

IBM 6x86 MICROPROCESSOR  
Sixth-Generation Superscalar  
Superpipelined x86-Compatible CPU



Bus Interface

3.0 IBM 6x86 BUS INTERFACE

The signals used in the IBM 6x86 CPU bus interface are described in this chapter. Figure 3-1 shows the signal directions and the major signal groupings. A description of each signal and their reference to the text are provided in Table 3-1 (Page 3-2).

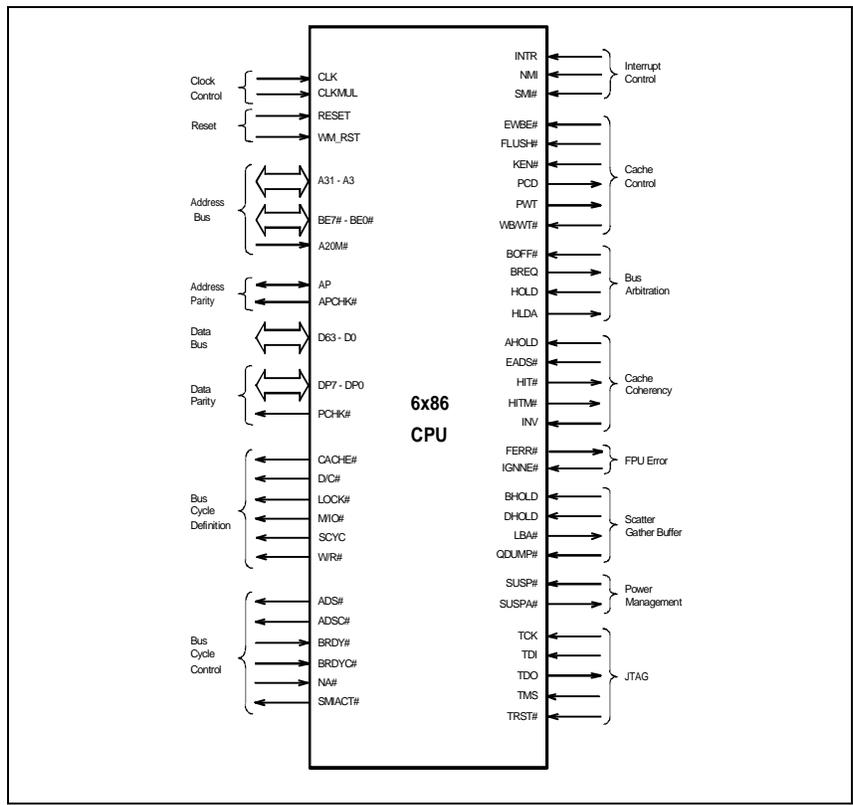


Figure 3-1. IBM 6x86 CPU Functional Signal Groupings



3.1 Signal Description Table

The Signal Summary Table (Table 3-1) describes the signals in their active state unless otherwise mentioned. Signals containing slashes (/) have logic levels defined as “1/0.” For example the signal W/R#, is defined as write when W/R#=1, and as read when W/R#=0. Signals ending with a “#” character are active low.

Table 3-1. IBM 6x86 CPU Signals Sorted by Signal Name

Signal Name	Description	I/O	Reference
<b>A20M#</b>	<b>A20 Mask</b> causes the CPU to mask (force to 0) the A20 address bit when driving the external address bus or performing an internal cache access. A20M# is provided to emulate the 1 MByte address wrap-around that occurs on the 8086. Snoop addressing is not effected.	Input	Page 3-9
<b>A31-A3</b>	The <b>Address Bus</b> , in conjunction with the Byte Enable signals (BE7#-BE0#), provides addresses for physical memory and external I/O devices. During cache inquiry cycles, A31-A5 are used as inputs to perform cache line invalidations.	3-state I/O	Page 3-9
<b>ADS#</b>	<b>Address Strobe</b> begins a memory/I/O cycle and indicates the address bus (A31-A3, BE7#-BE0#) and bus cycle definition signals (CACHE#, D/C#, LOCK#, M/IO#, PCD, PWT, SCYC, W/R#) are valid.	Output	Page 3-13
<b>ADSC#</b>	<b>Cache Address Strobe</b> performs the same function as ADS#.	Output	Page 3-13
<b>AHOLD</b>	<b>Address Hold</b> allows another bus master access to the IBM 6x86 CPU address bus for a cache inquiry cycle. In response to the assertion of AHOLD, the CPU floats AP and A31-A3 in the following clock cycle.	Input	Page 3-18
<b>AP</b>	<b>Address Parity</b> is the even parity output signal for address lines A31-A5 (A4 and A3 are excluded). During cache inquiry cycles, AP is the even-parity input to the CPU, and is sampled with EADS# to produce correct parity check status on the APCHK# output.	3-state I/O	Page 3-10
<b>APCHK#</b>	<b>Address Parity Check Status</b> is asserted during a cache inquiry cycle if an address bus parity error has been detected. APCHK# is valid two clocks after EADS# is sampled active. APCHK# will remain asserted for one clock cycle if a parity error is detected.	Output	Page 3-10
<b>BE7#-BE0#</b>	The <b>Byte Enables</b> , in conjunction with the address lines, determine the active data bytes transferred during a memory or I/O bus cycle.	3-state I/O	Page 3-9
<b>BHOLD</b>	<b>Byte Enable Hold</b> forces the byte enables (BE7#-BE0#) to float during the next clock cycle. The IBM 6x86 CPU continues to generate additional bus cycles while BHOLD is asserted. While BHOLD is asserted, the byte enables are driven by an external source and select which data bytes are accessed through the scatter/gather buffer. BHOLD is ignored if the scatter/gather interface is disabled.	Input	Page 3-20
<b>BOFF#</b>	<b>Back-Off</b> forces the IBM 6x86 CPU to abort the current bus cycle and relinquish control of the CPU local bus during the next clock cycle. The IBM 6x86 CPU enters the bus hold state and remains in this state until BOFF# is negated.	Input	Page 3-16

Table 3-1. IBM 6x86 CPU Signals Sorted by Signal Name (Continued)

Signal Name	Description	I/O	Reference
<b>BRDY#</b>	<b>Burst Ready</b> indicates that the current transfer within a burst cycle, or the current single transfer cycle, can be terminated. The IBM 6x86 CPU samples BRDY# in the second and subsequent clocks of a bus cycle. BRDY# is active during address hold states.	Input	Page 3-13
<b>BRDYC#</b>	<b>Cache Burst Ready</b> performs the same function as BRDY# and is logically ORed with BRDY# within the IBM 6x86 CPU.	Input	Page 3-13
<b>BREQ</b>	<b>Bus Request</b> is asserted by the IBM 6x86 CPU when an internal bus cycle is pending. The IBM 6x86 CPU always asserts BREQ, along with ADS#, during the first clock of a bus cycle. If a bus cycle is pending, BREQ is asserted during the bus hold and address hold states. If no additional bus cycles are pending, BREQ is negated prior to termination of the current cycle.	Output	Page 3-16
<b>CACHE#</b>	<b>Cacheability Status</b> indicates that a read bus cycle is a potentially cacheable cycle; or that a write bus cycle is a cache line write-back or line replacement burst cycle. If CACHE# is asserted for a read cycle and KEN# is asserted by the system, the read cycle becomes a cache line fill burst cycle.	Output	Page 3-11
<b>CLK</b>	<b>Clock</b> provides the fundamental timing for the IBM 6x86 CPU. The frequency of the IBM 6x86 CPU input clock determines the operating frequency of the CPU's bus. External timing is defined referenced to the rising edge of CLK.	Input	Page 3-7
<b>CLKMUL</b>	The <b>Clock Multiplier</b> input is sampled during RESET to determine the IBM 6x86 CPU core operating frequency. If CLKMUL=0 or is left unconnected, the core frequency is 2x the frequency of the CLK input. If CLKMUL=1, the core frequency is 3x the frequency of CLK.	Input	Page 3-7
<b>D63-D0</b>	<b>Data Bus</b> signals are three-state, bi-directional signals which provide the data path between the IBM 6x86 CPU and external memory and I/O devices. The data bus is only driven while a write cycle is active (state=T2). The data bus is floated when DHOLD is asserted.	3-state I/O	Page 3-10
<b>D/C#</b>	<b>Data/Control Status</b> . If high, indicates that the current bus cycle is an I/O or memory data access cycle. If low, indicates a code fetch or special bus cycle such as a halt, prefetch, or interrupt acknowledge bus cycle. D/C# is driven valid in the same clock as ADS# is asserted.	Output	Page 3-11
<b>DHOLD</b>	<b>Data Bus Hold</b> forces the IBM 6x86 CPU to float the data bus (D63-D0) and the data parity lines (DP7-DP0) in the next clock. While DHOLD is asserted, only the data and data parity buses are disabled. The current bus cycle remains active and is completed in the normal fashion in response to BRDY#. The IBM 6x86 CPU generates additional bus cycles while DHOLD is asserted. DHOLD is ignored if the scatter/gather interface is disabled.	Input	Page 3-21
<b>DP7-DP0</b>	<b>Data Parity</b> signals provide parity for the data bus, one data parity bit per data byte. Even parity is driven on DP7-DP0 for all data write cycles. DP7-DP0 are read by the IBM 6x86 CPU during read cycles to check for even parity. The data parity bus is only driven while a write cycle is active (state=T2).	3-state I/O	Page 3-10



Signal Description Table

Table 3-1. IBM 6x86 CPU Signals Sorted by Signal Name (Continued)

Signal Name	Description	I/O	Reference
<b>EADS#</b>	<b>External Address Strobe</b> indicates that a valid cache inquiry address is being driven on the IBM 6x86 CPU address bus (A31-A5) and AP. The state of INV at the time EADS# is sampled active determines the final state of the cache line. A cache inquiry cycle using EADS# may be run while the IBM 6x86 CPU is in the address hold or bus hold state.	Input	Page 3-18
<b>EWBE#</b>	<b>External Write Buffer Empty</b> indicates that there are no pending write cycles in the external system. EWBE# is sampled only during I/O and memory write cycles. If EWBE# is negated, the IBM 6x86 CPU delays all subsequent writes to on-chip cache lines in the "exclusive" or "modified" state until EWBE# is asserted.	Input	Page 3-14
<b>FERR#</b>	<b>FPU Error Status</b> indicates an unmasked floating point error has occurred. FERR# is asserted during execution of the FPU instruction that caused the error. FERR# does not float during bus hold states.	Output	Page 3-19
<b>FLUSH#</b>	<b>Cache Flush</b> forces the IBM 6x86 CPU to flush the cache. External interrupts and additional FLUSH# assertions are ignored during the flush. Cache inquiry cycles are permitted during the flush.	Input	Page 3-15
<b>HIT#</b>	<b>Cache Hit</b> indicates that the current cache inquiry address has been found in the cache (modified, exclusive or shared states). HIT# is valid two clocks after EADS# is sampled active, and remains valid until the next cache inquiry cycle.	Output	Page 3-18
<b>HITM#</b>	<b>Cache Hit Modified Data</b> indicates that the current cache inquiry address has been found in the cache and dirty data exists in the cache line (modified state). The IBM 6x86 CPU does not accept additional cache inquiry cycles while HITM# is asserted. HITM# is valid two clocks after EADS#.	Output	Page 3-18
<b>HLDA</b>	<b>Hold Acknowledge</b> indicates that the IBM 6x86 CPU has responded to the HOLD input and relinquished control of the local bus. The IBM 6x86 CPU continues to operate during bus hold as long as the on-chip cache can satisfy bus requests.	Output	Page 3-16
<b>HOLD</b>	<b>Hold Request</b> indicates that another bus master has requested control of the CPU's local bus.	Input	Page 3-16
<b>IGNNE#</b>	<b>Ignore Numeric Error</b> forces the IBM 6x86 CPU to ignore any pending unmasked FPU errors and allows continued execution of floating point instructions.	Input	Page 3-19
<b>INTR</b>	<b>Maskable Interrupt</b> forces the processor to suspend execution of the current instruction stream and begin execution of an interrupt service routine. The INTR input can be masked (ignored) through the IF bit in the Flags Register.	Input	Page 3-14
<b>INV</b>	<b>Invalidate Request</b> is sampled with EADS# to determine the final state of the cache line in the case of a cache inquiry hit. An asserted INV directs the processor to change the state of the cache line to "invalid". A negated INV directs the processor to change the state of the cache line to "shared."	Input	Page 3-18

Table 3-1. IBM 6x86 CPU Signals Sorted by Signal Name (Continued)

Signal Name	Description	I/O	Reference
<b>KEN#</b>	<b>Cache Enable</b> allows the data being returned during the current cycle to be placed in the CPU's cache. When the IBM 6x86 CPU is performing a cacheable code fetch or memory data read cycle (CACHE# asserted), and KEN# is sampled asserted, the cycle is transformed into a 32-byte cache line fill. KEN# is sampled with the first asserted BRDY# or NA# for the cycle.	Input	Page 3-15
<b>LBA#</b>	<b>Local Bus Access</b> indicates that the current bus cycle is for an address within the local bus address region. If LBA# is asserted during a CPU write cycle with BE3#-BE0# negated, the IBM 6x86 CPU automatically maps the upper DWORD of data to the lower DWORD of the data bus. LBA# floats if scatter/gather pins are disabled.	Output	Page 3-21
<b>LOCK#</b>	<b>Lock Status</b> indicates that other system bus masters are denied access to the local bus. The IBM 6x86 CPU does not enter the bus hold state in response to HOLD while LOCK# is asserted.	Output	Page 3-11
<b>M/IO#</b>	<b>Memory/IO Status.</b> If high, indicates that the current bus cycle is a memory cycle (read or write). If low, indicates that the current bus cycle is an I/O cycle (read or write, interrupt acknowledge, or special cycle).	Output	Page 3-11
<b>NA#</b>	<b>Next Address</b> requests the next pending bus cycle address and cycle definition information. If either the current or next bus cycle is a locked cycle, a line replacement, a write-back cycle, or if there is no pending bus cycle, the IBM 6x86 CPU does not start a pipelined bus cycle regardless of the state of NA#.	Input	Page 3-13
<b>NMI</b>	<b>Non-Maskable Interrupt Request</b> forces the processor to suspend execution of the current instruction stream and begin execution of an NMI interrupt service routine.	Input	Page 3-14
<b>PCD</b>	<b>Page Cache Disable</b> reflects the state of the PCD page attribute bit in the page table entry or the directory table entry. If paging is disabled, or for cycles that are not paged, the PCD pin is driven low. PCD is masked by the cache disable (CD) bit in CR0, and floats during bus hold states.	Output	Page 3-15
<b>PCHK#</b>	<b>Data Parity Check</b> indicates that a data bus parity error has occurred during a read operation. PCHK# is only valid during the second clock immediately after read data is returned to the IBM 6x86 CPU (BRDY# asserted) and is inactive otherwise. Parity errors signaled by a logic low on PCHK# have no effect on processor execution.	Output	Page 3-10
<b>PWT</b>	<b>Page Write Through</b> reflects the state of the PWT page attribute bit in the page table entry or the directory table entry. PWT pin is negated during cycles that are not paged, or if paging is disabled. PWT takes priority over WB/WT#.	Output	Page 3-15
<b>QDUMP#</b>	<b>Q Buffer Dump</b> is used to dump the contents of the scatter/gather buffer onto the data bus. The data bytes specified by the byte enables (BE7#-BE0#) are driven onto the data bus during the clock after QDUMP# is sampled asserted. QDUMP# is ignored if the scatter/gather pins are disabled.	Input	Page 3-22
<b>RESET</b>	<b>Reset</b> suspends all operations in progress and places the IBM 6x86 CPU into a reset state. Reset forces the CPU to begin executing in a known state. All data in the on-chip caches is invalidated.	Input	Page 3-7



Table 3-1. IBM 6x86 CPU Signals Sorted by Signal Name (Continued)

Signal Name	Description	I/O	Reference
SCYC	<b>Split Locked Cycle</b> indicates that the current bus cycle is part of a misaligned locked transfer. SCYC is defined for locked cycles only. A misaligned transfer is defined as any transfer that crosses an 8-byte boundary.	Output	Page 3-11
SMI#	<b>SMM Interrupt</b> forces the processor to save the CPU state to the top of SMM memory and to begin execution of the SMI service routine at the beginning of the defined SMM memory space. An SMI is a higher-priority interrupt than an NMI.	Input	Page 3-14
SMIACK#	<b>SMM Interrupt Active</b> indicates that the processor is operating in System Management Mode. SMIACK# does not float during bus hold states.	Output	Page 3-13
SUSP#	<b>Suspend Request</b> requests that the CPU enter suspend mode. SUSP# is ignored following RESET and is enabled by setting the SUSP bit in CCR2.	Input	Page 3-22
SUSPA#	<b>Suspend Acknowledge</b> indicates that the IBM 6x86 CPU has entered low-power suspend mode. SUSPA# floats following RESET and is enabled by setting the SUSP bit in CCR2.	Output	Page 3-22
TCK	<b>Test Clock (JTAG)</b> is the clock input used by the IBM 6x86 CPU's boundary scan (JTAG) test logic.	Input	Page 3-24
TDI	<b>Test Data In (JTAG)</b> is the serial data input used by the IBM 6x86 CPU's boundary scan (JTAG) test logic.	Input	Page 3-24
TDO	<b>Test Data Out (JTAG)</b> is the serial data output used by the IBM 6x86 CPU's boundary scan (JTAG) test logic.	Output	Page 3-24
TMS	<b>Test Mode Select (JTAG)</b> is the control input used by the IBM 6x86 CPU's boundary scan (JTAG) test logic.	Input	Page 3-24
TRST#	<b>Test Mode Reset (JTAG)</b> initializes the IBM 6x86 CPU's boundary scan (JTAG) test logic.	Input	Page 3-24
WB/WT#	<b>Write-Back/Write-Through</b> is sampled during cache line fills to define the cache line write policy. If high, the cache line write policy is write-back. If low, the cache line write policy is write-through. (PWT forces write-through policy when PWT=1.)	Input	Page 3-15
WM_RST	<b>Warm Reset</b> forces the IBM 6x86 CPU to complete the current instruction and then places the IBM 6x86 CPU in a known state. Once WM_RST is sampled active by the CPU, the reset sequence begins on the next instruction boundary. WM_RST does not change the state of the configuration registers, the on-chip cache, the write buffers and the FPU registers. WM_RST is sampled during reset.	Input	Page 3-9
W/R#	<b>Write/Read Status.</b> If high, indicates that the current memory, or I/O bus cycle is a write cycle. If low, indicates that the current bus cycle is a read cycle.	Output	Page 3-11

### 3.2 Signal Descriptions

The following paragraphs provide additional information about the IBM 6x86 CPU signals. For ease of this discussion, the signals are divided into 16 functional groups as illustrated in Figure 3-1 (Page 3-1).

#### 3.2.1 Clock Control

The Clock Input (CLK) signal, supplied by the system, is the timing reference use by the IBM 6x86 CPU bus interface. All external timing parameters are defined with respect to the CLK rising edge. The CLK signal enters the IBM 6x86 CPU where it is doubled or tripled to produce the IBM 6x86 CPU internal clock signal. During power on, the CLK signal must be running even if CLK does not meet AC specifications.

The Clock Multiplier (CLKMUL) input is sampled during RESET to determine the CPU's core operating frequency. If CLKMUL=0, the core frequency is 2x the frequency of the CLK input. If CLKMUL=1, the core frequency is 3x the frequency of the CLK input. The CLKMUL input is connected to an internal pull-down resistor. Therefore, if CLKMUL is left unconnected, the core frequency defaults to 2x the input CLK. CLKMUL should be connected to Vss, Vcc, or left unconnected. CLKMUL should not be connected to a switching signal.

#### 3.2.2 Reset Control

The IBM 6x86 CPU output signals are initialized to their reset states during the CPU reset sequence, as shown in Table 3-3 (Page 3-8). The signal states given in Table 3-3 assume that HOLD, AHOLD, and BOFF# are negated.

Asserting RESET suspends all operations in progress and places the IBM 6x86 CPU in a reset state. RESET is an asynchronous signal but must meet specified setup and hold times to guarantee recognition at a particular clock edge.

On system power-up, RESET must be held asserted for at least 1 msec after Vcc and CLK have reached specified DC and AC limits. This delay allows the CPU's clock circuit to stabilize and guarantees proper completion of the reset sequence.

During normal operation, RESET must be asserted for at least 15 CLK periods in order to guarantee the proper reset sequence is executed. When RESET negates (on its falling edge), the pins listed in Table 3-2 determine if certain IBM 6x86 CPU functions are enabled.

**Table 3-2. Pins Sampled During RESET**

SIGNAL NAME	DESCRIPTION
FLUSH#	If = 0, three-state test mode enabled.
QDUMP#	If = 0, scatter/gather interface enabled.
WM_RST	If = 1, built-in self test initiated.



Table 3-3. Signal States During RESET

SIGNAL LINE	STATE	SIGNAL LINE	STATE
A20M#	Ignored	INTR	Ignored
A31-A3	Undefined until first ADS#	INV	Ignored
ADS#	1	KEN#	Ignored
ADSC#	1	LBA#	1
AHOLD	Recognized	LOCK#	1
AP	Undefined until first ADS#	M/IO#	Undefined until first ADS#
APCHK#	1	NA#	Ignored
BE7#-BE0#	Undefined until first ADS#	NMI	Ignored
BHOLD	Ignored	PCD	Undefined until first ADS#
BOFF#	Recognized	PCHK#	1
BRDY#	Ignored	PWT	Undefined until first ADS#
BRDYC#	Ignored	QDUMP#	Enables scatter/gather interface pins
BREQ	0	RESET	1
CACHE#	Undefined until first ADS#	SCYC	Undefined until first ADS#
D(63-0)	Float	SMI#	Ignored
D/C#	Undefined until first ADS#	SMIACT#	1
DHOLD	Ignored	SUSP#	Ignored
DP(7-0)	Float	SUSPA#	Float
EADS#	Ignored	TCK	Recognized
EWBE#	Ignored	TDI	Recognized
FERR#	1	TDO	Responds to TCK, TDI, TMS, TRST#
FLUSH#	Initiates three-state test mode	TMS	Recognized
HIT#	1	TRST#	Recognized
HITM#	1	W/R#	Undefined until first ADS#
HLDA	Responds to HOLD	WB/WT#	Ignored
HOLD	Recognized	WM_RST	Initiates self-test
IGNNE#	Ignored		

**Warm Reset (WM\_RST)** allows the IBM 6x86 CPU to complete the current instruction and then places the IBM 6x86 CPU in a known state. WM\_RST is an asynchronous signal, but must meet specified setup and hold times in order to guarantee recognition at a particular CLK edge. Once WM\_RST is sampled active by the CPU, the reset sequence begins on the next instruction boundary.

WM\_RST differs from RESET in that the contents of the on-chip cache, the write buffers, the configuration registers and the floating point registers contents remain unchanged.

Following completion of the internal reset sequence, normal processor execution begins even if WM\_RST remains asserted. If RESET and WM\_RST are asserted simultaneously, WM\_RST is ignored and RESET takes priority. If WM\_RST is asserted at the falling edge of RESET, built-in self test (BIST) is initiated.

### 3.2.3 Address Bus

The **Address Bus (A31-A3)** lines provide the physical memory and external I/O device addresses. A31-A5 are bi-directional signals used by the IBM 6x86™ CPU to drive addresses to both memory devices and I/O devices. During cache inquiry cycles the IBM 6x86™ CPU receives addresses from the system using signals A31-A5.

Using signals A31-A3, the IBM 6x86™ CPU can address a 4-GByte memory address space. Using signals A15-A3, the IBM 6x86™ CPU can address a 64-KByte I/O space through the processor's I/O ports. During I/O accesses, signals A31-A16 are driven low. A31-A3 float during bus hold and address hold states.

The Byte Enable (BE7#-BE0#) lines are bi-directional signals that define the valid data bytes within the 64-bit data bus. The correlation between the enable signals and data bytes is shown in Table 3-4.

**Table 3-4. Byte Enable Signal to Data Bus Byte Correlation**

BYTE ENABLE	CORRESPONDING DATA BYTE
BE7#	D63-D56
BE6#	D55-D48
BE5#	D47-D40
BE4#	D39-D32
BE3#	D31-D24
BE2#	D23-D16
BE1#	D15-D8
BE0#	D7-D0

During a cache line fill, (burst read or “1+4” burst read) the IBM 6x86 CPU expects data to be returned as if all data bytes are enabled, regardless of the state of the byte enables. BE7#-BE0# float during bus hold and byte enable hold states.

**Address Bit 20 Mask (A20M#)** is an active low input which causes the IBM 6x86 CPU to mask (force low) physical address bit 20 when driving the external address bus or when performing an internal cache access. Asserting A20M# emulates the 1 MByte address wrap-around that occurs on the 8086. The A20 signal is never masked during write-back cycles, inquiry cycles, system management address space accesses or when paging is enabled, regardless of the state of the A20M# input.



### 3.2.4 Address Parity

Address Parity (AP) is a bi-directional signal which provides the parity associated with address lines A31-A5. (A4 and A3 are not included in the parity determination.) During IBM 6x86 CPU generated bus cycles, while the address bus lines are driven, AP becomes an output supplying even address parity. During cache inquiry cycles, AP becomes an input and is sampled by EADS#. During cache inquiry cycles, even-parity must be placed on the AP line to guarantee an accurate result on the APCHK# (Address Parity Check Status) pin.

Address Parity Check Status (APCHK#) is driven active by the CPU when an address bus parity error has been detected for a cache inquiry cycle. APCHK# is asserted two clocks after EADS# is sampled asserted, and remains valid for one clock only. Address parity errors signaled by APCHK# have no effect on processor execution.

### 3.2.5 Data Bus

**Data Bus (D63-D0)** lines carry three-state, bi-directional signals between the IBM 6x86 CPU and the system (i.e., external memory and I/O devices). The data bus transfers data to the IBM 6x86 CPU during memory read, I/O read, and interrupt acknowledge cycles. Data is transferred from the IBM 6x86 CPU during memory and I/O write cycles.

Data setup and hold times must be met for correct read cycle operation. The data bus is driven only while a write cycle is active.

### 3.2.6 Data Parity

The Data Parity Bus (DP7-DP0) provides and receives parity data for each of the eight data bus bytes (Table 3-5). The IBM 6x86 CPU generates even parity on the bus during write cycles and accepts even parity from the system during read cycles. DP7-DP0 is driven only while a write cycle is active.

Table 3-5. Parity Bit to Data Byte Correlation

PARITY BIT	DATA BYTE
DP7	D63-D56
DP6	D55-D48
DP5	D47-D40
DP4	D39-D32
DP3	D31-D24
DP2	D23-D16
DP1	D15-D8
DP0	D7-D0

**Parity Check (PCHK#)** is asserted when a data bus parity error is detected. Parity is checked during code, memory and I/O reads, and the second interrupt acknowledge cycle. Parity is not checked during the first interrupt acknowledge cycle.

Parity is checked for only the active data bytes as determined by the active byte enable signals except during a cache line fill (burst read or "1+4" burst read). During a cache line fill, the IBM 6x86 CPU assumes all data bytes are valid and parity is checked for all data bytes regardless of the state of the byte enables.

PCHK# is valid only during the second clock immediately after read data is returned to the IBM 6x86 CPU (BRDY# asserted). At other times PCHK# is not active. Parity errors signaled by the assertion of PCHK# have no effect on processor execution.

### 3.2.7 Bus Cycle Definition

Each bus cycle is assigned a bus cycle type. The bus cycle types are defined by six three-state outputs: CACHE#, D/C#, LOCK#, M/IO#, SCYC, and W/R# as listed in Table 3-6 (Page 3-12).

These bus cycle definition signals are driven valid while ADS# is active. D/C#, M/IO#, W/R#, SCYC and CACHE# remain valid until the clock following the earliest of two signals: NA# asserted, or the last BRDY# for the cycle.

LOCK# continues asserted until after BRDY# is returned for the last locked bus cycle. The bus cycle definition signals float during bus hold states.

**Cache Cycle Indicator (CACHE#)** is an output that indicates that the current bus cycle is a potentially cacheable cycle (for a read), or indicates that the current bus cycle is a cache line write-back or line replacement burst cycle (for a write). If CACHE# is asserted for a read cycle and the KEN# input is returned active by the system, the read cycle becomes a cache line fill burst cycle.

**Data/Control (D/C#)** distinguishes between data and control operations. When high, this signal indicates that the current bus cycle is a data transfer to or from memory or I/O. When low, D/C# indicates that the current bus cycle

involves a control function such as a halt, interrupt acknowledge or code fetch.

**Bus Lock (LOCK#)** is an active low output which, when asserted, indicates that other system bus masters are denied access to control of the CPU bus. The LOCK# signal may be explicitly activated during bus operations by including the LOCK prefix on certain instructions. LOCK# is also asserted during descriptor updates, page table accesses, interrupt acknowledge sequences and when executing the XCHG instruction. However, if the NO\_LOCK bit in CCR1 is set, LOCK# is asserted only during page table accesses and interrupt acknowledge sequences. The IBM 6x86 CPU does not enter the bus hold state in response to HOLD while the LOCK# output is active.

**Memory/IO (M/IO#)** distinguishes between memory and I/O operations. When high, this signal indicates that the current bus cycle is a memory read or memory write. When low, M/IO# indicates that the current bus cycle is an I/O read, I/O write, interrupt acknowledge cycle or special bus cycle.

**Split Cycle (SCYC)** is an active high output that indicates that the current bus cycle is part of a misaligned locked transfer. SCYC is defined for locked cycles only. A misaligned transfer is defined as any transfer that crosses an 8-byte boundary.

**Write/Read (W/R#)** distinguishes between write and read operations. When high, this signal indicates that the current bus cycle is a memory write, I/O write or a special bus cycle. When low, this signal indicates that the current cycle is a memory read, I/O read or interrupt acknowledge cycle.



**Table 3-6. Bus Cycle Types**

BUS CYCLE TYPE	M/IO#	D/C#	W/R#	CACHE#	LOCK#
Interrupt Acknowledge	0	0	0	1	0
Does not occur.	0	0	0	X	1
Does not occur.	0	0	1	X	0
Special Cycles: If BE(7-0)# = FEh: Shutdown If BE(7-0)# = FDh: Flush (INVD, WBINVD) If A4 = 0 and BE(7-0)# = FBh: Halt (HLT) If BE(7-0)# = F7h: Write-Back (WBINVD) If BE(7-0)# = EFh: Flush Acknowledge (FLUSH#) If A4 = 1 and BE(7-0)# = FBh: Stop Grant (SUSP#)	0	0	1	1	1
Does not occur.	0	1	X	X	0
I/O Data Read	0	1	0	1	1
I/O Data Write	0	1	1	1	1
Does not occur.	1	0	X	X	0
Cacheable Memory Code Read (Burst Cycle if KEN# Returned Active)	1	0	0	0	1
Non-cacheable Memory Code Read	1	0	0	1	1
Does not occur.	1	0	1	X	1
Locked Memory Data Read	1	1	0	1	0
Cacheable Memory Data Read (Burst Cycle if KEN# Returned Active)	1	1	0	0	1
Non-cacheable Memory Data Read	1	1	0	1	1
Locked Memory Write	1	1	1	1	0
Burst Memory Write (Writeback or Line Replacement)	1	1	1	0	1*
Single Transfer Memory Write	1	1	1	1	1

Note: X = Don't Care

\*Note: LOCK# continues to be asserted during a write-back cycle that occurs following an aborted (BOFF# asserted) locked bus cycle.

### 3.2.8 Bus Cycle Control

The bus cycle control signals (ADS#, ADSC#, BRDY#, BRDYC#, NA#, and SMIACT#) indicate the beginning of a bus cycle and allow system hardware to control bus cycle termination timing and address pipelining.

**Address Strobe (ADS#)** is an active low output which indicates that the CPU has driven a valid address and bus cycle definition on the appropriate output pins. ADS# floats during bus hold states.

**Cache Address Strobe (ADSC#)** performs the same function as ADS#. ADSC# is used to interface directly to a secondary cache controller.

**Burst Ready (BRDY#)** is an active low input that is driven by the system to indicate that the current transfer within a burst cycle or the current single transfer bus cycle can be terminated. The CPU samples BRDY# in the second and subsequent clocks of a cycle. BRDY# is active during address hold states.

**Cache Burst Ready (BRDYC#)** performs the same function as BRDY# and is logically ORed with BRDY internally by the CPU. BRDYC# is used to interface directly to a secondary cache controller.

**Next Address (NA#)** is an active low input that is driven by the system to request the next pending bus cycle address and cycle definition information even though all data transfers for the current bus cycle are not complete. This new bus cycle is referred to as a “pipelined” cycle. If either the current or next bus cycle is a locked cycle, a line replacement, a write-back

cycle or there is no pending bus cycle, the IBM 6x86 CPU does not start a pipelined bus cycle regardless of the state of the NA# input.

**System Management Mode Active (SMIACT#)** is an active low output which indicates that the CPU is operating in System Management Mode. SMIACT# is asserted in response to the assertion of SMI# or due to execution of the SMINT instruction. SMIACT# is also asserted during accesses to defined SMM memory if the SMAC bit in CCR1 is set. This bit allows access to SMM memory while not in SMM mode and is typically used for initialization purposes.

While servicing an SMI# interrupt or SMINT instruction, SMIACT# remains asserted until a RSM instruction is executed. The RSM instruction causes the IBM 6x86™ CPU to exit SMM mode and negate the SMIACT# output. If a cache inquiry cycle occurs while SMIACT# is active, any resulting write-back cycle is issued with SMIACT# asserted. This occurs even though the write-back cycle is intended for normal memory rather than SMM memory.

During RESET, the USE\_SMI bit in CCR1 is cleared. While USE\_SMI is zero, SMIACT# is always negated. SMIACT# does not float during bus hold states.

### 3.2.9 Interrupt Control

The interrupt control signals (INTR, NMI, SMI#) allow the execution of the current instruction stream to be interrupted and suspended.



**Maskable Interrupt Request (INTR)** is an active high level-sensitive input which causes the processor to suspend execution of the current instruction stream and begin execution of an interrupt service routine. The INTR input can be masked (ignored) through the IF bit in the Flags Register.

When not masked, the IBM 6x86 CPU responds to the INTR input by performing two locked interrupt acknowledge bus cycles. During the second interrupt acknowledge cycle, the IBM 6x86 CPU reads an 8-bit value, the interrupt vector, from the data bus. The 8-bit interrupt vector indicates the interrupt level that caused generation of the INTR and is used by the CPU to determine the beginning address of the interrupt service routine. To assure recognition of the INTR request, INTR must remain active until the start of the first interrupt acknowledge cycle.

**Non-Maskable Interrupt Request (NMI)** is a rising edge sensitive input which causes the processor to suspend execution of the current instruction stream and begin execution of an NMI interrupt service routine. The NMI interrupt cannot be masked by the IF bit in the Flags Register. Asserting NMI causes an interrupt which internally supplies interrupt vector 2h to the CPU core. Therefore, external interrupt acknowledge cycles are not issued.

Once NMI processing has started, no additional NMIs are processed until an IRET instruction is executed, typically at the end of the NMI service routine. If NMI is re-asserted prior to execution of the IRET, one and only one NMI rising edge is stored and then processed after execution of the next IRET.

**System Management Interrupt Request (SMI#)** is an interrupt input with higher priority than the NMI input. SMI# is a falling edge sensitive input and is sampled on every rising edge of the processor input clock. Asserting SMI# forces the processor to save the CPU state to the top of SMM memory and to begin execution of the SMI service routine at the beginning of the defined SMM memory space. After the processor internally acknowledges the SMI# interrupt, the SMIACK# output is driven low for the duration of the interrupt service routine.

Once SMI# servicing has started, no additional SMI# interrupts are processed until a RSM instruction is executed. If SMI# is re-asserted prior to execution of a RSM instruction, one and only one SMI# falling edge is stored and then processed after execution of the next RSM. SMI# is ignored following reset and recognition is enabled by setting the USE\_SMI bit in CCRI.

### 3.2.10 Cache Control

The cache control signals (EWBE#, FLUSH#, KEN#, PCD, PWT, WB/WT#) are used to indicate cache status and control caching activity.

**External Write Buffer Empty (EWBE#)** is an active low input driven by the system to indicate when there are no pending write cycles in the external system. The IBM 6x86 CPU samples EWBE# during write cycles (I/O and memory) only. If EWBE# is not asserted, the processor delays all subsequent writes to on-chip cache lines in the “exclusive” or “modified” state until EWBE# is asserted. Regardless of the state of EWBE#, all writes to

the on-chip cache are delayed until any previously issued external write cycle is complete. This ensures that external write cycles occur in program order and is referred to as “strong write ordering”. To enhance performance, “weak write ordering” may be allowed for specific address regions using the Address Region Registers (ARRs) and Region Control Registers (RCRs).

**Cache Flush (FLUSH#)** is a falling edge sensitive input that forces the processor to write-back all dirty data in the cache and then invalidate the entire cache contents. FLUSH# need only be asserted for a single clock but must meet specified setup and hold times to guarantee recognition at a particular clock edge.

Once FLUSH# is sampled active, the IBM 6x86™ CPU begins the cache flush sequence after completion of the current instruction. External interrupts and additional FLUSH# requests are ignored while the cache flush is in progress. However, cache inquiry cycles are permitted during the flush sequence. The IBM 6x86™ CPU issues a flush acknowledge special cycle to indicate completion of the flush sequence. If the processor is in a halt or shutdown state, FLUSH# is recognized and the IBM 6x86 CPU returns to the halt or shutdown state following completion of the flush sequence. If FLUSH# is active at the falling edge of RESET, the processor enters three state test mode.

**Cache Enable (KEN#)** is an active low input which indicates that the data being returned during the current cycle is cacheable. When the IBM 6x86 CPU is performing a cacheable code fetch or memory data read cycle and KEN# is sampled asserted, the cycle is transformed into

a cache line fill (4 transfer burst cycle) or a “1+4” cache line fill. KEN# is sampled with the first asserted BRDY# or NA# for the cycle. I/O accesses, locked reads, system management memory accesses and interrupt acknowledge cycles are never cached.

**Page Cache Disable (PCD)** is an active high output that reflects the state of the PCD page attribute bit in the page table entry or the directory table entry. If paging is disabled or for cycles that are not paged, the PCD pin is driven low. PCD is masked by the cache disable (CD) bit in CR0 (driven high if CD=1) and floats during bus hold states.

**Page Write Through (PWT)** is an active high output that reflects the state of the PWT page attribute bit in the page table entry or the directory table entry. If paging is disabled or for cycles that are not paged, the PWT pin is driven low. If PWT is asserted, PWT takes priority over the WB/WT# input. If PWT is asserted for either reads or writes, the cache line is saved in, or remains in, the shared (write-through) state. PWT floats during bus hold states.

The **Write-Back/Write-Through (WB/WT#)** input allows the system to define the write policy of the on-chip cache on a line-by-line basis. If WB/WT# is sampled high during a line fill cycle and PWT is low, the line is defined as write-back and is stored in the exclusive state. If WB/WT# is sampled high during a write to a write-through cache line (shared state) and PWT is low, the line is transitioned to write-back (exclusive state). If WB/WT# is sampled low or PWT is high, the line is defined as write-through and is stored in (line fill), or remains in (write), the shared state. Table 3-7 (Page 3-16) lists the effects of WB/WT# on the state of the cache line for various bus cycles.



**Table 3-7. Effects of WB/WT# on Cache Line State**

BUS CYCLE TYPE	PWT	WB/WT#	WRITE POLICY	MESI STATE
Line Fill	0	0	Write-through	Shared
Line Fill	0	1	Write-back	Exclusive
Line Fill	1	x	Write-through	Shared
Memory Write (Note)	0	0	Write-through	Shared
Memory Write (Note)	0	1	Write-back	Exclusive
Memory Write (Note)	1	x	Write-through	Shared

Note: Only applies to memory writes to addresses that are currently valid in the cache.

### 3.2.11 Bus Arbitration

The bus arbitration signals (BOFF#, BREQ, HOLD, and HLDA) allow the IBM 6x86 CPU to relinquish control of its local bus when requested by another bus master device. Once the processor has released its bus, the bus master device can then drive the local bus signals.

**Back-Off (BOFF#)** is an active low input that forces the IBM 6x86 CPU to abort the current bus cycle and relinquish control of the CPU's local bus in the next clock. The IBM 6x86 CPU responds to BOFF# by entering the bus hold state as listed in Table 3-8 (Page 3-17). The IBM 6x86 CPU remains in bus hold until BOFF# is negated. Once BOFF# is negated, the IBM 6x86 CPU restarts any aborted bus cycle in its entirety. Any data returned to the IBM 6x86 CPU while BOFF# is asserted is ignored. If BOFF# is asserted in the same clock that ADS# is asserted, the IBM 6x86™ CPU may float ADS# while in the active low state.

**Bus Request (BREQ)** is an active high output asserted by the IBM 6x86 CPU whenever a bus cycle is pending internally. The IBM 6x86 CPU always asserts BREQ in the first clock of a bus cycle with ADS# as well as during bus hold and address hold states if a bus cycle is pending. If no additional bus cycles are pending, BREQ is negated prior to termination of the current cycle.

**Bus Hold Request (HOLD)** is an active high input used to indicate that another bus master requests control of the CPU's local bus. After recognizing the HOLD request and completing the current bus cycle or sequence of locked bus cycles, the IBM 6x86 CPU responds by floating the local bus and asserting the hold acknowledge (HLDA) output. The bus remains granted to the requesting bus master until HOLD is negated. Once HOLD is sampled negated, the IBM 6x86 CPU simultaneously drives the local bus and negates HLDA.

**Hold Acknowledge (HLDA)** is an active high output used to indicate that the IBM 6x86 CPU has responded to the HOLD input and has relinquished control of its local bus. Table 3-8 (Page 3-17) lists the state of all the IBM 6x86 CPU signals during a bus hold state. The IBM 6x86 CPU continues to operate during bus hold states as long as the on-chip cache can satisfy bus requests. HLDA is asserted until HOLD is negated. Once HOLD is sampled negated, the IBM 6x86 CPU simultaneously drives the local bus and negates HLDA.

Table 3-8. Signal States During Bus Hold

SIGNAL LINE	STATE	SIGNAL LINE	STATE
A20M#	Recognized internally	INTR	Recognized internally
A31-A3	Float	INV	Recognized
ADS#	Float	KEN#	Ignored
ADSC#	Float	LBA#	Float
AHOLD	Ignored	LOCK#	Float
AP	Float	M/IO#	Float
APCHK#	Driven	NA#	Ignored
BE7#-BE0#	Float	NMI	Recognized internally
BHOLD	Ignored	PCD	Float
BOFF#	Recognized	PCHK#	Driven
BRDY#	Ignored	PWT	Float
BRDYC#	Ignored	QDUMP#	Recognized
BREQ	Driven	RESET	Recognized
CACHE#	Float	SCYC	Float
D/C#	Float	SMI#	Recognized
D63-D0	Float	SMIACT#	Driven
DHOLD	Ignored	SUSP#	Recognized
DP7-DP0	Float	SUSPA#	Driven
EADS#	Recognized	TCK	Recognized
EWBE#	Recognized internally	TDI	Recognized
FERR#	Driven	TDO	Responds to TCK, TDI, TMS, TRST#
FLUSH#	Recognized	TMS	Recognized
HIT#	Driven	TRST#	Recognized
HITM#	Driven	W/R#	Float
HLDA	Responds to HOLD	WB/WT#	Ignored
HOLD	Recognized	WM_RST	Recognized
IGNNE#	Recognized internally		



### 3.2.12 Cache Coherency

The cache coherency signals (AHOLD, EADS#, HIT#, HITM#, and INV) are used to initiate and monitor cache inquiry cycles. These signals are intended to be used to ensure cache coherency in a uni-processor environment only. Contact IBM for additional specifications on maintaining coherency in a multi-processor environment.

**Address Hold Request (AHOLD)** is an active high input which forces the IBM 6x86 CPU to float A31-A3 and AP in the next clock cycle. While AHOLD is asserted, only the address bus is disabled. The current bus cycle remains active and can be completed in the normal fashion. The IBM 6x86 CPU does not generate additional bus cycles while AHOLD is asserted except write-back cycles in response to a cache inquiry cycle.

**External Address Strobe (EADS#)** is an active low input used to indicate to the IBM 6x86 CPU that a valid cache inquiry address is being driven on the IBM 6x86 CPU address bus (A31-A5) and AP. The IBM 6x86 CPU checks the on-chip cache for this address. If the address is present in the cache the HIT# signal is asserted. If the data associated with the inquiry address is “dirty” (modified state), the HITM# signal is also asserted. If dirty data exists, a write-back cycle is issued to update external memory with the dirty data. Additional cache inquiry cycles are ignored while HITM# is asserted.

The state of the INV pin at the time EADS# is sampled active determines the final state of the cache line. If INV is sampled high, the final state of the cache line is “invalid”. If INV is

sampled low, the final state of the cache line is “shared”. A cache inquiry cycle using EADS# may be run while the IBM 6x86 CPU is in either an address hold or bus hold state. The inquiry address must be driven by an external device.

**Hit on Cache Line (HIT#)** is an active low output used to indicate that the current cache inquiry address has been found in the cache (modified, exclusive or shared states). HIT# is valid two clocks after EADS# is sampled active, and remains valid until the next cache inquiry cycle.

**Hit on Modified Data (HITM#)** is an active low output used to indicate that the current cache inquiry address has been found in the cache and dirty data exists in the cache line (modified state). If HITM# is asserted, a write-back cycle is issued to update external memory. HITM# is valid two clocks after EADS# is sampled active, and remains asserted until two clocks after the last BRDY# of the write-back cycle is sampled active. The IBM 6x86 CPU does not accept additional cache inquiry cycles while HITM# is asserted.

**Invalidate Request (INV)** is an active high input used to determine the final state of the cache line in the case of a cache inquiry hit. INV is sampled with EADS#. A logic one on INV directs the processor to change the state of the cache line to “invalid”. A logic zero on INV directs the processor to change the state of the cache line to “shared”.

### 3.2.13 FPU Error Interface

The FPU interface signals FERR# and IGNNE# are used to control error reporting for the on-chip floating point unit. These signals are typically used for a PC-compatible system implementation. For other applications, FPU errors are reported to the IBM 6x86 CPU core through an internal interface.

**Floating Point Error Status (FERR#)** is an active low output asserted by the IBM 6x86 CPU when an unmasked floating point error occurs. FERR# is asserted during execution of the FPU instruction that caused the error. FERR# does not float during bus hold states.

**Ignore Numeric Error (IGNNE#)** is an active low input which forces the IBM 6x86 CPU to ignore any pending unmasked FPU errors and allows continued execution of floating point instructions. When IGNNE# is not asserted and an unmasked FPU error is pending, the IBM 6x86 CPU only executes the following floating point instructions: FNCLEX, FNINIT, FNSAVE, FNSTCW, FNSTENV, and FNSTSW#. IGNNE# is ignored when the NE bit in CR0 is set to a 1.

### 3.2.14 Scatter/Gather Buffer Interface

The scatter/gather buffer interface signals (BHOLD, DHOLD, LBA#, QDUMP#), in conjunction with the byte enables (BE7#-BE0#) and address hold (AHOLD), can be used by the system hardware to transfer data to/from a 32-bit peripheral interface bus. A 64-bit buffer resides in the IBM 6x86 CPU to assist the system in these transfers. This buffer provides scatter/gather capability during four different types of transfers as listed in Table 3-9 (Page 3-20).



**Table 3-9. Scatter/Gather Cycles**

CYCLE TYPE	BHOLD USED	DHOLD USED	QDUMP# USED	DATA BUS TIMING
CPU Write to 32-Bit Bus	x	--	--	Data driven 1 clock after byte enables asserted.
CPU Read from 32-Bit Bus	x	--	--	Data sampled 1 clock after byte enables asserted.
32-Bit Bus Master Write to Memory * (1) Scatter/gather buffer load from 32-bit bus master.	x	x	--	Data sampled 1 clock after byte enables asserted.
(2) Scatter/gather buffer write to memory.	x	--	x	Data driven 1 clock after QDUMP# asserted.
32-Bit Bus Master Read from Memory * (1) Scatter/gather buffer load from memory.	x	x	--	Data sampled 1 clock after byte enables asserted.
(2) Scatter/gather buffer write to 32-bit bus master.	x	--	x	Data driven 1 clock after QDUMP# asserted.

\*Note: Bus master transfers using the scatter/gather buffer must be initiated while the CPU bus is in a bus hold state or an idle state. These cycles cannot occur during CPU initiated bus cycles.

**Byte Enable Hold (BHOLD)** is an active high input that causes the IBM 6x86 CPU to float the byte enable outputs (BE7#-BE0#) in the next clock. While BHOLD is asserted, only the byte enables are disabled. The current bus cycle remains active and can be completed in the normal fashion. The IBM 6x86 CPU continues to generate additional bus cycles while BHOLD is asserted, so BHOLD should only be asserted while AHOLD is asserted.

BHOLD is asserted by the external system during scatter/gather buffer cycles. While BHOLD is asserted, the byte enables are driven by an external source and indicate which bytes of the data bus should be loaded into/written out of the scatter/gather buffer. The IBM 6x86 CPU samples the byte enables at each rising clock edge while BHOLD is asserted. Table 3-10 (Page 3-21) lists the byte enable mappings for the scatter/gather cycles.

Table 3-10. Byte Enable Map for Scatter/Gather Cycles

CYCLE TYPE	BE7-BE0#	SOURCE	DESTINATION
CPU Read from 32-Bit Bus	F F	CPU Data Bus	Scatter/Gather Buffer
	F x	No Transfer	No Transfer
	x F	31-0	31-0
	x x	63-0	63-0
CPU Write to 32-Bit Bus*	F F	Scatter/Gather Buffer	CPU Data Bus
	F x	No Transfer	No Transfer
	x F	31-0	31-0
	x x	63-0	63-0
Scatter/Gather Buffer Load for 32-Bit Bus Master	F F	CPU Data Bus	Scatter/Gather Buffer
	F x	No Transfer	No Transfer
	x F	31-0	31-0
	x x	63-0	63-0
Scatter/Gather Buffer Dump using QDUMP#	F F	Scatter/Gather Buffer	CPU Data Bus
	F x	No Transfer	No Transfer
	x F	31-0	31-0
	x x	63-0	63-0

\*Note: If LBA# is active during a CPU write cycle with BE3-BE0# inactive, the IBM 6x86 CPU automatically maps the upper dword of data (D63-D32) to the lower dword of the data bus (D31-D0).

**Data Bus Hold (DHOLD)** is an active high input that forces the IBM 6x86 CPU to float the data bus lines (D63-D0) and the data parity lines (DP7-DP0) in the next clock. While DHOLD is asserted, only the data and data parity buses are disabled. The current bus cycle remains active and is completed in the normal fashion in response to BRDY#. The IBM 6x86 CPU generates additional bus cycles while DHOLD is asserted. To avoid writing invalid data, during a write cycle, DHOLD and BRDY# should not be asserted at the same time,

The external system asserts DHOLD during scatter/gather buffer load cycles

when the IBM 6x86 CPU is not the bus master. While DHOLD is asserted, the data bus is driven by an external source and the information is loaded into the scatter/gather buffer based on the state of the byte enables (BHOLD asserted). The data bus is sampled one clock after the clock edge at which an active byte enable is sampled.

**Local Bus Access (LBA#)** is an active low output asserted by the IBM 6x86 CPU for any I/O bus cycle or for any bus access that resides within a "local bus" address region as specified by the on-chip configuration registers. LBA# is asserted during



the clock that ADS# is asserted and remains asserted for only one clock. LBA# is used to indicate a cycle intended to address a device using the 32-bit peripheral bus. If LBA# is active during a CPU write cycle with BE(3-0)# inactive, the IBM 6x86 CPU automatically maps the upper dword of data to the lower dword of the data bus.

Q Buffer Dump (QDUMP#) is an active low input asserted by the external system to dump the contents of the scatter/gather buffer to the data bus. The data bytes specified by the asserted byte enables are driven onto the data bus during the clock after QDUMP# is sampled asserted. QDUMP# must be asserted at the falling edge of RESET to enable the scatter/gather interface pins.

### 3.2.15 Power Management Interface

The two power management signals (SUSP#, SUSPA#) allow the IBM 6x86 CPU to enter and exit suspend mode. The IBM 6x86 CPU also enters suspend mode as the result of executing a HALT instruction if the HALT bit is set in CCR2. Suspend mode circuitry forces the IBM 6x86 CPU to consume minimal power while maintaining the entire internal CPU state.

**Suspend Request (SUSP#)** is an active low input which requests that the IBM 6x86 CPU enter suspend mode. After recognition of an active SUSP# input, the IBM 6x86 CPU completes execution of the current instruction, any pending decoded instructions and associated bus cycles, issues a stop grant bus cycle, and then asserts the SUSPA# output. SUSP# is

ignored following RESET and is enabled by setting the SUSP bit in CCR2.

The **Suspend Acknowledge (SUSPA#)** output indicates that the IBM 6x86 CPU has entered low-power suspend mode as the result of either assertion of SUSP# or execution of a HALT instruction. SUSPA# remains asserted until SUSP# is negated, or until an interrupt is serviced if suspend mode was entered via the HALT instruction. If SUSP# is asserted and then negated prior to SUSPA# assertion, SUSPA# may toggle state after SUSP# negates.

The IBM 6x86 CPU accepts cache flush requests and cache inquiry cycles while SUSPA# is asserted. If FLUSH# is asserted, the CPU exits the low power state and services the flush request. After completion of all required write-back cycles, the CPU returns to the low power state. SUSPA# negates during the write-back cycles. Before issuing the write-back cycle, the CPU may execute several code fetches.

If AHOLD, BOFF# or HOLD is asserted while SUSPA# is asserted, the CPU exits the low power state in preparation for a cache inquiry cycle. After completion of any required write-back cycles resulting from the cache inquiry, the CPU returns to the low power state only if HOLD, BOFF# and AHOLD are negated. SUSPA# negates during the write-back cycle.

Table 3-11 (Page 3-23) lists the IBM 6x86 CPU signal states for suspend mode when initiated by either SUSP# or the HALT instruction. SUSPA# is disabled (three-state) following RESET and is enabled by setting the SUSP bit in CCR2.

Table 3-11. Signal States During Suspend Mode

SIGNAL LINE	SUSP# INITIATED/ HALT INITIATED	SIGNAL LINE	SUSP# INITIATED/ HALT INITIATED
A20M#	Ignored	INTR	Latched/Recognized
A31-A3	Driven	INV	Recognized
ADS#	1	KEN#	Ignored
ADSC#	1	LBA#	1
AHOLD	Recognized	LOCK#	1
AP	Driven	M/IO#	Driven
APCHK#	1	NA#	Ignored
BE7#-BE0#	Driven	NMI	Latched/Recognized
BHOLD	Ignored	PCD	Driven
BOFF#	Recognized	PCHK#	1
BRDY#	Ignored	PWT	Driven
BRDYC#	Ignored	QDUMP#	Ignored
BREQ	0	RESET	Recognized
CACHE#	Driven	SCYC	Driven
D/C#	Driven	SMI#	Latched/Recognized
D63-D0	Float	SMIACT#	1
DHOLD	Ignored	SUSP#	0 / Recognized
DP7-DP0	Float	SUSPA#	0
EADS#	Recognized	TCK	Recognized
EWBE#	Ignored	TDI	Recognized
FERR#	1	TDO	Responds to TCK, TDI, TMS, TRST#
FLUSH#	Recognized	TMS	Recognized
HIT#	Driven	TRST#	Recognized
HITM#	1	W/R#	Driven
HLDA	Driven in response to HOLD	WB/WT#	Ignored
HOLD	Recognized	WM_RST	Latched/Recognized
IGNNE#	Ignored		



### 3.2.16 JTAG Interface

The IBM 6x86 CPU can be tested using JTAG Interface (IEEE Std. 1149.1) boundary scan test logic. The IBM 6x86 CPU pin state can be set according to serial data supplied to the chip. The IBM 6x86 CPU pin state can also be recorded and supplied as serial data.

**Test Clock (TCK)** is the clock input used by the IBM 6x86 CPU boundary scan (JTAG) test logic. The rising edge of TCK is used to clock control and data information into the IBM 6x86 CPU using the TMS and TDI pins. The falling edge of TCK is used to clock data information out of the IBM 6x86 CPU using the TDO pin.

**Test Data Input (TDI)** is the serial data input used by the IBM 6x86 CPU boundary scan (JTAG) test logic. TDI is sampled on the rising edge of TCK.

**Test Data Output (TDO)** is the serial data output used by the IBM 6x86 CPU boundary scan (JTAG) test logic. TDO is output on the falling edge of TCK.

**Test Mode Select (TMS)** is the control input used by the IBM 6x86 CPU boundary scan (JTAG) test logic. TMS is sampled on the rising edge of TCK.

**Test Reset (TRST#)** is an active low input used to initialize the IBM 6x86 CPU boundary scan (JTAG) test logic.



NOTICE TO CUSTOMERS: Some of the information contained in this document was obtained through a third party and IBM has not conducted independent tests of all product characteristics contained herein. The product described in this document is sold under IBM's standard warranty.

The information contained in this document is subject to change without notice. The products described in this document are NOT intended for use in implantation or other life support applications where malfunction may result in injury or death to persons. The information contained in this document does not effect or change IBM's product specifications or warranties. Nothing in this document shall operate as an express or implied license or indemnity under the intellectual property rights of IBM or third parties. All the information contained in this document was obtained in specific environments, and is presented as an illustration. The results obtained in other operating environments may vary.

**THE INFORMATION CONTAINED IN THIS DOCUMENT IS PROVIDED ON AN "AS IS" BASIS. In no event will IBM be liable for any damages arising directly or indirectly from any use of the information contained in this document.**

© International Business Machines Corporation 1996.  
Printed in the United States of America  
2-96

All Rights Reserved

© Cyrix Corporation 1996.  
© IBM and the IBM logo are registered trademarks of the IBM Corporation.  
© Cyrix is a registered trademark of the Cyrix Corporation.  
IBM Microelectronics is a trademark of the IBM Corporation.  
6x86 is a trademark of Cyrix Corporation

Other company, product, and service names, which may be denoted by a double asterisk (\*\*), may be trademarks of service marks of others.

Product names used in this publication are for identification purposes only and may be trademarks of their respective companies.

IBM Corporation  
1000 River Street  
Essex Junction, Vermont 05452-4299  
United States of America