

## Overview

Resilient Packet Ring (RPR) is a ring-oriented Media Access Control (MAC) protocol standardized per IEEE 802.17 specifications.

The RPR network is a dual-ring-based architecture offering the best of Ethernet and SONET/SDH worlds, combining Ethernet's data transport efficiency, simplicity, familiarity, and cost advantage with SONET/SDH's efficient support for ring topology and fast recovery from failures.

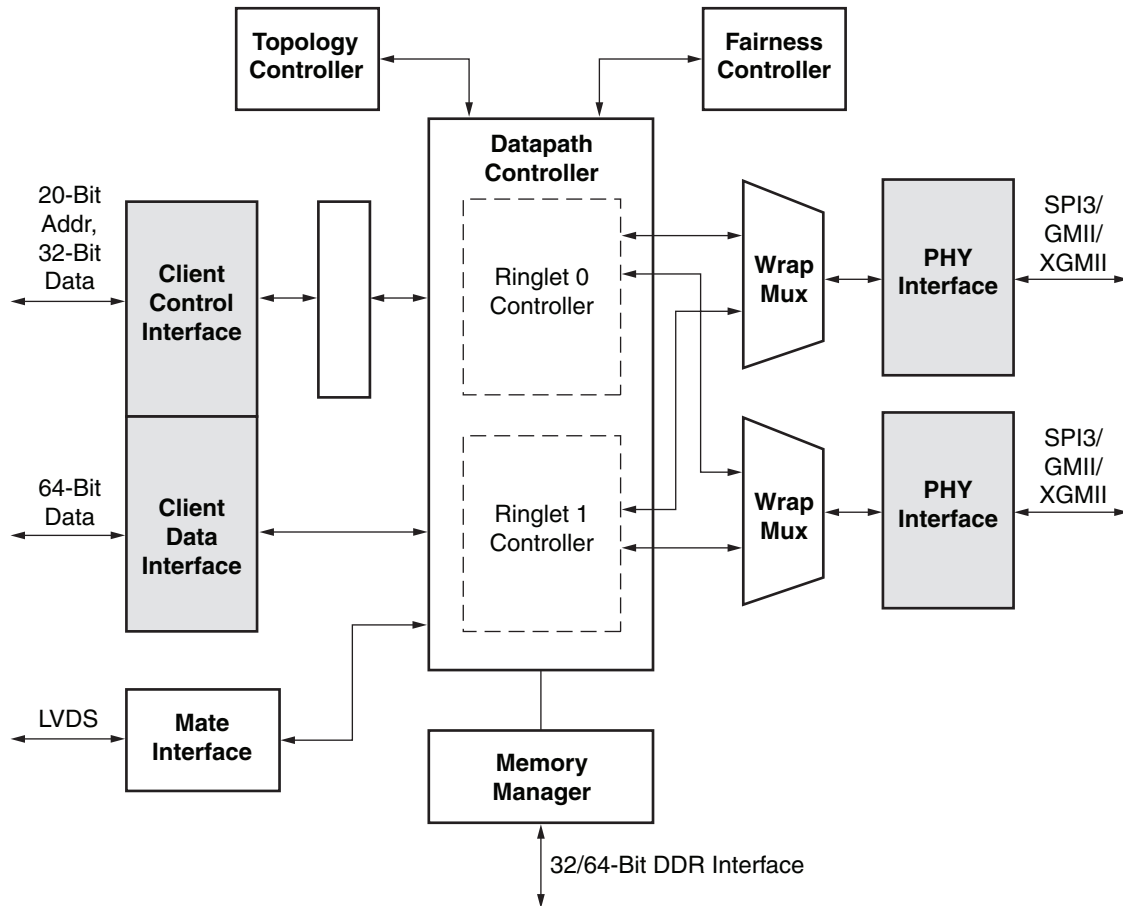
By effectively utilizing bandwidth on both rings, the RPR protocol significantly enhances the bandwidth efficiency of the service provider's networks, enabling optimal and fair use of bandwidth for bursty traffic through highly efficient statistical multiplexing, overbooking, and spatial reuse transport mechanisms.

## General Description

The RPR MAC is fully compliant to IEEE 802.17, September 2004 standards. The RPR MAC supports 1G, 2.5G, and 10G speeds and has been designed for flexibility to suit various implementation and system architectures.

## Features

- **Dual Ring, 1G, and 2.5G Speed Support**
  - GMII and SPI-3 PHY interface (32-bit-wide generic bus)
  - Client interface: 64-bit data, 1 clock, and control signals
  - Control interface port: 32-bit data, 20-bit address, and controls
- **Buffering**
  - Options for dual and single transit queues
  - Jumbo frame size support
  - DDR2 SDRAM based buffering support
  - The DDR memory can be configured to support client side data queue
  - Frame-aging checks
- **Fairness and Shaping**
  - Class A0 /A1 /B (CIR, EIR) /C
  - Options for conservative and aggressive fairness rate adjust mechanism
  - Active weights calculation (counting at 64-bytes interval)
  - Pre/post congestion shaping
  - Automatic fairness frame generation /processing
  - Frame-aging check
- **Dual Ring with Protection**
  - Under 50-ms ring reconfiguration
  - Automated topology and protection (TP) and topology checksum (TC) frame generation /processing
  - Automated topology checksum calculation and processing
  - Automated Loop Round Trip Time (LRTT) generation/processing
  - Steer, wrap (center and edge), and pass-through mode support for traffic protection
  - PHY LINK status-based alarm handling
  - Full support for IEEE 802.17 context containment
- **OAM Processing**
  - Partially managed by CPU
- **Other Features**
  - CRC16 and byte-enabled CRC32 computation and verification
  - All statistical counters per IEEE 802.17 standard. Additional counters can be added.
- **Self Test Features**
  - Loopback at PHY interface, PHY, and external
  - OAM organization frame
  - Self-verifying type memory interface - descriptor (256-byte size) has verification on queue number and parity (in both DDR lanes)
  - Optional ECC support for memory
  - FIFO underflow/overflow checks at all important data flow paths
  - Extensive statistics collection. 32-bit counters in hardware.
- **For 10G Applications**
  - XGMII, XAUI, and SPI-3 PHY interface
  - Dual 64-bit-wide DDR interface for transit data storage
  - One 64-bit DDR for client datapath
- **Linux-Based Device Driver and Application for the RPR MAC**
  - Interfacing the core to upper level software - initialization, power-on self tests, run-time tests, and interrupt handling
  - OAM and performance monitoring
  - Optional additional database based on Attribute Discovery (ATD) frames
  - RPR 802.17 MIB implementation (support for SNMP agent)



PB010\_01\_100305

Figure 1: RPR MAC Hardware Functional Block Diagram

## Functional Description

The RPR MAC is fully compliant to IEEE 802.17 specifications and supports dual queue operation with both wrap and steer protection functions.

The RPR MAC supports GMII for 1G, SPI-3 for 2.5G, and SPI-4.2/XGMII/XAUI/XSBI for 10G PHY interfaces. It also supports client-side interfaces with 32-bit data and 20-bit address. These interfaces can be configured during FPGA compilation. The RPR MAC also provides an optional mate interface that can be used for communication with a mate card in case of redundant-line card ring terminations. A Linux-based device driver for the RPR MAC is also provided. Following is a brief description of various blocks as shown in Figure 1.

### PHY Interface

The RPR MAC supports compile-configurable PHY interfaces for both Ethernet (PRS-1 and PRS-10) and SONET/SDH applications as listed in Table 1.

Table 1: RPR MAC Interfaces

Interface	1G	2.5G	10G
SONET/SDH PHY	NA	SPI-3	SPI-4.2
Ethernet PHY	GMII, TBI / RTBII	n/a	SPI-4.2 or XGMII or XAUI or XSBI
MATE	LVDS or RocketIO	LVDS or RocketIO	RocketIO - 3.125x4 or 10G
Client Interface	One or two links of 64-bit interface at 66 MHz	One or two links of 64-bit interface at 133 MHz	One or two links of 64-bit interface at 200 MHz
Control Path	32-bits for data, 20 bits for address and controls	32-bits for data, 20 bits for address and controls	32-bits for data, 20 bits for address and controls

A generic interface, provided for the PHY interface, includes a dual clock FIFO and clock generation logic. If the requirement is XGMII, SPI-3, or GMII, the PHY option is bundled together with the RPR MAC. XAUI, XSBI, and SPI-4.2 are

not part of the RPR MAC, but they are available as Xilinx cores.

## Datapath Controller

The datapath controller manages the flow of frames to and from the RPR MAC for both Ringlet 0 and Ringlet 1. The controller is comprised of receive datapath controller and transmit datapath controller blocks.

### Receive Datapath Controller

The receive datapath controller manages the data flow between the PHY layer and either the Control frame FIFO, Local Data FIFO, PTQ or the STQ, wherever the data needs to be stored. The receive datapath controller is comprised of frame classification, integrity verification logic, and the counter updates (as explained in chapter 7 of IEEE 802.17 document). Based on the classification results, a tag is appended to indicate drop/copy for the decision block. A 4K-deep FIFO is used for intermediate storage while the frame classification and header validation are in progress. The FIFO also provides intermediate storage for buffering requirements.

The receive datapath controller performs these functions:

- Header FCS check
- Frame FCS check
- Size check
- Address check
- Format check
- Hop consistency check
- Drop decisions based on Topology, SA, DA, and Errors
- Frame routing: Routing of frames to local, transit, or both local and transit paths based on address, flood, and frame type
- Statistics update

### Transmit Datapath Controller

The transmit datapath controller manages the datapath between the client interface, Primary Transit Queue (PTQ), Secondary Transit Queue (STQ), and the transmit PHY interface. It manages the add traffic to the ring and the addition of the data, control, fairness, and idle frames. Data frames based on the client QoS are written into the queue. Based on the inputs from the fairness controller and the local traffic shaper, the transmit datapath controller adds transmit frames from the PTQ, STQ, control frames, or the idle frames to the transmit FIFO. The transmit FIFO is a dual clock FIFO with the input side operating off the system clock and the output side operating off the PHY clock. The output is aligned as per the PHY bus width and tagged with an EOF delimiter.

The transmit datapath controller manages the wrap and steer functions based on the control bit setting. In addition, the transmit datapath controller manages these functions:

- Ring selection based on the control bit settings
- RPR framing for client frames
- Frame CRC calculation and insertion
- Header CRC calculation and insertion
- Time stamp verification for jitter computation
- Rate monitoring and rate computation. The rate computation is determined every 8 bytes and the rate monitor feeds the transmit rate input to the fairness controller.
- Buffer fullness and frame aging. The buffer fullness and the frame aging function provides inputs to the fairness controller on the buffer level in the PTQ/ STQ and aging of the frames to be used for idle shaping and congestion decision.

## Fairness Controller

The fairness controller implements the fairness algorithm as per section 10 of the IEEE 802.17 standards. The fairness algorithm distributes unallocated and unused reclaimable bandwidth fairly among the contending stations and allocates this bandwidth to fairness-eligible traffic, e.g., class B-EIR and class C traffic.

The RPR MAC maintains two instances of the fairness controller blocks, one block per ring. The blocks are independently identified by the station address and the ringlet ID. Following are the features and services of the fairness controller:

- Manages unallocated bandwidth and bandwidth reclamation
- Regulates only fairness-eligible (class C and class B-EIR) traffic. Separately regulates the add traffic and the transit traffic.
- Supports both single transit queue and dual transit queue deployment.
- Supports both aggressive and conservative rate adjustment methods.
- Computes fair rates in proportion to an administrative weight assigned to each fairness instance.
- Fair rate advertisement to the contributing upstream stations using the ringlet that opposes the data flow.
- Provides fair rate adjustment in less than 64 msec.

The fairness controller draws inputs from the topology controller (active weights and hop count), the transmit datapath controller, and system administration to calculate the fair rates. The fairness controller provides 512x68 frame buffers at the input and the output of the block.

## Topology Controller

The topology controller determines connectivity and the ordering of the stations around the ring. This is accomplished by collecting information about the stations and interconnecting links via the topology discovery

protocol. The collected information is stored in the topology databases of each station.

Following are the features and services of the topology controller:

- Protection switching: Wrapping and steering of traffic, switching within 50 msec without reordering/duplication of strict frames
- Supports up to a maximum of 255 stations
- Supports one primary and two secondary MAC addresses
- Supports strict order and relaxed order frames
- ATD, TC, TP, and LRTT frame processing and generation

## Client Interface

The RPR MAC provides a generic client-side interface that can be customized to specific requirements. The client interface provides independent interfaces for control and datapaths.

### Client Data Interface

The client data interface is generic and can be used for adapting to a vast variety of possible client interfaces. Custom logic can be added to connect to existing systems and backplanes.

This interface is a 64-bit interface that includes an embedded path for transferring the MA\_data\_request and MA\_data\_indication parameters. It resembles SPI-3 in signaling and flow control. The clock for this interface is supplied by the client. Following are the features and services of the client data interface:

- Data transfer: Data is transferred by invocation of MA\_DATA.request and MA\_DATA.indication.
- Burst FIFO: Provides for a fixed burst size of 256 bytes.
- Frame delineation: Frames are delineated by the SOF, EOF, and Byte\_Valid signals.
- Fault detection: FIFO overflow fault is indicated.

### Client Control Interface

The client control interface provides access for system management. The 32-bit interface can operate up to a maximum frequency of 133 MHz. This is the interface through which the device driver operates. Following are the features supported by the interface:

- Bus signals: 32-bit data bus, 20-bit address bus, ChipSelect, Read, Write, and Ready.
- Interrupt mask: One 32-bit-wide maskable interrupt port.
- Flexible clocking: Clock can be sourced to or from the interface.
- Data latching at I/O pads for better timing.
- Operations, Administration, and Maintenance (OAM):

Mandatory and organization specific frame reception and transfer FIFO interface. All OAM counters are implemented within the RPR MAC hardware. The counters can be accessed by the client via this interface. The CPU is expected to read the counters before they overflow once every 400 msec.

- For frames generated by the control path, HEC and FCS update are handled by hardware.

## Memory Manager

The memory manager manages the PTQ and STQ. In the case of single transit queue implementation, external memory is not needed. The PTQ is implemented using block RAMs and has provisions to store one jumbo frame of 9K or four frames of 1.6K each.

For a dual transit queue implementation, external DDR2 SDRAM storage is required to store the STQ. These are the memories required, depending on the rates:

- Single 32-bit-wide memory for up to 2.5 Gb/s with 200-MHz clock (for transit traffic)
- Dual 64-bit-wide memory for 10G systems with 275-MHz clock (one per ring)

Considering 256 Mb chips, two chips are needed for 1G and 2.5G implementations. Eight chips are required for 10G implementations. This provides for a total of 64 MB for 1G/2.5G and 256 MB for 10G.

The memory management block supports a self-verifying memory interface; packet descriptors (256-byte size) have verification on the queue number and parity in both DDR lanes. Optional ECC per buffer is 256 bytes in size and supports multi-buffer frame handling. Memory size per queue is configurable from the control path.

The memory manager can be extended to support a Ternary CAM interface (TCAM) for VLAN tag look-up for VLAN based QoS. The TCAM interface can be added as a customization to the RPR MAC.

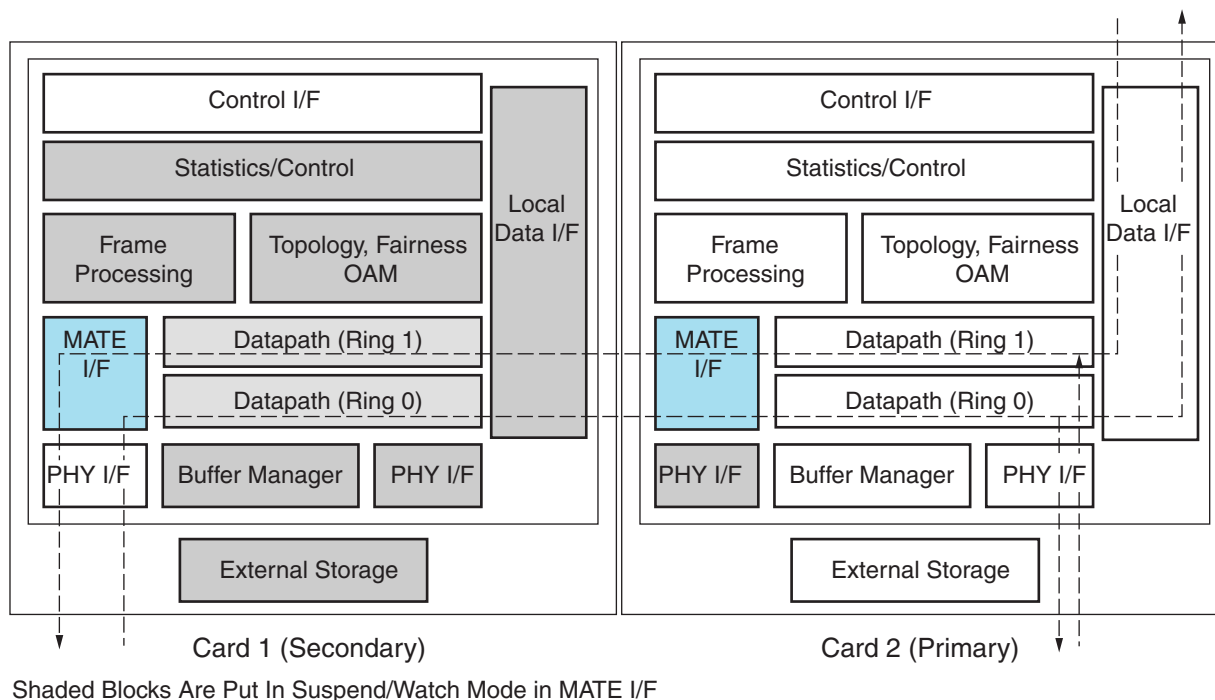
## Equipment Redundancy

The RPR MAC provides for two system configurations to implement equipment redundancy. One configuration uses a mate interface, internal to the equipment, with the cards interconnected for a master/slave mode of operation. Another configuration uses two separate RPR MACs with no interconnection.

### Equipment Redundancy via the Mate Interface

The mate interface is an implementation method especially popular with existing SONET interfaces to enable equipment redundancy. Two RPR rings are terminated on two uplink interface cards with SONET protection switching on the PHY layer. Two cards have a data interface through the backplane. Between the two cards, one is configured as the

primary card and the other is the secondary card. One of the RPR system implementations is shown in Figure 2.



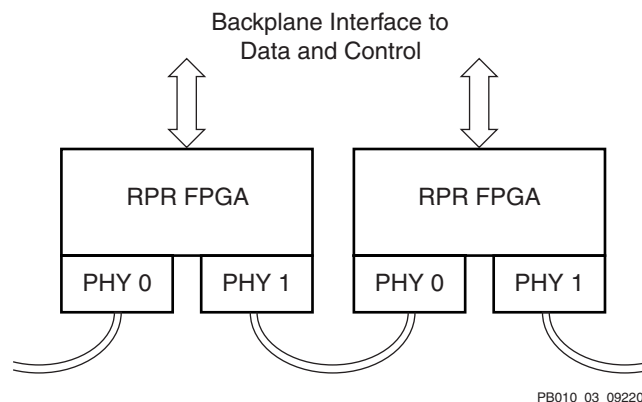
PB010\_02\_092205

Figure 2: Equipment Redundancy Using the Mate Interface

The mate interface uses 6 LVDS pairs connected to the backplane that links the two cards. Five of the pairs are for the data and one pair is for the clock. These pairs operate at approximately 320-350 MHz (-10 Speed grade), which is sufficient for 2.5G applications. The implementation of the mate interface depends on the system-level architecture. This interface can be customized for specific applications.

**Equipment Redundancy via Separate RPR MACs**

Another system configuration for equipment protection with dual-card/dual-node implementation has two cards, each with a separate RPR MAC. An external interface connects the two cards as shown in Figure 3. The primary card is operational and the secondary card is kept under pass-through.



PB010\_03\_092205

Figure 3: Equipment Redundancy with the Dual-Card, Dual-Node Option

**OAM and Management Interface**

The OAM block is for RPR ring integrity verification and checks. It provides for loopback verification between 2 RPR nodes. Organization frame support helps in sending and receiving special purpose test/information frames across the nodes. The OAM block initiates a flush of previously sent information in accordance with the standard, IEEE 802.17. The OAM block also provides all timeout functionality for operations.

The management block collects statistics from the 32-bit memory-based counters and passes it to the MIB interface. The management block also handles interrupts regarding topology and errors on the PHY interface to reconfigure the topology status. Most of the OAM functions are carried-out by the client with the use of the supplied device driver.

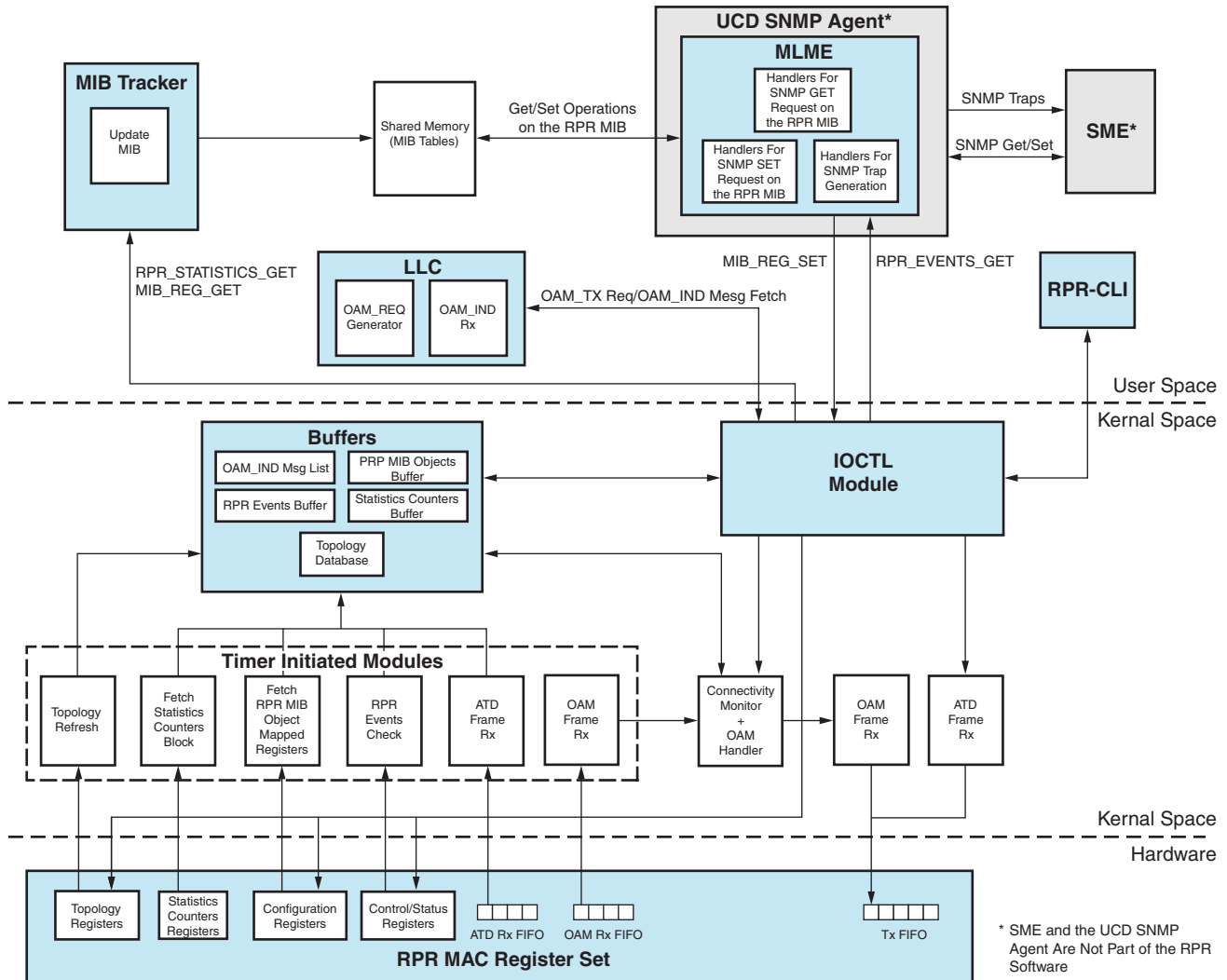
## FPGA Device Driver and Application Software

Most of the RPR sections are implemented in the FPGA logic, except for those functions that would consume substantial logic if implemented in hardware and have no real-time requirement. Therefore, these functions are implemented in software.

The RPR MAC is provided with a Linux device driver to implement these functions in software. Along with the device driver, application layer binaries are also provided that can be used by a Simple Network Management Protocol (SNMP) agent to configure/monitor the RPR MAC parameters. The Station Management Entity (SME) can be used to initiate the SNMP calls. Following is a list of device driver features and services:

- System initialization
  - Memory setup
  - Power-on self tests
- Station setup with myTopoInfo table at startup
  - Local address
  - Default ringlet
- Initial system weights
- MAX stations
- Periodic keep-alive checks (section 11.6 of IEEE 802.17 standards)
- OAM frame handling
  - Flush frame transmit when the conditions outlined in IEEE 802.17 section 5.17.2 have occurred.
  - ECHO frame handling (OAM) - transmits and responds to ECHO frames.
- ATD frame handling (generates and terminates as changes occur) and topology database maintenance and update
- Interrupt handling (events polling)
  - Interrupts are generated when MIB variables are altered. The software modifies MAC-layer control registers in response to the interrupt. The SNMP agent uses this functionality to enable network management.
  - Timers create interrupts that trigger polling events. Statistics must be read at regular polling intervals and various MIB variables are updated according to the statistics.
  - Interrupts are generated upon the occurrence of RPR events. The SME is notified about certain RPR events.

Figure 4 shows the software architecture.



\* SME and the UCD SNMP Agent Are Not Part of the RPR Software

PB010\_04\_092805

Figure 4: RPR MAC Driver and Management Software Modules

As shown in Figure 4, the software modules are divided into user parts, user space entities, and kernel space entities.

The kernel space entity is the RPR MAC driver. The MAC driver performs the following operations:

- FPGA initialization
- Topology refresh
- Fetch statistics counter values
- Fetch RPR MIB object mapped register values
- Check RPR events status
- Interrupt status polling (performs the following if the corresponding interrupts have occurred)
  - Receive ATD frame
  - Receive OAM frame

The user space entities are:

- The MIB Tracker - Performs the task of refreshing the MIB tables that are maintained in the shared memory

segment.

- MLME - An entity within the SNMP Agent (UCD-SNMP) that handles all SNMP requests to the RPR MIB.

The MLME is comprised of functions that handle the following scenarios:

- SNMP GET requests on the RPR MIB - When an SNMP GET request is received for a particular RPR MIB object, the appropriate MLME handler is called to fetch the object's value from the appropriate MIB table in the shared memory.
- SNMP SET requests on the RPR MIB - When an SNMP SET request is received for a particular RPR MIB object, the appropriate MLME handler is called. The handler not only modifies the object's value in the shared memory, but it also requests the RPR driver to set the corresponding MIB register in the hardware

abstraction layer (HAL).

- SNMP trap generations for RPR events - The RPR driver notifies the MLME asynchronously whenever certain RPR events occur. The MLME handler then fetches the RPR events status from the driver and generates SNMP traps that are sent to the SME.
- LLC - Performs the following operations
  - Checks connectivity with a particular station
  - Initiates a flush operation
- RPR-CLI - Provides a command line interface to the local user/administrator. This application is designed to provide a set of commands to the client to enable local administration and testing.

### FPGA Resources

The RPR MAC is targeted to work on Xilinx Virtex-4 FPGAs. Based on various system-level parameters, the implementation can be on either Virtex-4 FX or LX devices. **Table 2** lists the resource utilization of the RPR MAC.

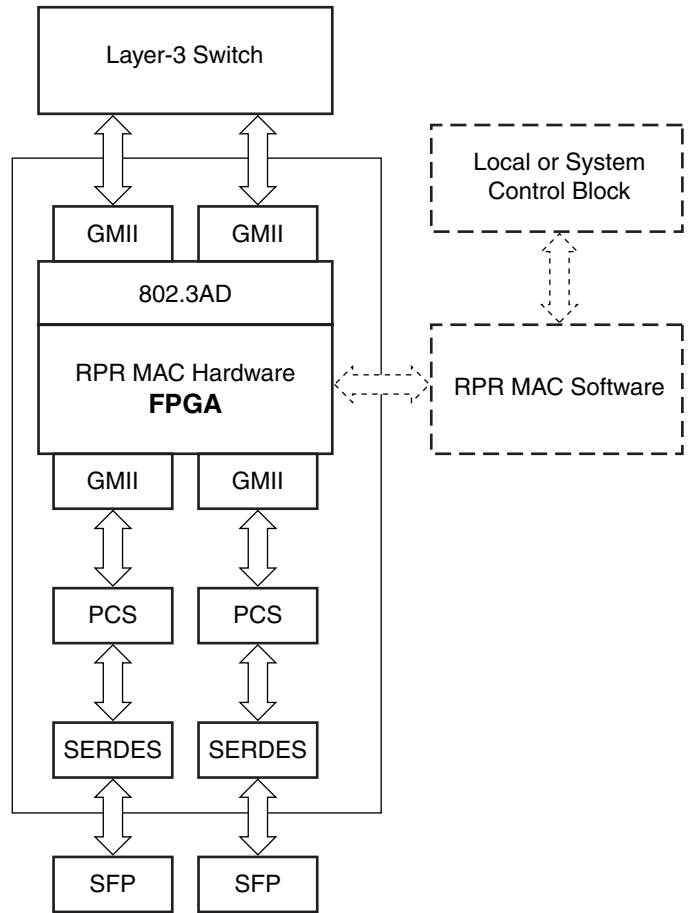
The RPR MAC fits in XC4VLX40/ XC4VFX40 with about 20% headroom available for customization.

Table 2: FPGA Resource Utilization

Device Family	Virtex-4 LX and FX Devices			
Speed Grades	-10 for 1G, 2.5G, and 10G			
Resources Used (exact value depends on configuration)	Slices	Block RAM	DCM	FFs
	15K – 16.5K	72 – 90	5 – 7	18744
Provided with core	User Guide Product Specifications Detailed Interface Guidelines			
Design File Formats	EDIF and NGC Netlist			
Test Bench Files	VHDL Testbench Perl Scripts			
Software	Device Driver for Linux RPR MIB Support Functions CLI			
Tools				
Xilinx Implementation Tools	ISE7.1 or Later			
Simulation	Modelsim PE 5.7 or Later			

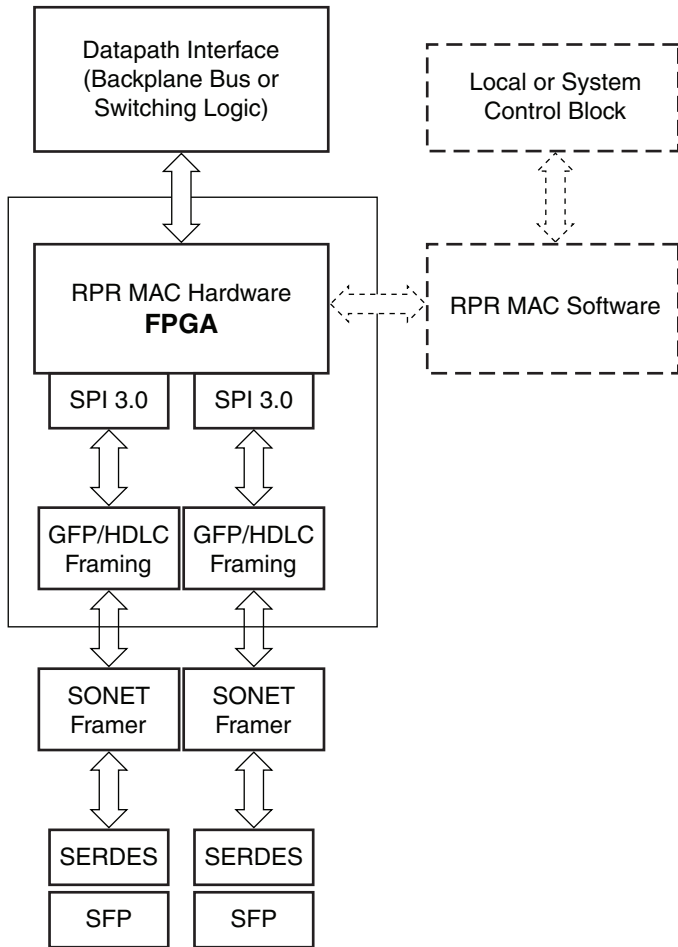
### Application Diagrams

Figure 5 and Figure 6 show example architectures for common RPR applications.



PB010\_05\_100605

Figure 5: For 1G Applications



PB010\_06\_100505

Figure 6: For 2.5G Applications

## Revision History

The following table shows the revision history for this document.

Date	Version	Revision
10/19/05	1.0	Initial Xilinx release.
08/13/07	1.0.1	Minor typographical edits.