XAPP695 (v1.0) December 16, 2003

Gigabit Ethernet Aggregation to SPI-4.2 with Optional GFP-F Adaptation

Author: Hamish Fallside

Summary

The Gigabit Ethernet Aggregation reference design (EARD) as shown in Figure 1 demonstrates the aggregation of up to eight Gigabit Ethernet ports to SPI-4.2 with optional frame-mapped Generic Framing Procedure (GFP-F). The reference design is targeted for Virtex-II Pro™ FPGA family of devices, and includes Xilinx LogiCORE™ IP blocks for Gigabit Ethernet MAC 1000Base-X [See Ref. 2] and SP-I4.2 functions [See Ref. 3]. The Gigabit Ethernet IP provides connection to standard external SFP modules via the Virtex-II Pro RocketIO™, and the embedded PowerPC 405 processor implements management access to port statistics and data plane status and control. The SPI-4.2 source and sink interfaces provide a 10 Gbp/s link to an external framer or network processor, and utilizes the SelectIO™ technology in the Virtex-II Pro family.

The design is available as Verilog source code, with a testbench. All LogiCORE IP for use with the reference design must be purchased separately from the EARD, and can be obtained from the Xilinx website. Details on how to purchase and download the IP may be found at the end of this document in "Download and Purchase Information," page 27.

The EARD source code can be downloaded from http://www.xilinx.com/esp/networks_telecom/optical/xinx_net/eard_download.htm. Please note that registration is required to download the source code.

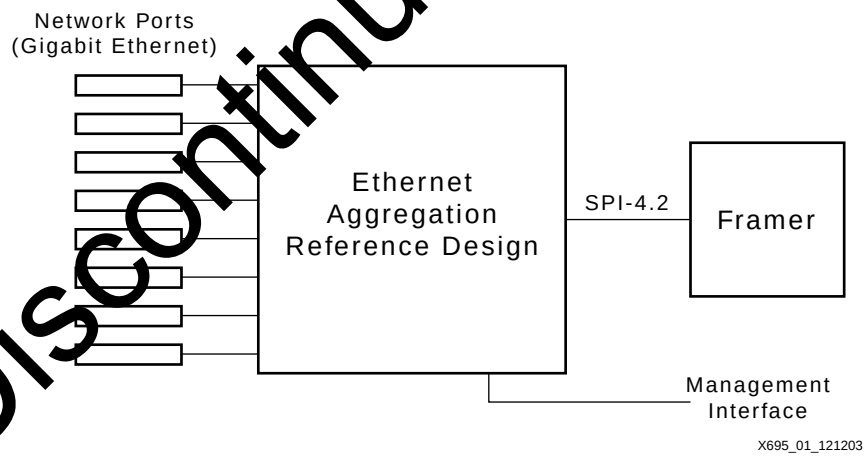

Figure 1: **Ethernet Aggregation Reference Design**

Features

The EARD features the following:

- Gigabit Ethernet 1000Base-X network ports
  - Uses Xilinx standard IP
  - Configurable as four or eight ports
  - Jumbo frame support
- SPI-4.2 framer port

  - Uses Xilinx standard IP
  - Dynamic phase alignment
- GFP-F client adaptation for Gigabit Ethernet
  - As detailed in ANSI Draft Specification T1X1.5/2000-024R3
  - The GFP-F adaptation function is optional, and independently selected for ingress and egress directions
- Internal block RAM FIFO on each port
  - 12 KB in framer-network direction (*ingress*)
  - 24 KB in network-framer direction (*egress*)
- Control plane software implemented on embedded Virtex-II Pro PPC405
  - Management interface accessible through UART command line
  - Performs initialization of the data plane
  - Provides data plane status and network ports statistics to management interface
- Compile-time loopback option for test
  - Network and framer ports can be set to loopback
- Status channel support in ingress path
- Statistics gathering per the Ethernet MAC document (Xilinx DS200)
- Performs multiplexing on egress
  - Each network port is logically mapped to a framer port via SPI-4.2
  - Performs simple round robin scheduling on SPI-4.2 segment basis
- Implements demultiplexing on ingress
  - Each framer port is logically mapped to the network ports
  - Uses channel-based mapping

Specific details of these features are described in "Architecture," page 4.

Introduction

The EARDis a Virtex-II Pro solution that aggregates multiple Gigabit Ethernet ports to a single SPI-4.2 interface with optional GFP-F encapsulation for Ethernet. The design includes a Virtex-II Pro PPC405-based control plane that provides various traffic management tasks. This functionality has been verified in hardware; a simulation testbench is provided with the design files. Actual applications, such as Ethernet over SONET or a front end to a network processor-based classification engine, may require additional customization.

Four or eight Gigabit Ethernet ports are selectable as a compile time option to the design. Each Ethernet port includes a Xilinx LogiCORE Gigabit Ethernet MAC with full statistics gathering capabilities, and 1000BaseX connection through the Virtex-II Pro embedded MGT. This allows the reference design to be used with standard SFP optical modules. Other interface options are available.

The designs implements a Xilinx LogiCORE SPI-4.2 block, comprising both source and sink. This provides an interface to an external framer or Network Processor. The network ports are mapped to virtual channels in the external device using the SPI-4.2 port numbers. The design provides optional GFP-F encapsulation/decapsulation on egress and ingress. FIFOs are provided within the MUX/DEMUX function and used for frame segmentation (on egress) and reassembly (on ingress).

The control plane is implemented on a Virtex-II Pro embedded PowerPC. A UART command line interface is provided with a set of simple commands that provide initialization, control, and status functions. In a full system these commands would form a simple API for an external

controller, or for an Operating System running on the PowerPC. Functions are included for initializing one or more network ports, initializing the SPI-4.2 interface, accessing the status registers for each of these blocks, obtaining statistics from the Ethernet MAC, monitoring system status, and trapping various events. The interface was invaluable during the hardware bring up of the system as a diagnosis and debugging tool.

Terminology

Certain terms are used with specific meaning within this document. These terms are defined in Table 1. Acronyms are spelled out in Table 2.

Note: In this document, the terms *ingress* and *egress* are used in the opposite sense than in the ANSI specification [See Ref. 1] They refer to the direction of travel as used by segments across the EARD SPI-4.2 interface.]

Table 1: **Definition of Terms**

Term	Definition
CAB	Client Adaptation Block.
Egress	Refers to network- framer direction. Direction of traffic that arrives at the network ports and is sent out the framer port.
Frame	Unit of transfer at the network ports. (Also applies after GFP encapsulation.)
Ingress	Refers to framer-network direction. Direction of traffic that arrives at the framer port and is sent out the network ports.
Packet	Synonymous with *frame* within the context of this document.
Segment	SPI-4.2 supports transmission of multiple virtual channels, using time division multiplexing. A segment is the amount of data allocated to a channel within each time slot.
Watermark	The term watermark is used to describe a threshold against which the state of a buffer is compared. A *watermark fill threshold* is compared against the amount of data in the buffer, while a *watermark empty threshold* is compared against the amount of space remaining.

Table 2: **Meaning of Acronyms**

Acronym	Meaning
CRC-16	Cyclic redundancy check (16-bit)
DCR	IBM CoreConnect Device Control Register interface to PowerPC
DEMUX	Demultiplexer
FCS	Frame check sequence
FIFO	First-in first-out (memory)
GFP	Generic framing procedure
GFP-F	Frame-mapped GFP
PLB	IBM CoreConnect Processor Local Bus interface to PowerPC
PHY	Physical device/layer (Ethernet)
MAC	Media access controller

Table 2: **Meaning of Acronyms *(Continued)***

Acronym	Meaning
MUX	Multiplexer
PCS/PMA	Physical coding sublayer/physical medium attachment
RTL	Register transfer level
SPI-4.2	System Packet Interface Level-4 Phase 2

Architecture

Figure 2 shows a block diagram of the basic operation of the EARD. For simplicity, only four ports are shown.

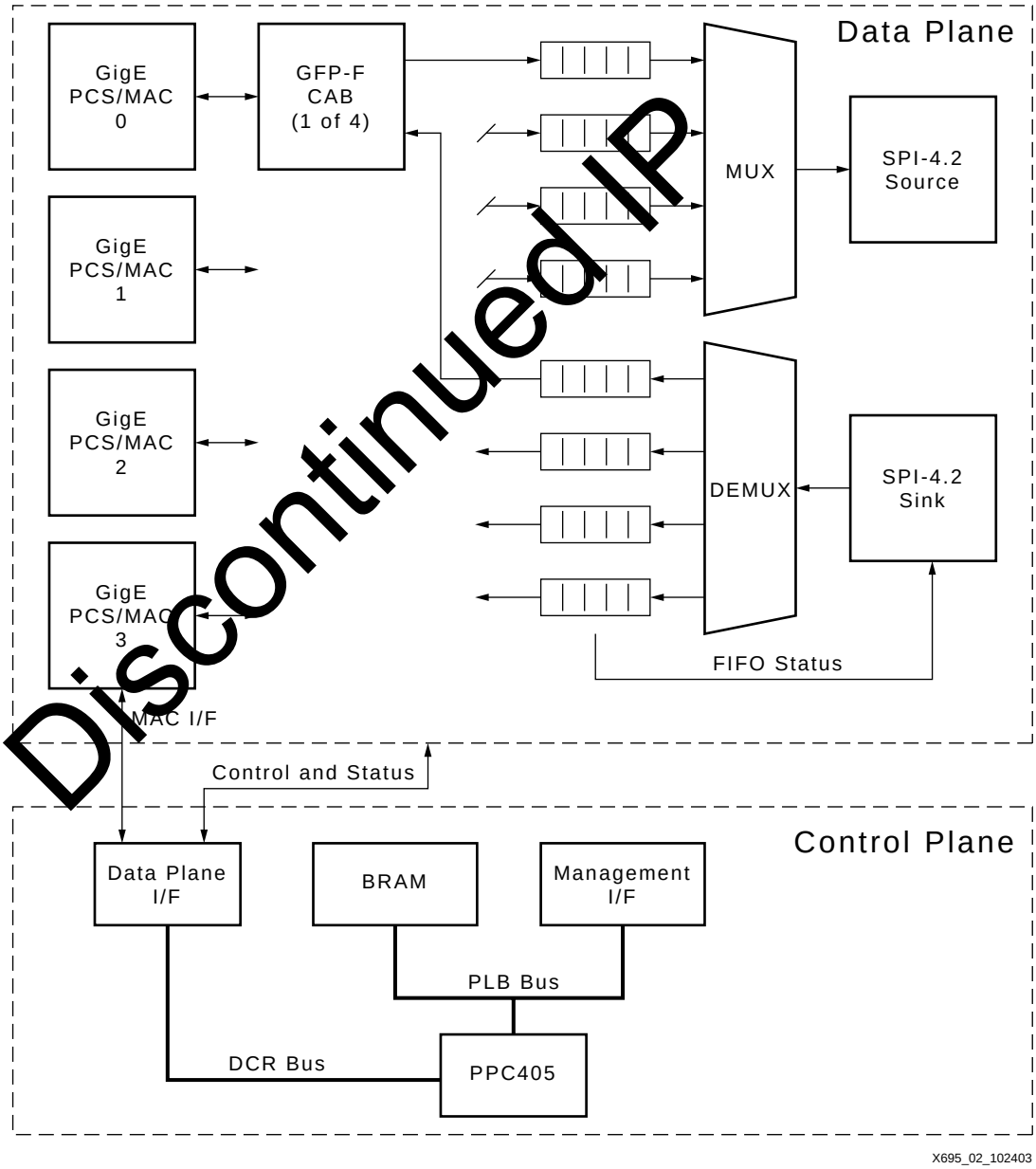

Figure 2: **High-Level Block Diagram**

The design comprises two top-level modules: a *data plane*, and a *control plane*. Verilog source code is provided for the parts of the design that connect the LogiCORE IP blocks together. All LogiCORE IP for use with the reference design must be purchased separately from the EARD, and can be obtained from the Xilinx website. Details on how to purchase and download the IP may be found at the end of this document in "Download and Purchase Information," page 27.

The data plane comprises either four or eight Gigabit Ethernet network ports. GFP-F client adaptation is an optional feature in the design, and is selected at compile time. In addition, the data plane can be configured for loopback on both the network and framer ports. MUX and DEMUX functions are provided for mapping frames and segments between the network and framer interfaces.

The control plane is provided as an EDK project, comprising a PowerPC sub-system. A UART interface is supplied for control of this subsystem via a simple command line. All software is stored and executed from on-chip block RAM, located on the PowerPC PLB bus. C source code is provided for this function as part of the EARD.

Framer Port

The *framer port* is implemented using the Xilinx SPI-4.2 v6.0 interface core. Table 3 gives an overview of the SPI-4.2 interface configuration. Dynamic phase alignment is used. Requests for phase realignment are handled automatically in hardware. Table 4, page 6 provides the complete list of the parameters for the interface.

The sink status channel is used by updating the sink status interface based on programmable FIFO thresholds. The source status channel is *not* used. Both calendars are initialized through configuration. There is no circuitry to allow for system calendar updates. Training is handled entirely by the SPI-4.2 core. There are *no* manual requests for idle control words. SPI-4.2 status signals are reflected in the Data Plane Status Register, as defined in "Embedded Programming Model," page 17.

Many of the SPI-4.2 core configuration parameters are dependent upon external factors, such as the target FPGA, PCB design, and the external SPI-4.2 device. The EARD imposes constraints on a few parameters, as described in Table 3.

Table 3: **Constrained SPI-4.2 Configuration Parameters**

Parameter	Usage
Number of Channels	This must equal the number of network ports.
Sink Load Init File Source Load Init File	Calendars must be initialized at compile-time using coefficient files.
Source Almost Full Assert Source Almost Full Negate	The minimum values allowed for the SPI-4.2 core are sufficiently large that latency in the client interface is not a problem. However, the *negate* threshold must be at least four more than the *assert* threshold, to avoid narrow pulses on the Source Almost Full signal.
Sink Clocking Implementation	Sink clocking must be configured for dynamic phase alignment.
Source Clocking Implementation	Source clocking must be configured for high-performance clocking.

Table 4: **SPI-4.2 Core Configuration Parameter**

Parameter	Setting	Parameter	Setting
BusFormat	BusFormatParen	core_type_option	Sink_and_Source
SimulationOutputProducts	Verilog	burst_size_in_credits	32
XilinxFamily	Virtex2P	sink_clk180	Use_inverted_CLK0_to_generate_CLK180
OutputOption	DesignFlow	sink_fifo_depth	512
DesignFlow	Verilog	Rsclkdiv	Status_Channel_Quarter_RDClk
FlowVendor	Synplicity	source_almost_full_assert	32
FormalVerification	None	sink_almost_full_negate	36
sink_almost_full_assert	32	source_number_of_dip2_matches	4
number_of_data_cycles_before_training	4096	sink_length_of_calendar_sequence	8
sink_io_placement	Bank7	source_fifo_depth	512
sink_load_init_file	True	source_length_of_calendar_sequence	8
source_io_placement	Bank6	source_load_init_file	True
number_of_training_patterns_during_training	2	sink_repeat_status_sequence_before_dip2	1
sink_status_io	LVTTL_Status_IO	source_repeat_status_sequence_before_dip2	1
source_almost_full_negate	36	source_status_io	LVTTL_Status_IO
enable_automatic_static_alignment	False	rsclk_phase	RStat_Changes_on_Rising_RSClk
package	FF1517	fifo_af_mode	Send_Satisfied_on_All_Channels
source_number_of_dip2_errors	4	Configuration	Dynamic_Alignment
tdclk_generation	HighPerformanceClocking	Device	2vp50
user_interface_data_width	64	sink_number_of_training_sequences	4
speed_grade	5	component_name	spi4_core
Performance	600_700 Mbp/s	clockmode	MasterClock
sink_init_filename	D:\IP04-017\coregen_ise\spi4_core.coe	sink_number_of_dip4_errors	
	4		
ldvs_type	2.5V	number_of_channels	8

Table 4: **SPI-4.2 Core Configuration Parameter**

Parameter	Setting	Parameter	Setting
		status_fifo_interface	Transparent
		source_init_filename	D:\IP04-017\coregen_ise\spi4_core.coe
		burst_mode	Complete_Burst

Network Ports

The *network ports* are implemented using the Xilinx Gigabit Ethernet MAC core, Version 3.0 [4]. Table 5, page 7 contains the core configuration parameter settings. Each MAC host interface is made available to the PowerPC control plane through the DCR bus, including access to PHY registers.

Transmit flow control in the MAC is enabled. Pause frames are requested when a watermark fill threshold in the egress FIFO is exceeded, triggered by the arrival of received frames. The watermark threshold is programmable from the control plane, and the pause value is a fixed configuration parameter. There is no back pressure link from the SPI-4.2 source interface status channel to this flow control mechanism, as the SPI-4.2 source port is under-subscribed with the maximum of eight network ports.

Receive flow control is enabled, and the *pause frame source* MAC address is programmable from the control plane. Back pressure from the ingress FIFO will be asserted to the SPI-4.2 status channel on the sink interface to indicate when framer traffic should be paused.

Table 5: **Gigabit Ethernet MAC Core Configuration**

Parameter	Setting
Compile-Time Configuration Parameters	
Physical Interface	PHY (PCS/PMA for 1000BASE-X)
Statistics Gathering	Enabled
Half-Duplex Capable	False
Memory Type	Block memory
Management Interface	True
MAC Register Configuration	
Pause Frame MAC Source Address	Software default as per "Software Configuration Parameters." Can be changed through the management interface.
Receiver Half Duplex	Disabled (Full-duplex only)
Receiver VLAN Enable	Enabled (Transparent to EARD)
Receiver In-Band FCS Enable	Enabled (Padding and FCS included in GFP payload)
Receiver Jumbo Frame Enable	Enabled
Transmitter Interframe Gap Adjust Enable	Disabled
Transmitter Half Duplex	Disabled (Full-duplex only)
Transmitter VLAN Enable	Enabled (Transparent to EARD)

Table 5: **Gigabit Ethernet MAC Core Configuration *(Continued)***

Parameter	Setting
Transmitter In-Band FCS Enable	Enabled (Padding and FCS included in GFP payload)
Transmitter Jumbo Frame Enable	Enabled
Transmit Flow Control Enable	Enabled
Receiver Flow Control Enable	Enabled
PHY Register Configuration All default values are used, with the following exceptions:	
Auto-Negotiation Enable	Disable auto-negotiation
Isolate	Normal operation

Data Flow

The following sections define certain characteristics of the data flow between the network ports and the framer port.

Buffer Management

Data buffering is provided by the EARD using simple FIFOs. Egress buffering is 24 KB per network port and ingress buffering is 12 KB per network port.

In the egress path, data is written to the FIFOs as frames. Data is read from the FIFOs as segments, which can cross multiple frame boundaries, and is transferred to the framer port a segment at a time. This is referred to as channel segmentation. The egress segment size is programmable from the control plane. In the ingress path, data is written to the FIFOs as segments.

The data is reassembled into Ethernet frames in the ingress FIFOs, and from subsequently transferred to the network ports. This process is referred to as channel reassembly. The ingress segment size is determined by the external SPI-4.2 device.

GFP Frame Encapsulation Enabled

When *GFP frame encapsulation* is enabled, each frame in the egress path has a GFP header prepended as described in "GFP Client Adaptation," page 10. Each frame must be buffered completely before transfer to the SPI-4.2 interface begins. In the ingress path, each frame has the GFP header stripped as described in "GFP Client Adaptation," page 10. Each frame is channel reassembled and buffered completely before transfer to the Gigabit Ethernet interface begins.

Note: GFP encapsulation can be selected independently for both egress and ingress.

Pass-Through Mode

When in pass-through mode, no GFP header is added to frames in the egress path. Transfer to the SPI-4.2 interface begins as soon as either one segment of data or an End-of-Packet is stored in the egress buffer. In the ingress path, there is no GFP header to strip. Transfer to the Gigabit Ethernet interface begins as soon as a watermark fill threshold is reached in the ingress buffer or an End-of-Packet is stored in the ingress buffer.

Note: Pass-through can be configured independently for both egress and ingress. MUX/DEMUX Function

When in *MUX/DEMUX mode*, data stored in the egress buffers is channel-segmented and multiplexed into the SPI-4.2 source interface on a per-segment basis. Scheduling is based on simple round-robin priority.

Data arriving at the SPI-4.2 sink interface is channel-reassembled in the ingress buffers. Mapping is based directly on channel numbers.

Loopback Function

Two loopback functions are provided as a compile-time option. This can be useful for verification purposes during interoperability testing or when porting the design to a new board. Figure 3 is a simplified illustration of loopback showing only four ports. When configured for loopback, the design handles data as follows:

- One loopback function allows Ethernet frames in the egress buffers to be looped back directly to the corresponding ingress buffers for the same port.
- The second loopback function provides SPI-4.2 loopback whereby data arriving at the SPI-4.2 sink interface is written directly to the SPI-4.2 source.

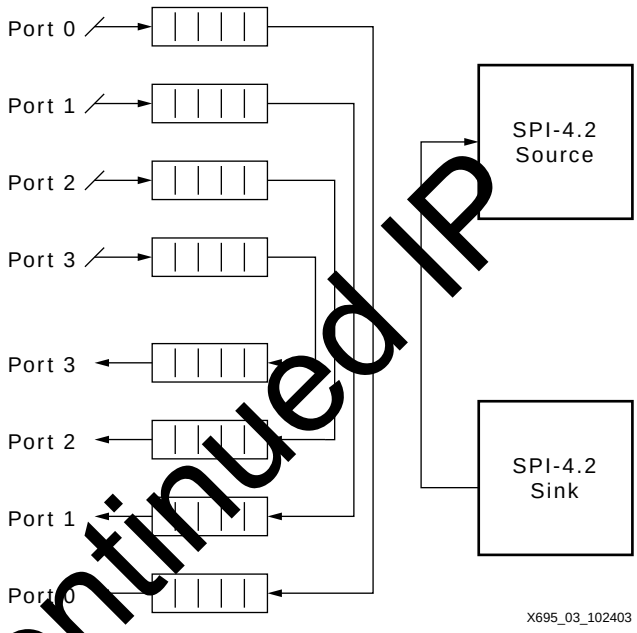

Figure 3: **Loopback Block Diagram**

GFP Client Adaptation

When GFP frame encapsulation is enabled, the EARD performs Ethernet MAC payload-specific *client adaptation*, as described in Section 7.1 of the *Generic Framing Procedure* document [See Ref. 1]. GFP headers are prepended to frames in the egress path and stripped from frames in the ingress path. Table 6 shows derivations of GFP header fields.

Table 6: **GFP Header Fields**

Header Field	Value
PLI	Contains the length of the GFP Payload Area, as defined in Section 6.1.1.1 of the *Generic Framing Procedure* document [See Ref. 1].
cHEC	Contains CRC-16 sequence computed over PLI field, as defined in Section 6.1.1.2 of the *Generic Framing Procedure* document [See Ref. 1].
Type	Contains a constant value defined as a configuration parameter. The default is 0001h, corresponding to *Ethernet with Null Extension Header & no Payload FCS* in Table 3 of the *Generic Framing Procedure* document [See Ref. 1].

Table 6: **GFP Header Fields *(Continued)***

Header Field	Value
tHEC	Contains a constant value defined as a configuration parameter, which must equal the CRC-16 sequence computed over the Type field.
Extension	The EARD assumes that no extension header is present.

Management Interface

The EARD employs a UART/serial port as the *management interface*. It is expected that different interfaces will be required for actual applications.

Control Plane Software

The *control plane software* performs data path management to manage the data plane. It also provides management interface access to network port statistics and data plane status and control.

Data Path Management

The data path is initialized, according to the following sequence, through access to the Data Plane control and status registers.

Initially, all **Reset** signals are asserted and all **Enable** signals are negated.

1. Release the Gigabit Ethernet DCM Reset signal.
2. Wait until the Gigabit Ethernet DCM Locked signal is asserted.
3. Release the TDClk DCM Reset signal.
4. Wait until the TDClk DCM Locked signal is asserted.
5. Release the RDClk DCM Reset signal.
6. Wait until the RDClk DCM Locked signal is asserted.
7. Release the TSClk DCM Reset signal.
8. Wait until the TSClk DCM Locked signal is asserted.
9. Release the Data Plane Core Reset signals.
10. Release the SPI-4.2 Sink and Source Reset signals.
11. Assert the SPI-4.2 Sink and Source Enable signals.
12. If needed, request phase alignment until the SPI-4.2 Sink Out of Frame signal is negated.
13. Release the Gigabit Ethernet Reset signal.
14. Configure all Gigabit Ethernet MAC and PHY registers as defined in "Network Ports," page 7.
15. Enable Gigabit Ethernet ports.

Management Interface Access

Control and monitoring is via the *management interface*, which typically varies across different applications. As a starting point, a simple message-passing protocol has been developed as part of the EARD, and accessed via simple commands, as summarized in Table 7, page 12 (this table spans multiple pages). The implementation should port easily to most processor interfaces. This is a layer of abstraction to hide the details of the interface from application-level software. The commands that are invoked at the UART interface are summarized in the table.

Traps are implemented, and the nature of the trap depends on the specific implementation of the management interface. It is expected to be an interrupt signal. Descriptions of specific register settings changed by the commands can be found in "Embedded Programming Model," page 17.

When using the command line interface to the EARD, the following notes apply:

- Command names are case sensitive.
- Command names & parameters should be space-separated.
- Return values, and trap occurrences, are simply printed to the screen.
- Parameters are all numeric, and can be encoded as:
  - Decimal (prefix with “0d”)
  - Binary (prefix with “0b”)
  - Hexadecimal (prefix with “0x”, or no prefix)

Table 7: **Message Passing**

Message	Command Line Format	Notes
Initialize System	initSystem [p0] [p1] [p2] [p3] [p4] [p5] [p6] [p7]	Initializes the data plane, as described in “Data Path Management,” page 11. pN = 0|1, where 0 disables and 1 enables the network port
Get Revision	getRevision	Returns contents of Revision Register, plus software revision ID.
Get Pause Threshold	getPauseThreshold	Returns the value of the egress FIFO watermark threshold used for pause frame generation. This is obtained from the Data Plane Control Register.
Set Pause Threshold	setPauseThreshold [8-bit threshold]	Set the value of the Egress FIFO watermark threshold used for pause frame generation. This is set in the Data Plane Control Register.
Get Error Disposition	getErrorDisposition	Returns the error frame disposition bit from the Data Plane Control Register.
Set Error Disposition	setErrorDisposition [1/0]	Sets the error frame disposition bit from the Data Plane Control Register.
Get Pause Frame SA	GetPauseFrameSa [port]	Returns the pause frame source address for the corresponding network port.
Set Pause Frame SA	setPauseFrameSa [port] AA BB CC DD EE FF	Sets the pause frame source address for the corresponding network port.
Get Configuration	getConfig	Returns the contents of the Data Plane Configuration Register.
Set Configuration	setConfig SS HT ST TX	Sets contents of Data Plane Configuration Register. • SS = Egress Segment Size • HT = Ingress Hungry Threshold • ST = Ingress Starving Threshold • TX = Ingress TX Threshold
Get Status	getStatus	Returns contents of Data Plane Status Register.
Clear Status	clearStatus [bit mask]	Clears the bits in the Data Plane Status Register that are indicated by the bit mask, and returns the resulting register contents.
Get Trap	getTrap	Returns Trap Mask, along with a masked version of the Data Plane Status Register.
Set Trap Mask	setTrapMask [32-bit mask]	Sets the Trap Mask to allow specific conditions to be monitored.
Enable Traps	EnaTrap	Enables trap on Data Plane Status Register contents.

Table 7: **Message Passing *(Continued)***

Message	Command Line Format	Notes
Disable Traps	DisTrap	Disables trap feature.
Get Statistic	getStatistic [port] [index]	Returns statistics value of given network port and statistics index. ***Note:*** Refer to the Ethernet MAC data sheet [See Ref. 4].
Get MAC Register	readPort [port] [offset]	Returns the contents of the GE MAC Control Register specified by port and management interface offset value. ***Note:*** Refer to the Ethernet MAC data sheet [See Ref. 4].
Enable Monitor	EnaMon	Enables monitor feature.
Disable Monitor	disMon	Disables monitor feature.
Help	Help	Prints to the screen the available commands and their syntax.

Error Handling

Table 8, page 13 summarizes *error handling* in the EARL. Frames with errors are treated according to the state of the error frame disposition register bit described in "Data Plane Control Register," page 19. The two options are:

Mark: Erroneous egress frames are marked by assertion of the **SrcFFErr** signal. Erroneous ingress frames are written unmodified to the Gigabit Ethernet transmit port.

Discard: Erroneous egress frames are deleted if possible, but if writing to the SPI-4.2 interface has already started at the time of detection they are marked as above. (This prevents egress discard during pass-through mode.) Erroneous ingress frames are discarded by the Gigabit Ethernet MAC in response to assertion of the **TX_UNDERRUN** signal.

In order to minimize complexity, some cases are handled minimally or not at all. FIFO parity checking is not performed. Checking for Ethernet *runt* frames is not performed. Frames are passed as-is regardless of size, with one exception; ingress frames of less than four bytes are padded to four bytes to simplify implementation.

Table 8: **Error Handling**

Error	Handling Procedure
Detection of erroneous frame in Gigabit Ethernet MAC (egress) (Assertion of RX_BAD_FRAME)	Handled according to error frame disposition.
Mid-frame egress FIFO overflow (FIFO overflows part way through frame)	Frame is truncated and handled according to error frame disposition. Minimum truncated frame consists of a 16-byte Ethernet packet (plus GFP header if in GFP mode). Remainder of frame is received from the Gigabit Ethernet MAC and discarded. Occurrence of this error is trapped in the Data Plane Status Register.
Full-frame egress FIFO overflow (FIFO is full at the start of a frame)	Entire frame is received from the Gigabit Ethernet MAC and discarded. Occurrence of this error is trapped in the Data Plane Status Register. (Mid-frame and full-frame overflow share the same register bit.)
Ingress FIFO underflow (Pass-through mode only)	Handled according to error frame disposition. Occurrence of this error is trapped in the Data Plane Status Register.

Table 8: **Error Handling *(Continued)***

Error	Handling Procedure
Detection of erroneous frame by SPI-4.2 Interface (ingress) (Assertion of SnkFFErr)	Handled according to error frame disposition. Occurrence of this error is trapped in the Data Plane Status Register.
SPI-4.2 sink out of frame (Assertion of SnkOof)	This error cannot be reliably associated with specific frames, so it does not cause frames to flagged as erroneous. Occurrence of this error is trapped in the Data Plane Status Register.
SPI-4.2 sink bus error (Assertion of SnkBusErr)	This error cannot be reliably associated with specific frames, so it does not cause frames to be flagged as erroneous. Occurrence of this error is trapped in the Data Plane Status Register.
SPI-4.2 sink FIFO DIP-4 error (Assertion of SnkFFDIP4Err)	The corresponding frame is handled according to error frame disposition. Occurrence of this error is trapped in the Data Plane Status Register.
SPI-4.2 sink FIFO payload DIP-4 error (Assertion of SnkFFPayloadDIP4)	The corresponding frame is handled according to error frame disposition. Occurrence of this error is trapped in the Data Plane Status Register.
SPI-4.2 sink FIFO burst error (Assertion of SnkFFBurstErr)	The corresponding frame is handled according to error frame disposition. Occurrence of this error is trapped in the Data Plane Status Register.
SPI-4.2 sink FIFO payload error (Assertion of SnkFFPayloadErr)	Each corresponding data word is discarded. Occurrence of this error is trapped in the Data Plane Status Register.
SPI-4.2 source out of frame error (Assertion of SrcOof)	Occurrence of this error is trapped in the Data Plane Status Register.
SPI-4.2 source DIP-2 error (Assertion of SrcDIP2Err)	This error condition is not handled or recorded, since the egress status path is not used.
SPI-4.2 source data pattern error (Assertion of SrcPatternErr)	Occurrence of this error is trapped in the Data Plane Status Register. (This error should be prevented by design.)

Hardware Configuration Parameters

The *hardware configuration parameters* supported by the EARD are described in Table 9. Configuration for the SPI-4.2 and Gigabit Ethernet MAC cores are defined in "Framer Port," page 5 and "Network Ports," page 7, respectively.

Table 9: **Hardware Configuration Parameters**

Parameter	Usage
MUX_8_TO_1	Eight network ports if defined, otherwise four network ports. Default: undefined (four ports)
LOOPBACK_MODE	Operating in loopback mode if parameter is defined, else in MUX/DEMUX Mode. Default: undefined (MUX/DEMUX mode)
INGRESS_PASSTHROUGH_MODE EGRESS_PASSTHROUGH_MODE	Operating in pass-through mode if parameter is defined, else in GFP mode. Applied independently to ingress and egress. Default: both defined (pass-through mode)

Table 9: **Hardware Configuration Parameters *(Continued)***

Parameter	Usage
PAUSE_FRAME_DURATION	The value passed to the Gigabit Ethernet MAC to be used in pause frames. 6-bit value Default: 32 (0020h)
GFP_TYPE GFP_THEC	The type value to be used in the GFP header as defined in the *Generic Framing Procedure* document, with the exception that header extension is not supported. The HEC field must be set to the correct CRC-16 for the type. 6-bit values Default GFP_TYPE: 0001h Default GFP_THEC: 1021h

Software Configuration Parameters

Table 10 lists the compile-time parameters supported by the control plane software.

Table 10: **Software Configuration Parameters**

Parameter	Usage
DEFAULT_PAUSE_MAC_SA0 to DEFAULT_PAUSE_MAC_SA7	Default expected source address for pause frames for each Gigabit Ethernet port.

Implementation

Clocking

Clock generation and the corresponding clock domains are shown in Figure 4 and Figure 5, page 17, respectively. **BUS_CLK** and **PPC_CLK** can be derived from a third external source if necessary.

Note: The initial EARD includes a clock doubler to generate a 312.5 MHz differential clock from a 156.25 MHz reference. The clock doubler is implemented using an MGT, and the resulting clock must be connected externally to the SPI-4.2 system clock input.

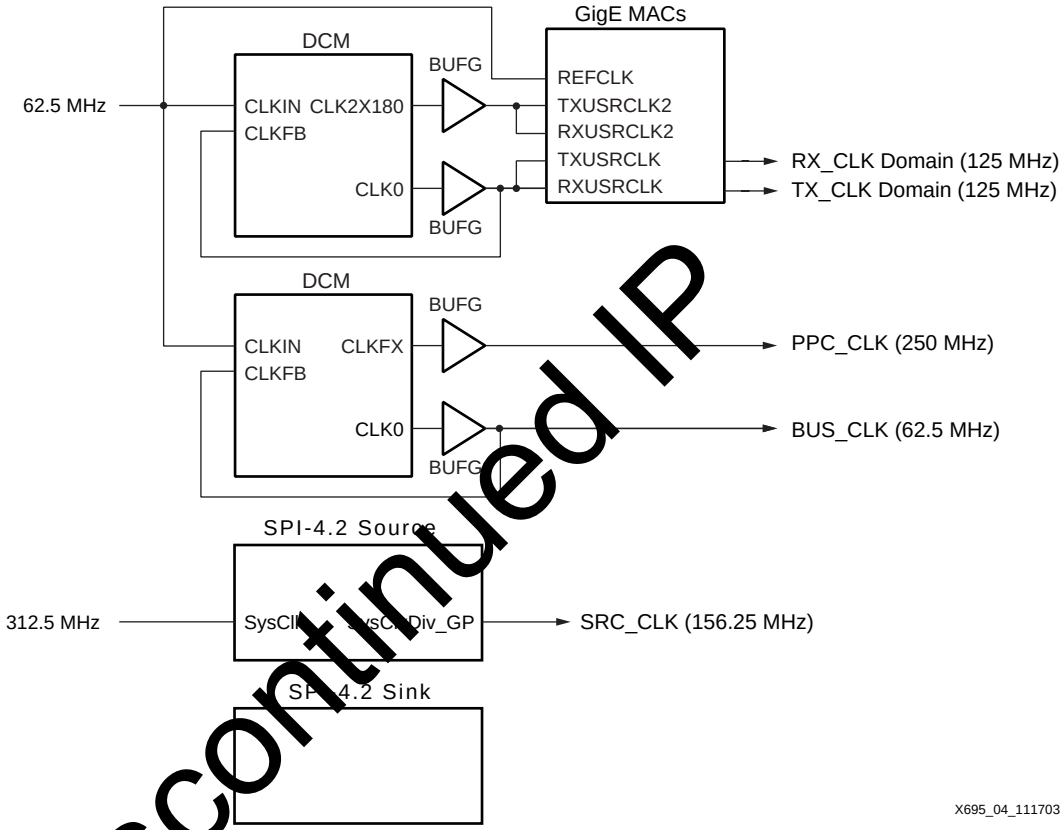

Figure 4: **Clock Generation**

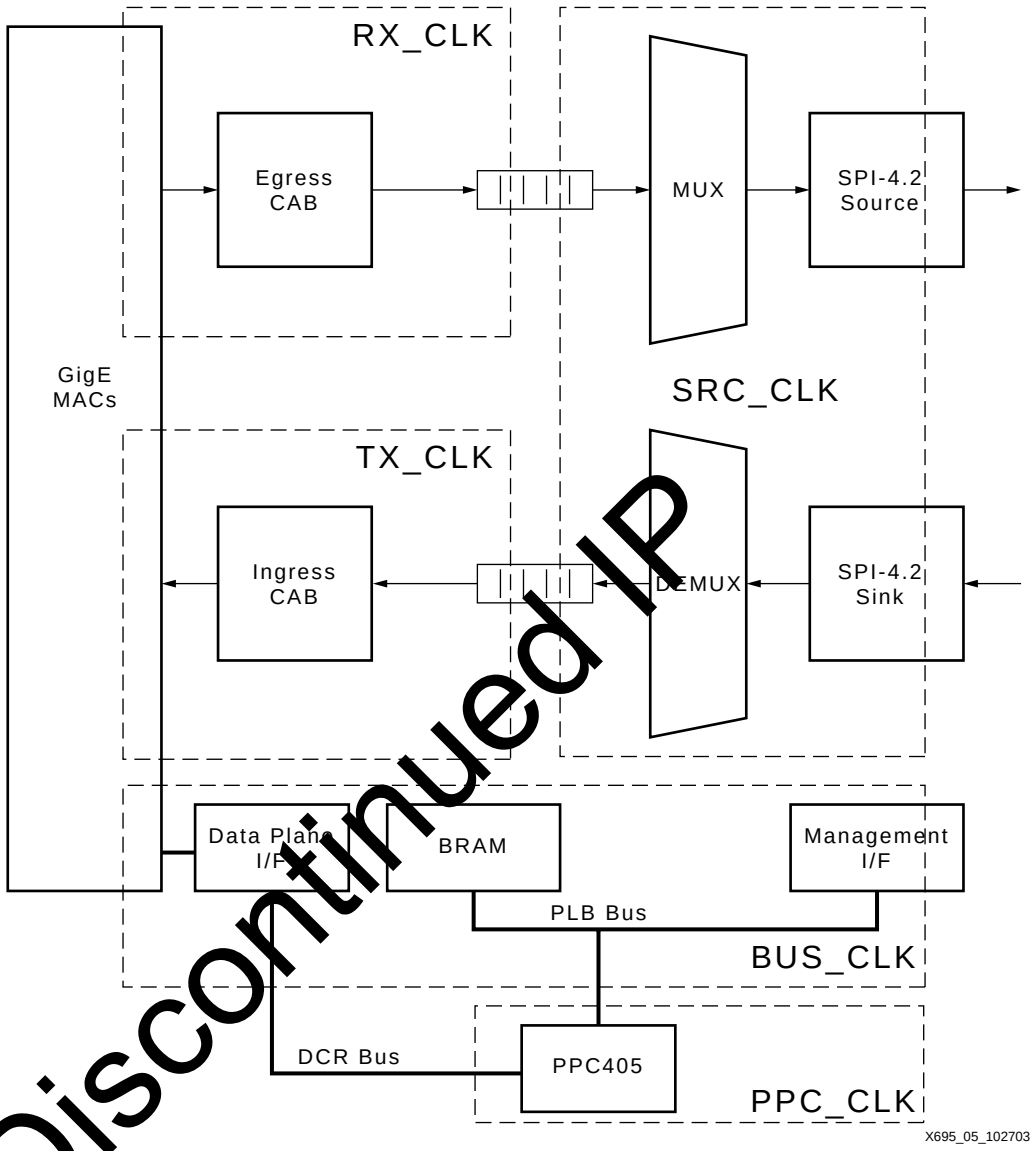

Figure 5: **Clock Domains**

Embedded Programming Model

The PLB memory map is shown in Table 11. The memory map of the UART can be found in the *PLB 16550 UART LogiCORE Product Specification* data sheet [See Ref. 5].

Table 11: **PLB Memory Map**

Address Range	Usage
FFFF8000h to FFFFFFFFh	BRAM
00000000h to 00001FFFh	UART 16550

Table 12 shows the memory map for the data plane interface, which is the sole DCR peripheral. Within the DCR bus, the data plane interface is located at Offset = 4.

Table 12: **Data Plane Interface Memory Map**

Address Offset	Usage
007h	Reserved
006h	Port Data Upper Register
005h	Port Data Lower Register
004h	Port Access Register
003h	Data Plane Configuration Register
002h	Data Plane Status Register
001h	Data Plane Control Register
000h	Revision Register

Data Plane Interface Registers

The *Data Plane interface registers* are defined in Table 13 through Table 16.

Revision Register

The *Revision Register*, shown in Table 13, is a read-only register that provides a means to access the key configuration status of the EARD.

Table 13: **Revision Register**

Bit Field	Reset	Access	Definition
31:16	-	RO	Revision ID This field can be used by system designers to designate different revisions and configurations of the EARD. The initial delivered system uses 0000h.
15:4	0	RO	Reserved
3	-	RO	EGRESS_PASSTHROUGH_MODE setting 0: GFP mode egress 1: Pass-through mode egress
2	-	RO	INGRESS_PASSTHROUGH_MODE setting 0: GFP mode ingress 1: Pass-through mode ingress
1	-	RO	LOOPBACK_MODE setting 0: MUX/DEMUX mode 1: Loopback mode
0	-	RO	MUX_8_TO_1 setting 0: Four network ports 1: Eight network ports

Data Plane Control Register

The *Data Plane Control Register* is shown in Table 14. All of the reset bits [7:0] are active low, i.e. the corresponding module is held in reset when the bit is low.

Table 14: **Data Plane Control Register**

Bit Field	Reset	Access	Definition
31:24	0	RO	Reserved
23:16	128	RW	Egress FIFO flow control threshold Allowable values **0..191** (Limited in hardware), representing number of 128 byte blocks. When the amount of data stored in an egress FIFO equals or exceeds this threshold, any received frame on the same port triggers a pause frame request.
15:12	0	RO	Reserved
11	0	RW	SPI-4.2 sink enable (SnkEn signal) 1: Enable SPI-4.2 sink core 0: Disable SPI-42. sink core
10	0	RW	SPI-4.2 source enable (SrcEn signal) 1: Enable SPI-4.2 source core 0: Disable SPI-42. source core
9	0	RW	Phase alignment request trigger Positive edge triggers a phase alignment request in the SPI-4.2 sink core (PhaseAlignRequest signal).
8	0	RW	Error frame disposition (discard on Error) 1: Erroneous frames are discarded on error, if possible 0: Erroneous frames are not discarded, unless necessary due to buffer overflow
7	0	RW	SPI-4.2 TSCLK DCM reset
6	0	RW	SPI-4.2 TDCLK DCM reset
5	0	RW	SPI-4.2 source reset
4	0	RW	SPI-4.2 RDCLK DCM reset
3	0	RW	SPI-4.2 sink reset
2	0	RW	Data plane core reset
1	0	RW	Gigabit Ethernet DCM reset
0	0	RW	Gigabit Ethernet reset

Notes: Access Abbreviations

1. **RW:** Read/write
2. **RO:** Read only

Data Plane Status Register

The *Data Plane Status Register* is shown in Table 15. The sense of these bits indicates normal operation with all zeros, as follows:

Clock status signals (bits [20:17], 0):

- 1: DCM is not locked (Bit 20: Clock is not active)
- 0: DCM is locked (Bit 20: Clock is active)

Error signals:

- Read 1: This error has occurred since the last clear
- Read 0: No error occurrences since the last clear
- Bitwise clear by writing1

Table 15: **Data Plane Status Register**

Bit Field	Reset	Access	Definition
31	0	RCW	SPI-4.2 sink FIFO burst error (SnkFFBurstErr assertion)
30	0	RCW	SPI-4.2 sink FIFO payload DIP4 error (SnkFFPayloadDIP4 assertion)
29	0	RCW	SPI-4.2 sink FIFO overflow error (SnkOverflow assertion)
28	0	RCW	SPI-4.2 sink FIFO error (SnkFFErr assertion)
27	0	RCW	SPI-4.2 sink FIFO payload error (SnkFFPayloadErr assertion)
26	0	RCW	SPI-4.2 sink FIFO DIP-4 parity error (SnkDIP4Err assertion)
25	0	RCW	SPI-4.2 sink Bus error (SnkBusErr assertion)
24	0	RCW	SPI-4.2 sink out of frame error (SnkOof assertion)
23	0	RCW	SPI-4.2 source FIFO overflow error (SrcFFOverflow assertion)
22	0	RCW	SPI-4.2 source data pattern error (SrcPatternErr assertion)
21	0	RCW	SPI-4.2 source out of frame error (SrcOof assertion)
20	N/A	RO	SPI-4.2 RDCLK inactive
19	N/A	RO	SPI-4.2 TSCLK DCM locked
18	N/A	RO	SPI-4.2 TDCLK DCM locked
17	N/A	RO	SPI-4.2 RDCLK DCM locked
16	0	RCW	Ingress FIFO Port 7 underflow
15	0	RCW	Ingress FIFO Port 6 underflow
14	0	RCW	Ingress FIFO Port 5 underflow

Table 15: **Data Plane Status Register *(Continued)***

Bit Field	Reset	Access	Definition
13	0	RCW	Ingress FIFO Port 4 underflow
12	0	RCW	Ingress FIFO Port 3 underflow
11	0	RCW	Ingress FIFO Port 2 underflow
10	0	RCW	Ingress FIFO Port 1 underflow
9	0	RCW	Ingress FIFO Port 0 underflow
8	0	RCW	Egress FIFO Port 7 overflow
7	0	RCW	Egress FIFO Port 6 overflow
6	0	RCW	Egress FIFO Port 5 overflow
5	0	RCW	Egress FIFO Port 4 overflow
4	0	RCW	Egress FIFO Port 3 overflow
3	0	RCW	Egress FIFO Port 2 overflow
2	0	RCW	Egress FIFO Port 1 overflow
1	0	RCW	Egress FIFO Port 0 overflow
0	N/A	RO	Gigabit Ethernet DCM locked

Notes: Access Abbreviations

1. **RO:** Read only
2. **RCW:** Read, bitwise clear by writing 1

Data Plane Configuration Register

The *Data Plane Configuration Register*, shown in Table 16, is used to control settings that influence data plane operation. These fields can only be modified when the data plane is reset.

Table 16: **Data Plane Configuration Register**

Bit Field	Reset	Access	Definition
31:26	0	RO	Reserved
25:24	00	RW	Egress segment size 00: 64B 01: 128B 10: 256B 11: 512B
23:16	64 (80h)	RW	Ingress hungry threshold If the available space in an ingress FIFO (64-bit words) is greater than or equal to this threshold, SPI-4.2 status for the corresponding port will be set to *hungry*.
15:8	128 (F0h)	RW	Ingress starving threshold If the available space in an ingress FIFO (64-bit words) is greater than or equal to this threshold, SPI-4.2 status for the corresponding port will be set to *starving*. **Note:** Starving takes precedence over hungry. If the hungry threshold exceeds the starving threshold, the SPI-4.2 status will never be set to hungry.

Table 16: **Data Plane Configuration Register *(Continued)***

Bit Field	Reset	Access	Definition
7:0	10 (0Ah)	RW	Ingress TX threshold Watermark fill threshold which triggers transmission in pass-through mode.

Notes: Access Abbreviations

1. **RW:** Read-write
2. **RO:** Read only

Port Access Register

The *Port Access Register*, shown in Table 17, is used to initiate accesses to the Gigabit Ethernet MAC registers.

Table 17: **Port Access Register**

Bit Field	Reset	Access	Definition
31	0	RW	Access Write 1: Triggers a write of the Port Data Lower Register to the location indicated by Port Select and Offset. Write 0: Triggers a read of the location indicated by Port Select and MAC Offset, with results stored in the port data register(s). Read: Returns the last value written.
30	0	RW	PHY select 0: MAC Access 1: PHY Access
29	0	RO	Port access busy Set upon initiation of any access Cleared upon completion
28:13	0	RO	Reserved
12:3	0	RW	Offset MAC Accesses: Offset[9:0] indicates the offset within the Gigabit Ethernet MAC to which this access should be directed. PHY Accesses: Offset[9:5] indicates which PHY should be accessed (always 00h). Offset[4:0] indicates the offset within the PHY to which this access should be directed.
2:0	0	RW	Port select Indicates the network port to which this access should be directed.

Notes: Access Abbreviations

1. **RW:** Read-write
2. **RO:** Read only

Port Data Registers

Reads and writes of Gigabit Ethernet MAC registers are channeled through the locations shown in Table 18 and Table 19.

Table 18: **Port Data Upper Register**

Bit Field	Reset	Access	Definition
31:0	0	RO	Contains the upper 32 bits of the most recent read from a Gigabit Ethernet statistics register.

Notes: Access Abbreviations:

1. **RO:** Read only

Table 19: **Port Data Lower Register**

Bit Field	Reset	Access	Definition
31:0	0	RW	Contains the data resulting from a read of a Gigabit Ethernet control register. Contains the lower 32 bits of data resulting from a read of a Gigabit Ethernet statistics register. Contains the data to be written to a Gigabit Ethernet control register.

Notes: Access Abbreviations:

1. **RW:** Read-write

Design Compilation and Simulation

Table 20 details the tools (including their version number) that were used in the implementation of the EARD.

Table 20: **Tools and Versions**

Tool	Version
EDK	3.2 SP2
ISE	Foundation 5.2.03i
CoreGen	Foundation 5.2.03i Cores Update #2
Synplify	7.30
Modelsim	PE 5.7d

Design Resources

Table 21 summarizes the resource requirements of the EARD. The 4-port design can be implemented in an XC2VP30, while the 8-port version fits in an XC2VP50. The design is currently limited by block RAM than by logic or I/O. Reducing the amount of block RAM in the design, either for the control plane software, and/or for the data plane FIFOs, will result in the EARD fitting into a smaller Virtex-II Pro device.

Note: The logic usage figures (LUT and Flip-Flops) are approximate. The I/O figures are based on an actual application using the existing management interface and differential SPI-4.2 status. The actual EARD in its current form has some differences to account for the target validation hardware.

Table 21: **Resource Usage**

Design	BRAM	LUT	Flip-Flop	DCM	GCLK	MGT	GPIO
4-port	123	18500	18100	5	11	4	96
8-port	215	29800	27300	5	11	8	96

Validation

The EARD was validated in hardware by AMIRIX Systems using the ML324 demonstration board with SPI-4.2 loopback [See Ref. 2]. The ML324 contains a 2VP50 FF1517 device, with RocketIO interfaces brought out to SMA connectors. These are connected to SMA2SFP adapter cards to provide the network port connectors. The board has standard 0.1-inch header pins that are used to bring out the SPI-4.2 source and sink pins. The traces to the pins are matched pairs, and connect to LVDS paired IOBs on the FPGA. A ribbon cable was used to provide a loopback connection between the SPI-4.2 source and sink interfaces.

The tests included all configurations for both 4-port and 8-port designs. This section includes a summary of the tests that were performed.

Sanity Test

The sanity test verified the basic functionality of the test harness. The features that were tested included reset conditions and data flows through the data plane using fixed-length packets of sequential data.

Single Packet Data Flow Test

The test for single packet data flow validated basic data path operations and the test environment using one packet at a time. This testing ensured that the data plane did not require a continuous stream of packets to operate correctly. Tests included packet flow on each port and handling of runt packets using both random-length and fixed-length packets of sequential data.

Back-to-Back Packet Data Flow Test

This test validated normal data path operations using all ports and full-rate traffic flow. Tests included packet flow on each port and handling of runt packets using both random-length and fixed-length packets of sequential data.

Congestion Test

The congestion test validated the behavior of the data plane under heavy congestion. The features that were tested included FIFO congestion and fill levels and the SPI-4.2 sink status interface using random-length packets of sequential data.

Flow Control and Wait States Test

The flow control and wait states test validated flow control handshaking along the data path. The features that were tested included flow control when all FIFOs were in full or almost-full condition using random-length packets of sequential data.

Pass-Through Mode Test

This test validated the reduced latency of the data plane when running in pass-through mode using fixed-length packets of sequential data. Pass-through mode allows packets to be sent without being fully buffered.

Segmentation and Channel Reassembly Test

This test validated the segmentation of egress traffic and the channel reassembly of ingress traffic using random-length packets of sequential data.

Egress MUX Scheduler Test

This test validated the round robin scheduler employed by the egress MUX. This included the segmentation of egress traffic and the channel reassembly of ingress traffic using fixed-length packets of sequential data.

Gigabit Ethernet Error Conditions Test

This test validated the correct handling of Gigabit Ethernet error conditions along the egress data path using random-length packets of sequential data.

SPI-4.2 Error Conditions Test

This test validated the correct error handling of SPI-4.2 error conditions along the ingress data path. This included random-length packets of sequential data and random error offsets.

Design Hierarchy

The structure of the release is shown below, with the various directories for the EDK control plane project and the ISE design project. Bit files are provided for the evaluation board.

Note: The Ethernet and SPI-4.2 IP must be purchased separately and the generated netlists inserted into the *source/cores* directory.

EDK Files

```
edk                     EDK Project (standard EDK file structures with
|                       the following notes)
|___xps                 option values (platgen/ simgen/ libgen/ xpsxflow/
|                       ppc405_i_compiler)
|___code                contains embedded software C files
|___data                <standard EDK directory>
|___etc                 <standard EDK directory>
|___myip                customized IP (VHDL files)
|   |___bram_block_v1_00_a
|   |___dcr_slave_V1_00_a  NOTE: contains the DCR interface files
|   |                      consisting of a VHDL wrapper bringing DCR bus
|   |                     signals out of the EDK system sub-module and the
|   |                      DCR implementation files
|   |___inverter_ip
|___pcores              standard versions of cores (Verilog/ VHDL files)
|___ppc405_i            bsp zip file
    |___bsp_ppc405_i    C files for bsp/ cpu drivers/ etc.
    |___include         H header files
    |___libsrc          C source files for bsp/ common_v1_00_a/
    |                   cpu_ppc405_v1_00_a/ uartns550_v1_00_b
    |___ppc_wrapper     NOTE: contains the custom PPC wrapper
                        [ppc405_top.v] that is used to interface the PPC
                        with the DCR bus. Ppc405_top.v needs to be copied
                        into the EDK install directory under the
                        $(EDK)\hw\iplib\pcores\ppc405_v1_00_a\hdl\Verilog
                        subdirectory.
```

Source Files

```
source                  Verilog source, related test and implementation files
|___{root}                global definitions file containing compile-time
|                         parameters
|___coregen               files generated by Coregen and EDK, used for
|                         implementation
|___implementation        user constraints file, bitfile configuration, and
|   |                     the ISE compile scripts
|   |___4p_gfp_lm         4 port GFP Framer Loopback implementation results
|   |___4p_pt_lm          4 port Passthrough Framer Loopback implementation
|   |                     results
|   |___8p_gfp_nlm        8 port GFP MUX/DEMUX implementation results
|   |___8p_pt_nlm       8 port Passthrough MUX/DEMUX implementation results
|___modelsim              simulation scripts and results (after simulation)
|___primitives            simulation primitives
|___regression            Verilog regression toolkit
|___rtl                   Verilog source tree
|___synplify              synthesis compile script and results (after
|                         synthesis)
|___test_bench            Verilog test benches for simulation
|___test_harness          Verilog test harness for simulation
```

Bit Files

```
bitfiles                downloadable bitfiles including blockRAM
|                       initialization for software
|___4p_gfp_lm           4 port GFP Framer Loopback
|___4p_pt_lm            4 port Passthrough Framer Loopback
|___8p_gfp_nlm          8 port GFP MUX/DEMUX
|___8p_pt_nlm           8 port Passthrough MUX/DEMUX
```

For a complete description of the various design processes (EDK integration, configuration, simulation, implementation) please refer to Version Description Document (`VDD_150ct03.doc`) in the provided zip file.

Download and Purchase Information

Downloading and Purchasing

The EARD has three parts (see Figure 6) that can be downloaded. Details about these parts are listed below.

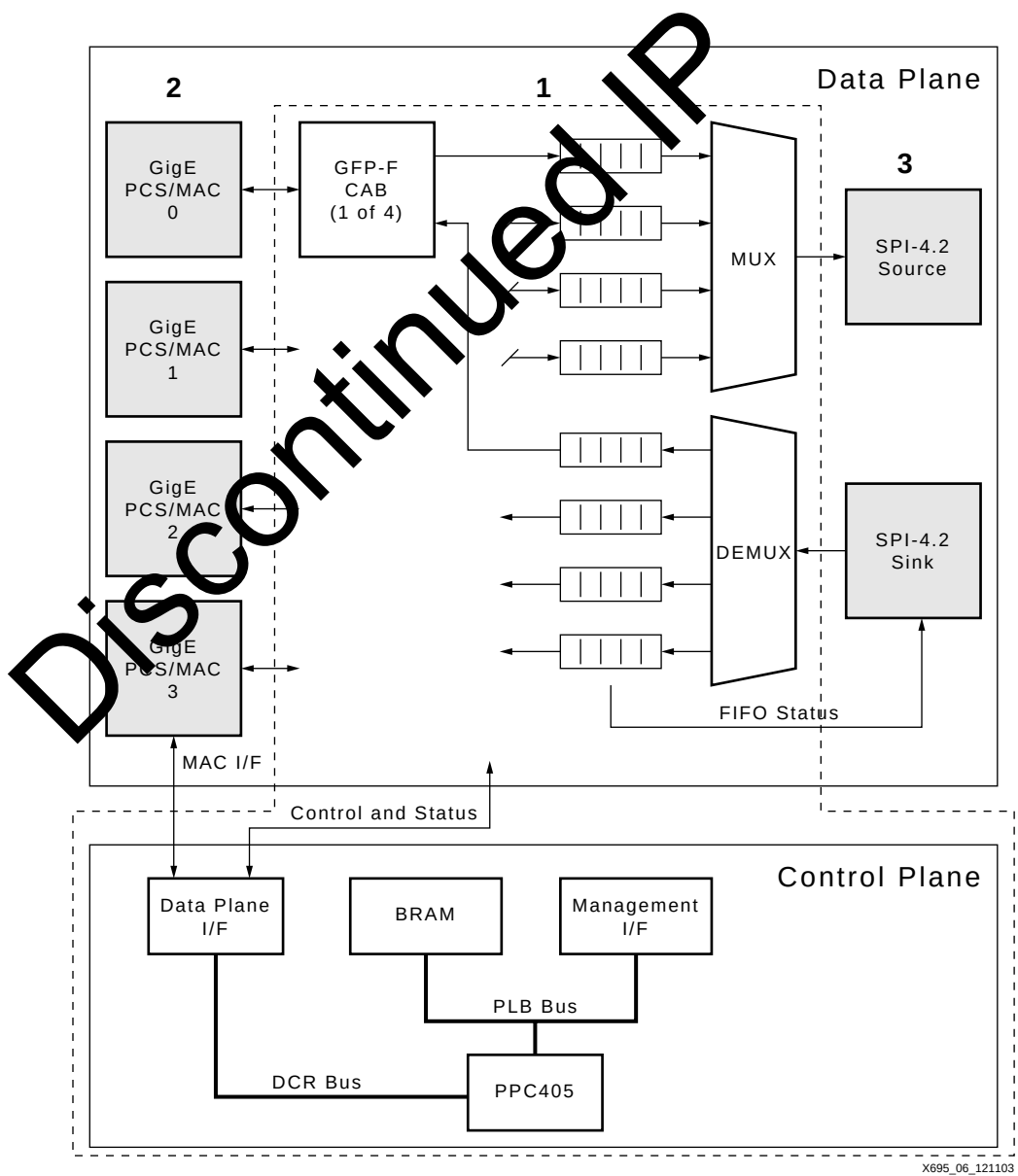

Figure 6: **Ehternet Reference Design Download Diagram**

1. Aggregation Logic Free Downloadable Reference Design includes:
   - MUX/DEMUX
   - Traffic management
   - Control plane
   - Optional frame-mapped Generic Framing Procedure, or GFP-F functions
   - Registration required to download reference design

   The Reference Design source code can be downloaded from http://www.xilinx.com/esp/networks_telecom/optical/xlnx_net/eard_download.htm.
2. 1-Gigabit Ethernet MAC (GMAC) LogiCORE™ Solutions

   Further information about the core and and purchasing information is available at http://www.xilinx.com/ipcenter/1ge/1ge_registration.htm#order

   Xilinx provides two ways to evaluate the GMAC core:

   - Simulation Only
   - Full System Hardware Evaluation

   Further information about these evaluations is available at http://www.xilinx.com/ipcenter/ipevaluation/1ge_mac_evaluation.htm

   Note: The reference design contains a limited behavioral model of the Ethernet MAC that can be used for the purposes of simulation.
3. SPI-4.2 (POS-PHY Level 4) LogiCORE™ Solutions

   Further information about the SPI-4.2 core and purchasing information is available at http://www.xilinx.com/ipcenter/posphy/spi42_registration.htm.

   You may purchase the SPI-4.2 Multi-Channel LogiCORE product from your local Xilinx sales representative (http://www.xilinx.com/company/contact.htm).

   Xilinx provides Verilog and VHDL Functional Simulation models for the evaluation of its SPI-4.2 core. Using the Xilinx SPI-4.2 solution, designers can significantly reduce implementation costs and accelerate their time-to-market over fixed function Application Specific Standard Product (ASSP) alternatives. For more information, go to http://www.xilinx.com/ipcenter/ipevaluation/pl4_evaluation.htm.

Contact us for more details at eard@xilinx.com.

Merging Cores with EARD

The EARD release contains all the files required to build the complete reference design, with the exception of the following files that are expected to be in the *source/cores* directory of the build:

- `ge_mac.edn`
- `ge_mac_gmac_gen_1.ngc`
- `pl4_snk_top.edf`
- `pl4_src_top.edf`

UCF and NCF files for the originally generated cores remain in the directory for reference.

Once the cores have been licensed, follow the steps in "GbE MAC Core Integration" and "SPI-4 Core Integration" to recreate these files.

GbE MAC Core Integration

1. Obtain the Gigabit Ethernet MAC core.
2. Create new Gigabit Ethernet MAC core project named `gmac_gen_1`.
3. Configure core per Table 5, page 7 (Gigabit Ethernet MAC Core Configuration).
4. Generate core output files:

   `ge_mac.edn`

   `ge_mac_gmac_gen_1.ngc`

5. Copy output files into source/cores directory of project.

SPI-4 Core Integration

1. Obtain the SPI-4 core.
2. Create new SPI-4 core project named `spi4_core`.
3. Configure core per Table 3, page 5 (Constrained SPI-4.2 Configuration Parameters).
4. Generate core output files:

   `pl4_src_top.edf`

   `pl4_src_top.ncf`

   `pl4_snk_top.edf`

   `pl4_snk_top.ncf`

   `spi4_core_pl4_wrapper.ucf`

5. Copy output files into source/cores directory of project.

References

1. T1X1.5/2000-024R3 (Draft ANSI T1.xxx.yy-200x), Generic Framing Procedure
2. AMIRIX Systems Inc., DOC-003329, GFP Grooming Demonstration FPGA Test Report
3. Xilinx, Inc., DS209, SPI-4.2 Core V6.0 LogiCORE Product Specification, July 31, 2003
4. Xilinx, Inc., DS200, 1-Gigabit Ethernet MAC Core with PCS/PMA Sublayers (1000BASE-X) or SMII V3.0 LogiCORE Product Specification, April 30, 2003, v1.1
5. Xilinx, Inc., DS481, PLB 16550 UART LogiCORE Product Specification, v2.2, May 15, 2003

See www.xilinx.com/ipcenter for Xilinx LogiCORE documentation.

Customization Services

AMIRIX Systems is a development partner with Xilinx and offers customization services on the Ethernet Aggregation reference design (EARD).

Contact:

AMIRIX Systems Inc.
77 Chain Lake Drive
Halifax, NS, Canada B3S 1E1

www.amirix.com

Tel: (902) 450-1700

Toll Free: 1-866-426-4749

Fax: (902) 450-1704

E-mail: info@amirix.com

Revision History

The following table shows the revision history for this document.

Date	Version	Revision
12/16/03	1.0	Initial Xilinx release.