SYNOPSYS®

DesignWare IBM PowerPC 405-S CPU Core

Design View Databook

Database Release 1.1a
December 2006

Copyright Notice and Proprietary Information

Synopsys, Inc.
700 E. Middlefield Road
Mountain View, CA 94043

www.synopsys.com

Contents

Preface

About This Manual

This manual describes the features, functions, and interfaces of the DesignWare IBM PowerPC 405-S CPU Core. The DesignWare IBM PowerPC 405-S CPU Core is a DesignWare coreKit implementation of the PPC 405-S CPU Core from International Business Machines Corporation (IBM). Throughout this manual, the term PowerPC 405-S CPU refers to the DesignWare IBM PowerPC 405-S CPU Core.

This manual describes the functionality of the PowerPC 405-S CPU as viewed from its external interfaces and is intended for use by chip designers who are integrating the PowerPC 405-S CPU into a chip-level design. Readers are assumed to be familiar with HDL-based chip design methodologies and tools.

Related Documents

This manual is part of the DesignWare IBM PowerPC 405-S CPU Core document set. The complete documentation set resides in the doc directory of your coreConsultant workspace.

Manual Overview

This manual contains the following chapters and appendixes:

Preface	Describes the manual and lists the typographical conventions and symbols used in it; tells how to get technical assistance.
Chapter 1 “Introduction”	Provides a general description of the DesignWare IBM PowerPC 405-S CPU Core and the associated DesignWare coreKits.

Chapter 2 "Input/Output Interfaces"	Identifies the DesignWare IBM PowerPC 405-S CPU Core interfaces, defines signal naming conventions, provides an alphabetized listing of all signals, and provides an overview of the interfaces.

Typographical and Symbol Conventions

The following conventions are used throughout this document:

Table 1: Documentation Conventions

Convention	Description and Example
%	Represents the UNIX prompt.
Bold	User input (text entered by the user). `% cd $LMC_HOME/hdl`
`Monospace`	System-generated text (prompts, messages, files, reports). `No Mismatches: 66 Vectors processed: 66 Possible"`
Italic or *Italic*	Variables for which you supply a specific value. As a command line example: `% setenv LMC_HOME prod_dir` In body text: In the previous example, *prod_dir* is the directory where your product must be installed.
| (Vertical rule)	Choice among alternatives, as in the following syntax example: `-effort_level low | medium | high`
[ ] (Square brackets)	Enclose optional parameters: *pin1* [*pin2 ... pinN*] In this example, you must enter at least one pin name (*pin1*), but others are optional ([*pin2* … *pinN*]).
TopMenu > SubMenu	Pulldown menu paths, such as: **File > Save As …**

Getting Help

For customer support:

- Logon to http://solvnet.synopsys.com using your Solvnet ID and password
  a. Under Support Resources, Click Enter a Call to the support center located at the right top corner
  b. Select Designware STAR IP from product entry
  c. Select Processors from sub product category
  d. Specify the subject and details of issues/errors
  e. Fill in all relevant details and then click Submit
- Send an e-mail message to support_center@synopsys.com.
- Telephone your local support center:
  - United States:

    Call 1-800-245-8005 from 7 AM to 5:30 PM Pacific Time, Monday through Friday.
  - Canada:

    Call 1-650-584-4200 from 7 AM to 5:30 PM Pacific Time, Monday through Friday.
  - All other countries:

    Find other local support center telephone numbers at the following URL:

    http://www.synopsys.com/support/support_ctr

Additional Information

General information about Synopsys and its products is available at this URL:

http://www.synopsys.com

1 Introduction

The DesignWare IBM PowerPC 405-S CPU Core is a DesignWare version of the IBM PPC 405F5 32-bit RISC CPU core.

Synopsys delivers the DesignWare IBM PowerPC 405-S CPU Core (referred to in this databook as the PowerPC 405-S CPU) as a DesignWare coreKit. Separate coreKits are available for evaluation (Design View coreKit) and implementation purposes (Implementation View coreKit).

This chapter introduces the features and architecture of the PowerPC 405-S CPU. The topics are:

Features

The PowerPC 405-S CPU provides high performance and low power consumption. The PowerPC 405-S CPU RISC CPU executes at sustained speeds approaching one cycle per instruction. On-chip instruction and data cache arrays can be implemented to reduce chip count and design complexity in systems and improve system throughput.

PowerPC 405-S CPU features include:

- PowerPC RISC fixed-point CPU
  - Implements the PowerPC User Instruction Set Architecture (UISA) and extensions for embedded applications
  - Thirty-two 32-bit general purpose registers (GPRs)
  - Static branch prediction
  - Five-stage pipeline with single-cycle execution of most instructions, including loads and stores
  - Unaligned load/store support to cache, main memory, and on-chip memory (OCM)
  - Hardware multiply/divide for faster integer arithmetic
  - Multiply-accumulate instructions
  - Enhanced string and multiple-word handling
  - True little endian operation
  - Parity detection and reporting for the instruction cache, data cache, and translation lookaside buffer (TLB)
- Storage Control
  - Separate, configurable instruction and data cache units, both two-way set-associative
  - Eight words (32 bytes) per cache line
  - Support for 16KB instruction- and data- cache arrays
  - Instruction cache unit (ICU) non-blocking under line fills, data cache unit (DCU) non-blocking under line fills and flushes
  - Instruction fetch hit from line buffer
  - Data load/store hit to line buffer
  - Programmable ICU prefetching of next sequential line into line buffer

  - Programmable ICU prefetching of non-cacheable instructions, full line (eight words) or half line (four words)
  - Operand forwarding during cache line fills
  - Write-back or write-through DCU write strategies
  - Programmable allocation on loads and stores.
  - Doubleword instruction fetch from cache improves branch latency
- Virtual mode memory management unit (MMU)
  - Translation of the 4GB logical address space into physical addresses
  - Software control of page replacement strategy
  - Supports multiple page sizes, ranging from 1KB–16MB, simultaneously
  - Independent enabling of instruction and data translation/protection
  - Page level access control using the translation mechanism
  - Additional control over protection using zones
  - WIU0GE (write-through, cacheability, user-defined 0, guarded, endian) storage attributes
- WIU0GE (write-back/write through, cacheability, user-defined 0, guarded, endian) storage attribute control for thirty-two 128MB regions
- On-chip memory (OCM) interface provides memory access performance identical to cache hits
- Full PowerPC floating-point unit (FPU) support using the auxiliary processor unit (APU) interface (the DesignWare IBM PowerPC 405-S CPU Core does not include an FPU)
- PowerPC timer facilities
  - 64-bit time base
  - Programmable interval timer (PIT)
  - Fixed interval timer (FIT)
  - Watchdog timer
  - Synchronous external time base clock input
- Debug Support
  - Enhanced debug support with logical operators
  - Four instruction address compares (IACs)

  - Two data address compares (DACs)
  - Two invasive data value points
  - New JTAG instructionforwrite to ICU
  - Trace and trace-back support
- Minimized interrupt latency
- Advanced power management support
- Static circuit design with extensive static and dynamic clock and power management support
- Core interfaces that support a wide range of function and performance:
  - Separate 32-bit instruction and 64-bit data interfaces to the processor local bus (PLB) or other bus interface unit (BIU) designs
    - Supports 4:1, 3:2, 3:1, 2:1, and 1:1 core to PLB clock frequency
  - 32-bit device-paced Device Control Register (DCR) interface for system and device control
  - Clock and power management
  - JTAG debug interface
  - Separate 64-bit instruction and 32-bit data interfaces to OCM for single-cycle memory accesses matching cache performance
  - 32-bit APU interface for hardware acceleration
    - Support for aligned subset of loads/stores
    - Support for attachment of a full PowerPC FPU

Target Applications

Primary applications for the PowerPC 405-S CPU are those in which low cost and low power are the critical selection criteria. The core is also well-suited for customers which have little need for extensive peripheral integration in their embedded controllers. A good example is customers that have existing systems and simply need a boost in microprocessor performance.

The PowerPC 405-S CPU is intended for use in:

- Consumer video applications including digital cameras, video games, and set-top boxes
- Portable products such as advanced cellular phones, PDAs, and hand held GPS receivers
- Office automation products such as ink jet printers, X-terminals, and FAX machines
- Networking and storage products such as disk drive controllers, routers, ATM switches, high performance modems and network interface cards

Architecture

This section begins with a brief description of the PowerPC architecture and the standards that define it. Following the architecture description, a discussion of the PowerPC 405-S CPU implementation of the architecture is provided.

PowerPC Architecture

The PowerPC Architecture comprises three levels of standards:

- PowerPC User Instruction Set Architecture, including the base user-level instruction set, user-level registers, programming model, data types, and addressing modes. This is referred to as Book I of the PowerPC Architecture.
- PowerPC Virtual Environment Architecture, describing the memory model, cache model, cache-control instructions, address aliasing, and related issues. While accessible from the user level, these features are intended to be accessed from within library routines provided by the system software. This is referred to as Book II of the PowerPC Architecture.
- PowerPC Operating Environment Architecture, including the memory management model, supervisor-level registers, and the exception model. These features are not accessible from the user level. This is referred to as Book III of the PowerPC Architecture.

Book I and Book II define the instruction set and facilities available to the application programmer. Book III defines features, such as system-level instructions, that are not directly accessible by user applications.

The PowerPC Architecture guarantees application code compatibility across all PowerPC implementations to help maximize the cross-platform portability of applications developed for PowerPC processors. This is accomplished through compliance with the first level of architectural standard, the PowerPC User Instruction Set Architecture, which is common for all PowerPC implementations.

PowerPC 405-S CPU Implementation

The PowerPC 405-S CPU implements the PowerPC User Instruction Set Architecture, user-level registers, programming model, data types, and addressing modes for 32-bit fixed-point operations. The PowerPC 405-S CPU fully complies with specifications for 32-bit implementations of the PowerPC User Instruction Set Architecture. These operations are trapped and can be emulated in software.

Most of the architected features of the PowerPC 405-S CPU are compatible with the specifications for the PowerPC Virtual Environment and Operating Environment Architectures, as specified for processors such as the 6xx family of PowerPC processors. The PowerPC 405-S CPU also provides a number of optimizations and extensions to the lower layers of the PowerPC Architecture. The full architecture of the PowerPC 405-S CPU is defined by the PowerPC Embedded Environment and the PowerPC User Instruction Set Architecture.

The primary extensions of the PowerPC Architecture defined in the Embedded Environment are:

- Simplified memory management mechanism with enhancements for embedded applications
- Enhanced, dual-level interrupt structure
- Architected DCR address space for integrated system control functions
- Additional instructions to support these modified and extended resources

Some of the specific implementation features of the PowerPC 405-S CPU are beyond the scope of the PowerPC Architecture. These features are included to enhance performance, integrate functionality, and reduce system complexity in embedded control applications.

Functional Description

The following sections contain information on the structure and function of the components that comprise the PowerPC 405-S CPU.

PowerPC 405-S CPU Organization

The PowerPC 405-S CPU consists of a five-stage pipelined processor core, virtual memory management unit (MMU), separate instruction and data cache units, JTAG, debug, trace logic, and three timers. The PowerPC UISA and special purpose registers (SPRs) provide a high degree of user control over configuration and operation of the functional units, both interface and core.

Figure 1 illustrates the logical organization of the PowerPC 405-S CPU.

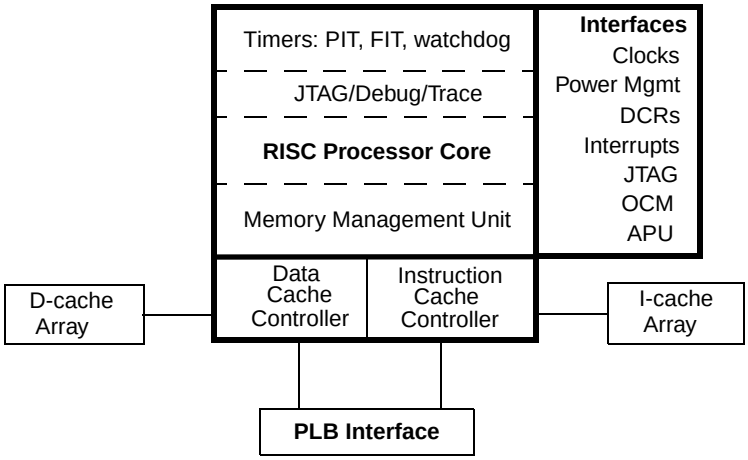

Figure 1: PowerPC 405-S CPU Block

CPU

The RISC processor core comprises a five-stage instruction pipeline.

Cache Controllers

The PowerPC 405-S CPU core provides an instruction cache unit (ICU) and a data cache unit (DCU) that allow concurrent accesses and minimize pipeline stalls. The storage capacity of the cache units, which can range from 0KB–16KB, depends upon the implementation. Both cache units are two-way set associative, use a 32-byte line

size, and provide user supplied RAM BIST for manufacturing. The instruction set provides a rich assortment of cache control instructions, including instructions to read tag information and data arrays.

The cache units, optimized for minimal size and power consumption, maintain high performance. The cache units are PLB-compliant for use in the IBM Core+ASIC program.

Instruction Cache Unit

The instruction cache unit (ICU) provides one or two instruction per cycle to the EXU over a 64-bit bus. A line buffer (built into the output of the array for manufacturing test) enables the ICU to be accessed only once for every four instructions, to reduce power consumption by the array.

The ICU can forward any or all of the words of a line fill to the EXU to minimize pipeline stalls caused by cache misses. The ICU aborts speculative fetches abandoned by the EXU, eliminating unnecessary line fills and enabling the ICU to handle the next EXU fetch. Aborting abandoned requests also eliminates unnecessary external bus activity to increase external bus utilization.

Data Cache Unit

The data cache unit (DCU) transfers one, two, three, or four bytes per cycle, depending on the number of byte enables presented by the CPU. The DCU contains a single-element command and store data queue to reduce pipeline stalls; this queue enables the DCU to independently process load/store and cache control instructions. Dynamic PLB request prioritization reduces pipeline stalls even further. When the DCU is busy with a low-priority request while a subsequent storage operation requested by the CPU is stalled, the DCU automatically increases the priority of the current request to the PLB.

The DCU uses a two-line flush queue to minimize pipeline stalls caused by cache misses. Line flushes are postponed until after a line fill is completed. Registers comprise the first position of the flush queue; the line buffer built into the output of the array for manufacturing test serves as the second position of the flush queue. Pipeline stalls are further reduced by forwarding the requested word to the CPU during the line fill. Single-queued flushes are non-blocking. When a flush operation is pending, the DCU can continue to access the array to determine subsequent load or store hits. Under these conditions, load hits can occur concurrently with store hits to write-back memory without stalling the pipeline. Requests abandoned by the CPU can also be aborted by the cache controller.

The DCU provides two additional features that allow the programmer to tailor its performance for a given application. The DCU can function in write-back or write-through mode, as controlled by the Data Cache Write-through Register (DCWR) or the

translation look-aside buffer (TLB); performance of the cache controller can be tuned for a balance of performance and memory coherency. Write-on-allocate, controlled by the CCR0[SWOA] field, can inhibit line fills caused by a store miss to further reduce potential pipeline stalls and unwanted external bus traffic. A similar statement is valid for CCR0[LWOA] and a load miss.

Memory Management Unit

The PowerPC Architecture provides WIU0GE (write-back/write through, cacheability, user-defined 0, guarded, endian) storage attributes that control memory accesses, using bits in the TLB or, when address translation is disabled, storage attribute control registers.

The storage attribute control bits in the TLB control the storage attributes associated with the current page. Each storage attribute control register contains 32 fields. Each field sets the associated storage attribute for a 128MB memory region.

Timers

The PowerPC 405-S CPU contains a time base and three timers: the Programmable Interval Timer (PIT), the Fixed Interval Timer (FIT), and a watchdog timer. The time base is a 64-bit counter incremented either by an internal signal equal to the CPU clock rate or by a separate external timer clock signal. No interrupts are generated when the time base rolls over.

The PIT is a 32-bit register that is decremented at the same rate as the time base is incremented. The user loads the PIT register with a value to create the desired delay. When the register is decremented to zeros, the timer stops decrementing, a bit is set in the Timer Status Register (TSR), and a PIT interrupt is generated. Optionally, the PIT can be programmed to reload automatically the last value written to the PIT register, after which the PIT begins decrementing again.The Timer Control Register (TCR) contains the interrupt enable for the PIT interrupt.

The FIT generates periodic interrupts based on selected bits in the time base. Users can select one of four intervals for the timer period by setting the appropriate bits in the TCR. When the selected bit in the time base changes from 0 to 1, a bit is set in the TSR and a FIT interrupt is generated. The FIT interrupt enable is contained in the TCR.

The watchdog timer generates a periodic interrupt based on selected bits in the time base. Users can select one of four time periods for the interval and the type of reset generated if the watchdog timer expires twice without an intervening clear from software.

JTAG

The PowerPC 405-S CPU JTAG port is enhanced to support the attachment of a debug tool such as the RISCWatch product from IBM Microelectronics. Through the JTAG test access port, a debug workstation can single-step the processor and interrogate the internal processor state to facilitate software debugging. The enhancements comply with the IEEE 1149.1 specification for vendor-specific extensions, and are therefore compatible with standard JTAG hardware for boundary-scan system testing.

Processor Local Bus

The processor local bus (PLB) interface provides separate 32-bit address and data buses for the instruction and data sides. The PLB is the primary interface between the PowerPC 405-S CPU and high-speed peripherals.

Clock and Power Management

The clock and power management (CPM) interface supports several methods of clock distribution and power management. Three modes of operation that reduce power consumption below the normal operational level are available.

I/O Interfaces

The PowerPC 405-S CPU interfaces support a range of I/O interfaces the simplify the attachment of on-chip and off-chip devices. Chapter 2 provides descriptions of the available interfaces and their signals.

Device Control Register

The device control register (DCR) interface supports the attachment of registers for device control. These registers are accessed using the **mfdcr** and **mtdcr** instructions.

Interrupts

The PowerPC 405-S CPU provides an interface to an interrupt controller that is logically outside the PowerPC 405-S CPU core. This controller combines the asynchronous interrupt inputs and presents them to the core as a single interrupt signal. The sources of asynchronous interrupts are external signals, the JTAG/debug unit, and any implemented peripherals.

Auxiliary Processor Unit

The auxiliary processor unit (APU) interface supports the attachment of auxiliary processor hardware and the implementation of the associated instructions for improved performance in specialized applications.

On-Chip Memory

The on-chip memory (OCM) interface supports the implementation of instruction- and data-side memory that can be accessed at performance levels matching the cache arrays.

Register Set Summary

The registers can be grouped into basic categories based on function and access mode: general purpose registers (GPRs), special purpose registers (SPRs), the machine state register (MSR), the condition register (CR), and, in standard products, device control registers (DCRs).

General Purpose Registers

The PowerPC 405-S CPU contains 32 GPRs. Each GPR contains 32 bits. The contents of the GPRs can be transferred from memory using load instructions and stored to memory using store instructions. GPRs, which are specified as operands in many PowerPC 405-S CPU instructions, can also hold instruction results and the contents of other registers.

Special Purpose Registers

Special Purpose Registers (SPRs), which are part of the PowerPC Architecture, are accessed using the **mtspr** and **mfspr** instructions. SPRs control the use of the debug facilities, timers, interrupts, storage control attributes, and other architected processor resources.

The only SPRs that are not privileged for read and write access are the Count Register (CTR), Link Register (LR), SPR General Purpose Registers 4–7 (SPRG4–SPRG7), and Fixed Point Exception Register (XER). User-mode programs have read-only access to the Time Base Lower (TBL) and Time Base Upper (TBU) time base registers.

Machine State Register

The PowerPC 405-S CPU contains a 32-bit machine state register (MSR). The contents of a GPR can be written to the MSR using the **mtmsr** instruction, and the MSR contents can be read into a GPR using the **mfmsr** instruction. The MSR contains fields that control the operation of the DesignWare IBM PowerPC 405-S CPU Core.

Condition Register

The PowerPC 405-S CPU contains a 32-bit Condition Register (CR). These bits are grouped into eight 4-bit fields, CR[CR0]–CR[CR7]. Instructions are provided to perform logical operations on CR fields and bits within fields and to test CR bits within fields. The CR fields, which are set by compare instructions, can be used to control branches. CR[CR0] can be set implicitly by arithmetic instructions.

Device Control Registers

Device Control Registers (DCRs), which are architecturally outside of the processor core, are accessed using the **mtdcr** and **mfdcr** instructions. DCRs are used to control, configure, and hold status for various functional units that are not part of the processor core,. Although the PowerPC 405-S CPU does not contain DCRs, the **mtdcr** and **mfdcr** instructions are provided.

The **mtdcr** and **mfdcr** instructions are privileged, for all DCRs; therefore, all accesses to DCRs are privileged.

All DCR numbers are reserved, and should be neither read nor written, unless they are part of an IBM Core+ASIC implementation.

Addressing Modes

The PowerPC 405-S CPU supports the following addressing modes to allow efficient retrieval and storage of data in memory:

- Base plus displacement addressing
- Indexed addressing
- Base plus displacement addressing and indexed addressing, with update

In the base plus displacement addressing mode, an effective address (EA) is formed by adding a displacement to a base address contained in a GPR (or to an implied base of 0). The displacement is an immediate field in an instruction.

In the indexed addressing mode, the EA is formed by adding an index contained in a GPR to a base address contained in a GPR (or to an implied base of 0).

The base plus displacement and the indexed addressing modes also have a “with update” mode. In “with update” mode, the effective address calculated for the current operation is saved in the base GPR, and can be used as the base in the next operation. The “with update” mode relieves the processor from repeatedly loading a GPR with an address for each piece of data, regardless of the proximity of the data in memory.

Parity

As IC fabrication processes change and more transistors are instantiated on a chip, the soft error rate (SER) due to alpha particles may increase. Such an SER increase may induce error conditions during normal operation of the PowerPC 405-S CPU. To offset possible SER effects, the PowerPC 405-S CPU utilizes parity error detection in the instruction cache, data cache, and unified translation lookaside buffer. Whenever a parity error is detected, the PowerPC 405-S CPU generates a machine check interrupt so that a software handler can be invoked to recover from the error.

Data Types

PowerPC 405-S CPU operands are bytes, halfwords, or words. Multiple words or strings of bytes can be transferred using the load/store multiple and load/store string instructions. Data are represented in two's complement notation or in unsigned fixed-point format.

The address of a multi-byte operand is always the lowest memory address occupied by that operand. Byte ordering can be selected as big endian (the lowest memory address of an operand contains its most significant byte) or as little endian (the lowest memory address of an operand contains its least significant byte).

The PowerPC 405-S CPU supports little-endian addressing and data types.

Exception Handling

Exceptiosn are events which, if enabled, cause the processor to take an interrupt. PowerPC 405-S CPU exceptions are generated by signals from external interrupt controller, instructions, the internal timer facilities, debug events, and error conditions. Two external interrupt signals are provided in the PowerPC 405-S CPU, one critical and one noncritical. Both external interrupts are maskable. The Machine State Register (MSR) enables critical and noncritical external interrupt signals.

An interrupt is the action in which the processor saves its old context (MSR and instruction pointer) and begins execution at a pre-determined interrupt-handler address, with a modified MSR.

In addition to the MSR, other registers that support interrupt handling and control are:

- Data Exception Address Register (DEAR)
- Exception Syndrome Register (ESR)
- Exception Vector Prefix Register (EVPR)
- Save/Restore Registers (SRR0–SRR3)

All interrupts, except for machine checks, are handled precisely. The PowerPC 405-S CPU processes interrupts as noncritical and critical.When a *noncritical* interrupt is taken, Save/Restore Register 0 (SRR0) is written with the address of the excepting instruction or the next sequential instruction to be processed. Save/Restore Register 1 (SRR1) is written with the contents of the MSR; the MSR is then updated to reflect the new machine context. The new MSR contents take effect beginning with the first instruction of the interrupt handling routine.

Interrupt handling routine instructions are fetched at an address determined by the interrupt type. The address of the interrupt handling routine is formed by concatenating the 16 high-order bits of the EVPR and the interrupt vector offset.

At the end of the interrupt handling routine, execution of an **rfi** instruction forces the contents of SRR0 and SRR1 to be written to the program counter and the MSR, respectively. Execution then begins at the address in the program counter.

Critical interrupts are processed similarly. When a critical interrupt is taken, Save/Restore Register 2 (SRR2) and Save/Restore Register 3 (SRR3) hold the next sequential address to be processed when returning from the interrupt, and the contents of the MSR, respectively. At the end of the critical interrupt handling routine, execution of an **rfci** instruction writes the contents of SRR2 and SRR3 into the program counter and the MSR, respectively.

Debug

The PowerPC 405-S CPU debug facilities include debug modes for the various types of debugging used during hardware and software development. Also included are debug events that allow developers to control the debug process. Debug modes and debug events are controlled using debug registers in the chip. The debug registers are accessed either through software running on the processor, or through the JTAG port. The JTAG port can also be used for board test.

The debug modes, events, controls, and interfaces provide a powerful combination of debug facilities for a complete set of hardware and software development tools such as RISCWatch™ and OS Open™ from IBM.

Development Tool Support

The PowerPC 405-S CPU provides powerful debug support for a wide range of hardware and software development tools.

The OS Open Real-Time Operating System (RTOS) Debugger is an example of an operating system-aware debugger, implemented using software traps.

RISCWatch an example of a development tool that uses the external debug mode, debug events, and the JTAG port to support hardware and software development and debugging.

Logic analyzers from Hewlett-Packard and Tektronix provide PowerPC 405-S CPU disassembler support and support for the RISCTrace™ feature of RISCWatch.

Debug Modes

The PowerPC 405-S CPU supports two debug modes, internal and external; each mode supports a different type of debug tool used in embedded systems development. Internal debug mode supports ROM monitors, and external debug mode supports emulators. Both modes can be enabled simultaneously. The debug modes are controlled by the Debug Control Register (DBCR).

Internal debug mode supports accessing architected processor resources, setting hardware and software break points, and monitoring processor status. In internal debug mode, debug events can generate debug exceptions, which can interrupt normal program flow so that monitor software can collect processor status and alter processor resources.

Internal debug mode relies on exception-handling software, running in the processor, and an external communications path, to debug software problems. This mode is used while the processor is executing instructions and enables debugging of problems in application or operating system code.

Access to debugger software executing in the processor, while in internal debug mode, is through a communications port on the processor board, such as a serial port.

External debug mode, accessed through a JTAG port, supports stopping and starting the processor, accessing architected processor resources, setting hardware and software break points, and monitoring processor status. In external debug mode, debug events can architecturally “freeze” the processor. While the processor is frozen, normal instruction execution stops, and the architected processor resources can be accessed and altered.

External debug mode relies only on internal processor resources to debug system hardware and software problems. This mode can also be used for software development on systems without a control program, or to debug control program code.

2
Input/Output Interfaces

The DesignWare IBM PowerPC 405-S CPU Core provides input/output (I/O) signals that are grouped functionally into the following interfaces:

- Clock and Power Management (CPM)
- CPU Control
- Test
- Reset
- Instruction-Side Processor Local Bus (PLB)
- Data-side Processor Local Bus (PLB)
- Instruction-Side On-Chip Memory (OCM)
- Data-Side OCM
- Device Control Register (DCR)
- External Interrupt Controller (EIC)
- Joint Test Action Group (Jtag) Test Access Port (TAP)
- Debug (DBG)
- Trace
- Auxiliary Processor Unit (APU)

Signal Naming Conventions

Signal names used in this document follow the format defined below:

PREFIX1_prefix2SigName1[_sigName2][_ NEG][(*m:n*)]

where:

- PREFIX1 is an *uppercase* prefix identifying the source of the signal, either by unit name (for example, CPU, DCU) or by the interface type (for example, DCR, TEST).
- _prefix2 is a *lowercase* prefix identifying the destination of the signal, either by unit name (for example, cpu, dcu) or by the interface type (for example, dcr, test).
- SigName1 is a *mixed case* name reflecting the primary function of the signal.
- [sigName2] (optional) is a *mixed case* name reflecting the secondary function of the signal, and is separated from SigName1 by an underscore.
- [_NEG] (optional) denotes that a signal is active low. Unless so denoted, all signals are active high.
- [(*m:n*)] (optional) indicates a bussed signal. This notation is for reading convenience only; the actual core signal names must be unbussed, 1-bit signal names. “Bussed” signals appear on the macro symbol in an expanded format, each including the base name and a one or two digit suffix (depending on whether the highest bit position must be represented by one or two digits) identifying that signal’s bit position on the bus.

Actual PowerPC 405-S CPU hard macro signal names do not follow the format above. These names must be unbussed, and contain only uppercase letters and numbers. Other characters, such as underscores, are illegal. These restrictions make the hard core macro compatible with a variety of vendor development tools for chip design, simulation, synthesis, timing, and so on.

Signal names that follow the naming conventions are used throughout this chapter to ease reading and more clearly identify function.

Table 2 defines the prefixes used in the signal names. The second column in the table identifies whether the prefix refers to a logic unit that resides *on* the DesignWare IBM PowerPC 405-S CPU Core or *off* of the DesignWare IBM PowerPC 405-S CPU Core. For example, the DCU unit, which resides on the PowerPC 405-S CPU, is an on-core unit. The CPM unit, which does not reside on the PowerPC 405-S CPU, is an off-core unit; it is not part of the DesignWare IBM PowerPC 405-S CPU Core.

Table 2: Signal Name Prefix Definitions

Prefix1 (prefix2)	Definition	On/Off Core
APU (apu)	Auxiliary processor unit	Off
CPM (cpm)	Clock and power management	Off
C405 (c405)	PPC405xx CPU	On
DBG (dbg)	Debug unit	On
DCR (dcr)	Device control register	Off
DSOCM (dsocm)	Data-side OCM	Off
EIC (eic)	External interrupt controller	Off
ICU (icu)	Instruction cache unit	On
ISOCM (isocm)	Instruction-side OCM	Off
JTG (jtg)	JTAG	On
TEST(test)	scan design	Off
BIST(bist)	RAM BIST interface	Off
PLB (plb)	Processor local bus	On
RST (rst)	Reset	On
TIE (tie)	Tie (static connection to GND or V_{DD})	Off
TRC (trc)	Trace	On
XXX (xxx)	Unspecified ASIC unit	Off

Signal List

Table 3 lists the PowerPC 405-S CPU signals in alphabetical order. Table 3 also lists the hard macro name, I/O type, termination if unused, and timing.

Table 3: Signal Names in Alphabetical Order

Signal	Hard Macro Signal	I/O	If Unused
APU_c405DcdApuOp	APUC405DCDAPUOP	I	0
APU_c405DcdCREn	APUC405DCDCREN	I	0
APU_c405DcdForceAlgn	APUC405DCDFORCEALGN	I	0
APU_c405DcdForceBESteering	APUC405DCDFORCEBESTEERING	I	0
APU_c405DcdFpuOp	APUC405DCDFPUOP	I	0
APU_c405DcdGprWrite	APUC405DCDGPRWRITE	I	0
APU_c405DcdLdStByte	APUC405DCDLDSTBYTE	I	0
APU_c405DcdLdStDw	APUC405DCDLDSTDW	I	0
APU_c405DcdLdStHw	APUC405DCDLDSTHW	I	0
APU_c405DcdLdStQw	APUC405DCDLDSTQW	I	0
APU_c405DcdLdStWd	APUC405DCDLDSTWD	I	0
APU_c405DcdLoad	APUC405DCDLOAD	I	0
APU_c405DcdPrivOp	APUC405DCDPRIVOP	I	0
APU_c405DcdRaEn	APUC405DCDRAEN	I	0
APU_c405DcdRbEn	APUC405DCDRBEN	I	0
APU_c405DcdStore	APUC405DCDSTORE	I	0
APU_c405DcdTrapBE	APUC405DCDTRAPBE	I	0
APU_c405DcdTrapLE	APUC405DCDTRAPLE	I	0
APU_c405DcdUpdate	APUC405DCDUPDATE	I	0
APU_c405DcdValidOp	APUC405DCDVALIDOP	I	0
APU_c405DcdXerCAEn	APUC405DCDXERCAEN	I	0
APU_c405DcdXerOVEn	APUC405DCDXEROVEN	I	0
APU_c405Exception	APUC405EXCEPTION	I	0
APU_c405ExeBlockingMCO	APUC405EXEBLOCKINGMCO	I	0
APU_c405ExeBusy	APUC405EXEBUSY	I	0
APU_c405ExeCR(0:3)	APUC405EXECR*n*	I	0
APU_c405ExeCRField(0:2)	APUC405EXECRFIELD*n*	I	0
APU_c405ExeLdDepend	APUC405EXELDDEPEND	I	0

Table 3: Signal Names in Alphabetical Order (Continued)

Signal	Hard Macro Signal	I/O	If Unused
APU_c405ExeNonBlockingMCO	APUC405EXENONBLOCKINGMCO	I	0
APU_c405ExeResult(0:31)	APUC405EXERESULT*nn*	I	0
APU_c405ExeXerCA	APUC405EXEXERCA	I	0
APU_c405ExeXerOV	APUC405EXEXEROV	I	0
APU_c405FpuException	APUC405FPUEXCEPTION	I	0
APU_c405LwbLdDepend	APUC405LWBLDDEPEND	I	0
APU_c405SleepReq	APUC405SLEEPREQ	I	1
APU_c405WbLdDepend	APUC405WBLDDEPEND	I	0
BIST_c405dcuBistDebugEn(0:3)	BISTC405DCUBISTDEBUGENnn	I	0
BIST_c405dcuBistDebugSi(0:3)	BISTC405DCUBISTDEBUGSInn	I	0
BIST_c405dcuBistMbRun	BISTC405DCUBISTMBRUN	I	0
BIST_c405dcuBistModeRegIn(0:18)	BISTC405DCUBISTMODEREGINnn	I	0
BIST_c405dcuBistModeRegSi	BISTC405DCUBISTMODEREGSI	I	0
BIST_c405dcuBistParallelDr	BISTC405DCUBISTPARALLELDR	I	0
BIST_c405dcuBistShiftDr	BISTC405DCUBISTSHIFTDR	I	0
BIST_c405icuBistDebugEn(0:3)	BISTC405ICUBISTDEBUGENnn	I	0
BIST_c405icuBistDebugSi(0:3)	BISTC405ICUBISTDEBUGSInn	I	0
BIST_c405icuBistMbRun	BISTC405ICUBISTMBRUN	I	0
BIST_c405icuBistModeRegIn(0:18)	BISTC405ICUBISTMODEREGINnn	I	0
BIST_c405icuBistModeRegSi	C405BISTICUBISTMODEREGSI	I	0
BIST_c405icuBistParallelDr	BISTC405ICUBISTPARALLELDR	I	0
BIST_c405icuBistShiftDr	BISTC405ICUBISTSHIFTDR	I	0
C405_apuDcdFull	C405APUDCDFULL	O	No connect
C405_apuDcdHold	C405APUDCDHOLD	O	No connect
C405_apuDcdInstruction(0:31)	C405APUDCDINSTRUCTION*nn*	O	No connect
C405_apuExeFlush	C405APUEXEFLUSH	O	No connect
C405_apuExeHold	C405APUEXEHOLD	O	No connect
C405_apuExeLoadDBus(0:31)	C405APUEXELOADDBUS*nn*	O	No connect
C405_apuExeLoadDValid	C405APUEXELOADDVALID	O	No connect
C405_apuExeRaData(0:31)	C405APUEXERADATA*nn*	O	No connect
C405_apuExeRbData(0:31)	C405APUEXERBDATA*nn*	O	No connect

Table 3: Signal Names in Alphabetical Order (Continued)

Signal	Hard Macro Signal	I/O	If Unused
C405_apuExeWdCnt(0:1)	C405APUEXEWDCNT*n*	O	No connect
C405_apuMsrFE0	C405APUMSRFE0	O	No connect
C405_apuMsrFE1	C405APUMSRFE1	O	No connect
C405_apuWbByteEn(0:3)	C405APUWBBYTEEN*n*	O	No connect
C405_apuWbEndian	C405APUWBENDIAN	O	No connect
C405_apuWbFlush	C405APUWBFLUSH	O	No connect
C405_apuWbHold	C405APUWBHOLD	O	No connect
C405_apuXerCA	C405APUXERCA	O	No connect
C405_bistdcuBistDebugSo(0:3)	C405BISTDCUBISTDEBUGSOnn	O	No connect
C405_bistdcuBistModeRegOut(0:18)	C405BISTDCUBISTMODEREGOUTnn	O	No connect
C405_bistdcuBistModeRegSo	C405BISTDCUBISTMODEREGSO	O	No connect
C405_bisticuBistDebugSo(0:3)	C405BISTICUBISTDEBUGSOnn	O	No connect
C405_bisticuBistModeRegOut(0:18)	C405BISTICUBISTMODEREGOUTnn	O	No connect
C405_bisticuBistModeRegSo	C405BISTICUBISTMODEREGSO	O	No connect
C405_cpmCoreSleepReq	C405CPMCORESLEEPREQ	O	No connect
C405_cpmMsrCE	C405CPMMSRCE	O	No connect
C405_cpmMsrEE	C405CPMMSREE	O	No connect
C405_cpmTimerIRQ	C405CPMTIMERIRQ	O	No connect
C405_cpmTimerResetReq	C405CPMTIMERRESETREQ	O	No connect
C405_dbgLoadDataOnApuDBus	C405DBGLOADDATAONAPUDBUS	O	No connect
C405_dbgMsrWE	C405DBGMSRWE	O	No connect
C405_dbgStopAck	C405DBGSTOPACK	O	No connect
C405_dbgWbComplete	C405DBGWBCOMPLETE	O	No connect
C405_dbgWbFull	C405DBGWBFULL	O	No connect
C405_dbgWbIar(0:29)	C405DBGWBIAR*nn*	O	No connect
C405_dcrABus(0:9)	C405DCRABUS*n*	O	No connect
C405_dcrDBusOut(0:31)	C405DCRDBUSOUT*nn*	O	No connect, or wrap to DBus In
C405_dcrRead	C405DCRREAD	O	No connect
C405_dcrWrite	C405DCRWRITE	O	No connect
C405_dsocmAbortOp	C405DSOCMABORTOP	O	No connect

Table 3: Signal Names in Alphabetical Order (Continued)

Signal	Hard Macro Signal	I/O	If Unused
C405_dsocmAbortReq	C405DSOCMABORTREQ	O	No connect
C405_dsocmABus(0:29)	C405DSOCMABUS*nn*	O	No connect
C405_dsocmByteEn(0:3)	C405DSOCMBYTEEN*n*	O	No connect
C405_dsocmCacheable	C405DSOCMCACHEABLE	O	No connect
C405_dsocmGuarded	C405DSOCMGUARDED	O	No connect
C405_dsocmLoadReq	C405DSOCMLOADREQ	O	No connect
C405_dsocmStoreReq	C405DSOCMSTOREREQ	O	No connect
C405_dsocmStringMultiple	C405DSOCMSTRINGMULTIPLE	O	No connect
C405_dsocmU0Attr	C405DSOCMU0ATTR	O	No connect
C405_dsocmWait	C405DSOCMWAIT	O	No connect
C405_dsocmWrDBus(0:31)	C405DSOCMWRDBUS*nn*	O	No connect
C405_dsocmXlateValid	C405DSOCMXLATEVALID	O	No connect
C405_isocmAbort	C405ISOCMABORT	O	No connect
C405_isocmABus(0:29)	C405ISOCMABUS*nn*	O	No connect
C405_isocmCacheable	C405ISOCMCACHEABLE	O	No connect
C405_isocmContextSync	C405ISOCMCONTEXTSYNC	O	No connect
C405_isocmIcuReady	C405ISOCMICUREADY	O	No connect
C405_isocmReqPending	C405ISOCMREQPENDING	O	No connect
C405_isocmU0Attr	C405ISOCMU0ATTR	O	No connect
C405_isocmXlateValid	C405ISOCMXLATEVALID	O	No connect
C405_jtgCaptureDR	C405JTGCAPTUREDR	O	No connect
C405_jtgExtest	C405JTGEXTEST	O	No connect
C405_jtgPgmOut	C405JTGPGMOUT	O	No connec
C405_jtgShiftDR	C405JTGSHIFTDR	O	No connect
C405_jtgTDO	C405JTGTDO	O	No connect
C405_jtgTDOEn	C405JTGTDOEN	O	No connect
C405_jtgUpdateDR	C405JTGUPDATEDR	O	No connect
C405_plbDcuAbort	C405PLBDCUABORT	O	No connect
C405_plbDcuABus(0:31)	C405PLBDCUABUS*nn*	O	No connect
C405_plbDcuBE(0:7)	C405PLBDCUBE*n*	O	No connect
C405_plbDcuCacheable	C405PLBDCUCACHEABLE	O	No connect
C405_plbDcuGuarded	C405PLBDCUGUARDED	O	No connect

Table 3: Signal Names in Alphabetical Order (Continued)

Signal	Hard Macro Signal	I/O	If Unused
C405_plbDcuPriority(0:1)	C405PLBDCUPRIORITYn	O	No connect
C405_plbDcuRequest	C405PLBDCUREQUEST	O	No connect
C405_plbDcuRNW	C405PLBDCURNW	O	No connect
C405_plbDcuSize2	C405PLBDCUSIZE2	O	No connect
C405_plbDcuU0Attr	C405PLBDCUU0ATTR	O	No connect
C405_plbDcuWrDBus(0:63)	C405PLBDCUWRDBUS*nn*	O	No connect
C405_plbDcuWriteThru	C405PLBDCUWRITETHRU	O	No connect
C405_plbIcuAbort	C405PLBICUABORT	O	No connect
C405_plbIcuABus(0:29)	C405PLBICUABUS*nn*	O	No connect
C405_plbIcuCacheable	C405PLBICUCACHEABLE	O	No connect
C405_plbIcuPriority(0:1)	C405PLBICUPRIORITYn	O	No connect
C405_plbIcuRequest	C405PLBICUREQUEST	O	No connect
C405_plbIcuSize(2:3)	C405PLBICUSIZE(*n*	O	No connect
C405_plbIcuU0Attr	C405PLBICUU0ATTR	O	No connect
C405_rstChipResetReq	C405RSTCHIPRESETREQ	O	Required
C405_rstCoreResetReq	C405RSTCORERESETREQ	O	Required
C405_rstSystemResetReq	C405RSTSYSTEMRESETREQ	O	Required
C405_testDiagAbistDone	C405TESTDIAGABISTDONE	O	No connect
C405_testScanOut(0:7	C405TESTSCANOUT*n*	O	Required
C405_trcCycle	C405TRCCYCLE	O	No connect
C405_trcEvenExecutionStatus(0:1)	C405TRCEVENEXECUTION STATUS*n*	O	No connect
C405_trcOddExecutionStatus(0:1)	C405TRCODDEXECUTIONSTATUS*n*	O	No connect
C405_trcTraceStatus	C405TRCTRACESTATUS	O	No connect
C405_trcTriggerEventOut	C405TRCTRIGGEREVENTOUT	O	Wrap to Trigger EventIn
C405_trcTriggerEventType(0:10)	C405TRCTRIGGEREVENTTYPEnn	O	No connect
C405_xxxMachineCheck	C405XXXMACHINECHECK	O	No connect
CPM_c405Clock	CPMC405CLOCK	I	Required
CPM_c405CoreClkInactive	CPMC405CORECLKINACTIVE	I	0
CPM_c405CpuClkEn_CClk	CPMC405CPUCLKENCCLK		
CPM_c405JtagClkEn_CClk	CPMC405JTAGCLKENCCLK		

Table 3: Signal Names in Alphabetical Order (Continued)

Signal	Hard Macro Signal	I/O	If Unused
CPM_c405PlbSampleCycle	CPMC405PLBSAMPLECYCLE	I	1
CPM_c405TimerClkEn_CClk	CPMC405TIMERCLKENCCLK	I	1
CPM_c405TimerTick	CPMC405TIMERTICK	I	1
DBG_c405DebugHalt	DBGC405DEBUGHALT	I	1
DBG_c405ExtBusHoldAck	DBGC405EXTBUSHOLDACK	I	0
DBG_c405UncondDebugEvent	DBGC405UNCONDDEBUGEVENT	I	0
DCR_c405Ack	DCRC405ACK	I	0
DCR_c405DBusIn(0:31)	DCRC405DBUSIN	I	0x00000000 or wrapped DBus Out
DSOCM_c405Complete	DSOCMC405COMPLETE	I	0
DSOCM_c405DisOperandFwd	DSOCMC405DISOPERANDFWD	I	0
DSOCM_c405Hold	DSOCMC405HOLD	I	0
DSOCM_c405RdDBus(0:31))	DSOCMC405RDDBUS*nn*	I	0x00000000
EIC_c405CritInputIRQ	EICC405CRITINPUTIRQ	I	0
EIC_c405ExtInputIRQ	EICC405EXTINPUTIRQ	I	0
ISOCM_c405Hold	ISOCMC405HOLD	I	0
ISOCM_c405RdDBus(0:63)	ISOCMC405RDDBUS*nn*	I	Tie to GND
ISOCM_c405RdDValid(0:1)	ISOCMC405RDDVALID*n*	I	0
JTG_c405BndScanTDO	JTGC405BNDSCANTDO	I	0
JTG_c405TCK	JTGC405TCK	I	See IEEE 1149.1
JTG_c405TDI	JTGC405TDI	I	1
JTG_c405TMS	JTGC405TMS	I	1
JTG_c405TRST_NEG	JTGC405TRSTNEG	I	Required
PLB_c405DcuAddrAck	PLBC405DCUADDRACK	I	0
PLB_c405DcuBusy	PLBC405DCUBUSY	I	0
PLB_c405DcuErr	PLBC405DCUERR	I	0
PLB_c405DcuRdDAck	PLBC405DCURDDACK	I	0
PLB_c405DcuRdDBus(0:63)	PLBC405DCURDDBUS*nn*	I	0x0000000000000000
PLB_c405DcuRdWdAddr(1:3)	PLBC405DCURDWDADDR*n*	I	0b000
PLB_c405DcuSSize1	PLBC405DCUSSIZE1	I	0
PLB_c405DcuWrDAck	PLBC405DCUWRDACK	I	0

Table 3: Signal Names in Alphabetical Order (Continued)

Signal	Hard Macro Signal	I/O	If Unused
PLB_c405IcuAddrAck	PLBC405ICUADDRACK	I	0
PLB_c405IcuBusy	PLBC405ICUBUSY	I	0
PLB_c405IcuErr	PLBC405ICUERR	I	0
PLB_c405IcuRdDAck	PLBC405ICURDDACK	I	0
PLB_c405IcuRdDBus(0:63)	PLBC405ICURDDBUS*nn*	I	0x00000000
PLB_c405IcuRdWdAddr(1:3)	PLBC405ICURDWDADDR*n*	I	0b000
PLB_c405IcuSSize1	PLBC405ICUSSIZE1	I	0
RST_c405ResetChip	RSTC405RESETCHIP	I	Required
RST_c405ResetCore	RSTC405RESETCORE	I	Required
RST_c405ResetSystem	RSTC405RESETSYSTEM	I	Required
TEST_c405BistCClk	TESTC405BISTCCLK	I	1
TEST_c405BistCE0StClk	TESTC405BISTCE0STCLK	i	0
TEST_c405CntlPoint	TESTC405CNTLPOINT	I	0
TEST_c405ScanEnable(0:7)	TESTC405SCANENABLE	I	0
TEST_c405ScanIn(0:7)	TESTC405SCAN0-7	I	0b00000000 00
TEST_c405TestM1	TESTC405TESTM1	I	0
TEST_c405TestM3	TESTC405TESTM3	I	0
TEST_c405TestMode	TESTC405TESTMODE	I	0
TIE_c405ApuDivEn	TIEC405APUDIVEN	I	0
TIE_c405ApuPresent	TIEC405APUPRESENT	I	0
TIE_c405DeterministicMult	TIEC405DETERMINISTICMULT	I	Required
TIE_c405DisOperandFwd	TIEC405DISOPERANDFWD	I	Required
TIE_c405MmuEn	TIEC405MMUEN	I	Required
TIE_c405PVR(0:31)	TIEC405PVR*nn*	I	Required
TRC_c405TraceDisable	TRCC405TRACEDISABLE	I	0
TRC_c405TriggerEventIn	TRCC405TRIGGEREVENTIN	I	Wrap to Trigger Event Out

Clock and Power Management (CPM)

The Clock And Power Management (CPM) interface provides clock control capabilities for power-sensitive applications.

CPM Interface I/O Symbol

illustrates the inputs and outputs of the CPM interface.

Figure 2: CPM Interface I/O Symbol

CPM Interface I/O Signal Summary

Table 4 summarizes the CPM interface signals:

Table 4: CPM Interface I/O Signals

Signal	I/O Type	If Unused	Function
CPM_c405Clock	I	Required	Main core clock input (for all non-JTAG logic).
CPM_c405CoreClkInactive	I	0	Indicates that CPM logic disabled the clocks to the core.
CPM_c405CpuClkEn_CClk	I	Required	Enables the core clock
TEST_c405BistCE1Mode	I	Required	Enables the internal generation of clocks for most of the core logic.
CPM_c405JTAGClkEn_CClk	I	Required	Enables Jtag clock internally
CPM_c405PLBSampleCycle	I	1	Used to synchronize the interfaces between the ICU/DCU and a slower PLB
CPM_c405TimerTick	I	1	Changes the PPC405 timers one increment or decrement every time it is active with the CPM_c405Clock.
CPM_c405TimerClkEn_CClk	I	Required	Enables the internal generation of clocks for the timer logic.
C405_cpmMsrEE	O	No connect	Indicates that MSR[EE] = 1.
C405_cpmMsrCE	O	No connect	Indicates that MSR[CE] = 1.
C405_cpmTimerIRQ	O	No connect	Indicates that a timer interrupt request was generated; can wake the CPU from a sleep mode.
C405_cpmTimerResetReq	O	No connect	Indicates that the watchdog timer generated a reset request; can wake the CPU from a sleep mode.
C405_cpmCoreSleepReq	O	No connect	Indicates that the core is requesting to be put into a sleep mode.

CPU Control

The CPU Control interface is used primarily to provide CPU setup information to the PowerPC 405 CPU. It is also used to report the detection of a machine check condition within the PowerPC 405 CPU.

CPU Control Interface I/O Symbol

illustrates the inputs and outputs of the CPU Control interface.

Figure 3: CPU Interface Symbol

CPU Control Interface I/O Signal Summary

Table 5 summarizes the CPU Control interface signals.

Table 5: CPU Control Interface I/O Signals

Signal	I/O Type	If Unused	Function
TIE_c405PVR(0:31)	I	Required	Unique chip ID; assigned by IBM.
TIE_c405MmuEn	I	Required	Enables the MMU functionality of the core
TIE_c405DeterministicMult	I	Required	Determines whether all CPU multiply operations will take a deterministic number of cycles, or have an early-out capability.
TIE_c405DisOperandFwd	I	Required	Allows the CPU to run at higher frequencies by adding a latch between the Data cache and the CPU for load instructions.
TIE_c405ClockEnable	I	1	Clock output enable should always be connected to a logic 1
TIE_c405DutyEnable	I	1	Clock output enable should always be connected to a logic 1
C405_xxxMachineCheck	O	No connect	Indicates that a machine check error has been detected by the core.

Test

The Test interface is comprised of the signals necessary to select the mode of operation of the PowerPC 405-S CPU and control manufacturing test on the PowerPC 405-S CPU and the customer's ASIC logic.

Test Interface I/O Symbol

The Test interface symbol in illustrates the connection between the PowerPC 405-S CPU and the Test interface.

Figure 4: Test Interface I/O Symbol

Test Interface I/O Signal Summary

Table 6 summarizes the Test interface signals.

Scan flush initialization does not reset the PowerPC 405-S CPU. Rather, it initializes the scan rings to some determinate state before application of a reset.

Table 6: Test Interface I/O Signals

Signal	I/O Type	Functional Mode	Timing	Function
TEST_c405BistCClk	I	1	N/A	Unused
TEST_c405TestM1	I	0	N/A	Unused
c405TestM3	I	0	N/A	Set to 1 for UTLB BIST.
TEST_c405ScanIn[0:7]	I	0x00	N/A	The core's scan in ports.
TEST_c405CntlPoint	I	0	N/A	Tie to logic 0
C405_DiagABistDone	O		N/A	Used for system self-test (optional output).
C405_testScanOut[0:7]	O		N/A	The core's corresponding scan out ports.
TEST_c405ScanEnable	I	0	N/A	Logic high allows shifting in scan chaines.
TEST_c405TestMode	I	0	N/A	Logic high for scan test

Reset

Resetting the PowerPC 405-S CPU is accomplished by asserting reset inputs into the core. The core is not reset using a scan-flush mechanism.

The PowerPC 405-S CPU supports three types of reset, defined in Table 7, to provide granularity for resetting system resources.

Table 7: Reset Levels Supported by the PowerPC 405-S CPU

Reset Type	Definition
Core Reset	Resets the core, including the data and instruction caches.
Chip Reset	Resets the ASIC, including the core and on-chip peripherals and logic.
System Reset	Intended to reset the entire system including the core, the ASIC, and all logic external to the ASIC.

Sample On-Chip Reset Connection

PowerPC 405-S CPU Reset Interface I/O Symbol

Signals shown with dashed lines represent signals that are involved in reset but are not fully described in this section.

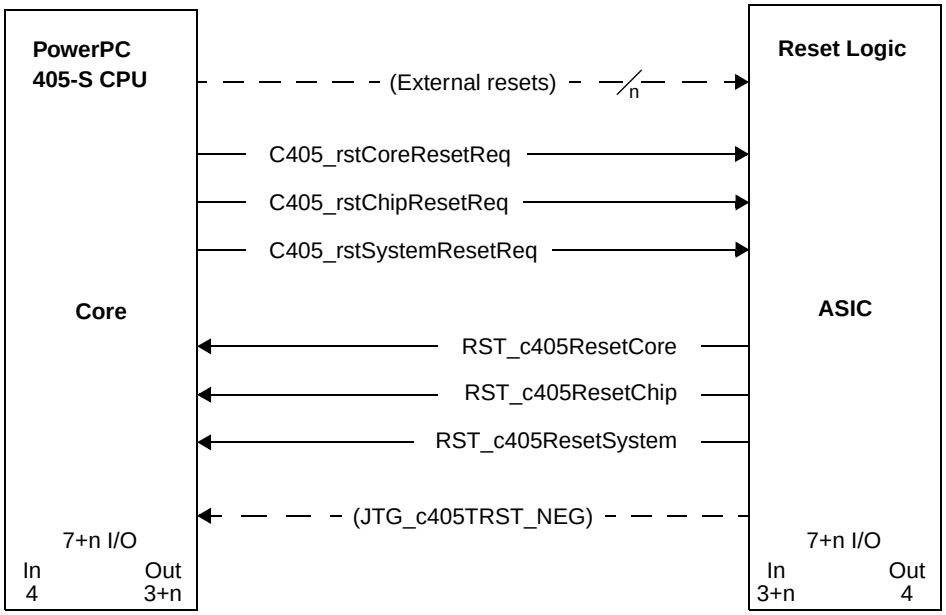

Figure 5: Reset Interface I/O Symbol

Reset Interface I/O Signal Table

Table 8 summarizes the Reset interface signals.

Table 8: Reset Interface I/O Signal Table

Signal	I/O Type	If Unused	Function
C405_rstCoreResetReq	O	Required	Indicates that an event has occurred to generate a core reset *request*
C405_rstChipResetReq	O	Required	Indicates that an event has occurred to generate a chip reset *request*
C405_rstSystemResetReq	O	Required	Indicates that an event has occurred to generate a system reset *request*
RST_c405ResetCore	I	Required	Core reset input that resets internal core logic. A minimum of four cycles is required.
RST_c405ResetChip	I	Required	Chip reset input that alerts the core to the fact that a chip level reset has been driven.
RST_c405ResetSystem	I	Required	System reset input that 1) alerts the core that a system reset has been driven and 2) resets internal core debug logic in the JTAG unit.
JTG_c405TRST_NEG	I	Required	JTAG Test Reset ($\overline{\text{TRST}}$) *must* be asserted at POR; can be used as $\overline{\text{TRST}}$ (negative active) thereafter.

Instruction-Side Processor Local Bus (PLB)

The Instruction-side Processor Local Bus (ISPLB) interface enables the instruction cache unit (ICU) to read instructions from any memory device connected to the processor local bus (PLB). The ICU cannot write to memory across the PLB interface. This interface has a dedicated 30 bit address bus output, C405_plbIcuABus(0:29), and a dedicated 64-bit read data bus input, PLB_c405IcuRdDBus(0:63) that allows the ICU to attach to a 64-bit or 32-bit slave. This interface was designed as a 64-bit PLB interface, however, it can be used with 32-bit PLB components as well. In addition, the PLB system design enables this 64-bit interface to be used in even larger PLB systems of the future. The interface is capable of one data transfer (64 or 32 bits) every cycle.

Note

The instruction-side read data bus may be combined with the data-side read data bus at the chip level to create a shared read data bus. This can be done if a single PLB arbiter services both of these PLB masters and the PLB arbiter design only returns data across the RdDBus to one PLB master at a time. This is true of IBM's PLB arbiter soft macro, which is a component of the IBM Blue Logic family of cores for system-on-a-chip designs.

Instruction-Side PLB Interface I/O Symbol

Figure 6: Instruction-Side PLB Interface I/O Symbol

Instruction-Side PLB I/O Signal Table

Table 9 summarizes the Instruction-Side PLB interface signals.

Table 9: Instruction-Side PLB Interface Signal Summary

Signal	I/O Type	If Unused	Timing	Function
C405_plbIcuRequest	O	No connect	Begin	Requests an instruction fetch transfer between the ICU and PLB
C405_plbIcuABus(0:29)	O	No connect	Early	Indicates the address of the memory which is being accessed for a read transfer with the ICU
C405_plbIcuCacheable	O	No connect	Early	Indicates the value of the cacheability storage attribute for the target address
C405_plbIcuU0Attr	O	No connect	Early	Indicates the value of the user-defined storage attribute for the target address
C405_plbIcuSize(2:3)	O	No connect	Early	Indicates the size of the transfer being requested as either a four or eight word *line* transfer
C405_plbIcuAbort	O	No connect	Early	Indicates that an unacknowledged ICU transfer request is being aborted
C405_plbIcuPriority(0:1)	O	No connect	Early	Indicates the priority of ICU to PLB
CPM_c405PLBSampleCycle	I	1	End	Used to synchronize the interface between the ICU and a slower PLB
PLB_c405IcuAddrAck	I	0	End	Indicates that the current transfer being requested by the ICU is being acknowledged/accepted by the PLB

Table 9: Instruction-Side PLB Interface Signal Summary (Continued)

Signal	I/O Type	If Unused	Timing	Function
PLB_c405IcuSSize1	I	0	Late	Indicates the size of the PLB Slave that is accepting the request as a 32-bit (0) or a 64-bit (1) slave
PLB_c405IcuRdDAck	I	0	Middle	Indicates that a requested instruction word or doubleword is being presented on the ICU read data bus
PLB_c405IcuRdDBus(0:63)	I	0x00000000	End	ICU read data bus - The 64-bit memory-aligned data bus used to transfer instructions from the PLB to the ICU
PLB_c405IcuRdWdAddr(1:3)	I	0b000	Middle	Indicates which word or doubleword of a four or eight word line transfer is being transferred across the ICU read data bus
PLB_c405IcuBusy	I	0	Late	Indicates that the PLB is busy performing an operation that was initiated by the ICU
PLB_c405IcuErr	I	0	Late	Indicates that an error was detected by the PLB Slave during the transfer of the instruction word(s) on the ICU read data bus (This signal MUST be presented when PLB_c405IcuRdDAck is active)

Data-side Processor Local Bus (PLB)

The Data-Side Processor Local Bus (DSPLB) interface enables the data cache unit (DCU) to perform read and write transfers with memory devices connected to the processor local bus (PLB). The DCU interface to the PLB has a dedicated 32-bit address bus output, C405_plbDcuABus(0:31), a dedicated 64-bit write data bus output, C405_plbDcuWrDBus(0:63), and a dedicated 64-bit read data bus input, PLB_c405DcuRdDBus(0:63). This interface was designed as a 64-bit PLB interface, however, it is able to be used with 32-bit PLB components as well. In addition, the PLB system design enables this 64-bit interface to be used in even larger PLB systems of the future. The interface is capable of one data transfer (64 or 32 bits) every cycle.

Note

The data-side read data bus may be combined with the instruction-side read data bus at the chip level to create a shared read data bus. This can be done if a single PLB arbiter services both of these PLB masters and the PLB arbiter design only returns data across the RdDBus to one PLB master at a time. This is true of IBM's PLB arbiter soft macro, which is a component of the IBM Blue Logic family of cores for system-on-a-chip designs.

Data-Side PLB Interface I/O Symbol

Figure 7: Data Side PLB Interface I/O Symbol

Data-Side PLB Interface I/O Signal Table

Table 10 summarizes the Data-Side PLB interface signals.

Table 10: Data-Side PLB Interface I/O Signal Summary

Signal	I/O Type	If Unused	Timing	Function
C405_plbDcuRequest	O	No connect	Begin	Requests a data transfer between the DCU and PLB
C405_plbDcuRNW	O	No connect	Early	Indicates whether the requested transfer is a read or a write transfer
C405_plbDcuABus(0:31)	O	No connect	Early	Indicates the address of the memory to be read from or written to
C405_plbDcuSize2	O	No connect	Early	Indicates the size of the data being requested as a one to four byte *word* transfer (0) or an eight word *line* transfer (1)
C405_plbDcuCacheable	O	No connect	Early	Indicates the value of the cacheability storage attribute for the target address
C405_plbDcuWriteThru	O	No connect	Early	Indicates the value of the write-through storage attribute for the target address
C405_plbDcuU0Attr	O	No connect	Early	Indicates the value of the user-defined storage attribute for the target address

Table 10: Data-Side PLB Interface I/O Signal Summary (Continued)

Signal	I/O Type	If Unused	Timing	Function
C405_plbDcuGuarded	O	No connect	Early	Indicates the value of the guarded storage attribute for the target address
C405_plbDcuBE(0:7)	O	No connect	Early	Indicates which bytes in a word are to be transferred during *word* transfers. These bits are ignored for *line* transfers
C405_plbDcuPriority(0:1)	O	No connect	Early	Indicates the current DCU request priority. The priority will increase when the CPU pipeline is stalled, waiting for the data transfer to complete
C405_plbDcuAbort	O	No connect	Early	Indicates that an unacknowledged DCU transfer request should be aborted because of a core reset
C405_plbDcuWrDBus(0:63)	O	No connect	Early	DCU write data bus. The 64-bit memory-aligned data bus used to transfer write data from the DCU to the PLB
CPM_c405PLBSampleCycle	I	1	End	Synchronizes the interface between the DCU and a slower PLB
PLB_c405DcuAddrAck	I	0	Late	Indicates that the current transfer being requested by the DCU is being acknowledged/accepted by the PLB
PLB_c405DcuSSize1	I	0	Late	Indicates the size of the PLB slave that is being accessed, as a 32-bit (0) or 64-bit (1) slave
PLB_c405DcuRdDAck	I	0	Middle	Indicates that a data word or doubleword associated with a read request is being presented on the DCU read data bus
PLB_c405DcuRdDBus(0:63)	I	0x0000000000000000	End	DCU read data bus. The 64-bit memory-aligned data bus used to transfer read data from the PLB to the DCU
PLB_c405DcuRdWdAddr(1:3)	I	0b000	Late-	During *line* transfers, indicates which word or doubleword of an eight word cache line is being transferred across the DCU read data bus

Table 10: Data-Side PLB Interface I/O Signal Summary (Continued)

Signal	I/O Type	If Unused	Timing	Function
PLB_c405DcuWrDAck	I	0	Late	Indicates that a data word or doubleword associated with a write request is being accepted by the PLB from the DCU write data bus
PLB_c405DcuBusy	I	0	Late-	Indicates that the PLB is busy performing an operation that was initiated by the DCU
PLB_c405DcuErr	I	0	End	Indicates that an error was detected by the PLB slave during the transfer of data associated with a DCU request (Should only be presented while the Busy signal is asserted)

Instruction-Side On-Chip Memory (OCM)

The Instruction-Side On-chip Memory (ISOCM) interface provides the system-on-a-chip designer with the capability to reserve an area of code storage whose access is characterized by low-latency instruction fetches that exhibit cycle performance identical to cache hits.

Instruction-Side OCM Interface I/O Symbol

Figure 8 illustrates the instruction-side interface between the PowerPC 405-S CPU and the instruction-side OCM.

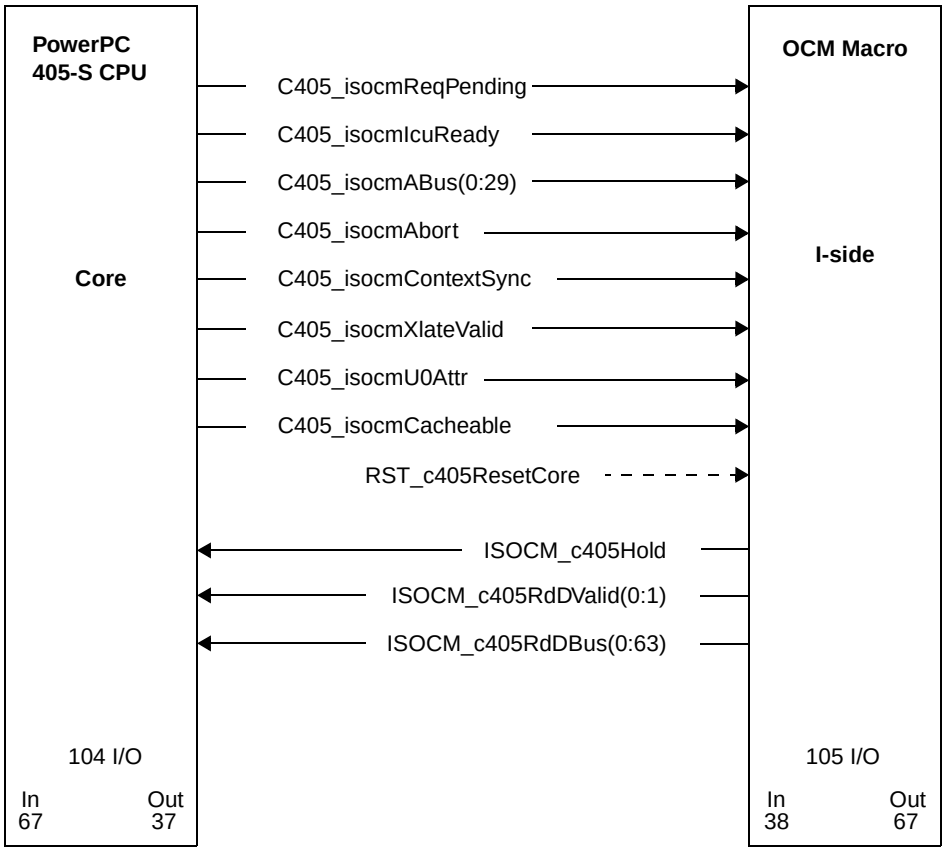

Figure 8: Instruction-Side OCM Interface I/O Symbol

Instruction-Side OCM Interface Signal Summary

Table 11 lists and provides summary information about the instruction-side OCM interface.

Table 11: Instruction-Side OCM Interface Signals

Signal	I/O Type	If Unused	Timing	Function
C405_isocmReqPending	O	No connect	Early	CPU's instruction fetcher has a request pending (not necessarily a request cycle).
C405_isocmIcuReady	O	No connect	Late	ICU is able to accept request.
C405_isocmABus(0:29)	O	No connect	Late	Address of the instruction(s) being requested by the CPU.
C405_isocmXlateValid	O	No connect	Late	Authorizes access of requested address and validates storage attributes.
C405_isocmCacheable	O	No connect	Late	Value of the cacheability storage attribute for the target address.
C405_isocmU0Attr	O	No connect	Late	Value of the user-defined storage attribute for the target address.
C405_isocmAbort	O	No connect	Late	An outstanding instruction-side OCM transfer request is being aborted.
C405_isocmContextSync	O	No connect	Early+	Context synchronization required is required.
ISOCM_c405Hold	I	0	Middle–	Instruction-side OCM is working on the request but is unable to complete the transfer in the current cycle.
ISOCM_c405RdDValid(0:1)	I	0	Middle	Validates instruction-side OCM read data. Bit 0: Even data valid on RdDBus(0:31) Bit 1: Odd data valid on RdDBus(32:63)
ISOCM_c405RdDBus(0:63)	I	Tie to GND	Late	Instruction-side OCM read data bus - the 64-bit memory-aligned data bus used to transfer instructions to the CPU.

Data-Side OCM

The data-side on-chip memory (DSOCM) interface provides low-latency data accesses to the on-chip memory (OCM) capable of cycle performance identical to cache hits. Because data-side OCM data does not flow through the data cache unit (DCU), the DCU remains available for caching data from other sources.

Data-Side OCM Interface I/O Symbol

Figure 9 illustrates the data-side OCM signals.

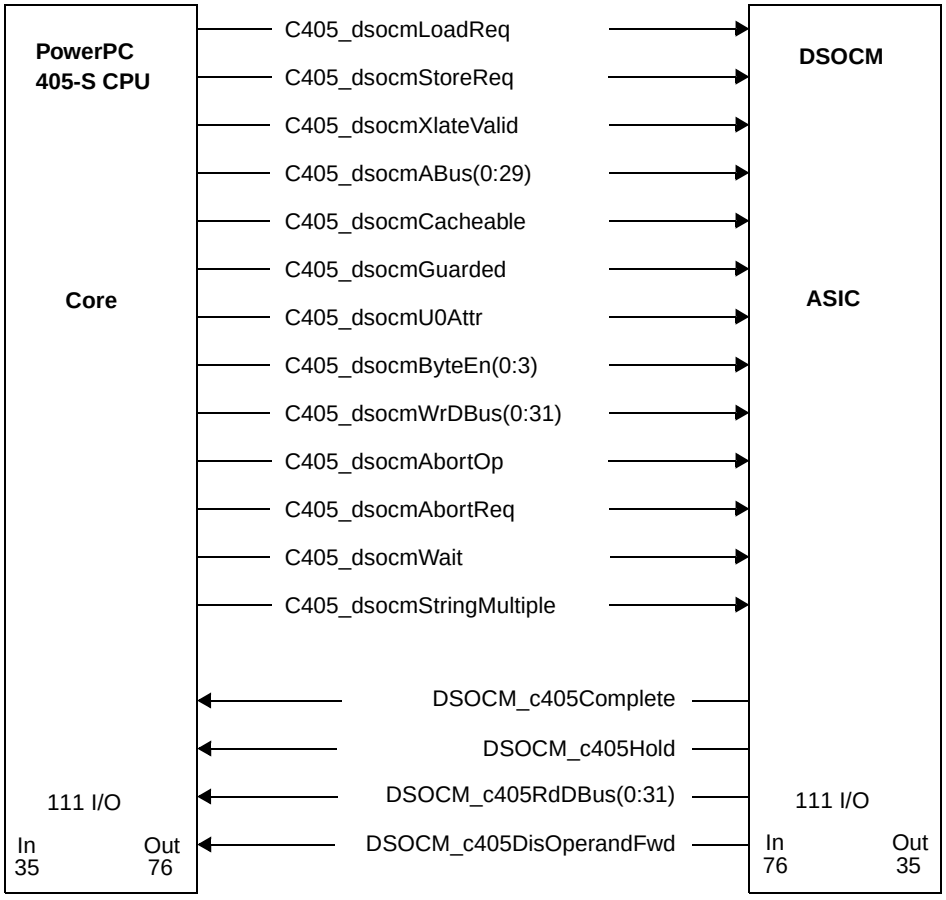

Figure 9: Data-Side OCM Interface I/O Symbol

Data-Side OCM Interface I/O Signal Summary

Table 12 lists the data-side OCM signals and provides summary descriptions.

Table 12: Data-Side OCM Interface I/O Signal Summary

Signal	I/O Type	If Unused	Timing	Function
C405_dsocmLoadReq	O	No Connect	Middle+	Valid load request
C405_dsocmStoreReq	O	No Connect	Middle+	Valid store request
C405_dsocmXlateValid	O	No Connect	Begin	Authorizes access of requested address and validates storage attributes and store data.
C405_dsocmABus(0:29)	O	No Connect	End	Load/store effective address
C405_dsocmCacheable	O	No Connect	Begin	Cacheability of load/store address
C405_dsocmGaurded	O	No Connect	Begin	Guarded load/store address
C405_dsocmU0Attr	O	No Connect	Begin	Indicates the value of the user-defined storage attribute for the target address
C405_dsocmByteEn(0:3)	O	No Connect	Middle+	Load/store byte(s) requested
C405_dsocmWrDBus(0:31)	O	No Connect	Late	Store data bus to data-side OCM
C405_dsocmAbortOp	O	No Connect	Late	Aborts new requests and operations after request cycle.
C405_dsocmAbortReq	O	No Connect	Late	Aborts any new load or store request.
C405_dsocmWait	O	No Connect	Begin	Wait for DCU load to finish.
C405_dsocmStringMultiple	O	No Connect	Early	String/multiple load/store address
DSOCM_c405Complete	I	0	Early+	Indicates that the store is moving to write stage, or that load data is available
DSOCM_c405Hold	I	0	Early+	Operation in OCM address space, but OCM is not ready.
DSOCM_c405RdDBus(0:31)	I	0x00000000	Middle	Data-side OCM read (load) data bus to core.
DSOCM_c405DisOperandFwd	I	0	DC	Disables operand forwarding to improve read data path timing.

Device Control Register (DCR)

The Device Control Register (DCR) interface provides a mechanism for the DesignWare IBM PowerPC 405-S CPU Core to set up non-core (peripheral) facilities that reside on-chip. For example, programmable facilities in an external bus interface unit can be configured for use with external memory devices, according to their transfer characteristics and address assignments. The DCRs are accessed using of the PowerPC instructions **mfdcr** and **mtdcr**.

The DCR interface comprises an address bus, an input data bus and output data bus, controls indicating whether an operation is a read (move from) or write (move to) operation, and an acknowledge signal indicating whether a move-to operation is complete, or whether move-from data is on the bus. The acknowledge signal interlocks with the control signals; a transfer takes at least three CPU cycles.

The interlock mechanism enables the DCR interface to be connected to peripheral units that are clocked at different frequencies than the core clock.

Note

The rising edge of the slower clock (either the CPU clock or the peripheral clock) must always correspond to the rising edge of the faster clock. This implies that the clocks for the ASIC logic that contains the DCR, and the clocks for the PowerPC 405-S CPU, are derived from a common source. The common source frequency can be multiplied or divided before being sent to the PowerPC 405-S CPU or ASIC logic.

DCR Interface I/O Symbol

Figure 10 illustrates the I/O signals for the DCR interface.

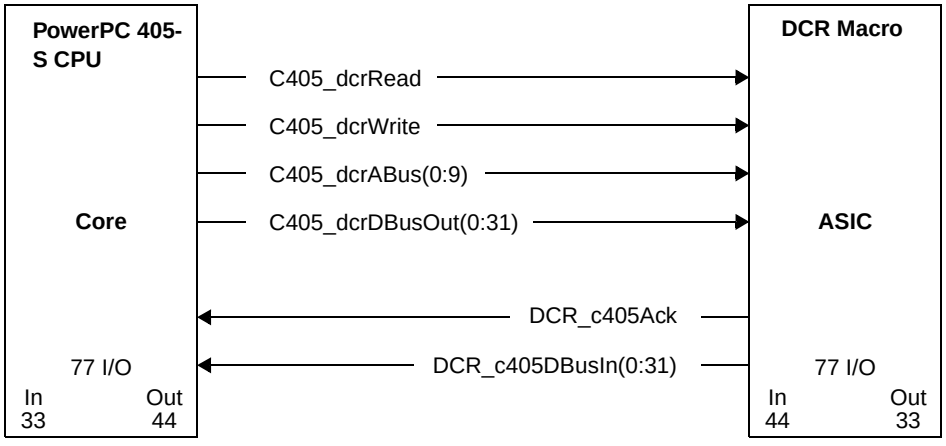

Figure 10: DCR Interface I/O Symbol

DCR Interface I/O Signal Summary

Table 13 summarizes the DCR interface signals.

Table 13: DCR Interface I/O Signals

Signal	I/O Type	If Unused	Timing	Function
C405_dcrRead	O	No connect	Latch	Indicates that the CPU is requesting to read data from a DCR into the CPU.
C405_dcrWrite	O	No connect	Latch	Indicates that the CPU is requesting to write data from the CPU to a DCR.
C405_dcrABus(0:9)	O	No connect	Middle–	Indicates the address of the DCR which is being accessed for a read or write DCR operation.
C405_dcrDBusOut(0:31)	O	No connect, or wrap to DBus In	Middle	The 32-bit data bus output used to transfer write data from the CPU to a DCR. This bus also drives 0x00000000 during DCR read operations.
DCR_c405Ack	I	0	Middle	Indicates that the target DCR data is available on DCR_c405DBusIn (read), or that the new DCR data is latched (write).
DCR_c405DBusIn(0:31)	I	0x00000000 or wrapped DBus Out	Middle–	The 32-bit data bus input used to transfer read data from a DCR to the CPU.

External Interrupt Controller (EIC)

This section describes the External Interrupt Controller interface (EIC). The IBM PowerPC Embedded Environment defines two architected interrupts, the external interrupt and the critical interrupt. Off-core non-critical interrupts should feed into the PowerPC 405-S CPU using the EIC_c405ExtInputIRQ signal. Off-core critical interrupts should feed into the PowerPC 405-S CPU using the EIC_c405CritInputIRQ signal.

EIC Interface I/O Symbol

Figure 11 illustrates the I/O signals for the EIC interface.

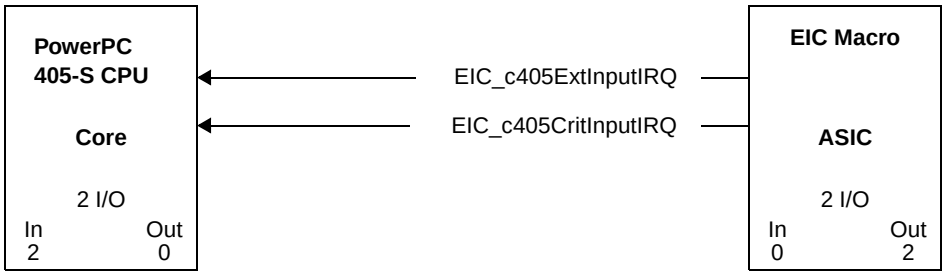

Figure 11: EIC Interface I/O Symbol

EIC Interface I/O Signal Summary

Table 14 summarizes the EIC interface signals.

Table 14: EIC Interface I/O Signals

Signal	I/O Type	If Unused	Timing	Function
EIC_c405ExtInputIRQ	I	0	Early	Indicates that an external non-critical interrupt occurred.
EIC_c405CritInputIRQ	I	0	Early	Indicates that a critical interrupt occurred.

Joint Test Action Group (Jtag) Test Access Port (TAP)

The JTAG (Joint Test Action Group) interface provides basic JTAG chip testing functionality. It also provides the ability for an external debug tool, such as RISCWatch, to gain control of the processor for debug purposes.

JTAG Interface I/O Symbol

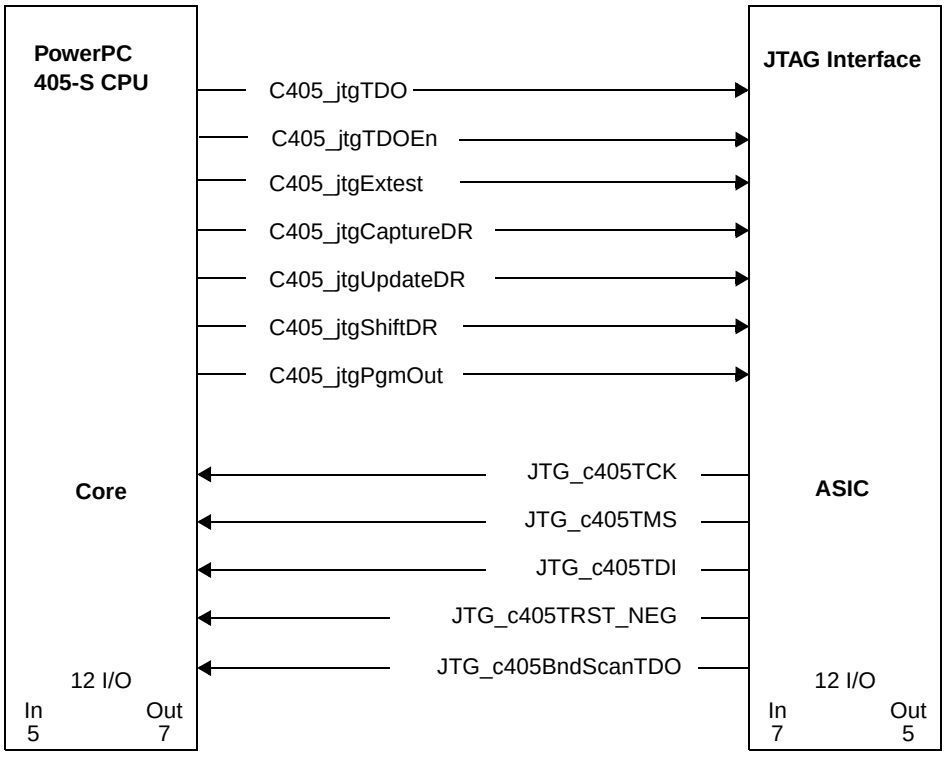

Figure 12: JTAG Interface I/O Symbol

JTAG Interface I/O Signal Table

Table 15 summarizes the JTAG interface signals.

Table 15: JTAG Interface I/O Signals

Signal	I/O Type	If Unused	Timing	Function
JTG_c405TCK	I	See IEEE 1149.1	CLK	JTAG Test Clock (TCK)—the core's JTAG logic source clock. This input is also used as an LSSD B clock during Core_LSSD mode.
JTG_c405TMS	I	1	End	JTAG Test Mode Select (TMS) determines the mode in which the TAP operates.
JTG_c405TDI	I	1	End	JTAG Test Data In (TDI) is a JTAG serial data input port.
JTG_c405TRST_NEG	I	Required	Middle	JTAG Test Reset ($\overline{\text{TRST}}$) *must* be asserted at POR. It can be used as $\overline{\text{TRST}}$ (negative active) thereafter.
JTG_c405BndScanTDO	I	0	End	This is an input fed from the last JTAG boundary scan latch's ScanOut port. This signal feeds the TDO signal when scanning.
C405_jtgTDO	O	No connect	Late–	JTAG Test Data Out (TDO) is a JTAG serial data output port.
C405_jtgTDOEn	O	No connect	Late–	This is the core's enable signal for the JTAG C405_jtgTDO signal.
C405_jtgExtest	O	No connect	Late–	This signal indicates that the JTAG EXTEST instruction is selected.
C405_jtgCaptureDR	O	No connect	Early+	This signal indicates that the JTAG state machine is in the Capture-DR state.
C405_jtgShiftDR	O	No connect	Early+	This signal indicates that the JTAG state machine is in the Shift-DR state.
C405_jtgUpdateDR	O	No connect	Early+	This signal indicates that the JTAG state machine is in the Update-DR state.
C405_jtgPgmOut	O	No connect	Early+	This signal indicates the state of a general purpose program bit in the JDCR that can be set through the external JTAG interface.

Debug (DBG)

The Debug (DBG) interface inputs into the PowerPC 405-S CPU may be used to provide additional debug enhancements for the ASIC customer. The signals on this interface may be used to provide information and control to an external debug tool, such as RISCWatch, and also allows customer debug logic to interrupt the normal processor flow through detection and reporting of an off-core debug event.

Debug Interface I/O Symbol

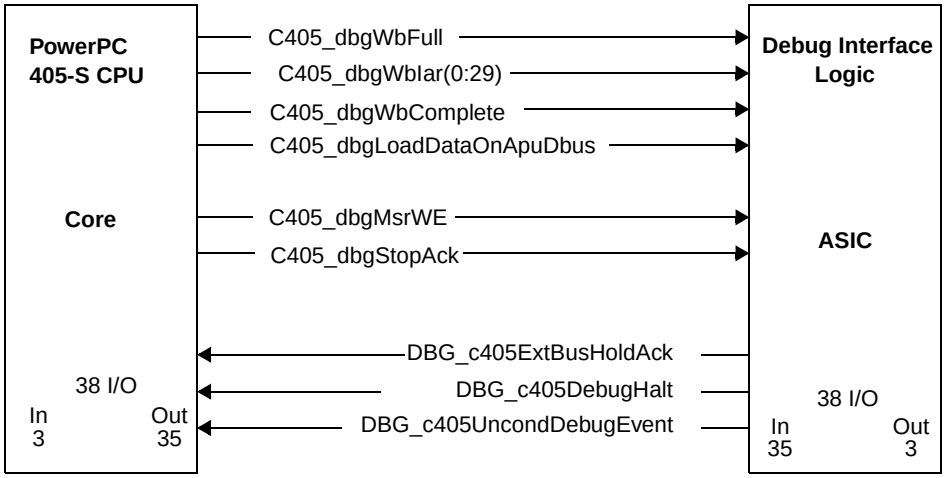

Figure 13: PowerPC 405-S CPU Debug Interface I/O Symbol

Debug Interface I/O Signal Summary

Table 16 summarizes the Debug interface signals, which are described in detail in text following the table.

Table 16: Debug Interface I/O Signals

Signal	I/O Type	If Unused	Timing	Function
DBG_c405ExtBusHoldAck	I	0	Begin	Indicates that a bus controller or PLB has given control of its external bus to an external master and the bus is unavailable for debug purposes.
DBG_c405DebugHalt	I	0	End	Indicates that an external (to the core) source is requesting to stop the processor for debug purposes.
DBG_c405UncondDebugEvent	I	0	N/A	Indicates a request for a UDE debug event and sets the UDE bit of the DBSR if enabled. Allows customer debug logic to interrupt normal CPU flow.
C405_dbgWbFull	O	No Connect	Begin	Indicates that the write back stage of the CPU pipeline is full.
C405_dbgWbIar(0:29)	O	No Connect	Begin	The address of the instruction currently in the write back stage of the CPU.
C405_dbgWbComplete	O	No Connect	End	Indicates the instruction in write back will be complete or moving to load write back.
C405_dbgLoadDataOnApuDbus	O	No Connect	Begin	Indicates that the C405_apuExeLoadDbus contains valid data from a load.
C405_dbgMsrWE	O	No Connect	Begin	Indicates that the CPU is in the Wait State.
C405_dbgStopAck	O	No Connect	Middle	Indicates that the CPU is stopped.

Trace

The PowerPC 405-S CPU provide a Trace interface that enables the connection of an external trace tool, such as RISCWatch, and allows for user-extended trace functions. It is important to note that users can have full trace capability *without* adding ASIC logic, although including some trace control logic in the ASIC can provide some benefits.

Trace Interface I/O Symbol

Figure 14 illustrates the Trace interface.

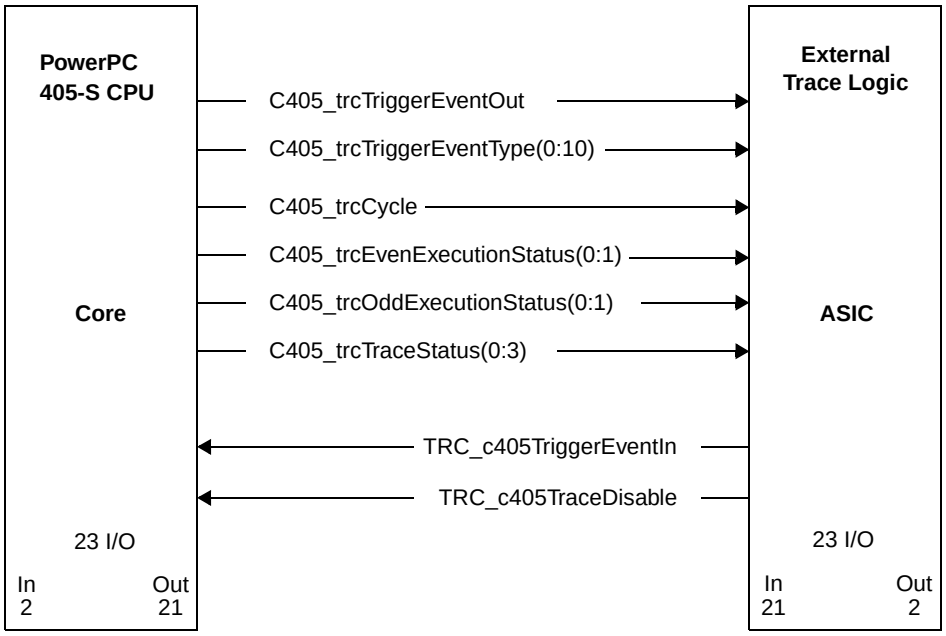

Figure 14: Trace Interface I/O Symbol

Trace Interface Signal Summary

Table 17 lists and provides summary information about the trace interface signals.

Table 17: Trace Interface Signals

Signal	I/O Type	If Unused	Timing	Function
C405_trcTriggerEventOut	O	Wrap to TriggerEventIn	Begin	CPU (debug) trigger event indication for trace logic.
C405_trcTriggerEventType(0:10)	O	No Connect	Begin	Identifies the debug event that caused TriggerEventOut.
C405_trcCycle	O	No connect	Begin	Signal that represents the trace cycle. This is not a clock signal.
C405_trcEvenExecutionStatus(0:1)	O	No connect	Begin	Encoded execution status bus for the first instruction represented in the trcCycle.
C405_trcOddExecutionStatus(0:1)	O	No connect	Begin	Encoded execution status bus for the second instruction represented in the trcCycle.
C405_trcTraceStatus(0:3)	O	No connect	Begin	Encoded trace status bus.
TRC_c405TriggerEventIn	I	Wrap to TriggerEventOut	Middle	Trigger event input to trace logic
TRC_c405TraceDisable	I	0	End	Allows an external signal to disable the trace function

Auxiliary Processor Unit (APU)

The Auxiliary Processor Unit (APU) interface enables a Core + ASIC implementation to use an auxiliary processor to execute instructions that are not part of the instruction set of the PowerPC Architecture or IBM PowerPC Embedded Environment. Implementations can control user-specific operations using user-defined instructions. An auxiliary processor can also replace the PowerPC 405-S CPU internal multiply or divide logic with faster hardware. User-defined instructions would appear as illegal op codes to the execute unit (EXU) if not validated by an auxiliary processor.

APU Interface I/O Symbol

Figure 15 and Figure 16 illustrate the APU interface.

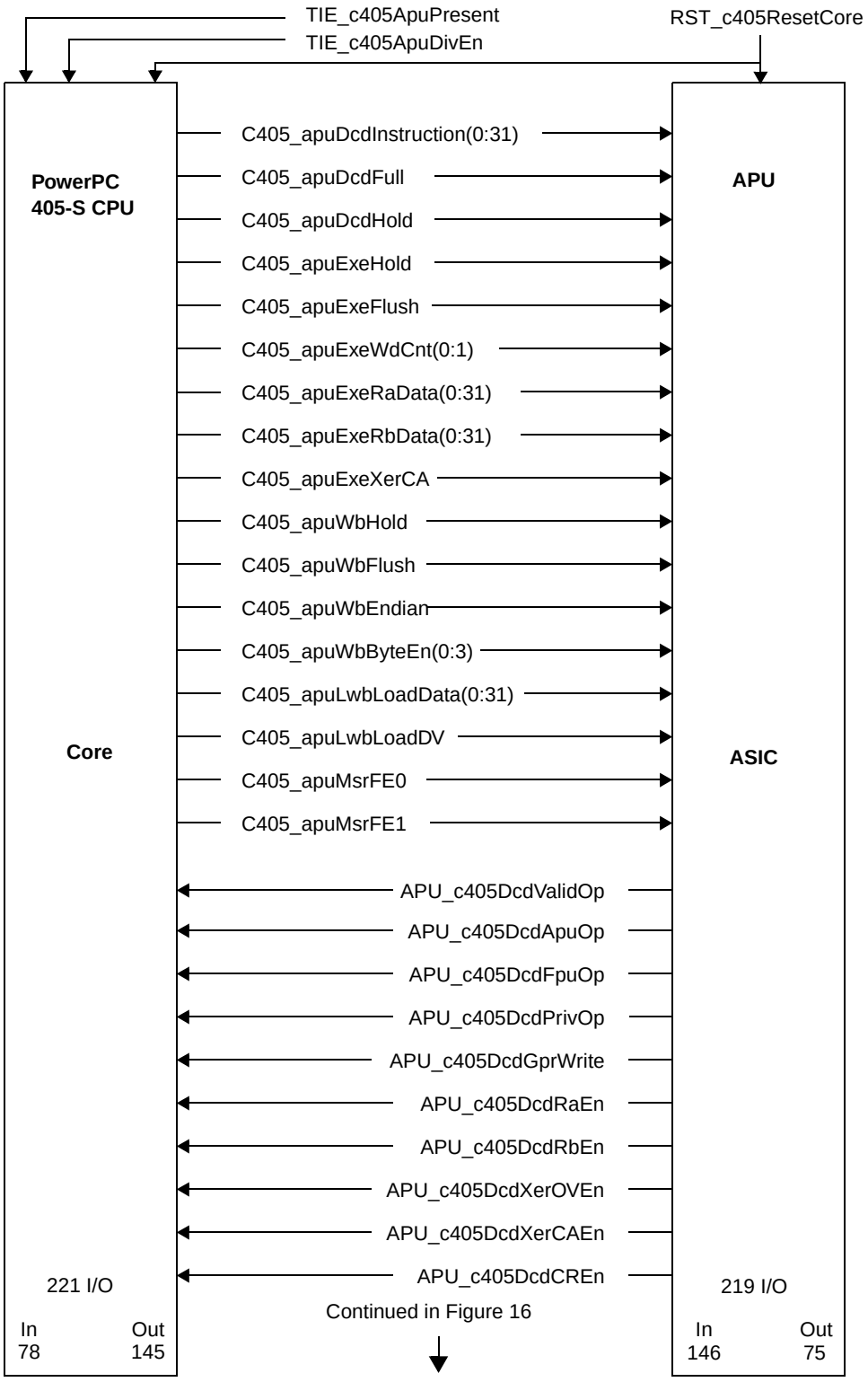

Figure 15: APU Interface I/O Symbol (Part 1)

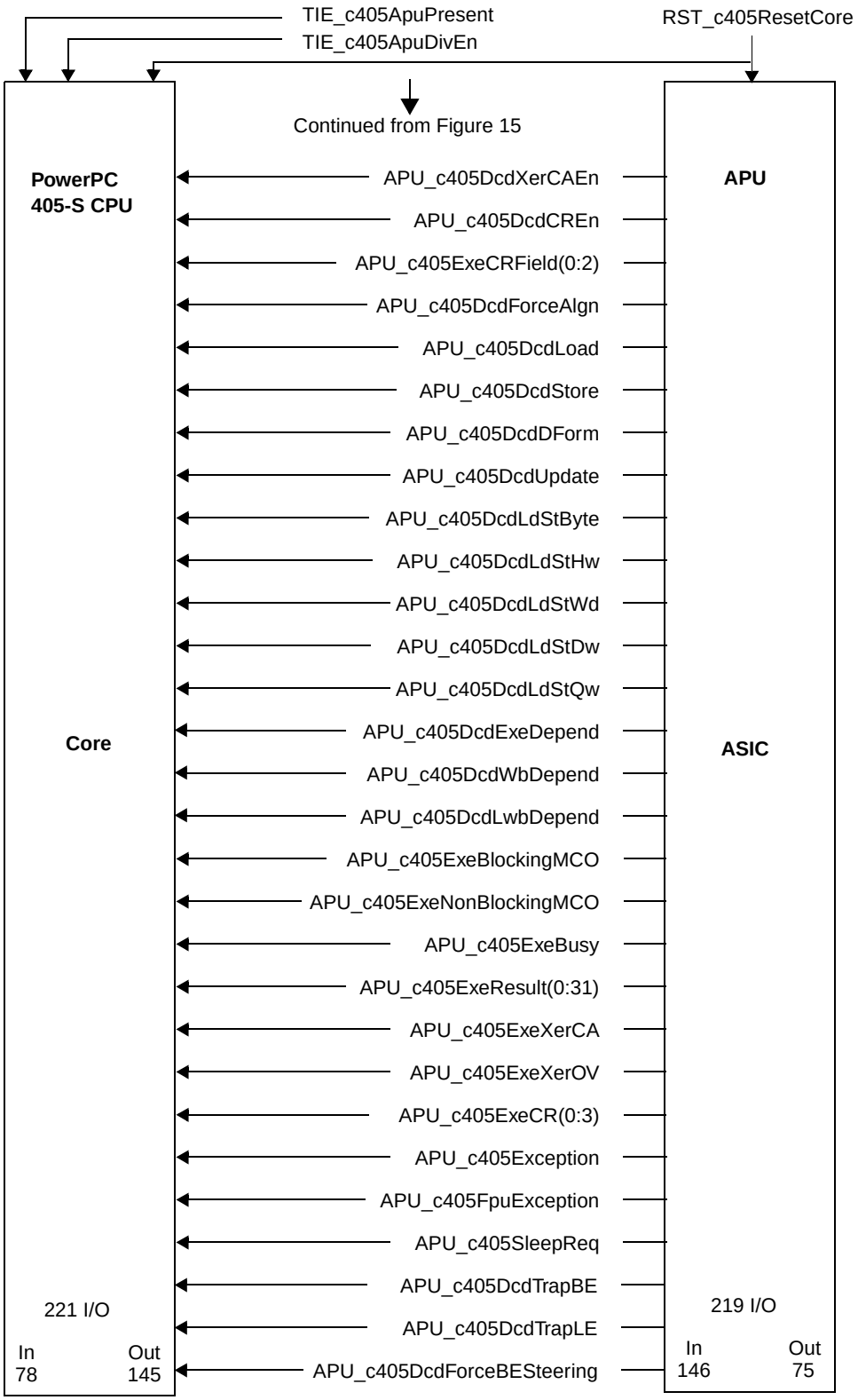

Figure 16: APU Interface I/O Symbol (Part 2)

APU Interface I/O Signal Summary

Table 18 lists and provides summary information about the APU interface signals.

Table 18: APU Interface Signals

Signal	I/O Type	If Unused	Timing	Function
C405_apuDcdInstruction(0:31)	O	No connect	Begin	Current instruction in decode.
C405_apuDcdFull	O	No connect	Begin	0 = No instruction in DCD, 1 = instruction in DCD.
C405_apuDcdHold	O	No connect	Late	When asserted instruction in DCD should not be allowed to move to execute stage.
C405_apuExeHold	O	No connect	Middle+	When asserted an APU instruction in execute must hold in execute stage.
C405_apuExeFlush	O	No connect	Late	When asserted execute stage must be flushed.
C405_apuExeWdCnt(0:1)	O	No connect	Early	00 = Word or less, 01 = double word, 10 = triple word, 11 = quad word, load/store. The count is decremented as each word transfer is passed through the pipeline. The instruction is held in EXE till the word count goes to zero.
C405_apuExeRaData(0:31)	O	No connect	Early+	Data read from GPR file using RA field of instruction.
C405_apuExeRbData(0:31)	O	No connect	Early+	Data read from GPR file using RB field of instruction.
C405_apuXerCA	O	No connect	Middle-	This is the xerCA bit used for extended arithmetic.
C405_apuWbHold	O	No connect	Early+	When asserted an APU instruction in write back must hold in write back stage.
C405_apuWbFlush	O	No connect	Middle+	When asserted write back stage must be flushed.
C405_apuWbEndian	O	No connect	Early	When asserted a load/store instruction in write back has true little Endian storage attribute.
C405_apuWbByteEn(0:3)	O	No connect	Early	Specifies the valid bytes for load instruction in write back stage.
C405_apuExeLoadDBus(0:31)	O	No connect	Early	Data loaded from storage for the instruction in load write back.

Table 18: APU Interface Signals (Continued)

Signal	I/O Type	If Unused	Timing	Function
C405_apuExeLoadDValid	O	No connect	Early	When asserted, the load data is valid on C405_apuExeLoadDBus(0:31).
C405_apuMsrFE0	O	No connect	Early	This is the msrFE0 bit used for FPU exception.
C405_apuMsrFE1	O	No connect	Early	This is the msrFE1 bit used for FPU exception.
APU_c405DcdValidOp	I	0	Late	When asserted, a valid APU instruction is in decode.
APU_c405DcdApuOp	I	0	End	When asserted, an APU instruction is in decode.
APU_c405DcdFpuOp	I	0	End	When asserted, an FPU instruction is in decode.
APU_c405DcdPrivOp	I	0	End	When asserted, the APU instruction in decode is privileged.
APU_c405DcdGprWrite	I	0	End	Indicates an APU result is to be placed in the GPR file.
APU_c405DcdRaEn	I	0	End	When asserted, an APU instruction is in decode read GPR file using RA field of instruction.
APU_c405DcdRbEn	I	0	End	When asserted, an APU instruction is in decode read GPR file using RB field of instruction.
APU_c405DcdXerOVEn	I	0	End	Indicates overflow is to be recorded from an APU.
APU_c405DcdXerCAEn	I	0	End	Indicates carry is to be recorded from an APU.
APU_c405DcdCREn	I	0	End	Indicates condition code is to be recorded from an APU.
APU_c405ExeCRField(0:2)	I	0	End	Specifies the CR field to be set by the APU instruction in decode stage.
APU_c405DcdForceAlgn	I	0	End	When asserted, forces an alignment for load/store.
APU_c405DcdLoad	I	0	Late	When asserted, the APU instruction in decode is a Load.
APU_c405DcdStore	I	0	Late	When asserted, the APU instruction in decode is a Store.

Table 18: APU Interface Signals (Continued)

Signal	I/O Type	If Unused	Timing	Function
APU_c405DcdUpdate	I	0	Late	When asserted, the APU instruction in decode has an update form.
APU_c405DcdLdStByte	I	0	End	When asserted, the APU instruction in decode is Load/Store byte.
APU_c405DcdLdStHw	I	0	End	When asserted, the APU instruction in decode is Load/Store halfword.
APU_c405DcdLdStWd	I	0	End	When asserted, the APU instruction in decode is Load/Store word.
APU_c405DcdLdStDw	I	0	End	When asserted, the APU instruction in decode is Load/Store doubleword.
APU_c405DcdLdStQw	I	0	End	When asserted, the APU instruction in decode is Load/Store quadword.
APU_c405DcdTrapBE	I	0	End	When asserted, the APU/FPU load/store instruction in decode will take alignment exception if the storage Endian attribute is 0.
APU_c405DcdTrapLE	I	0	End	When asserted, the APU/FPU load/store instruction in decode will take alignment exception if the storage Endian attribute is 1.
APU_c405DcdForceBESteering	I	0	End	When asserted, the APU/FPU load/store instruction in decode will force Big Endian steering.
APU_c405ExeLdDepend	I	0	Late	When asserted the APU instruction in decode has a dependency on APU load instruction in execute stage.
APU_c405WbLdDepend	i	0	Late	When asserted the APU instruction in decode has a dependency on APU load instruction in write back stage.
APU_c405LwbLdDepend	I	0	Late	When asserted the APU instruction in decode has a dependency on APU load instruction in load write back stage.
APU_c405ExeBlockingMCO	I	0	Middle	When asserted, holds CPU pipe and is non-interruptible after 1st cycle.
APU_c405ExeNonBlockingMCO	I	0	Middle	When asserted, holds CPU pipe and is interruptible at any time.

Table 18: APU Interface Signals (Continued)

Signal	I/O Type	If Unused	Timing	Function
APU_c405ExeBusy	I	0	Middle	When asserted, APU is busy with an AMCO and cannot receive additional instructions.
APU_c405ExeResult(0:31)	I	0	Late	Result or store data from APU.
APU_c405ExeXerCA	I	0	Late	Carry bit from APU/FPU to XER.
APU_c405ExeXerOV	I	0	Late	Overflow bit from APU.
APU_c405ExeCR(0:3)	I	0	Late	APU condition code as a result of APU instruction in execute.
APU_c405Exception	I	0	Middle-	When asserted, and if enabled, generates program exception (vctr 0x0700).
APU_c405FpuException	I	0	Middle-	When asserted, and if enabled, generates program exception (vctr 0x0700).
APU_c405SleepReq	I	1	Late	0 = no sleep req, 1 = sleep req. Must bea 1 for CPU to sleep
TIE_c405apuPresent	I	0	N/A	1 = an auxiliary processor is included in the system for use with the CPU.
TIE_c405apuDivEn	I	0	N/A	0 = Use internal divide, 1 = APU supplied divide.
RST_c405ResetCore				