Advance Information

MPC8245TS/D
Rev. 2, 8/2002

MPC8245
Integrated Processor
Technical Summary

This technical summary provides an overview of the MPC8245 integrated processor for high-performance embedded systems. The MPC8245 is a cost-effective, general-purpose integrated processor for applications using PCI in networking infrastructure, telecommunications, and other embedded markets. It can be used for control processing in applications such as network routers and switches, mass storage subsystems, network appliances, and print and imaging systems.

To locate any published errata or updates for this document, refer to the web site at http://www.motorola.com/semiconductors.

Part I Integrated Processor Overview

The MPC8245 integrated processor is comprised of a peripheral logic block and a 32-bit superscalar processor core, as shown in Figure 1.

The peripheral logic integrates a PCI bridge, dual universal asynchronous receiver/transmitter (DUART), memory controller, DMA controller, PIC interrupt controller, a message unit (and I_2O interface), and an I^2C controller. The processor core is a full-featured, high-performance processor with floating-point support, memory management, 16-Kbyte instruction cache, 16-Kbyte data cache, and power management features. The integration reduces the overall packaging requirements and the number of discrete devices required for an embedded system.

The MPC8245 contains an internal peripheral logic bus that interfaces the processor core to the peripheral logic. The core can operate at a variety of frequencies, allowing the designer to trade-off performance for power consumption. The processor core is clocked from a separate PLL, which is referenced to the peripheral logic PLL. This allows the microprocessor and the peripheral logic block to operate at different frequencies, while maintaining a synchronous bus interface. The interface uses a 64- or 32-bit data bus (depending on memory data bus width) and a 32-bit address bus along with control signals that enable the interface between the processor and peripheral logic to be optimized for performance. PCI accesses to the MPC8245 memory space are passed to the processor bus for snooping when snoop mode is enabled.

The processor core and peripheral logic are general-purpose in order to serve a variety of embedded applications. The MPC8245 can be used as either a PCI host or PCI agent controller.

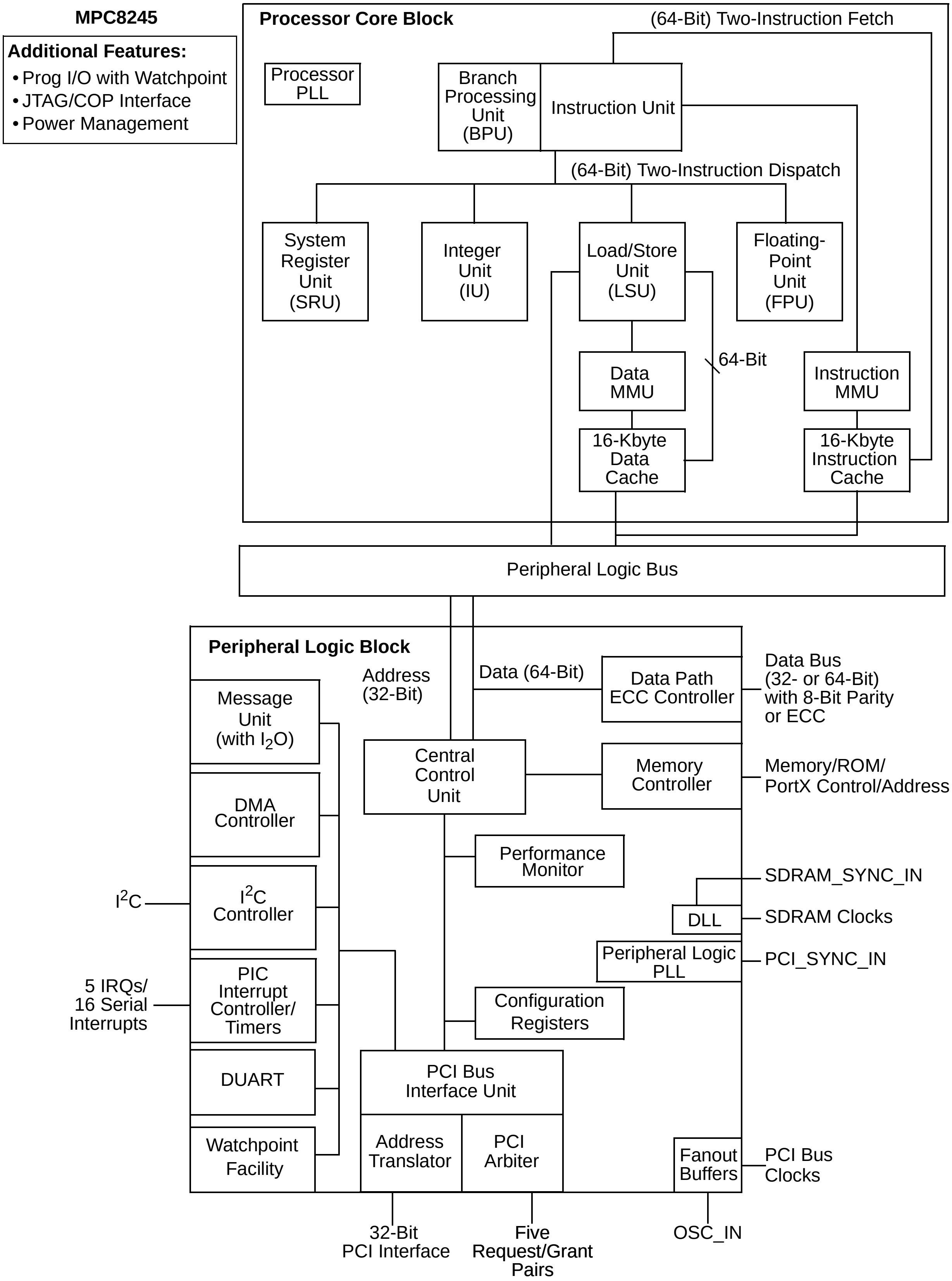

Figure 1. MPC8245 Integrated Processor Functional Block Diagram

1.1 Features

This section summarizes the features of the MPC8245. Major features of the MPC8245 are as follows:

- Processor core
  - High-performance, superscalar processor core
  - Integer unit (IU), floating-point unit (FPU) (software enabled or disabled), load/store unit (LSU), system register unit (SRU), and a branch processing unit (BPU)
  - 16-Kbyte instruction cache
  - 16-Kbyte data cache
  - Lockable L1 caches—entire cache or on a per-way basis up to three of four ways
  - Dynamic power management—supports 60x nap, doze, and sleep modes
- Peripheral logic
  - Peripheral logic bus
    - Supports various operating frequencies and bus divider ratios
    - 32-bit address bus, 64-bit data bus
    - Supports full memory coherency
    - Decoupled address and data buses for pipelining of peripheral logic bus accesses
    - Store gathering on peripheral logic bus-to-PCI writes
  - Memory interface
    - Supports up to 2 Gbytes of SDRAM memory
    - High-bandwidth data bus (32- or 64-bit) to SDRAM
    - Programmable timing supporting SDRAM
    - Supports 1 to 8 banks of 16-, 64-, 128-, 256-, or 512-Mbit memory devices
    - Write buffering for PCI and processor accesses
    - Supports normal parity, read-modify-write (RMW), or ECC
    - Data-path buffering between memory interface and processor
    - Low-voltage TTL logic (LVTTL) interfaces
    - 272 Mbytes of base and extended ROM/Flash/PortX space
    - Base ROM space supports 8-bit data path or same size as the SDRAM data path (32- or 64-bit)
    - Extended ROM space supports 8-, 16-, 32-bit gathering data path, 32- or 64-bit (wide) data path
    - PortX: 8-, 16-, 32-, or 64-bit general-purpose I/O port using ROM controller interface with programmable address strobe timing, data ready input signal ($\overline{\text{DRDY}}$), and 4 chip selects
  - 32-bit PCI interface
    - Operates up to 66 MHz
    - PCI 2.2-compatible
    - PCI 5.0-V tolerance
    - Support for dual address cycle (DAC) for 64-bit PCI addressing (master only)
    - Support for PCI locked accesses to memory

    - Support for accesses to PCI memory, I/O, and configuration spaces
    - Selectable big- or little-endian operation
    - Store gathering of processor-to-PCI write and PCI-to-memory write accesses
    - Memory prefetching of PCI read accesses
    - Selectable hardware-enforced coherency
    - PCI bus arbitration unit (five request/grant pairs)
    - PCI agent mode capability
    - Address translation with two inbound and outbound units (ATU)
    - Some internal configuration registers accessible from PCI
  - Two-channel integrated DMA controller (writes to ROM/PortX not supported)
    - Supports direct mode or chaining mode (automatic linking of DMA transfers)
    - Supports scatter gathering—read or write discontinuous memory
    - 64-byte transfer queue per channel
    - Interrupt on completed segment, chain, and error
    - Local-to-local memory
    - PCI-to-PCI memory
    - Local-to-PCI memory
    - PCI memory-to-local memory
  - Message unit
    - Two doorbell registers
    - Two inbound and two outbound messaging registers
    - I_2O message interface
  - I^2C controller with full master/slave support that accepts broadcast messages
  - Programmable interrupt controller (PIC)
    - Five hardware interrupts (IRQs) or 16 serial interrupts
    - Four programmable timers with cascade
  - Two (dual) universal asynchronous receiver/transmitters (UARTs)
  - Integrated PCI bus and SDRAM clock generation
  - Programmable PCI bus and memory interface output drivers
- System level performance monitor facility
- Debug features
  - Memory attribute and PCI attribute signals
  - Debug address signals
  - $\overline{\text{MIV}}$ signal: marks valid address and data bus cycles on the memory bus
  - Programmable input and output signals with watchpoint capability
  - Error injection/capture on data path
  - IEEE 1149.1 (JTAG)/test interface

1.2 Applications

The MPC8245 can be used for control processing in applications such as routers, switches, multi-channel modems, network storage, image display systems, enterprise I/O processor, internet access device (IAD), disk controller for RAID systems, and copier/printer board control.

Figure 2 shows the MPC8245 in the role of host processor.

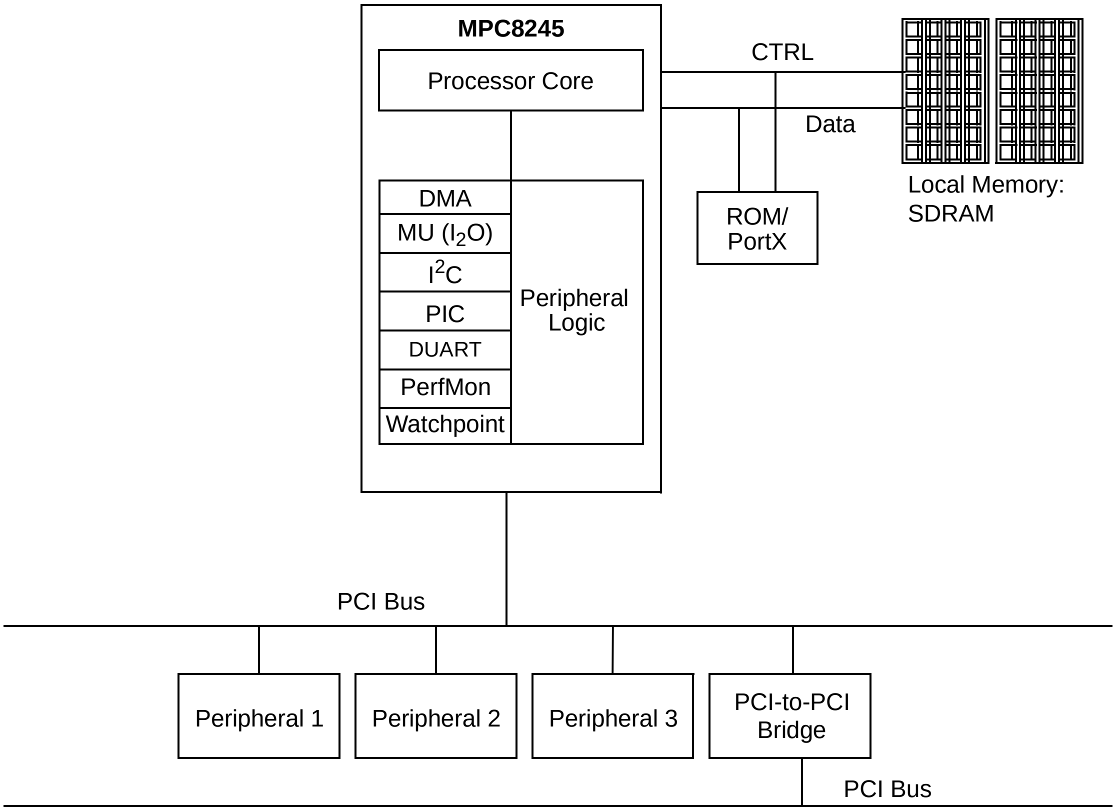

Figure 2. System Using an Integrated MPC8245 as a Host Processor

Figure 3 shows the MPC8245 in a peripheral processor application.

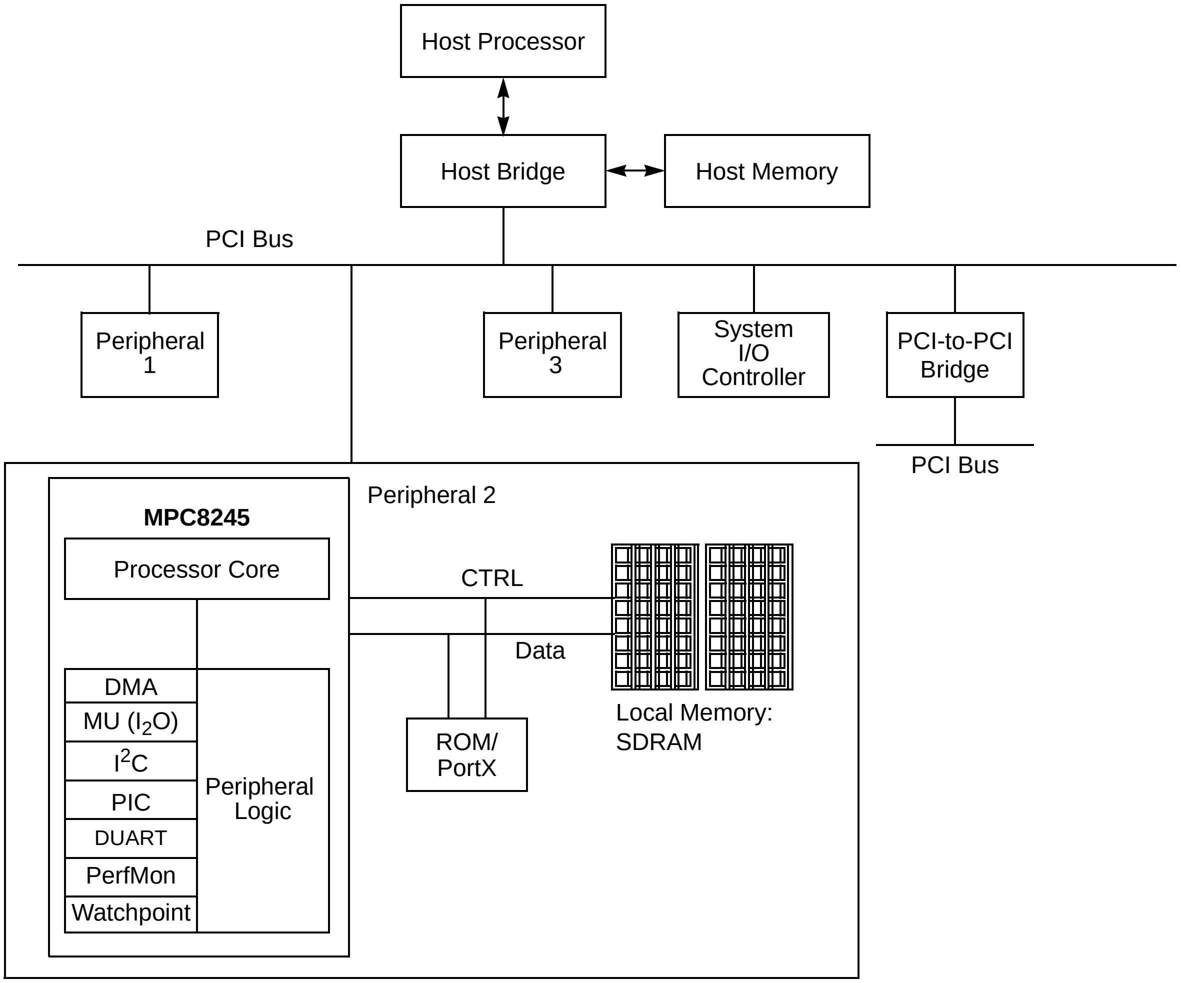

Figure 3. Embedded System Using an MPC8245 as a Peripheral Processor

Figure 4 shows the MPC8245 as a distributed I/O processing device. The PCI-to-PCI bridge shown could be of the PCI type 0 variety. The MPC8245 would not be part of the system configuration map. This configuration is useful in applications such as RAID controllers, where the I/O devices shown are SCSI controllers, or multi-port network controllers where the devices shown are Ethernet controllers.

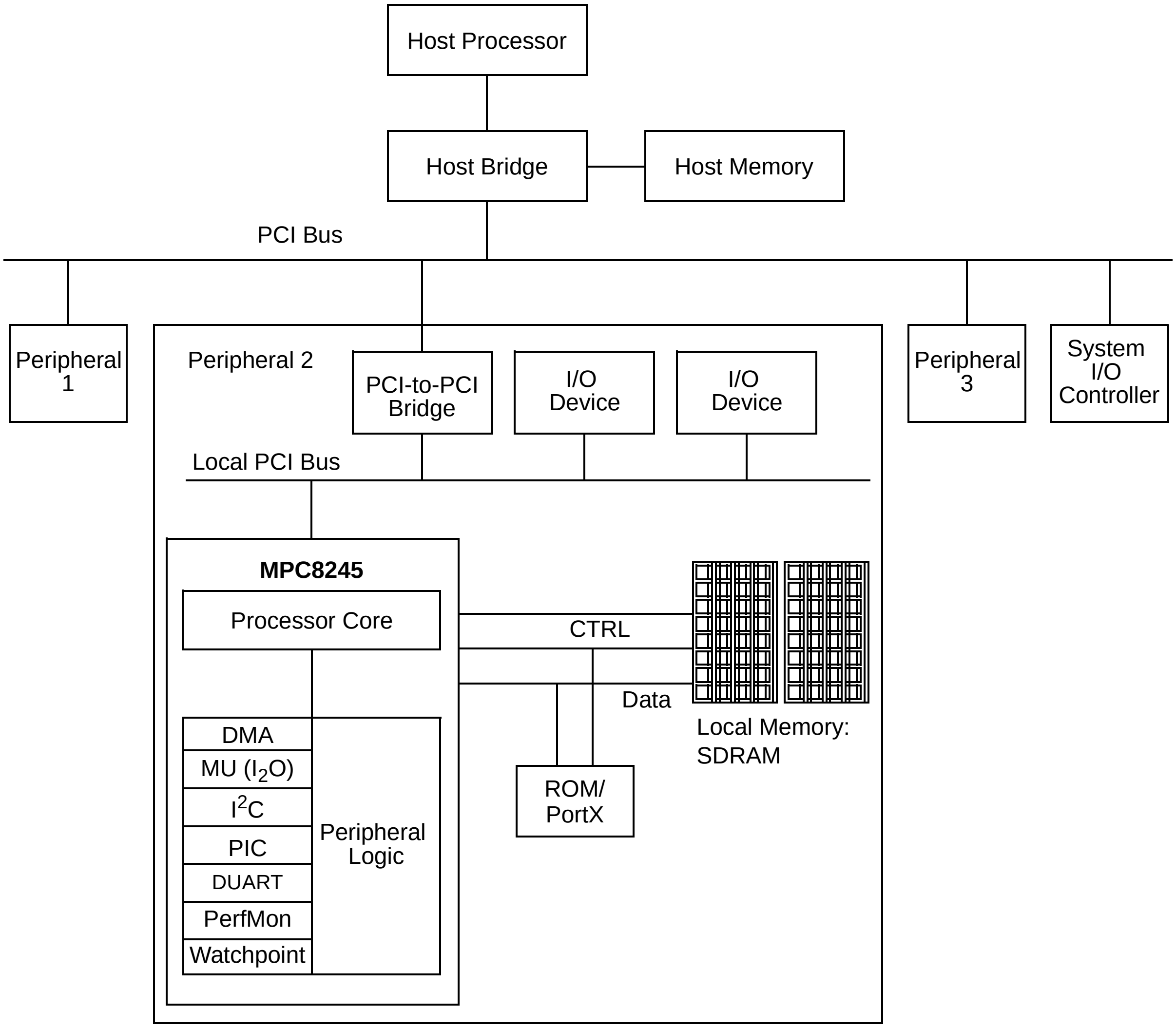

Figure 4. Embedded System Using an MPC8245 as a Distributed Processor

Part II Processor Core Overview

The MPC8245 contains an embedded version of the MPC603e processor (G2 processor core). For detailed information regarding the processor refer to the following:

- The *MPC603e RISC Microprocessor User's Manual* (those chapters that describe the programming model, cache model, memory management model, exception model, and instruction timing)
- The *Programming Environments Manual*

This section is an overview of the processor core, provides a block diagram showing the major functional units, and describes briefly how those units interact.

The processor core is a low-power implementation of the family of microprocessors that implement the PowerPC architecture. The processor core implements the 32-bit portion of the PowerPC architecture, which provides 32-bit effective addresses, integer data types of 8, 16, and 32 bits, and floating-point data types of 32 and 64 bits.

The processor core is a superscalar processor that can issue and retire as many as three instructions per clock. Instructions can execute out of order for increased performance, however, the processor core makes completion appear sequential.

The processor core integrates five execution units—an integer unit (IU), a floating-point unit (FPU), a branch processing unit (BPU), a load/store unit (LSU), and a system register unit (SRU). The ability to execute five instructions in parallel and the use of simple instructions with rapid execution times yield high efficiency and throughput. Most integer instructions execute in one clock cycle. On the processor core, the FPU is pipelined so a single-precision multiply-add instruction can be issued and completed every clock cycle.

The processor core supports integer data types of 8, 16, and 32 bits, and floating-point data types of 32 and 64 bits.

The processor core provides independent on-chip, 16-Kbyte, four-way set-associative, physically-addressed instruction and data caches. The processor also features independent on-chip instruction and data memory management units (MMUs). The MMUs contain 64-entry, two-way set-associative, data and instruction translation lookaside buffers (DTLB and ITLB) that provide support for demand-paged virtual memory address translation and variable-sized block translation. The TLBs and caches use a least recently used (LRU) replacement algorithm. The processor also supports block address translation through the use of two independent instruction and data block address translation (IBAT and DBAT) arrays of four entries each. Effective addresses are compared simultaneously with all four entries in the BAT array during block translation. In accordance with the PowerPC architecture, if an effective address hits in both the TLB and BAT array, the BAT translation takes priority.

As an added feature to the processor core, the MPC8245 can lock the contents of one to three ways in the instruction and data cache (or an entire cache). For example, this allows embedded applications to lock interrupt routines or other important (time-sensitive) instruction sequences into the instruction cache. It allows data to be locked into the data cache, which may be important to code that must have deterministic execution.

The processor core has a selectable 32- or 64-bit data bus and a 32-bit address bus. The processor core supports single-beat and burst data transfers for memory accesses, and supports memory-mapped I/O operations.

Figure 5 provides a block diagram of the MPC8245 processor core that shows how the execution units (IU, FPU, BPU, LSU, and SRU) operate independently and in parallel. Note that this is a conceptual diagram and does not attempt to show how these features are physically implemented on the chip.

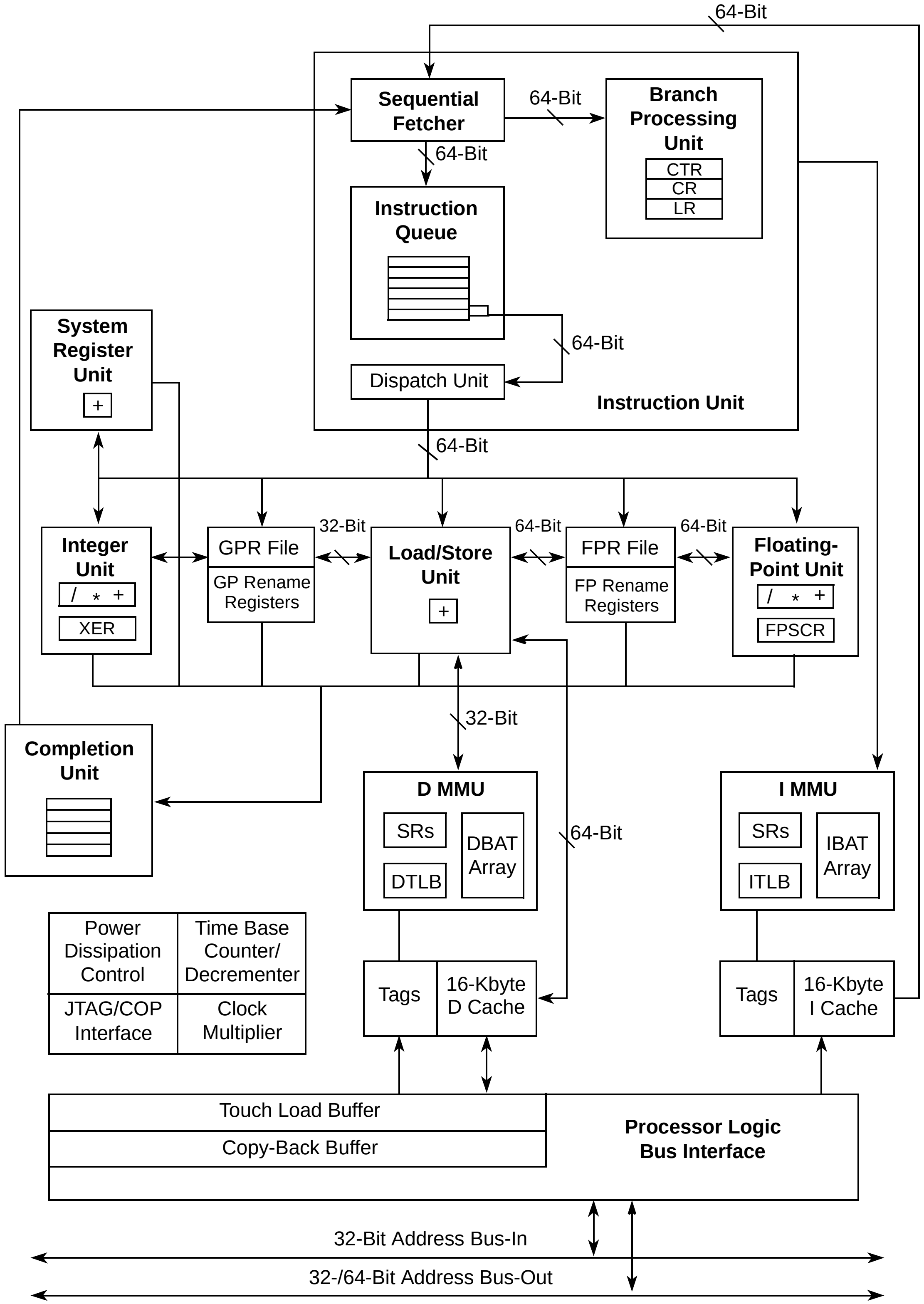

Figure 5. MPC8245 Integrated Processor Core Block Diagram

Part III Peripheral Logic Bus

The MPC8245 contains an internal peripheral logic bus that interfaces the processor core to the peripheral logic. The core can operate at a variety of frequencies allowing the designer to balance performance and power consumption. The processor core is clocked from a separate PLL, which is referenced to the peripheral logic PLL. This allows the microprocessor and the peripheral logic to operate at different frequencies while maintaining a synchronous bus interface.

The processor core-to-peripheral logic interface includes a 32-bit address bus, a 32- or 64-bit data bus as well as control and information signals. The peripheral logic bus allows for internal address-only transactions as well as address and data transactions. The processor core control and information signals include the address arbitration, address start, address transfer, transfer attribute, address termination, data arbitration, data transfer, data termination, and processor state signals. Test and control signals provide diagnostics for selected internal circuits.

The peripheral logic interface supports bus pipelining, which allows the address tenure of one transaction to overlap the data tenure of another. PCI accesses to the memory space are monitored by the peripheral logic bus to allow the processor to snoop these accesses (when snooping not explicitly disabled).

As part of the peripheral logic bus interface, the processor core’s data bus is configured at power up to either a 32- or 64-bit width. When the processor is configured with a 32-bit data bus, memory accesses on the peripheral logic bus interface allow transfer sizes of 8, 16, 24, or 32 bits in one bus clock cycle. Data transfers occur in either single-beat transactions, or two- or eight-beat burst transactions, with a single-beat transaction transferring as many as 32 bits. Single- or double-beat transactions are caused by noncached accesses that access memory directly (that is, reads and writes when caching is disabled, caching-inhibited accesses, and stores in write-through mode). Eight-beat burst transactions, which always transfer an entire cache line (32 bytes), are initiated when a line is read from or written to memory.

When the peripheral logic bus interface is configured with a 64-bit data bus, memory accesses allow transfer sizes of 8, 16, 24, 32, or 64 bits in one bus clock cycle. Data transfers occur in either single-beat transactions or four-beat burst transactions. Single-beat transactions are caused by noncached accesses that access memory directly (that is, reads and writes when caching is disabled, caching-inhibited accesses, and stores in write-through mode). Four-beat burst transactions, which always transfer an entire cache line (32 bytes), are initiated when a block is read from or written to memory.

Part IV Peripheral Logic Overview

The peripheral logic block integrates a PCI bridge, memory controller, DMA controller, PIC unit/timers, a message unit with an intelligent input/output (I_2O) message interface, an inter-integrated circuit (I^2C) controller, and a dual universal asynchronous receiver/transmitter (DUART), performance monitor, and watchpoint facility. The integration reduces the overall packaging requirements and the number of discrete devices required for an embedded system.

Figure 6 shows the major functional units within the peripheral logic block. Note that this is a conceptual block diagram intended to show the basic features rather than an attempt to show how these features are physically implemented.

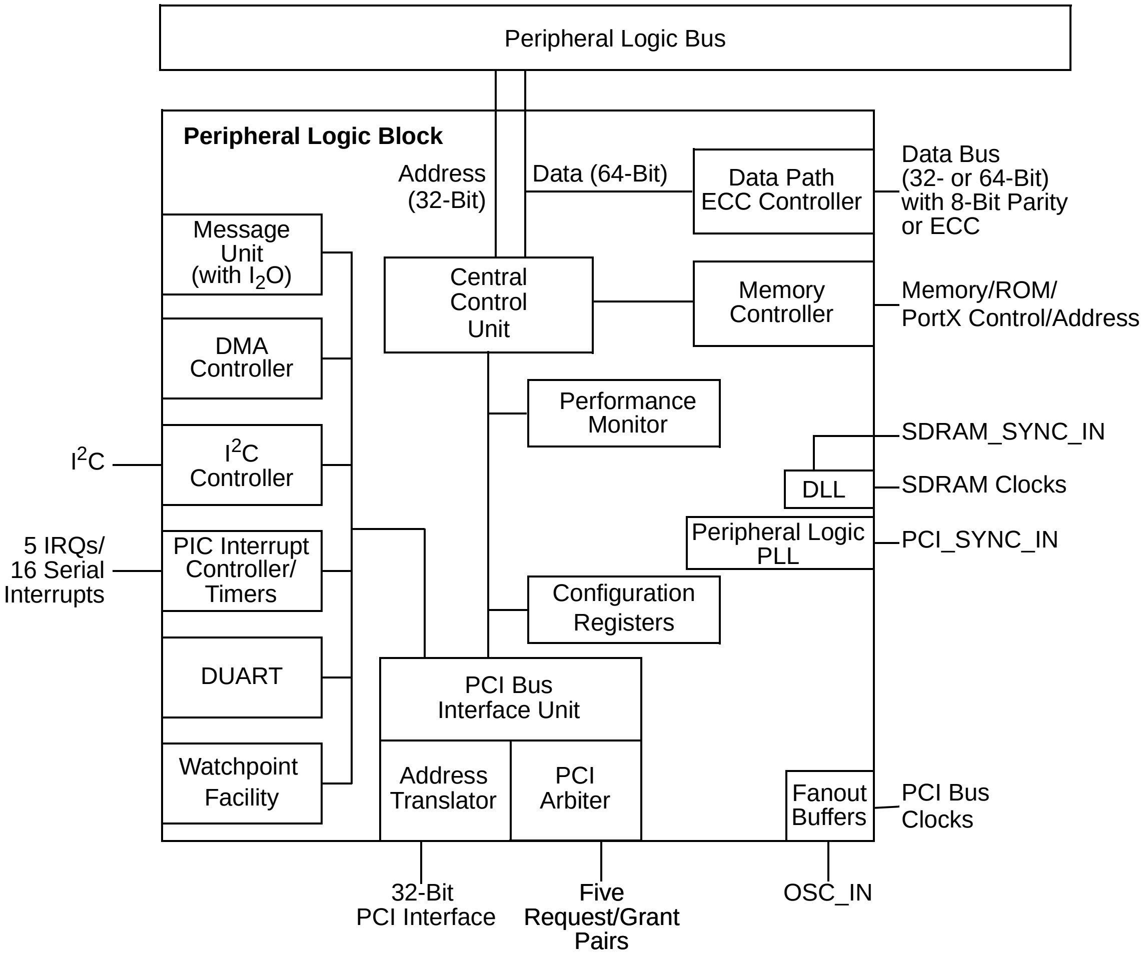

Figure 6. MPC8245 Peripheral Logic Block Diagram

4.1 Memory System Interface

The MPC8245 memory interface controls processor and PCI interactions to main memory. It supports a variety of Flash or ROM configurations. The MPC8245 supports synchronous DRAM (SDRAM). The maximum supported memory size is 2 Gbytes of SDRAM and 272 Mbytes of ROM/Flash. SDRAM must comply with the JEDEC SDRAM specification.

The MPC8245 is designed to control a 32- or 64-bit data path to main memory SDRAM. For a 32-bit data path, the MPC8245 can be configured to check and generate byte parity using 4 parity bits. For a 64-bit data path, the MPC8245 can be configured to support parity or ECC checking and generation with 8 parity/syndrome bits checked and generated. Note that the data bus width (32- or 64-bit) chosen at reset for the 60x bus interface is also used for the memory interface.

The MPC8245 supports SDRAM bank sizes from 1 to 512 Mbytes and provides bank start address and end address configuration registers. Note that the MPC8245 does not support DRAM. The MPC8245 can be configured so that appropriate row and column address multiplexing occurs according to the accessed memory bank. Addresses are provided to SDRAM through a 14-bit interface.

The ROM/Flash interface of the MPC8245 controls two areas of memory—base ROM space, which is a 16-Mbyte area; and extended ROM space, which is a 256-Mbyte area. Four chip selects, 1 write enable, 1 output enable, and up to 25 address signals are provided for ROM/Flash systems.

The MPC8245 implements PortX, a memory bus interface that facilitates the connection of general-purpose I/O devices. The PortX functionality allows the designer to connect external registers, communication devices, and other such devices directly to the MPC8245. Some devices may require a small amount of external logic to generate properly address strobes, chip selects, and other signals.

4.2 Peripheral Component Interconnect (PCI) Interface

The MPC8245 PCI interface provides mode-selectable, big- to little-endian conversion and can operate at speeds up to 66 MHz.

PCI bus commands as implemented by the MPC8245 include memory, I/O, and configuration reads and writes and special cycle, interrupt acknowledge, dual-address cycle, and other initiator-caused commands. The PCI interface is compatible with the *Peripheral Component Interconnect Specification*, Rev. 2.2 and includes the following:

- PCI agent capability
- PCI bus arbitration unit
- Address maps and translation
- Big- and little-endian modes
- PCI bus clock buffers and bus ratios

4.2.1 PCI Agent Capability

The MPC8245 PCI interface can be configured as host or agent. In host mode, the interface acts as the main memory controller for the system and responds to all host memory transactions.

In certain applications, the embedded system architecture dictates that the MPC8245 act as a peripheral processor. In this case, the peripheral logic must not act like a host bridge for the PCI bus. Instead, it functions as a configurable device that is accessed by a host bridge. This capability allows multiple MPC8245 devices to coexist with other PCI peripheral devices on a single PCI bus. The MPC8245 has PCI 2.2-compatible configuration capabilities.

In agent mode, the MPC8245 can be configured to respond to a programmed window of PCI memory space. A variety of initialization modes are provided to boot the device.

4.2.2 PCI Bus Arbitration Unit

The MPC8245 contains a PCI bus arbitration unit, which eliminates the need for an external unit, thus lowering system complexity and cost. It has the following features:

- Five external arbitration signal pairs. The MPC8245 is the sixth member of the arbitration pool.
- The bus arbitration unit allows fairness as well as a priority mechanism.
- A two-level round-robin scheme is used in which each device can be programmed within a pool of a high- or low-priority arbitration. One member of the low-priority pool is promoted to the high-priority pool. As soon as it is granted the bus, it returns to the low-priority pool.
- The unit can be disabled to allow a remote arbitration unit to be used.

4.2.3 Address Maps and Translation

The MPC8245 processor bus supports memory-mapped accesses. The address space is divided between memory and PCI according to address map B.

The MPC8245 allows remapping of PCI to local memory (inbound) transactions and processor core to PCI (outbound) transactions. Agent mode supports both inbound and outbound translation. Host mode only supports outbound translation. Note that address translation is supported only in address map B because agent mode is supported only for address map B.

When the MPC8245 is configured to be a PCI agent, the amount of local memory visible to the system is programmable. In addition, it may be necessary to map the local memory to a different system memory address space. The address translation unit handles the mapping of both inbound and outbound transactions for these cases.

4.2.4 Byte Ordering

The MPC8245 allows the processor to run in either big- or little-endian mode (except for the initial boot code which must run in big-endian mode).

4.2.5 Bus Clock Buffers and Bus Ratios

Refer to Section 4.8, “Integrated PCI Bus and SDRAM Clock Generation,” for information on clock buffers and ratios.

4.3 DMA Controller

The integrated DMA controller contains two independent units. Note that the DMA writing capability for local memory is available for SDRAM, but writing is not available for the ROM/PortX interface. Each DMA unit is capable of performing the following types of transfers:

- PCI-to-local memory
- Local-to-PCI memory
- PCI-to-PCI memory
- Local-to-local memory

The DMA controller allows chaining through local memory-mapped chain descriptors. Transfers can be scatter-gathered and misaligned. Interrupts are provided on completed segment, chain, and error conditions. PCI dual address cycle (DAC) support is provided.

4.4 Message Unit (MU)

Many embedded applications require handshake algorithms to pass control, status, and data information from one owner to another. This is made easier with doorbell and message registers. The MPC8245 has a message unit (MU) that implements doorbell and message registers as well as an I_2O interface. The MU has many conditions that can cause interrupts, and it routes external interrupts to the PCI interface and internal interrupts through PIC to the processor core.

4.4.1 Doorbell Registers

The MPC8245 MU contains one 32-bit inbound doorbell register and one 32-bit outbound doorbell register. The inbound doorbell register allows a remote processor to set a bit in the register from the PCI bus. This, in turn, generates an interrupt to the processor core. Only the processor core can clear the inbound doorbell register bits.

The processor core can write to the outbound register, causing the outbound interrupt signal $\overline{\text{INTA}}$ to assert, thus interrupting the host processor. When $\overline{\text{INTA}}$ is generated, it can be cleared only by the host processor by writing ones to the bits that are set in the outbound doorbell register.

4.4.2 Inbound and Outbound Message Registers

The MPC8245 contains two 32-bit inbound message registers and two 32-bit outbound message registers. The inbound registers allow a remote host or PCI master to write a 32-bit value, causing an interrupt to the processor core. The outbound registers allow the processor core to write an outbound message that causes the outbound interrupt signal $\overline{\text{INTA}}$ to assert.

4.4.3 Intelligent Input/Output Controller (I_2O)

The intelligent I/O specification is an open standard that defines an abstraction layer interface between the OS and subsystem drivers. Messages are passed between the message abstraction layer from one device to another.

The I_2O specification describes a system as being made up of host processors and input/output platforms (IOPs). The host processor is a single processor or a collection of processors working together to execute a homogenous operating system. An IOP consists of a processor, memory, and I/O interfaces. The IOP functions separately from other processors within the system to handle system I/O functions.

The I_2O controller of the MU enhances communication between hosts and IOPs within a system. There are two paths for messages—an inbound queue is used to transfer messages from a remote host or IOP to the processor core, and an outbound queue is used to transfer messages from the processor core to the remote host. Each queue is implemented as a pair of FIFOs. The inbound and outbound message queues each consists of a free_list FIFO and a post_list FIFO.

Messages are transferred between the host and the IOP using PCI memory-mapped registers. The MPC8245 I_2O controller facilitates moving the messages to and from the inbound and outbound registers and local IOP memory. Interrupts signal the host and IOP to indicate the arrival of new messages.

4.5 Inter-Integrated Circuit (I^2C) Controller

The I^2C serial interface has become an industry de facto standard for communicating with low-speed peripherals. Typically, it is used for system management functions and EEPROM support. The MPC8245 contains an I^2C controller with full master and slave functionality.

4.6 Programmable Interrupt Controller (PIC)

The MPC8245 integrated programmable interrupt controller (PIC) reduces the overall component count in embedded applications. The PIC unit is designed to collect external and internal hardware interrupts, prioritize them, and deliver them to the processor core.

The module operates in one of three modes:

- In direct mode, five level- or edge-triggered interrupts can be connected directly to an MPC8245.
- In pass-through mode, interrupts detected at the IRQ0 input are passed directly to the processor core. Also in this case, interrupts generated by the I_2O, I^2C, DMA, controllers, watchpoint monitor, doorbell and message registers, and DUART are passed to the $\overline{\text{L\_INT}}$ output signal.
- The MPC8245 provides a serial delivery mechanism when more than five external interrupt sources are needed. The serial mechanism allows for up to 16 interrupts to be serially scanned into the MPC8245. This mechanism increases the number of interrupts without increasing the number of pins but with increasing the interrupt latency.

The outbound interrupt request signal, $\overline{\text{L\_INT}}$, is used to signal interrupts to the host processor when the MPC8245 is configured for agent mode. The MPC8245 PIC includes four programmable timers that can be used for system timing or for generating periodic interrupts.

4.7 Dual Universal Asynchronous Receiver/Transmitter (DUART)

The MPC8245 DUART controls the processor core interface to the serial devices attached to the UART signals. Each UART is capable of converting the parallel data from the processor core into a single serial bit stream for outbound transmission. On inbound transmission, the UART converts the serial bit stream into the bytes for handling by the processor core.

Some of the features of the MPC8245 DUART unit include:

- Full-duplex operation

- Program model compatible with the original 16450 UART and the PC16550D, an improved version of the 16450 which can be put into an alternate mode (FIFO mode)
- 16450 register reset values
- FIFO mode for both transmitter and receiver provide 16-byte FIFOs
- Serial data encapsulation and decapsulation with standard asynchronous communication bits (START, STOP, and parity)
- Maskable transmit, receive, line status, and MODEM status interrupts
- Software-programmable baud generators divide SDRAM_CLK*n* by 1 to (2^{16} – 1) and generate a 16x clock
- Clear to send ($\overline{\text{CTS}}$) and ready to send ($\overline{\text{RTS}}$) MODEM control functions
- Software-selectable serial-interface data format (data length, parity, 1/1.5/2 STOP bit, baud rate)

4.8 Integrated PCI Bus and SDRAM Clock Generation

There are two PCI bus clocking solutions that are directed towards a wide range of operating frequencies with different system configurations and requirements. Trade-offs between operating frequency (for performance) and power consumption are easily managed. For systems where the MPC8245 is the host controller with a minimum number of clock loads, five clock fanout buffers are provided on-chip.

For systems requiring more clock fanout or where the MPC8245 is an agent device, external clock buffers may be used.

The MPC8245 provides an on-chip delay-locked loop (DLL) that supplies the external memory bus clock signals to SDRAM banks. The memory bus clock signals are of the same frequency and synchronous with the internal peripheral bus clock.

The four SDRAM clock outputs are generated by the internal DLL and can account for the trace length between the SDRAM_SYNC_OUT signal and SDRAM_SYNC_IN signal.

The MPC8245 requires a single clock input signal, PCI_SYNC_IN, which can be driven by the PCI clock fanout buffers—specifically, the PCI_SYNC_OUT output. PCI_SYNC_IN can also be driven by an external clock driver.

PCI_SYNC_IN is driven by the PCI bus frequency. An internal PLL, using PCI_SYNC_IN as a reference, generates an internal *sys_logic_clk* signal that is used for the internal logic. The peripheral bus clock frequency is configured at reset (by the MPC8245 PLL configuration signals, PLL_CFG[0:4]) to be a multiple of the PCI_SYNC_IN frequency.

The internal clocking of the processor core is generated from and synchronized to the internal peripheral bus clock by means of a second PLL. The core's PLL provides multiples of the internal processor core clock rates as specified in the *MPC8245 Integrated Processor Hardware Specifications*.

4.9 Performance Monitor

The MPC8245 core logic contains a performance facility that monitors bridge logic events such as SDRAM or PCI bus traffic, DUART, or a number of interrupts emanating from an interrupt controller. The performance monitor can be used for the following:

- To optimize overall system performance by monitoring bridge logic events.

- To understand the MPC8245 behavior in any system or software environment, because some systems or software environments are not easily characterized by signal traces or benchmarks.
- To help system developers bring up and debug their systems.

The performance monitor uses the following runtime registers in the embedded utility memory block:

- Performance monitor counter registers (PMC0–PMC3) are 32-bit counters used to count the times a software-selectable event has occurred.
- Command registers (CMDR0–CMDR3) that select the counter, type of event, event to be counted, and threshold for that event.
- Monitor mode control register (MMCR) that controls the operation of the performance monitor counters.

Part V Power Management

The MPC8245 provides both automatic and program-controllable power reduction modes for progressive reduction of power consumption.

The MPC8245 has independent power management functionality for both the processor core and the peripheral logic. The MPC8245 provides hardware support for three levels of programmable power reduction for both the processor and the peripheral logic. Doze, nap, and sleep modes are invoked by register programming—HID0 in the case of the processor core and configuration registers in the case of the peripheral logic block.

The processor and peripheral logic blocks are both fully static, allowing internal logic states to be preserved during all power-saving modes. The following sections describe the programmable power modes.

5.1 Programmable Processor Power Management Modes

Table 1 summarizes the programmable power-saving modes for the processor core. These are very similar to those available in the MPC603e device.

Table 1. Peripheral Logic Power Modes Summary

PM Mode	Functioning Units	Activation Method	Full-Power Wake Up Method
Full power	All units active	—	—
Full power (with DPM)	Requested logic by demand	By instruction dispatch	—
Doze	Bus snooping Data cache as needed Decrementer timer	Controlled by software (write to HID0)	External asynchronous exceptions (assertion of $\overline{\mathrm{SMI}}$ or $\overline{int}$) Decrementer exception Hard or soft reset Machine check exception ($\overline{mcp}$)

Table 1. Peripheral Logic Power Modes Summary (continued)

PM Mode	Functioning Units	Activation Method	Full-Power Wake Up Method
Nap	Decrementer timer	Controlled by software (write to HID0) and qualified with $\overline{\text{QACK}}$ from peripheral logic	External asynchronous exceptions (assertion of $\overline{\text{SMI}}$, or $\overline{int}$) Decrementer exception Negation of $\overline{\text{QACK}}$ by peripheral logic Hard or soft reset Machine check exception ($\overline{mcp}$)
Sleep	None	Controlled by software (write to HID0) and qualified with $\overline{\text{QACK}}$ from peripheral logic	External asynchronous exceptions (assertion of $\overline{\text{SMI}}$, or $\overline{int}$) Negation of $\overline{\text{QACK}}$ by peripheral logic Hard or soft reset Machine check exception ($\overline{mcp}$)

5.2 Programmable Peripheral Logic Power Management Modes

The following sections describe the power management modes of the peripheral logic. Table 2 summarizes the programmable power-saving modes for the peripheral logic block.

Table 2. Programmable Peripheral Logic Power Modes Summary

PM Mode	Functioning Units	Activation Method	Full-Power Wake Up Method
Full power	All units active	—	—
Doze	PCI address decoding and bus arbiter System RAM refreshing Processor bus request and NMI monitoring PIC unit I^2C unit PLL	Controlled by software (write to PMCR1)	PCI access to memory Processor bus request Assertion of NMI [1] Interrupt to PIC Hard reset
Nap	PCI address decoding and bus arbiter System RAM refreshing Processor bus request and NMI monitoring PIC unit I^2C unit PLL	Controlled by software (write to PMCR1) and processor core in nap or sleep mode ($\overline{\text{QREQ}}$ asserted)	PCI access to memory [2] Processor bus request Assertion of NMI [1] Interrupt to PIC Hard reset
Sleep	PCI bus arbiter System RAM refreshing (can be disabled) Processor bus request and NMI monitoring PIC unit I^2C unit PLL (can be disabled)	Controlled by software (write to PMCR1) and processor core in nap or sleep mode ($\overline{\text{QREQ}}$ asserted)	Processor bus request Assertion of NMI [1] Interrupt to PIC Hard reset

[1] Programmable option based on value of PICR1[MCP_EN] = 1.

[2] A PCI access to memory in nap mode does not cause $\overline{\text{QACK}}$ to negate, consequently, it does not wake up the processor core, and the processor core won't snoop this access. After servicing the PCI access, the peripheral logic automatically returns to the nap mode.

Part VI Programmable I/O Signals with Watchpoint

The MPC8245 programmable I/O facility allows the system designer to monitor the peripheral logic bus. Up to two watchpoints and their respective 4-bit countdown values can be programmed. When the programmed threshold of the selected watchpoint is reached, an external trigger signal is generated and the states of the peripheral logic address, control, and data buses are latched into user-readable registers.

Part VII Debug Features

The MPC8245 includes the following debug features:

- Memory attribute and PCI attribute signals
- Debug address signals
- $\overline{\text{MIV}}$ signal: marks valid address and data bus cycles on the memory bus
- Error injection/capture on data path
- IEEE 1149.1 (JTAG)/test interface

7.1 Memory Attribute and PCI Attribute Signals

The MPC8245 provides additional information corresponding to memory and PCI activity on several signals to assist with system debugging. The two types of attribute signals are described as follows:

- The memory attribute signals are associated with the memory interface and provide information concerning the source of the memory operation being performed by the MPC8245.
- The PCI attribute signals are associated with the PCI interface and provide information concerning the source of the PCI operation being performed by the MPC8245.

7.2 Memory Debug Address

When enabled, the debug address provides software disassemblers a simple way to reconstruct the 30-bit physical address for a memory bus transaction to SDRAM and ROM, Flash, or PortX. For SDRAM, these 16 debug address signals are sampled with the column address and chip-selects. For ROMs, Flash, and PortX devices, the debug address pins are sampled at the same time as the ROM address and can be used to recreate the 25-bit physical address in conjunction with ROM address. The granularity of the reconstructed physical address is limited by the bus width of the interface; double-words for 64-bit interfaces, words for 32-bit interfaces, and bytes for 8-bit interfaces.

7.3 Memory Interface Valid ($\overline{\text{MIV}}$)

The memory interface valid signal, $\overline{\text{MIV}}$, is asserted whenever SDRAM, Flash, or ROM addresses or data are present on the external memory bus. It is intended to help reduce the number of bus cycles that logic analyzers must store in memory during a debug trace.

7.4 Error Injection/Capture on Data Path

The MPC8245 provides hardware to exercise and debug the ECC and parity logic by allowing the user to inject multi-bit stuck-at faults onto the peripheral logic or memory data/parity buses and to capture the data/parity output on receipt of an ECC or parity error.

7.5 IEEE 1149.1 (JTAG)/Test Interface

The processor core provides IEEE 1149.1 functions for facilitating testing and software debugging. The IEEE 1149.1 test interface provides a means for boundary-scan testing the MPC8245 and the board to which it is attached.

Part VIII Differences Between the MPC8245 and the MPC8240

The design philosophy of the MPC8245 is to maintain the MPC8240 base, but improve some features and add a few others for enhanced capability in the embedded market. The MPC8245 core is essentially the same as the MPC8240, but new features such as a DUART have been added to the peripheral logic block. Differences are summarized in Table 3.

Table 3. Differences Between the MPC8245 and the MPC8240

Supported Feature	Additions to MPC8245
SDRAM	Supports register and inline buffer modes (flow through no longer supported)
	Support for up to 2-Gbyte, 133 MHz SDRAM memory
	Supports up to 256-Mbit memory technology
ROM/Flash	Supports 272 Mbytes available ROM space (the added extended ROM mode supports 256 Mbytes of ROM or PortX)
	Supports a 16-bit width ROM data bus in addition to 8-, 32-, and 64-bit widths (MPC8240)
PortX	Supports a 16-bit ROM I/O port in addition to 8-, 32-, and 64-bit ports (MPC8240)
	Supports the $\overline{\text{DRDY}}$ signal and two additional chip selects (now four)
	Allows PCI writes to PortX
PCI interface	PCI 2.2-compatible
	Adds a dual address cycle for 64-bit addressing
Address translation unit	Has two ATUs
PIC	Adds a cascade function for the four PIC timers and counters
I^2C controller	Accepts broadcast messages
Performance monitor	Adds a system level performance monitor with interrupts and PCI arbitration monitor
Core and I/O voltages	The electrical characteristics of the MPC8245 are different from those of the MPC8240. See the corresponding hardware specifications for each device.
DUART	Has dual 2-pin UARTs
	Configurable to single 4-pin UART mode
	Functionality selectable by reset configuration signal

Part IX Revision History

This section provides a list of only the major changes made to this document. Table 4 provides a revision history for this technical summary.

Table 4. Document Revision History

Rev. No.	Substantive Change(s)
0–1	Previously released documents.
2	Updated template.
	Changed all instances of EPIC to PIC, register names EICR to ICR and EVI to IVI.
	Changed the fourth item under the ‘Memory interface’ heading
	Changed the second item under the ‘32-bit PCI Interface’ heading.
	Changed the first five paragraphs under Part II, “Processor Core Overview.”

THIS PAGE INTENTIONALLY LEFT BLANK

HOW TO REACH US:

USA/EUROPE/LOCATIONS NOT LISTED:

Motorola Literature Distribution
P.O. Box 5405, Denver, Colorado 80217
1-303-675-2140
(800) 441-2447

JAPAN:

Motorola Japan Ltd.
SPS, Technical Information Center
3-20-1, Minami-Azabu Minato-ku
Tokyo 106-8573 Japan
81-3-3440-3569

ASIA/PACIFIC:

Motorola Semiconductors H.K. Ltd.
Silicon Harbour Centre, 2 Dai King Street
Tai Po Industrial Estate, Tai Po, N.T., Hong Kong
852-26668334

TECHNICAL INFORMATION CENTER:

(800) 521-6274

HOME PAGE:

www.motorola.com/semiconductors

Information in this document is provided solely to enable system and software implementers to use Motorola products. There are no express or implied copyright licenses granted hereunder to design or fabricate any integrated circuits or integrated circuits based on the information in this document.

Motorola reserves the right to make changes without further notice to any products herein. Motorola makes no warranty, representation or guarantee regarding the suitability of its products for any particular purpose, nor does Motorola assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. “Typical” parameters which may be provided in Motorola data sheets and/or specifications can and do vary in different applications and actual performance may vary over time. All operating parameters, including “Typicals” must be validated for each customer application by customer’s technical experts. Motorola does not convey any license under its patent rights nor the rights of others. Motorola products are not designed, intended, or authorized for use as components in systems intended for surgical implant into the body, or other applications intended to support or sustain life, or for any other application in which the failure of the Motorola product could create a situation where personal injury or death may occur. Should Buyer purchase or use Motorola products for any such unintended or unauthorized application, Buyer shall indemnify and hold Motorola and its officers, employees, subsidiaries, affiliates, and distributors harmless against all claims, costs, damages, and expenses, and reasonable attorney fees arising out of, directly or indirectly, any claim of personal injury or death associated with such unintended or unauthorized use, even if such claim alleges that Motorola was negligent regarding the design or manufacture of the part.

MOTOROLA