Rambus ASIC Cell (RAC)

Description

The Rambus™ ASIC Cell (RAC™) is a standard macrocell used in ASIC designs to interface the core logic of a CMOS ASIC to a high speed Rambus Channel capable of transferring data at 2 nanoseconds per byte. The RAC makes use of Rambus Signaling Logic (RSL) technology to enable Channel communication at this 500 MHz rate while utilizing standard ASIC design methodologies. The ASIC designer is provided full access and control over the Rambus Channel while data handling requirements are eased through the use of eight-byte wide internal data paths. The RAC is flexible enough to be used for implementations ranging from a simple graphics interface to a multiport high performance memory controller.

RAC Features

- Rambus Interface:
  - 500 MB/second peak transfer rate
  - RSL I/O levels used for Channel interface
  - Standard CMOS I/O levels used for ASIC core interface
  - Synchronous protocol for fast block-oriented transfers
- Implemented in standard CMOS processes
- Resides in ASIC I/O pad ring and therefore does not affect the number of core gates available
- Interfaces to ASIC core at 1/8th the Rambus data rate
- Both synchronous and asynchronous ASIC interfaces allow forflexible system design
- Built-in testability support

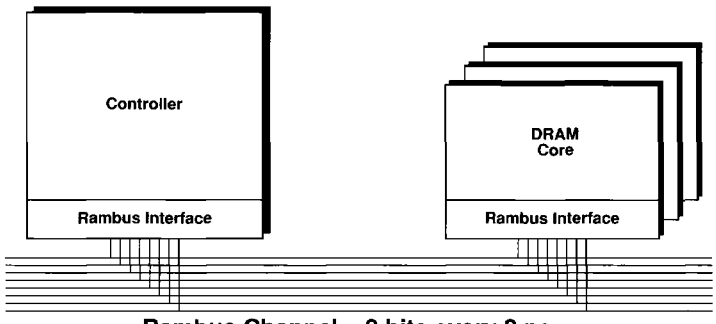
Rambus System Overview

A typical Rambus memory system has three main elements: the Rambus Channel, one or more Rambus DRAMs (RDRAM™), and a Rambus Interface on a controller. The logical representation of this is shown in the figure below.

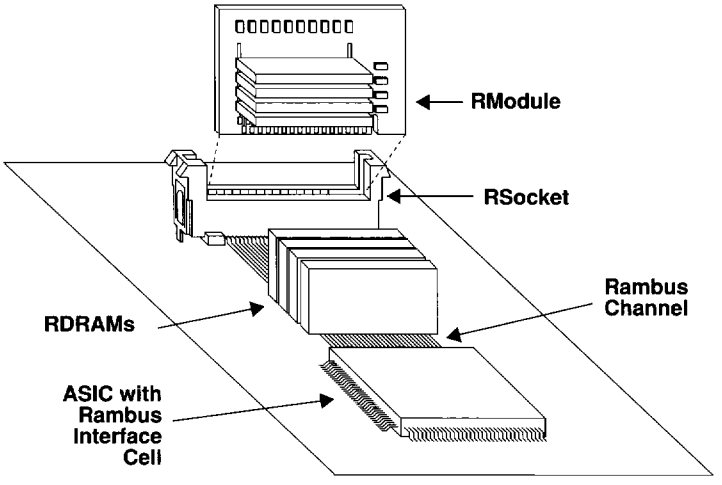
The Rambus Channel is a synchronous, high-speed, byte-wide bus that is used to directly connect Rambus devices together. Using only 13 high-speed signals, the Channel carries all address, data, and control information to and from devices through the use of a high level block-oriented protocol.

The Rambus Interface is implemented on both master and slave devices. Masters contain intelligence and are the only devices that generate transaction requests. Rambus masters can be ASIC devices, memory controllers, graphics engines, peripheral chips, or microprocessors. RDRAMs are slave devices and only respond to requests from master devices.

The figure on the next page shows a typical physical implementation of a Rambus system. It includes a controller ASIC that acts as the Channel master and a base set of RDRAMs soldered directly to the board. An RSocket™ is included on the Channel for memory upgrade using RModule™ expansion cards.



Rambus Channel = 9 bits every 2 ns

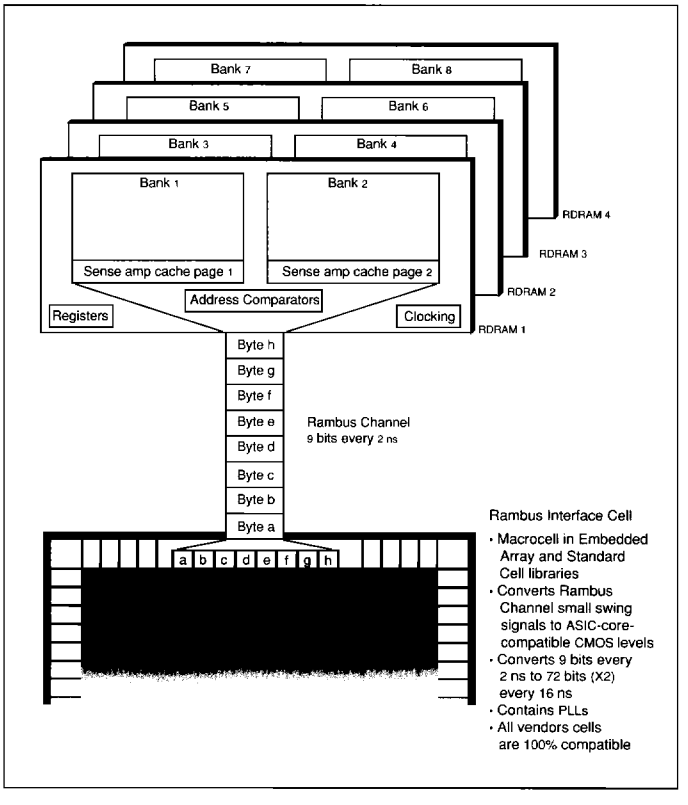


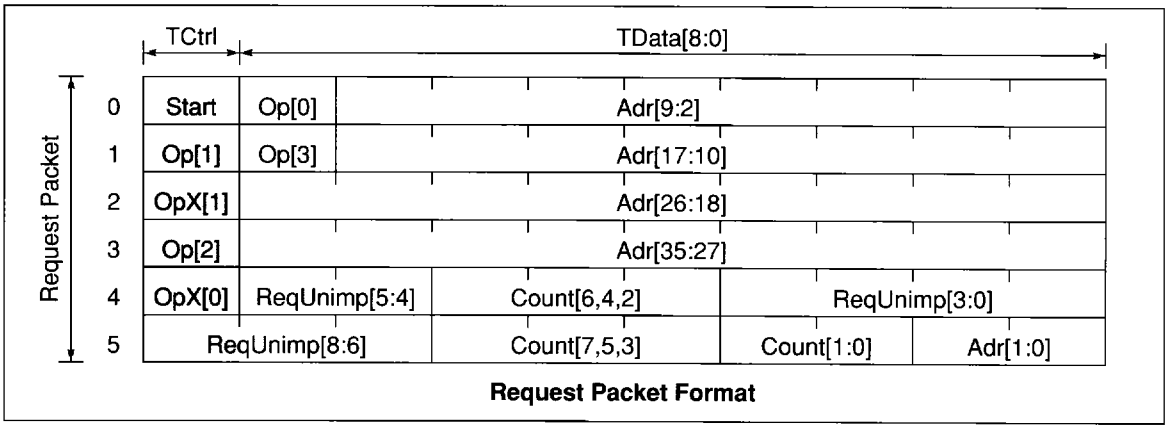
A Rambus System Example

Rambus Signaling Logic

RSL technology is the key to attaining the high data rates available in Rambus systems. By employing high quality transmission lines, current-mode drivers, low capacitive loading, low-voltage signaling, and precise clocking, systems reliably transfer data at 2 nanosecond intervals on a Rambus Channel with signal quality that is superior to TTL or GTL-based interfaces.

All Rambus Interfaces incorporate special logic to convert signals from RSL to CMOS levels for internal use. In addition, these interfaces convert the Channel data rate of one byte every 2 nanoseconds to an internal data rate of 8 bytes every 16 nanoseconds as shown in the figure below. Although the bandwidth remains the same, the use of a wide internal bus eases internal timing requirements for chip designers.



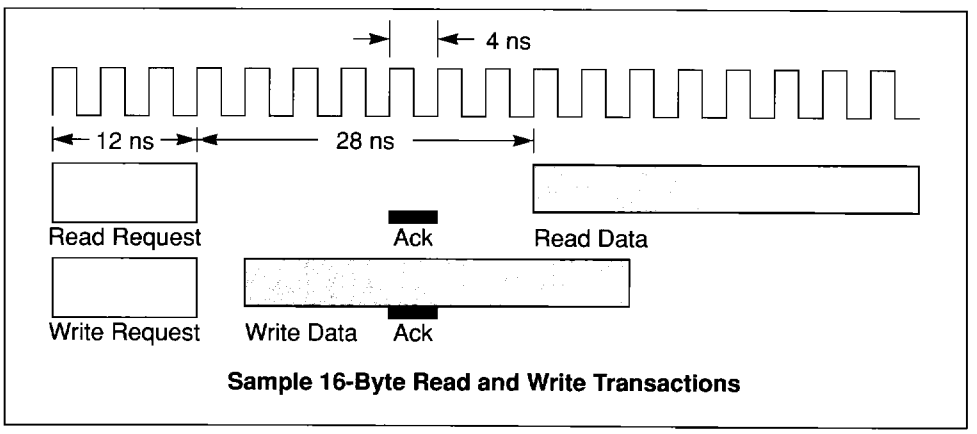


Request Packet Format

Protocol

The high-level transaction protocol used in Rambus systems is built from several types of information packets. These include the request, acknowledge, serial mode, and data packets. A master device initiates a transaction by generating a six byte request packet containing address, control, and byte count information as shown in the figure above.

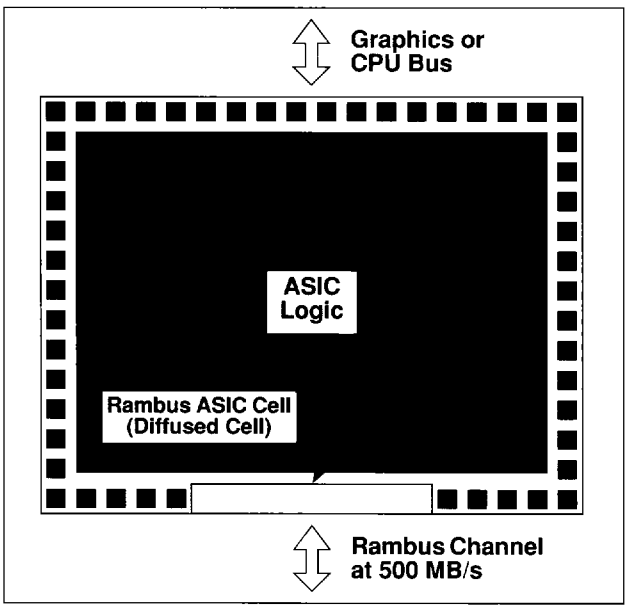
All slave devices constantly monitor the Channel for a request to access their assigned memory range. The device matching the address range requested then drives an acknowledge packet back to the master. The RDRAM also drives a data packet back to the master in the case of a read, or accepts a data packet from the master in the case of a write. The figure below shows example 16 byte read and write transactions. The actual timing from the end of a request packet to data and acknowledge packets is adjustable through RDRAM register settings.



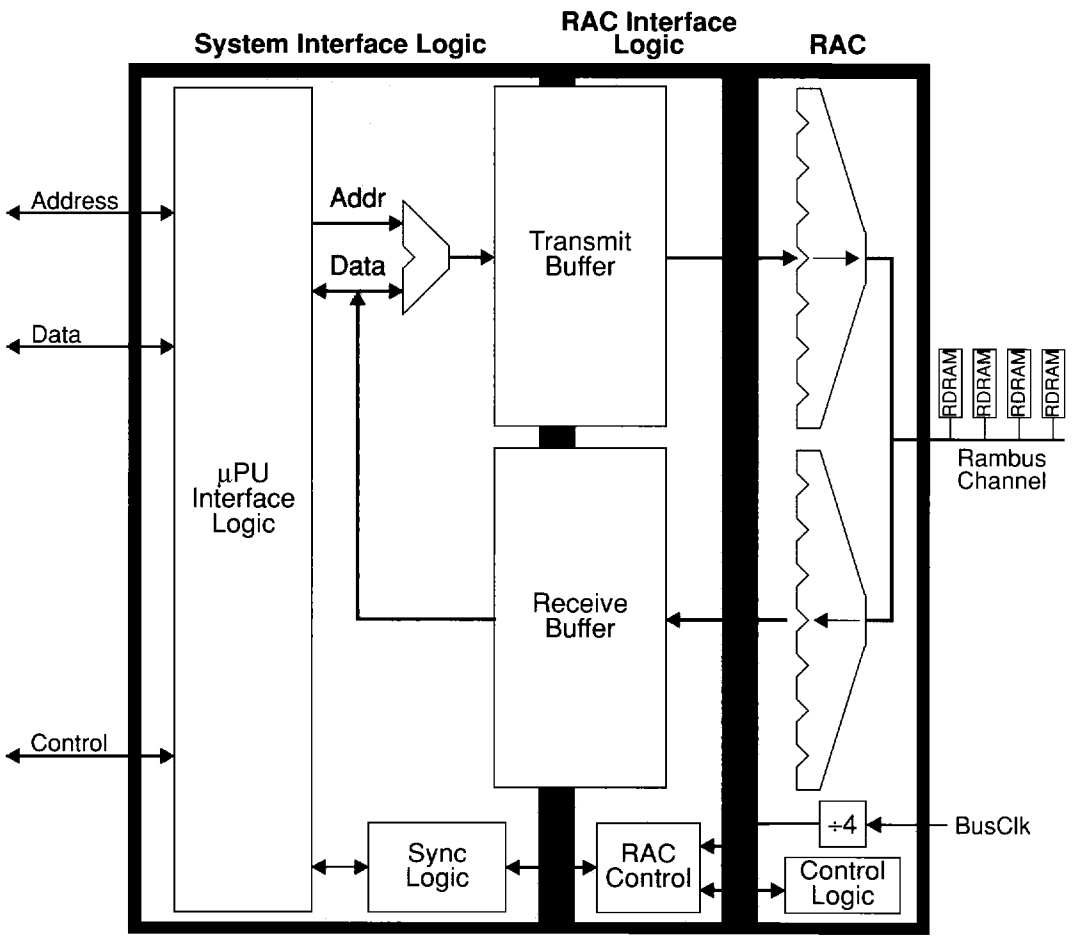
Sample 16-Byte Read and Write Transactions

RAC Overview

The RAC is a diffused ASIC cell that resides in a portion of the ASIC's I/O pad ring as shown in the figure to the right. This cell converts the high speed RSL signals of the Rambus Channel into lower speed CMOS level signals usable by the ASIC designer. The RAC is able to slow the data rate by functioning as a high performance parallel-to-serial and serial-to-parallel converter. A typical Rambus Channel can deliver one byte of data every 2 nanoseconds. Inside the ASIC, the designer sees this as eight bytes of data every 16 nanoseconds.



The figure below shows a simple block diagram of a Rambus memory controller ASIC for an example microprocessor application. The RAC moves blocks of data to and from the ASIC core through eight-byte wide transmit and receive buffers. These are used to buffer data flow and match transfer rates between the Rambus and CPU interfaces. The depth required for

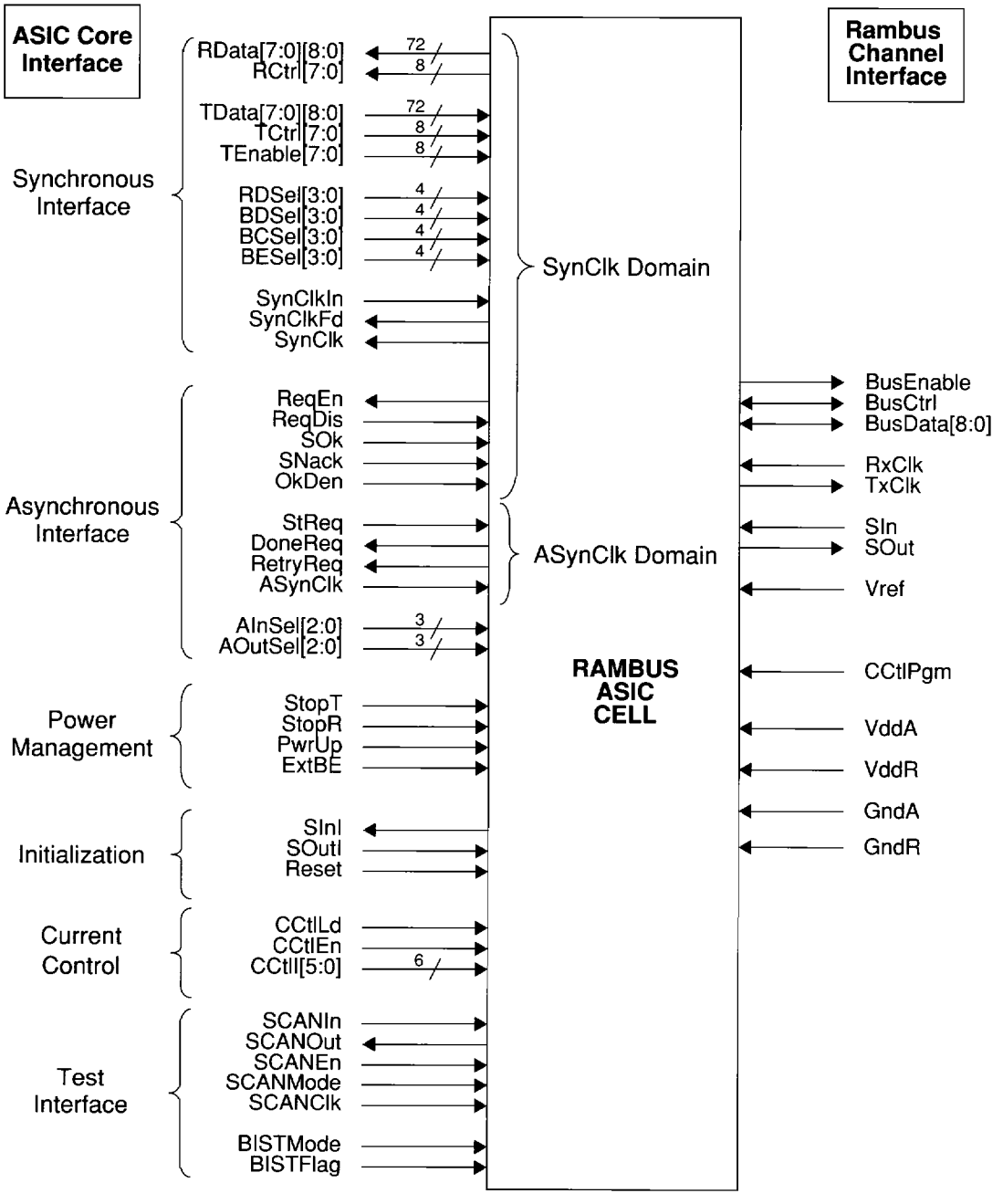


Rambus Memory Controller ASIC Block Diagram

these buffers depends on the nature and complexity of the design. Aside from these buffers, only a small amount of control logic is required to interface the memory controller core to the RAC. Because data is handled in eight-byte blocks (octbytes) on the ASIC side of the RAC, control logic runs at 1/8 of the Rambus Channel data rate, or 62.5 MHz for a 500 MB/s Rambus System.

RAC Interface Signals

The next figure shows all of the RAC I/O signals. There are two primary sets of signals, those that connect to the Rambus Channel and those that connect internally to the ASIC core. The signals which connect to the Rambus Channel conform to Rambus electrical and timing parameters. The signals which connect to the ASIC core all switch at CMOS I/O levels and are designed to connect directly to the user's internal logic.



Synchronous Interface

The synchronous interface of the RAC is the primary means for accessing a Rambus Channel. This interface generates a synchronous clock, SynClk, by dividing the Rambus Channel clock signals by four. Since one byte of data is transferred every half clock cycle on the Channel, eight bytes of data can be transferred to and from the ASIC core once every SynClk cycle via the 72-bit RData and TData buses. All control and data path circuitry connecting to the synchronous interface should be clocked by SynClk.

In a minimum Rambus ASIC, this is the only major interface that must be used. Use of the asynchronous interface, power management controls, and test interface are all optional.

Asynchronous Interface

In a simple system, SynClk can be used to clock all ASIC circuitry. In most cases though, a controller will need to connect to a CPU or other device that is clocked asynchronous to the ASIC's SynClk signal. This means at some point, control signals will need to be synchronized to the SynClk clock domain in order to avoid metastability in the design.

Although synchronization can be done through the use of double rank synchronizers, these circuits can introduce additional latency to a memory access. The asynchronous interface on the RAC has been designed to address this problem by providing a very low latency cross-domain handshake that allows circuitry in the asynchronous clock domain to easily communicate with circuitry in the SynClk clock domain. This is done by using the high frequency Channel clock to sample and synchronize handshake signals.

Since this interface only handles handshaking, the synchronous interface is still used to actually transfer data to and from the Channel.

Power Management

Power management controls are used to power down portions of the RAC while not in use. Internally, the RAC has separate clocks driving the transmit and receive logic. The StopT input is used to disable the clocks to the transmit logic while StopR is used to disable the clocks to the receive logic. Asserting one or both of these can reduce power consumption.

The PwrUp control signal can be used in low power applications to completely power down the entire RAC cell. In order to be powered up again, the RAC cell will need to be reinitialized. This results in a significant power up latency that must be considered when using this feature.

Current Control

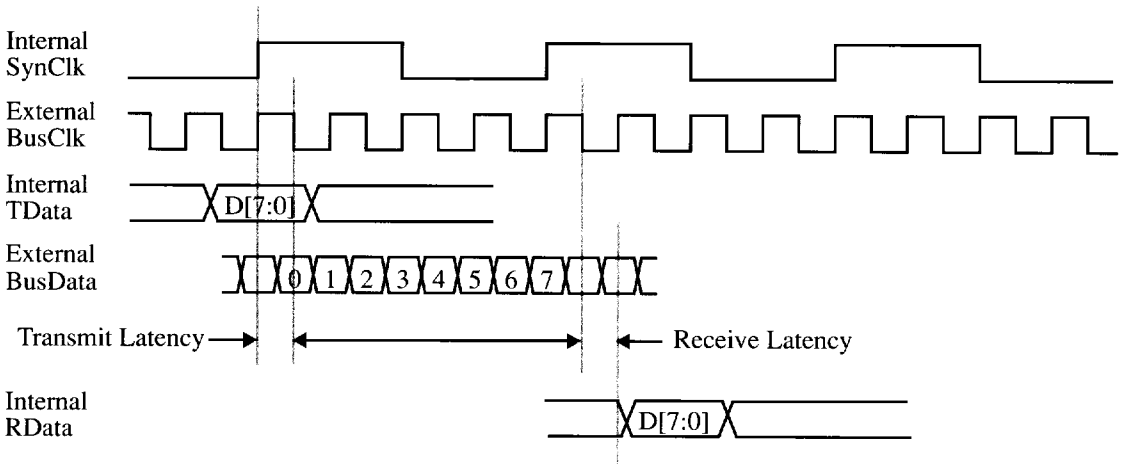
The Rambus Channel output drivers are open-drain current sinks. They are designed to sink the amount of current required to provide the proper voltage swing on a terminated transmission line (the Channel) with a characteristic impedance that may vary from system to system. In order to work with all system variations, the output drivers have been designed to be calibrated for each system. Each output driver is constructed from six binary weighted transistors that are driven in combination to provide the desired output current.

Because the drive current of the individual transistors will vary from RAC to RAC, and because the drive current will change with voltage and temperature, internal calibration circuitry is provided to help set and maintain constant current output over time.

Testability Support

To support testing of the RAC, all logic elements are connected in a scan chain which can be connected by the ASIC vendor into the internal ASIC scan chain and included in the ATPG. There is also a built-in self test function (BIST) which exercises the multiplexers, clock recovery circuits, output drivers and input samplers. The BIST loops back its test across the Rambus Channel, so that continuity on the Channel is verified during the test along with proper RAC operation.

Data Timing Diagram



Rambus Channel Signals

Signal Name	Type	Description
RxClk, TxClk (BusClk)	Input	Receive Clock and Transmit Clock. Both inputs connect to same signal and are jointly referred to as BusClk. All Rambus Channel transactions are synchronized to this clock. Low-swing signal referenced to Vref.
BusEnable	Output	Bus Enable. Carries control information to the slave devices. Low-swing signal referenced to Vref. Active low signal.
BusCtrl	I/O	Bus Control. Carries control information to and from slave devices. Low-swing signal referenced to Vref. Active low signal.
BusData[8:0]	I/O	Bus Data. Carries control and data information to and from slave devices. Low-swing signal referenced to Vref. Active low signal.
SOut	Output	Serial Out. Source of initialization daisy chain. TTL signal levels. Active high.
SIn	Input	Serial In. End of initialization daisy chain. TTL signal levels. Active high.
CCtlPgm	Input	Current Control Program. Provides a calibrated current into the RAC.
Vref	Input	Input threshold reference for low-swing signals.
VddR	Input	Power supply for the RAC cell.
GndR	Input	Ground for the RAC cell.
VddA	Input	Separate power supply for clock receivers.
GndA	Input	Separate ground for clock receivers.

ASIC Interface Signals - Synchronous Interface

Signal Name	Type	Description
RData[7:0][8:0]	Output	Receive Data. Eight bytes of BusData received from the Rambus Channel.
RCtrl[7:0]	Output	Receive Control. Eight bits of BusCtrl received from the Rambus Channel.
TData[7:0][8:0]	Input	Transmit Data. Eight bytes of data to be transmitted on the Rambus Channel BusData lines.
TCtrl[7:0]	Input	Transmit Control. Eight bits of data to be transmitted on the Rambus Channel BusCtrl line.
TEnable[7:0]	Input	Transmit Enable. Eight bits of data to be transmitted on the Rambus Channel BusEnable line.
RDSel[3:0]	Input	RData Select. Selects the valid data window timing of RData.
BDSel[3:0]	Input	BusData Select. Selects the sampling time of TData[7:0][8:0].
BCSel[3:0]	Input	BusCtrl Select. Selects the sampling time of TCtrl[7:0].
BESel[3:0]	Input	BusEnable Select. Selects the sampling time of TEnable[7:0].
SynClkIn	Input	Synchronous Clock In. Connected to SynClk or SynClkFd to set the phase of SynClk.
SynClkFd	Output	Synchronous Clock Feed. Connected to SynClkIn input to achieve synchronization between multiple RACs on the same chip.
SynClk	Output	Synchronous Clock. All RAC synchronous interface transfers are synchronous to this clock.

ASIC Interface Signals - Asynchronous Interface

Signal Name	Type	Description
ReqEn	Output	Request Enable. Synchronous output signalling that the RAC is in transmit mode and will begin transmitting data.
ReqDis	Input	Request Disable. Synchronous input signalling the end of the transmission and putting the RAC in standby mode.
SOk	Input	Synchronous Okay. Synchronous input signalling the successful (Okayed) completion of a request to RAC. SOk is used only when OkDEn is inactive.
SNack	Input	Synchronous Nack. Synchronous input signalling the unsuccessful (Nacked) completion of a request.
OkDe	Input	Okay Detect Enable. Enables the Okay detection feature of the RAC cell.
StReq	Input	Start Request. Asynchronous input indicating that the request packet is ready.
DoneReq	Output	Done Request. Asynchronous output signalling the successful completion of a request.
RetryReq	Output	Retry Request. Asynchronous output signalling the unsuccessful (Nacked) completion of a request.
ASynClk	Input	Asynchronous Clock. Clock used by the RAC to sample Asynchronous Interface signals.
AInSel[2:0]	Input	Asynchronous domain Input Select. Selects the sampling time of inputs from the ASynClk domain.
AOutSel[2:0]	Input	Asynchronous domain Output Select. Selects the valid data window timing of outputs to the ASynClk domain.

ASIC Interface Signals - Power Management

Signal Name	Type	Description
StopT	Input	Stop Transmit Clocks. Disables the transmit logic clocks.
StopR	Input	Stop Receive Clocks. Disables the receive logic clocks.
PwrUp	Input	Power Up. Applies power to the RAC when asserted. Negating PwrUp turns the RAC off to conserve power.
ExtBE	Input	External BusEnable. Alternate means along with TEnable[7:0] of driving BusEnable.

ASIC Interface Signals - Initialization

Signal Name	Type	Description
SInI	Output	Serial In Internal. Logically same as the SIn input at the Rambus interface.
SOutI	Input	Serial Out Internal Logically same as the SOut output at the Rambus interface
Reset	Input	Reset. Asynchronous input to initialize the RAC.

ASIC Interface Signals - Current Control

Signal Name	Type	Description
CCtlLd	Input	Current Control Load. Loads a new value into the current control register.
CCtlEn	Input	Current Control Enable. Selects whether the value on CCtlI[5:0] or an internally generated value will be loaded into the current control register. The internal value is used when CCtlEn is high.
CCtlI[5:0]	Input	Current Control Input. Data bus carrying the value to be loaded into the current control register when CCtlEn is low.

ASIC Interface Signals - Test Interface

Signal Name	Type	Description
SCANIn	Input	Scan Input. Serial Scan input data.
SCANOut	Output	Scan Output. Serial Scan output data.
SCANEn	Input	Scan Enable. Asserting SCANEn allows scanning in of new state and scanning out of old state
SCANMode	Input	Scan Test Mode Select. Asserting SCANMode puts the RAC into Scan test mode.
SCANClk	Input	Scan Clock. Clock signal for the RAC scan logic.
BISTMode	Input	BIST Mode Select. Asserting BISTMode initiates the RAC Built In Self Test.
BISTFlag	Output	BIST Flag. Indicates the result of the Built In Self Test. BISTFlag is asserted if the self test has been passed successfully.