asynchronous data in 4 bytes increments, or quadlets.
- The total amount of synchronous and asynchronous data available in a frame is 60 bytes. For this reason, the asynchronous portion of the frame is 60 bytes minus the synchronous data (60-(SBD*4) bytes).
- The remainder of the usable data in the MOST frame (2 bytes) is reserved for control data.

Streaming Port

The MOST NIC core supports a streaming port that provides access to the synchronous data portion of the MOST stream in real time without requiring processing by the host. This port allows both *ingress* (write to the NIC) and *egress* (read from the NIC) transactions. Both transmit and receive data can be accessed by logical channel (see below for more information about logical channels). The data is available *per byte* for this Xilinx LocalLink interface.

- **Receive**. The Receive Streaming port can be used to optionally intercept and/or insert data into the MOST NIC receive path. This allows for additional processing of the data where the external module can act as a sink, source, or a preprocessor of the stream, offloading operations from the host.
- **Transmit**. Similarly, the Transmit Streaming port can be used to optionally intercept and/or insert data into the MOST NIC transmit path. This allows for additional processing of the data where the external module can act as a sink, source, or a preprocessor of the stream, offloading operations from the host.
- **Register Control**. The MOST NIC core contains dedicated memory-mapped register space for the streaming port. Any accesses to these locations is forwarded to the streaming port. The external module can then implement control registers, accessible through the MOST NIC core. Typical applications for the streaming port are synchronous data encryption and decryption.

Logical Channel Mapping

Audio or video data is typically spread over several synchronous timeslots, and as such, grouping related timeslots into a logical channel is efficient for software processing. The MOST NIC core groups physical timeslot data into logical channel data under user control and automatically synchronizes the multiple timeslots on a per-frame basis. The MOST NIC core contains 32 logical channel buffers: sixteen reserved for transmit operations, and sixteen for receive operations. A logical channel can aggregate from one to 60 time slots in any of the synchronous physical channels or can be mapped to the asynchronous channel.

For the receive path, 4 logical channels are dedicated to streaming port ingress access, 4 logical channels for streaming port egress, and 8 logical channels for OPB receive. Similarly, for the transmit path, 4 logical channels are dedicated to streaming port ingress access, 4 logical channels for streaming port egress, and 8 logical channels for OPB transmit. Logical channel # 0 is reserved for asynchronous data in both directions.

Each logical channel buffer is capable of storing up to 32 words of 32-bits each. All buffers can assert interrupts to an external microprocessor to indicate that the buffer requires processing. The word count at which the interrupt is asserted is configurable by the user.

The MOST NIC core provides a host interface for access to control and status registers. The interface is a 32-bit wide OPB bus. The OPB bus is also used to transfer transmit data and receive MOST frame data.

Loopback

The transmit path can be internally connected to the MOST controller receive path for diagnostic tests, allowing the user to receive all frames transmitted. A user can validate that the configuration of the core and the data path are as expected before insertion on the MOST ring.

Applications

The MOST NIC core can be used in many applications to interface to a MOST network. The core can function as a standalone MOST interface or be combined with other LogiCORE IP and EDK cores to build a variety of embedded systems. The MOST NIC solution allows flexible partitioning between software and hardware functions.

Applications include:

- A customized, scalable MOST network controller that unburdens the host processor from network-specific overhead, ideal for applications demanding maximum host application performance
- Augmenting the MOST NIC core with custom IP to process the MOST streaming data with no host processor overhead

The following sections describe common MOST NIC applications in conjunction with additional Xilinx IP cores.

Basic MOST Controller

Figure 1 illustrates a typical application for a basic MOST network interface utilizing the MOST NIC and MicroBlaze cores. The MicroBlaze core contains the MOST driver and the MOCEAN Network Services stack. The MOST NIC core supports transmission and reception of synchronous data, asynchronous data, and control messages.

Synchronous and asynchronous data are aggregated and mapped by the core from physical channels (synchronous channel timeslots of one byte each or asynchronous data) into logical channels. Audio transmission is typically distributed among several discrete *quadlets* on the MOST bus. Grouping these related quadlets into logical channels allows software to process the data more efficiently. In this application, a variety of data types are supported for transmission and reception on the MOST network including audio, video, and control types.

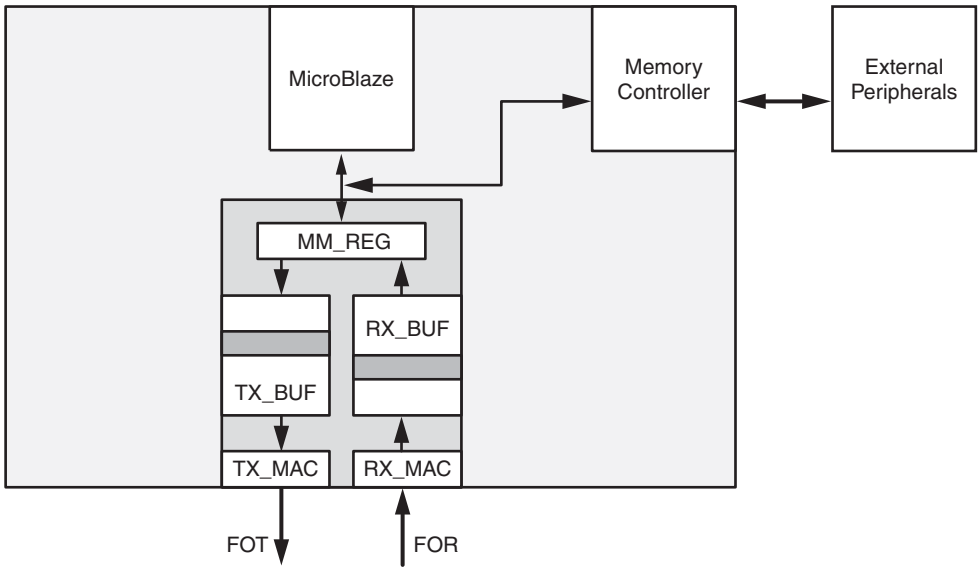

Figure 1: **Basic MOST NIC Configuration**

Audio / Video Encryption and Decryption

Figure 2 illustrates an application involving the MOST NIC core streaming port. The streaming port allows an external core to process the synchronous data portion of the MOST stream in real time. Synchronous channel data can be accessed over the streaming port on a logic channel basis. In this example, a decryption core accesses received synchronous data using the streaming port. After decryption is performed the decrypted data is written to the MOST NIC core receive buffer using the streaming port.

The received data is then accessed using the user interface and processed using software running on the MicroBlaze. Similarly, transmit data is encrypted by an encryption core using the streaming port. The encryption core reads unencrypted synchronous transmit data, encrypts it and then writes the encrypted data to the MOST NIC core transmit buffer using the streaming port. The encrypted data is then queued for transmission on the MOST ring.

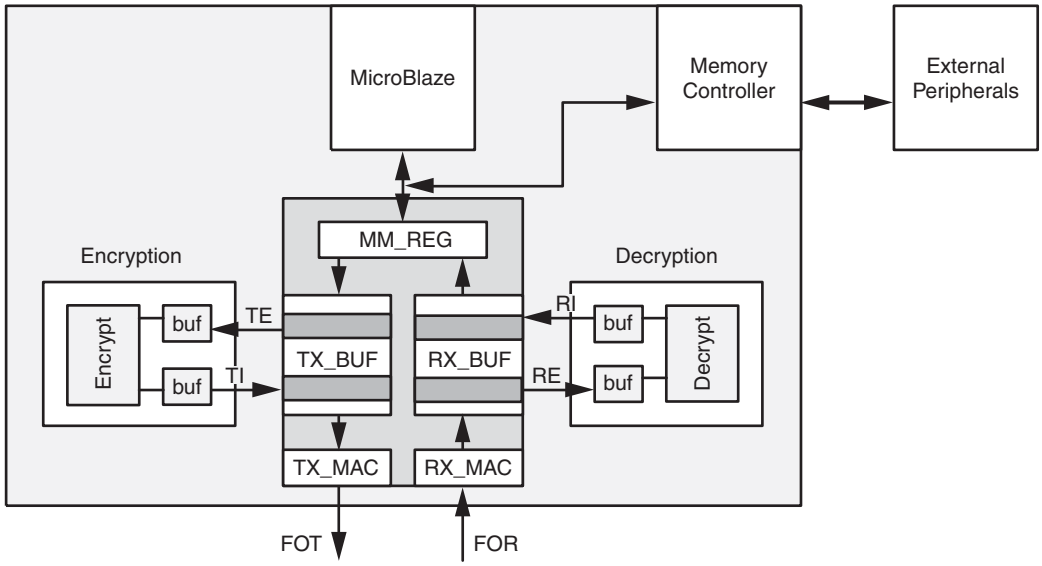

Figure 2: **Encryption/Decryption using the Streaming Port**

Standalone MOST NIC

Figure 3 illustrates a standalone MOST NIC. This example utilizes the MOST NIC core, MicroBlaze core, Centralized DMA controller, I2C, and S/PDIF cores. The MOST NIC transmits and receives synchronous, asynchronous, and control messages to and from the MOST network. Data is organized on a logical channel basis and transferred between system buffers using the 32-bit OPB bus.

Synchronous data is sinked and sourced from external multimedia devices through the S/PDIF core. Control data is transferred to and from external devices through the I2C core. Routing of data between system memory, the MOST NIC, S/PDIF and I2C cores is managed by the Centralized DMA controller.

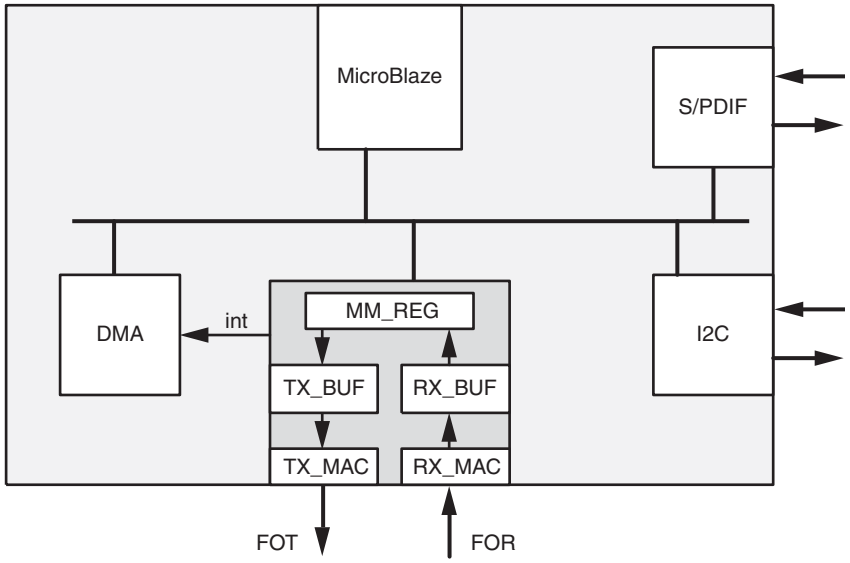

Figure 3: **Standalone MOST NIC**

Functional Overview

The MOST NIC core is a full-featured soft IP core incorporating all necessary logic to interface to a MOST ring. The core supports the transmission and reception of synchronous, asynchronous, and control data to and from the MOST ring.

Module Architecture

Figure 4 illustrates the major sub-modules of the MOST NIC.

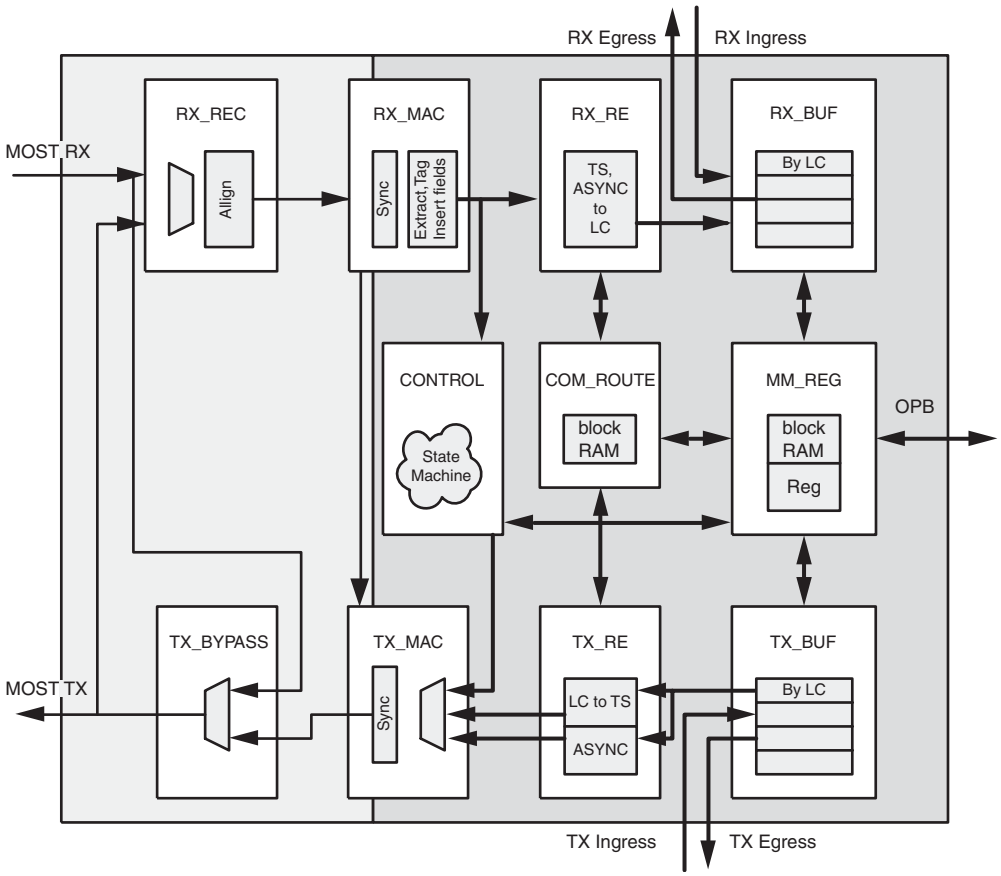

Figure 4: **MOST NIC Block Diagram**

Receive Recovery

The Receive Recovery (RX_REC) module realigns the incoming MOST data to the recovered/corrected MOST clock from the off-chip PLL.

Receive MAC

The Receive MAC (RX_MAC) module decodes the received MOST frame. It has several functions including frame decode, serial-to-parallel conversion, and timing decode.

Receive Routing Engine

The Receive Routing Engine (RX_RE) receives the decoded MOST frame and allocates data to the appropriate buffer for further processing. The RX_RE module utilizes a shared lookup table with the following information:

- Indication of which data to keep and which to discard
- Mapping of received synchronous timeslot data to logical channels

Asynchronous and Synchronous data is written to the appropriate location in the receive buffer (RX_BUF) based on the look up table.

Receive Buffer

The Receive Buffer (RX_BUF) contains sufficient storage for synchronous and asynchronous receive data. Data is organized on a logical channel basis.

Transmit Buffer

The Transmit Buffer (TX_BUF) contains sufficient storage for synchronous and asynchronous transmit data. Data is organized on a logical channel basis.

Transmit Routing Engine

The Transmit Routing Engine (TX_RE) maps logical channels to timeslots.

Transmit MAC

The Transmit MAC (TX_MAC) module encodes synchronous, asynchronous and control data into the transmit MOST frame. This module has several functions including frame encode, and parallel to serial conversion.

Transmit Bypass

The Transmit Bypass (TX_BYPASS) module muxes in the receive serial stream, in the event that this MOST NIC core is operating in a bypass mode.

Common Routing Table

This Common Routing Table (COM_ROUTE) module is a common RAM resource shared by both the receive and transmit paths as a look-up table.

Control Processor

The control processor (CONTROL) handles control messages. Processing of control messages is contained within the MOST NIC core. This submodule decodes the control messages and also generates a suitable response to received control messages. Any received normal message is forwarded through the Memory Mapped Register to the external microprocessor.

The control processor can hold up to two receive messages and up to two transmit messages in addition to the most recent transmit message with the response received on the ring. Any messages that require external processing are forwarded via the Memory Mapped Register interface. These messages are processed via an interrupt driven register interface.

Memory Mapped Registers

The Memory Mapped Register (MM_REG) module contains all registers for control and status of the MOST NIC. All registers are 32-bits wide. For detailed information about the MOST NIC programming and status registers and information about programming them, see "Chapter 6, Configuration Space," and "Chapter 7, Programming the Core," in the *MOST NIC User Guide.*

Clocking and Reset

Clocking

The MOST Controller contains three input clocks: MOST Clocks (`MOST_PLL_CLK`, `MOST_COM_CLK`) and OPB Clock (`OPB_Clk`). The following conditions apply to clock frequencies:

- `MOST_PLL_CLK` and `MOST_COM_CLK` can be either 45.1584 MHz or 49.152 MHz and are frequency-locked to each other.
- `OPB_Clk` is asynchronous to the MOST clocks, and may have a minimum clock frequency of 0.7 times the `MOST_PLL_CLK`. The maximum will depend on the device selected, but all devices will allow at least 75 MHz. The requirements for the DCM must also be met.
- An external clock source is required for Master and Loopback modes.
- An external clock and data recovery PLL is required for `MOST_PLL_CLK` operation.
- The streaming port clock (`STR_CLK`) is an output and identical to the OPB Clock, where the core forwards this clock on behalf of the user.

Reset Mechanism

Two reset mechanisms are provided for the MOST NIC: the `OPB_Rst` input is a hard reset (Table 5), and a software-controlled soft reset is provided through a user configuration register. Both the hard and soft resets cause all logic within the MOST NIC to return to the default state.

Hard Reset

Assertion of `OPB_Rst` causes all logic within the MOST NIC to return to the default state. All configuration registers not located in block RAM are returned to their default values as indicated in the memory map description. Read/Write transactions cannot be performed while the `OPB_Rst` input is asserted.

Soft Reset

Assertion of the soft reset signal causes all logic within the MOST NIC to return to the default state. All configuration registers not located in block RAM with the exception of the soft reset control register are returned to their default values as indicated in the memory map description. Read/Write OPB transactions can be performed starting at the next valid transaction window.

MOST Controller Design Parameters

The MOST NIC is configurable through a set of design parameters accessible through the CORE Generator Graphical User Interface (GUI).

Operating Modes

The core supports two configuration modes: Full master/slave, and slave with pseudo-master modes only (ring break diagnostic and loopback modes).

Word Count Triggers

The MOST NIC core supports 16 logical channel buffers for the receive and transmit directions, for a total of 32 logical channel buffers. In each direction, eight logical channels are dedicated for OPB, four for streaming ingress, and four for streaming egress. All buffers can assert interrupts to an external microprocessor to indicate that the buffer requires processing. The word count at which the interrupt is asserted is configurable, as defined in Table 1.

In addition, all streaming ports also assert a trigger to assist in processing for any logic directly connected to the port. Word counts can be chosen separately for ingress buffers and egress buffers. All ingress buffers use the same word counts, and all egress buffers use the same word counts. The available interrupts for each group of logical channel are also illustrated in Figure 5.

Table 1: **Word Count Parameters**

Name	Possible Values	Default	Description
C_FWC	8, 16	16	**Full Word Count**. The received word count at which an interrupt is generated.
C_EWC	16, 8	16	**Empty Word Count**. The available transmit word count at which an interrupt is generated.

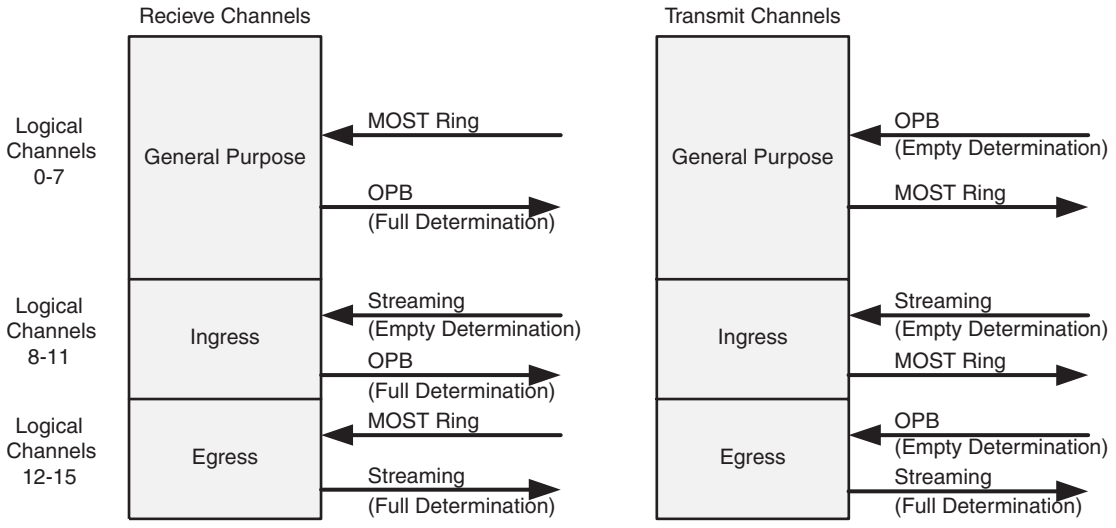

Figure 5: **Logical Channel Support**

Ingress / Egress

Ingress and egress are defined from the reference point of the core. An *ingress* transaction is synonymous with a *write* from the user interface to the core; an *egress* transaction is synonymous with a *read* to the user interface from the core. Table 2 summarizes the user interface access types (as illustrated in Figure 5), and identifies them as either ingress or egress buffers.

Table 2: **Ingress / Egress Buffer Mapping**

Access	Buffer Type	Word Count
RX OPB Read	Egress	Full word count
RX Streaming Read	Egress	Full word count
RX Streaming Write	Ingress	Empty word count
TX OPB Write	Ingress	Empty word count
TX Streaming Read	Egress	Full word count
TX Streaming Write	Ingress	Empty word count

Full Word Count

An interrupt is generated every time a total of C_FWC words have been filled in the egress buffer. Setting the C_FWC to 8 generates an interrupt when eight words (32 bytes) have been written to the buffer by the source. Another interrupt is generated when an additional eight words are written by the source, or an end-of- packet is received for asynchronous data. On initialization, all egress buffers are empty, and as a result, no interrupts are generated until the appropriate data is written.

Empty Word Count

An interrupt is generated every time a total of C_EWC have been removed in the ingress buffer. Setting the C_EWC to 8 generates an interrupt when there is room for at least eight words (32 bytes) to be written to the ingress buffer. When the interrupt is asserted, there is an assumption that the external processor will write eight words to the ingress buffer. In essence, eight locations of the buffer are now reserved and no longer considered unused. When or if there are eight more unused word locations, an additional interrupt is generated.

On initialization, all ingress buffers are empty. The user application must prime the ingress buffer with data, and then enable the logical channel. Interrupts are then generated when there is C_EWC words free in the buffer. After the interrupt is cleared, an additional interrupt can be generated depending on the setting of C_EWC and the number of unused word locations remaining.

OPB Support

The MOST NIC is an OPB slave. The address of the MOST NIC can be configured using the CORE Generator GUI.

Table 3: **OPB Base Address Parameter**

Name	Possible Values	Default Value	Description
C_BASEADDR	Valid address range: 0x00000000 - 0xffffc000, where bits 18-31 must be 0b	h'0000_0000	Base Address for the core

Core Interfaces

This section describes the interface signals of the MOST NIC core. The major MOST NIC interfaces include the following:

- MOST Ring Interface
- Host Interface
- Streaming Port Interface

MOST NIC Interfaces

Figure 6 illustrates the MOST NIC interfaces. All signals are defined in their respective sections following the illustration.

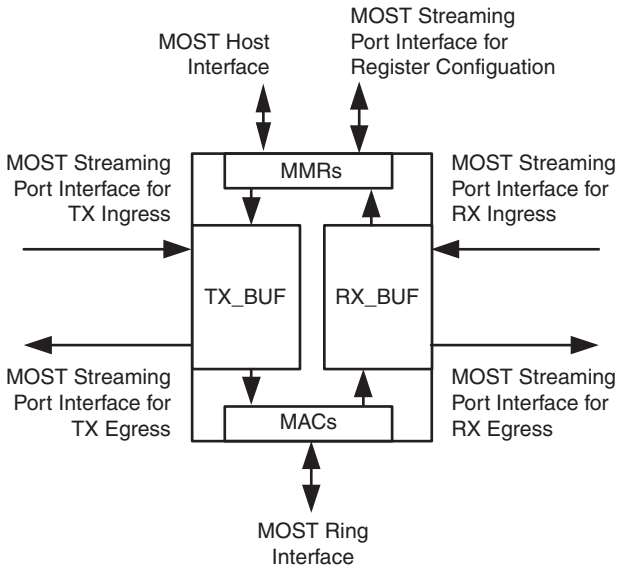

Figure 6: **MOST Core Interfaces**

MOST Ring Interface

Table 4 defines the signals (MOST_) interfacing to the MOST ring via an external PHY and DCM.

Table 4: **MOST Ring Interface**

Signal Name	Direction	Description
MOST_TX	Output	**Transmit MOST Data**: The serial bit stream transmitted onto the MOST ring.
MOST_RX	Input	**Receive MOST Data**: The serial bit stream received from the MOST ring.
MOST_PLL_CLK	Input	**MOST PLL Clock**: The clock recovered from the external PLL. Received data is sampled with this clock.
MOST_COM_CLK	Input	**MOST Common Clock**: The clock used internally to the core. For a slave-only core, the MOST common clock can be connected to MOST_PLL_CLK. For a master-slave core, the source when in Master mode is the external crystal, and the source when in slave mode is MOST_PLL_CLK.

Table 4: **MOST Ring Interface *(Cont'd)***

Signal Name	Direction	Description
MOST_EXT_NRESET	Output	**MOST External Not-Reset**: This active low signal can be used to drive a reset externally to the core. It is suggested to connect this to the reset of the PLL, in order to reset the PLL directly from software.
MOST_EXT_BYPASS	Output	**MOST External Bypass**: This signal can be used to control the state of the external PLL by placing it into bypass mode. This signal is not related to the programming of the MODE select register.
MOST_FOR_STATUS	Input	**Fiber Optic Receiver Status**: Indicates availability of the external FOR.
MOST_PLL_LOCK	Input	**PLL Lock**: The external PLL lock status negates this signal to indicate the receive clock is locked. This connection is not mandatory as the core examines the quality of the input clock to detect a loss of lock.
MOST_MASTER	Output	**MOST Master**: Indicates if this node is acting as the master. This is used to control the input of the off chip PLL to be either the RX line (for acting slaves) or an off chip crystal (for acting masters). This is applicable for master configured nodes as well as slaves in Ring Break Diagnostics acting as a master.

Host Interface

Table 5 defines the MOST host interface signals (`OPB_*/Sln_*`). The MOST NIC contains a 32-bit OPB slave interface bus to interface to the PowerPC, MicroBlaze, or other microprocessors. There are separate read and write data buses with a shared address bus. The host interfaces is used to access control registers as well as transmit / receive data.

Byte Enable Functionality

The MOST NIC core implements a superset of the standard OPB interface with regards to byte enables. All OPB accesses are considered long word aligned accesses, that is, bits 0 and 1 of the address are ignored, and assumed to be 00b. The byte enable signals are used to qualify which byte(s) are selected within the long word access.

If `OPB_BE[0]` is asserted, `OPB_DBus[0:7]` are selected for access

If `OPB_BE[1]` is asserted, `OPB_DBus[8:15]` are selected for access

If `OPB_BE[2]` is asserted, `OPB_DBus[16:23]` are selected for access

if `OPB_BE[3]` is asserted, `OPB_DBus[24:31]` are selected for access

The `OPB_BE[0:3]` bus can be driven with any value from 0x0 to 0xF.

Burst Support

The OPB interface accepts both burst and single cycle transactions to the control registers. However, only the TX Buffer (`TXBUFF`), RX Buffer (`RXBUFF`), Common Routing Table (`CRT`), and Master Allocation Tables (`MAT`) provide optimized pipelined data acknowledgement for burst requests. Burst requests to all other registers are treated as consecutive single-cycle access.

Table 5: **MOST Host Interface**

Signal Name	Direction	Description
System Signals		
OPB_Clk	Input	**Clock**: All host interface signals are synchronous to this clock.
OPB_Rst	Input	**Reset**: The MOST core is reset to the default state upon assertion of this signal.
MOST_Irpt	Output	**Interrupt**: Asserted to indicate a MOST Controller Status interrupt condition to the microprocessor or interrupt controller.
BUFFER_Irpt	Output	**Interrupt**: Asserted to indicate a Buffer interrupt condition to the microprocessor or interrupt controller.
OPB Slave Request Signals		
OPB_select	Input	**Select**: Indicates an active read or write access. This signal qualifies all bus inputs from the OPB master.
OPB_RNW	Input	**Read not Write**: A logic '1' indicates a read access to the location address by OPB_Abus. A logic '0' indicates a write access to the location addressed by OPB_Abus.
OPB_seqAddr	Input	**Sequential Transfer**: Indicates that the transfer being performed will be followed with a transfer to the next sequential address in the same direction, read or write. For any given burst transaction, OPB_seqAddr needs to be asserted for every address except the last
OPB_ABus[0:31]	Input	**Address**: Bus used to specify the address being accessed either for a read or write.
OPB_DBus[0:31]	Input	**Write Data Bus**: Data to be written to the address specified by OPB_Abus. The write is acknowledged by Sln_xferAck when complete.
OPB_BE[0:3]	Input	**Byte Enable**: Selects which byte lane of the data bus is being accessed.
Sln_DBus[0:31]	Output	**Read_Data**: Data read from the address OPB_ABus. Data is valid when a read request followed by an acknowledge (Sln_xfer Ack) is asserted.
Sln_xferAck	Output	**Acknowledge**: Acknowledgement of a completed read or write transfer. Following a read request, indicates that the data on Sln_Dbus is valid. Following a write request, indicates that the data on OPB_DBus was accepted.
Sln_Retry	Output	**Retry**: Asserted to indicate the MOST core is unable to perform the transfer requested at this time. This signal will be asserted instead of Sln_xferAck when required. This signal is not used for this release of the core.
Sln_ToutSup	Output	**Time Out Suppress**: Asserted to indicate the OPB arbiter that the bus operation will be delayed for an extended period of time. This signal is not used for this release of the core.
Sln_ErrAck	Output	**Error Acknowledge**: Asserted to indicate an error was encountered during the requested transfer. Asserted coincident with Sln_xferAck if asserted. This signal will be asserted when a transaction forwarded to the streaming memory map does not respond within the required time

Streaming Port

Table 6 defines the MOST NIC steaming port signals (STR_*). The streaming port has five main components: the Memory Map Register and the four Data Transfer interfaces.

Memory Map Register

A subset of the MOST NIC memory map is reserved for the streaming port. Access to these locations are forwarded to the streaming port. An external module interface to the streaming port can then implement control register specific to that module (but accessed through the MOST NIC memory map). This group also contains an interrupt port such that the external module can assert an interrupt source within the MOST NIC.

Data Transfer

The MOST NIC core includes one receive and one transmit streaming port based on the LocalLink interface for data transfer. The streaming ports are used to access synchronous channel data in the transmit and receive FIFOs on a logical channel basis. Both the transmit and receive streaming ports have ingress and egress capabilities.

As a general rule, the logical channel width is 4 bits. However, for the streaming ports, the logical channels are split by port type. Internal to the core, all egress logical channel values have a prefix of 11b and all ingress logical channel values have a prefix of 10b.

The streaming port contains interrupt signals to indicate logical channel buffer word count status. The word count interrupt signals indicate when the logical channel buffers have enough data to be processed, as defined by the full and empty word count parameters in Table 1, "Word Count Parameters," on page 10. The interrupt is a one-clock wide active high pulse synchronous to the STR_CLK.

The interrupts are generated by this controller and processed as required by the external module. In the case of egress buffers, an interrupt is asserted when there is sufficient data in the FIFO that can be read by the streaming port. In the case of ingress buffers, an interrupt is asserted when there is sufficient free space in the FIFO to allow for additional write data from the streaming port.

Table 6: **MOST Streaming Port Interface**

Signal Name	Direction	Description
Common to All Interfaces		
STR_CLK	Output	**Clock:** All signals on the streaming port are synchronous to this clock
STR_RST	Output	**Reset:** An active high reset signal that is asserted when ever the MOST core is reset (by either hard or soft reset)
Memory Map		
STR_MMR_REQ	Output	**Memory Map Request:** Assertion indicates an active read or write to the streaming port memory mapped signals. The REQ signal qualifies all other streaming port memory map signals. The REQ signal is deasserted when the transfer is complete (STR_MMR_ACK) sampled high.
STR_MMR_RNW	Output	**Memory Map Read / Write:** Indicates the direction of the memory map transfer. Logic '1' indicates reads. Logic '0' indicates writes. Asserted coincident with STR_MMR_REQ. The STR_MMR_RNW signal must remain constant during the memory map access.
STR_MMR_ADDR[0:7]	Output	**Memory Map Address:** The address being written to or read from.

Table 6: **MOST Streaming Port Interface *(Cont'd)***

Signal Name	Direction	Description
STR_MMR_BE[0:3]	Output	**Memory Map Byte Enable:** Selects which byte lane of the data bus is being accessed.
STR_MMR_WDATA[0:31]	Output	**Memory Map Write Data:** The data being written to the streaming port. Asserted coincident with STR_MMR_REQ. The STR_MMR_WDATA signals will remain constant during the memory map access. This signal is only valid when MMR_RNW = '0.'
STR_MMR_RDATA[0:31]	Input	**Memory Map Read Data:** Data read from the streaming port external module. Data is valid when a read request followed by an acknowledgment STR_MMR_ACK is asserted
STR_MMR_ACK	Input	**Memory Map Acknowledge:** Asserted by the target to indicate completion of a write or read cycle. When asserted the cycle is complete and returned read data is latched. The STR_MMR_REQ signal is deasserted immediately after assertion of ACK to complete the transaction.
STR_MMR_INT	Input	**Interrupt In:** The target asserts this signal for 1 clock duration to set the MOST interrupt status register bit assigned to the streaming port. If enabled this will cause an interrupt to the external microprocessor. The STR_MMR_INT signal must be asserted for 1 clock duration only.
Data Transfer: Receive Streaming Port–Egress		
STR_RE_LC[0:1]	Input	**Receive Egress Logical Channel:** Logical channel for the data being requested
STR_RE_DATA [0:7]	Output	**Receive Egress Data:** Data read from to the logical channel selected on STR_RE_LC. Data is valid when a read request followed by an acknowledgment by requestor (SRC_RDY) and target (DST_RDY) is asserted
STR_RE_BIF_AVAIL[0:3]	Output	**Receive Egress Buffer Interface Available**: When asserted, data is available in the given logical channel buffer. This signal is deasserted when the core is not enabled and triggers low in the event of the logical channel flush from the Host Interface.
STR_RE_SRC_RDY	Output	**Receive Egress Source Ready:** Read data on the STR_RE_DATA is available and valid.
STR_RE_DST_RDY	Input	**Receive Egress Destination Ready:** The requestor is ready for data for this logical channel (STR_RE_LC)
STR_RE_WCINT[0:3]	Output	**Receive Egress Interrupt:** Asserted for 1 clock when the receive egress logical channel buffer has crossed the word count as configured by the full word count parameter (FWC). There is one signal for each receive read logical channel.
Data Transfer: Receive Streaming Port–Ingress		
STR_RI_LC[0:1]	Input	**Receive Ingress Logical Channel:** Logical channel for the data being written.
STR_RI_DATA[0:7]	Input	**Receive Ingress Data:** Data being written the logical channel selected on STR_RI_LC. Data is accepted when a write request followed by an acknowledgment by requestor (SRC_RDY) and target (DST_RDY) is asserted

Table 6: **MOST Streaming Port Interface *(Cont'd)***

Signal Name	Direction	Description
STR_RI_BIF_AVAIL[0:3]	Output	**Receive Ingress Buffer Interface Available**: When asserted, data is available in the given logical channel buffer. This signal is deasserted when the core is not enabled and triggers low in the event of the logical channel flush from the Host Interface.
STR_RI_SRC_RDY	Input	**Receive Ingress Source Ready:** Write data on the STR_RI_DATA is available and valid.
STR_RI_DST_RDY	Output	**Receive Ingress Destination Ready:** The target has accepted the write data for this logical channel (STR_RI_LC)
STR_RI_WCINT[0:3]	Output	**Receive Ingress Interrupt:** Asserted for 1 clock when the receive ingress logical channel buffer has crossed the word count as configured by the empty word count parameter (EWC). There is one signal for each receive write logical channel.
Data Transfer: Transmit Streaming Port–Egress		
STR_TE_LC[0:1]	Input	**Transmit Egress Logical Channel:** Logical channel for the data being requested
STR_TE_DATA [0:7]	Output	**Transmit Egress Data:** Data read from the logical channel selected on STR_TE_LC. Data is valid when a read request followed by an acknowledgment by requestor (SRC_RDY) and target (DST_RDY) is asserted
STR_TE_BIF_AVAIL[0:3]	Output	**Transmit Egress Buffer Interface Available**: When asserted, data is available in the given logical channel buffer. This signal is deasserted when the core is not enabled and triggers low in the event of the logical channel flush from the Host Interface.
STR_TE_SRC_RDY	Output	**Transmit Egress Source Ready:** Read data on the STR_TE_DATA is available and valid.
STR_TE_DST_RDY	Input	**Transmit Egress Destination Ready:** The requestor is ready for data for this logical channel (STR_TE_LC)
STR_TE_WCINT[0:3]	Output	**Transmit Egress Interrupt:** Asserted for 1 clock when the transmit egress logical channel buffer has crossed the word count as configured by the full word count parameter (FWC). There is one signal for each transmit read logical channel
Data Transfer: Transmit Streaming Port–Ingress		
STR_TI_LC[0:1]	Input	**Transmit Ingress Logical Channel:** Logical channel for the data being written
STR_TI_DATA[0:7]	Input	**Transmit Ingress Data:** Data being written to the logical channel selected on STR_TI_LC. Data is accepted when a write request followed by an acknowledgment by requestor (SRC_RDY) and target (DST_RDY) is asserted
STR_TI_BIF_AVAIL[0:3]	Output	**Transmit Ingress Buffer Interface Available**: When asserted, data is available in the given logical channel buffer. This signal is deasserted when the core is not enabled and triggers low in the event of the logical channel flush from the Host Interface.

Table 6: **MOST Streaming Port Interface *(Cont'd)***

Signal Name	Direction	Description
STR_TI_SRC_RDY	Input	**Transmit Ingress Source Ready:** Write data on the STR_TI_DATA is available and valid.
STR_TI_DST_RDY	Output	**Transmit Ingress Destination Ready:** The target has accepted the write data for this logical channel (STR_TI_LC)
STR_TI_WCINT[0:3]	Output	**Transmit Ingress Interrupt:** Asserted for 1 clock when the transmit ingress logical channel buffer has crossed the word count as configured by the empty word count parameter (EWC). There is one signal for each transmit write logical channel.

Streaming Port Receive Egress (Read from Streaming Port)

The streaming ports use LocalLink signaling, where either the core or the external requesting interface are free to negate ready, indicating that a transmission is stalled for this cycle. Only when both source and destination ready are asserted does a transfer occur. The user can change the logical channel at any time as long `*_DST_RDY` is deasserted the previous cycle, although this can result in less than full throughput. Typically, the number of bytes read should match four times the word count as configured by the full word count parameter (`FWC`). Figure 7 illustrates two example transactions where the external requestor stalls the access on logical channel 0, and the MOST NIC stalls the access on logical channel 1.

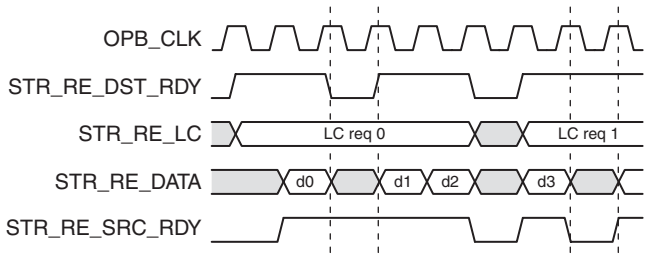

Figure 7: **Receive Streaming Port Read**

Streaming Port Receive Ingress (Write to Streaming Port)

The write port is similar to the read port. Typically, the number of bytes written should match four times the word count, as configured by the empty word count parameter (EWC). An application can watch the standard word count flags to trigger this access, or, alternatively, the hardware component can keep accessing data from any given channel, which the core provides as long as data exists. Figure 8 illustrates sample transactions where the external requestor is stalling on logical channel 0, and the MOST NIC is stalling on logical channel 1.

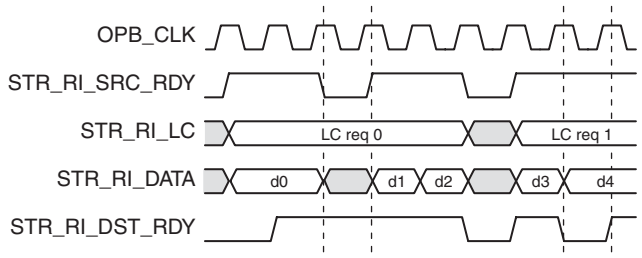

Figure 8: **Receive Streaming Port Write**

Streaming Port Transmit Egress and Ingress

The operation of the Transmit Egress streaming port is similar to the Receive Egress streaming port. The operation of the Transmit Ingress streaming port is similar to the Receive Ingress streaming port.

Configuration Space

Table 7 defines the MOST NIC control registers, including those that support byte enables. In addition, information about supported byte-enable locations is provided. All registers are 32-bits wide. For detailed information about the control registers, including those located in block RAM and base address offsets, see Chapter 6, "Configuration Space," in the *MOST NIC User Guide*.

Byte Enable Support

Locations that do not support byte enables ignore the `OPB_BE[0:3]` signals and provide 32-bit reads and writes. Locations that support byte enables qualify writes with the assertion of `OPB_BE[0:3]`. Read access ignores the `OPB_BE[0:3]` signal and provides 32-bit read data.

Reserved Bits

Reads to write-only registers, reserved registers, and reserved bits return zero(s). Writes to any read-only registers or bits are ignored.

Table 7: **MOST Controller Registers**

Name	Register Name	Byte Enable	Description
MOST Controller Registers			
Version	VR	N	Indicates capability and version information about the core.
Soft Reset	SRR	Y	Places the MOST NIC in reset mode and enables the core for active operation.
Mode Select	MSR	Y	Configures the operating mode of the controller. This register can be read back in order to determine that it has been programmed correctly. However, in order to determine the actual state of the core, the Status Register (SR) must be read.Configures the operating mode of the controller.
Status	SR	N	Indicates the functional state of the MOST NIC.
Channel Control	CCR	Y	Configures information regarding the synchronous and asynchronous channels. Specifically, the Channel Control Register sets the boundary of the MOST frame dedicated to synchronous and asynchronous data when a Master. This value determines the boundary between the synchronous and asynchronous data area in the frame. When a slave, this register is read-only.
Maximum and Current Position and Delay	MCPDR	N	Provides the maximum number of devices (position) and the maximum delay value of the MOST network. It also provides the current position and delay value of the device.
Logical Address	LAR	Y	Configures the logical address of this node.
Alternate Address	AAR	Y	Configures the alternate address of this node.
Group Address	GAR	Y	Configures the group address of this node.

Table 7: **MOST Controller Registers *(Cont'd)***

Name	Register Name	Byte Enable	Description
Flush Control	FCR	Y	Register allows the Synchronous, Asynchronous and Control Buffers to be flushed.
Transmit Buffer Error	TXBER	N	Indicates which transmit logical channel buffers have underflow or overflow errors.
Receive Buffer Error	RXBER	N	Indicates which receive logical channel buffers have underflow or overflow errors.
MOST Control Processing			
Message Retry Count and Delay	MRCDR	Y	Contains the retry count and delay value for the re-transmission of the control messages. The count value configures how many times the re-transmission should take place and the delay value configures the gap between the re-transmission.
Control Transmit Status	CTXS	Y	Register contains status and control for the transmission of control messages.
Control Receive Status	CRXS	N	Register contains status and control for the reception of control messages.
Control Transmit FIFO	CTXFIFO	N	A write buffer used to add control messages for transmission. Control messages can be 2 to 6 words in length. The control transmit message FIFO contains sufficient storage to hold one complete message. The control module within the core will also queue a message, for a total storage ability of 2 messages. The FIFO is accessed by writing words sequentially to the CTXFIFO.
Control Transmit Response FIFO	CTXRESFIFO	N	A read buffer used to read transmit message responses and status. Control messages can be up to from 2 to 6 words in length. A seventh word is used to return the transmission status. The control transmit response FIFO contains sufficient storage to hold one complete message, including the request, response and status. The buffer is accessed by reading words sequentially from the CTXRESFIFO.
Control Receive FIFO	CRXFIFO	N	A read buffer used to read received control messages. Control messages can be up to from 2 to 6 words in length. The control receive message FIFO contains sufficient storage to hold two complete messages. The buffer is accessed by reading words sequentially from the CRXFIFO.
Interrupt Status/Control			
MOST Interrupt Status	MISR	Y	Contains bits that are set when a particular interrupt condition occurs. If the corresponding mask bit in the MOST Interrupt Enable Register (MIER) is set, an interrupt is generated.
MOST Interrupt Pending	MIPR	N	Indicates the interrupts that are actually indicated to software. In effect, it is the MISR register bit wise ANDed with the MIER register.
MOST Interrupt Enable	MIER	Y	Used to enable or disable the generation of the interrupts. Interrupts are enabled by setting the appropriate bit in the MIER.
MOST Interrupt Clear	MICR	Y	Used to clear interrupt status bits. Writes of '1' clear the appropriate bit in the MOST Interrupt Status Register (MISR) and correspondingly the MOST Interrupt Pending Register (MIPR).

Table 7: **MOST Controller Registers *(Cont'd)***

Name	Register Name	Byte Enable	Description
Buffer Interrupt Status	BISR	Y	Contains bits that are set when a particular interrupt condition occurs. If the corresponding mask bit in the Buffer Interrupt Enable Register (BIER) is set, the BUFFER_irpt interrupt is generated.
Buffer Interrupt Pending	BIPR	N	Indicates the interrupts that are actually indicated to software. In effect, it is bit-wise the BISR register bit ANDed with the BIER register.
Buffer Interrupt Enable	BIER	Y	Used to enable or disable the generation of the interrupts. Interrupts are enabled by setting the appropriate bit in the BIER.
Buffer Interrupt Clear	BICR	Y	Used to clear interrupt status bits. Writes of '1' clear the appropriate bit in the BISR and the BIPR.
Routing Table			
Common Routing Table	CRT	N	Used to configure the mapping of timeslots to logical channels for both the transmit and receive data paths of the MOST controller. The MOST frame consists of up to 60 synchronous timeslots or asynchronous data. Each word in the routing table controls the mapping of 4 timeslots. Logical channel #0 is reserved for asynchronous data and as such can not be used for any synchronous timeslots.
Logical Channel Enable	LCE	Y	Used to enable/disable channels on a logical channel basis.
Slave Active 1	SAR1	N	Reflects the active bits set by slave nodes during the transmission of the Network Information Channel on the MOST Ring. Not all bits are necessarily accurate, depending on the node's position within the ring.
Slave Active 2	SAR2	N	
Master Allocation Table	MAT	N	Used to read the assignments of synchronous timeslots to various controllers on the ring. This table is only valid when this node is a Master.
Streaming Port MMR			
Streaming Port MMR	STR_MMR	Y	This portion of the memory map is dedicated to external modules connected to the streaming ports. Any accesses to this portion of the memory mapped are forwarded to the streaming port through the Streaming Port MMR bus (STR_MMR).
MOST PLL Control Registers			
PLL Reference Count Register	PRCR	Y	Used by the PLL clock monitoring logic to evaluate the quality of the clock generated by the PLL. If a problem was detected, the core will automatically reset the external PLL.
External PLL Reset Count	EPRC	N	Indicates the number of times reset was applied to the external PLL.

Table 7: **MOST Controller Registers *(Cont'd)***

Name	Register Name	Byte Enable	Description
Transmit/Receive Buffer			
Transmit Buffer	TXBUFF	N	This portion of the memory map allows for FIFO based access to the transmit buffers. The transmit buffer is organized on a logical channel basis. The transmit buffer supports a total of 16 logical channels. Each buffer contains 32 words. To ease in programming the Logical Channel # forms the 3rd Most Significant Nibble of the Transmit Buffer Address. As such the address space supports 64 words per logical channel. However, each logical channel FIFO is only 32 words deep. Figure 5 illustrates the memory map with respect to channel type.
Receive Buffer	RXBUFF	N	This portion of the memory map allows for FIFO based access to the receive buffers. The receive buffer is organized on a logical channel basis. The receive buffer supports a total of 16 logical channels. Each buffer contains 32 words. To ease in programming the Logical Channel # forms the 3rd Most Significant Nibble of the Receive Buffer Address. As such the address space supports 64 words per logical channel. However, each logical channel FIFO is only 32 words deep. Figure 5 illustrates the memory map with respect to channel type.

Resource Utilization and Performance

The MOST NIC core can be generated in several configurations to provide maximum flexibility for specific user applications. For this reason, resource utilization depends on the core configuration selected by the user. Table 8 provides sample resource utilization numbers for the MOST NIC core The most effective method for determining resource requirements for a user application is to generate a core with the required feature set, and then run the provided implementation script.

Table 8: **MOST NIC Core Resource Utilization**

Configuration	LUTs	FFs
Maximum: Full Master/Slave core, word counts set to 16	4,590	2,670
Minimum: Slave only core, word counts set to 8	4,450	2,620

The maximum operating frequency of the MOST NIC core depends on the selected target device, where all supported device families operate at a minimum of 75 MHz in the lowest speed grades. Note that the User Constraints File (UCF) provided with the core is set to a default value equivalent to the MOST_PLL_CLK and will not match the maximum performance numbers defined. However, the user can adjust the constraints through the UCF, as described in the *MOST NIC User Guide*.

Verification

Verification of the MOST NIC core has been conducted using both simulation and hardware testing to ensure compatibility with industry-standard MOST controllers:

- **Simulation**. The MOST NIC has been extensively verified in functional simulations utilizing advanced industry-standard verification techniques.
- **Hardware Validation**. The MOST NIC core and an example design have been tested in hardware. Hardware validation includes interoperability testing with other MOST compatible controllers.

References

1. *MOST Specification revisions 2.4 and 2.5*

2. *On-Chip Peripheral Bus Architecture Specification version 2.1*

3. *OPB IPIF (v2.00.j) Product Specification*

4. *LocalLink Specification* (To access the LocalLink specification, you must register on the LocalLink product page.)

Support

Xilinx provides technical support for this LogiCORE product when used as described in the product documentation. Xilinx cannot guarantee timing, functionality, or support of product if implemented in devices that are not defined in the documentation, if customized beyond that allowed in the product documentation, or if changes are made to any section of the design labeled *DO NOT MODIFY*.

Ordering Information

The Xilinx MOST NIC core is provided under the SignOnce IP Site License and can be generated using the CORE Generator v10.1 SP3 or higher, shipped with the Xilinx ISE Foundation Series software at no additional charge. For more information about the MOST NIC core, please visit the MOST NIC product page.

A Simulation Only evaluation license is provided with the core through the CORE Generator. To access the full functionality of the core, including FPGA bitstream generation, a Full System Hardware Evaluation or Full license must be obtained from Xilinx after satisfying the following requirements:

- You must be a member of the MOST Cooperation (www.mostcooperation.com)
- You must complete the Xilinx MOST license agreement

Instructions for completing these requirements are included on the MOST NIC product page.

Please contact your local Xilinx sales representative for pricing and availability of additional Xilinx LogiCORE modules and software. Information about additional Xilinx LogiCORE modules is available on the Xilinx IP Center.

Revision History

Date	Version	Revision
05/17/07	1.1	Initial Xilinx release.
08/08/07	1.2	Support for Spartan-3A DSP added.
04/25/08	1.3	No functional changes.
09/19/08	1.4	Automatic lock detection enhancement.