# 8087

Numeric Data Coprocessor
iAPX86 Family

## DISTINCTIVE CHARACTERISTICS

- High performance arithmetic and transcendental functions in hardware
- Supports 8-, 16-, 32-, 64-bit integer
- Performs 32-, 64-, 80-bit floating point calculations conforming to IEEE standard
- Standard 8086 instruction set and addressing modes
- Built-in exception handling functions
- Multibus* system compatible

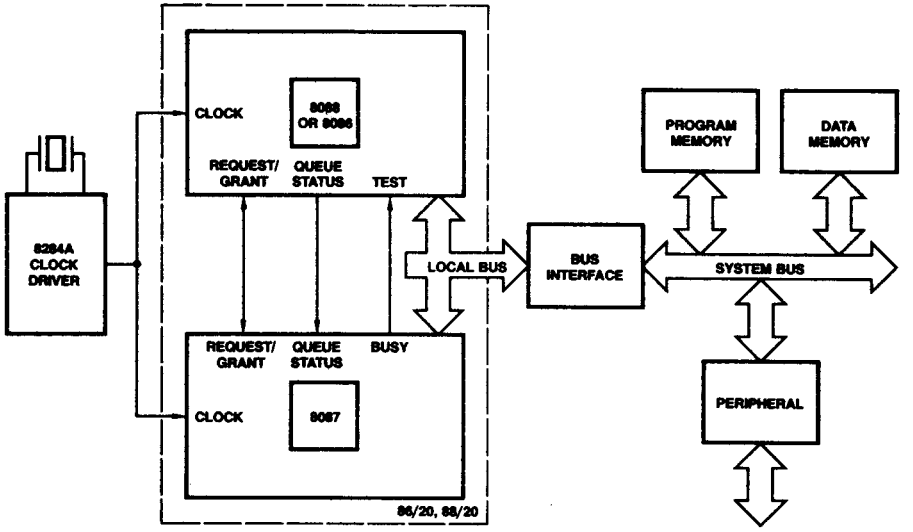
## GENERAL DESCRIPTION

The 8087 is designed to do high performance numeric processing in hardware. It operates as the coprocessor to an 8086 or 8088 CPU and can improve numeric throughput by a factor of 100 over the stand-alone CPU. It is programmed with the same instruction set as the 8086/88.

The 8087 does trigonometric, logarithmic, and exponential functions, which are essential in many scientific and military applications. The 8087 can also process BCD numbers up to 18 digits with no round-off error.

The 8087 is built in N-channel depletion load technology in a 40-pin package.

## BLOCK DIAGRAM




BD003720

*Multibus is a registered trademark of Intel Corporation.



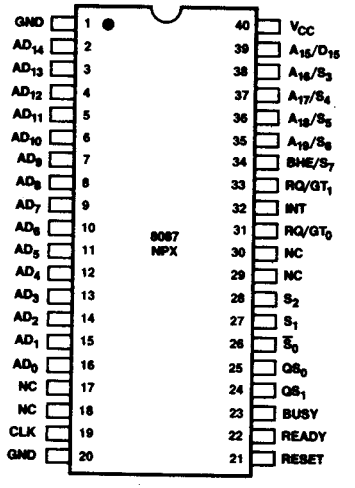
02037B



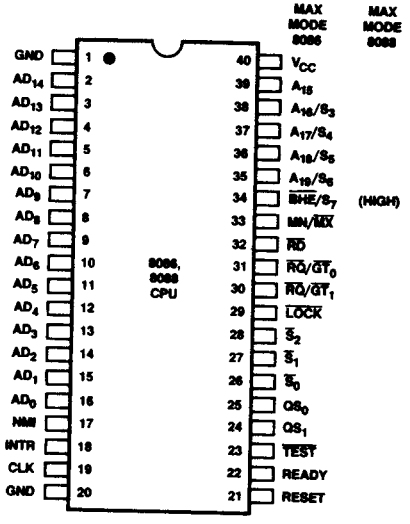
Refer to page 7-1 for Essential Information on Military Devices

## CONNECTION DIAGRAM
## Top View

**D-40**




CD005490

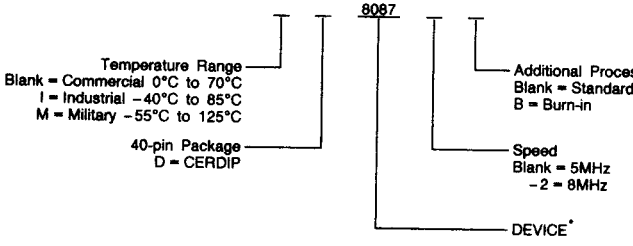



CD005500

Note: Pin 1 is marked for orientation

## ORDERING INFORMATION

AMD products are available in several packages and operating ranges. The order number is formed by a combination of the following: Device number, speed option (if applicable), package type, operating range and screening option (if desired).




**Valid Combinations**

Consult the local AMD sales office to confirm availability of specific valid combinations, check for newly released valid combinations and/or obtain additional data on AMD's standard military grade product.

*A "C" in the middle of the device type denotes CMOS version of the product.

## PIN DESCRIPTION

| Pin No. | Name | I/O | Description |
|---|---|---|---|
| 39, 2-16 | $AD_{15}$-$AD_0$ | I/O | Address Data. These lines constitute the time multiplexed memory address ($T_1$) and data ($T_2$, $T_3$, $T_W$, $T_4$) bus. $A_0$ is analogous to $\overline{BHE}$ for the lower byte of the data bus, pins $D_7$-$D_0$. It is LOW during $T_1$ when a byte is to be transferred on the lower portion of the bus in memory operations. Eight-bit oriented devices tied to the lower half of the bus would normally use $A_0$ to condition chip select functions. These lines are active HIGH. They are input/output lines for 8087 driven bus cycles and are inputs which the 8087 monitors when the 8086/8088 is in control of the bus. |
| 35, 36, 37, 38 | $A_{19}/S_6$, $A_{18}/S_5$, $A_{17}/S_4$, $A_{16}/S_3$ | I/O | Address Memory. During $T_1$ these are the four most significant address lines for memory operations. During memory operations, status information is available on these lines during $T_2$, $T_3$, $T_W$, and $T_4$. For 8087 controlled bus cycles, $S_6$, $S_4$, and $S_3$ are reserved and currently one (HIGH), while $S_5$ is always LOW. These lines are inputs which the 8087 monitors when the 8086/8088 is in control of the bus. |
| 34 | $\overline{BHE}/S_7$ | I/O | Bus High Enable. During $T_1$ the bus high enable signal ($\overline{BHE}$) should be used to enable data onto the most significant half of the data bus, pins $D_{15}$-$D_8$. Eight-bit oriented devices tied to the upper half of the bus would normally use $\overline{BHE}$ to condition chip select functions. $\overline{BHE}$ is LOW during $T_1$ for read and write cycles when a byte is to be transferred on the high portion of the bus. The $S_7$ status information is available during $T_2$, $T_3$, $T_W$, and $T_4$. The signal is active LOW. $S_7$ is an input which the 8087 monitors during 8086/8088 controlled bus cycles. |
| 26, 27, 28 | $S_0$, $S_1$, $S_2$ | I/O | Status. For 8087 driven bus cycles, these status lines are encoded as follows:<br>**Table 1.**<br>$\overline{S}_2$ / $\overline{S}_1$ / $\overline{S}_0$<br>0 (LOW) X X Unused<br>1 (HIGH) 0 0 Unused<br>1 0 1 Read Memory<br>1 1 0 Write Memory<br>1 1 1 Passive<br>Status is driven active during $T_4$, remains valid during $T_1$ and $T_2$, and is returned to the passive state (1, 1, 1) during $T_3$ or during $T_W$ when READY is HIGH. This status is used by the 8288 Bus Controller to generate all memory access control signals. Any change in $\overline{S}_2$, $\overline{S}_1$, or $\overline{S}_0$ during $T_4$ is used to indicate the beginning of a bus cycle, and the return to the passive state in $T_3$ or $T_W$ is used to indicate the end of a bus cycle. These signals are monitored by the 8087 when the 8086/8088 is in control of the bus. |
| 31 | $\overline{RQ}/\overline{GT}_0$ | I/O | Request/Grant. This request/grant pin is used by the NDP to gain control of the local bus from the CPU for operand transfers or on behalf of another bus master. It must be connected to one of the two processor request/grant pins. The request/grant sequence on this pin is as follows:<br>1. A pulse one clock wide is passed to the CPU to indicate a local bus request by either the 8087 or the master connected to the 8087 $\overline{RQ}/\overline{GT}_1$ pin.<br>2. The NDP waits for the grant pulse and when it is received will either initiate bus transfer activity in the clock cycle following the grant or pass the grant out on the $\overline{RQ}/\overline{GT}_1$ pin in this clock if the initial request was for another bus master.<br>3. The 8087 will generate a release pulse to the CPU one clock cycle after the completion of the last NDP bus cycle or on receipt of the release pulse from the bus master on $\overline{RQ}/\overline{GT}_1$. |
| 30 | $\overline{RQ}/\overline{GT}_1$ | I/O | Request/Grant. This request/grant pin is used by another local bus master to force the NDP to release the local bus at the end of the processor's current bus cycle. If the NDP is not in control of the bus when the request is made, the request/grant sequence is passed through the NDP on the $\overline{RQ}/\overline{GT}_0$ pin one cycle later. Subsequent grant and release pulses are also passed through the NDP with a two and one clock delay, respectively, for resynchronization. $\overline{RQ}/\overline{GT}_1$ has an internal pull-up resistor and may be left unconnected. If the NDP has control of the bus, the request/grant sequence is as follows:<br>1. A pulse 1 CLK wide from another local bus master indicates a local bus request to the 8087 (pulse 1).<br>2. During the NDP's next $T_4$ or $T_1$, a pulse 1 CLK wide from the 8087 to the requesting master (pulse 2) indicates that the 8087 has allowed the local bus to float and that it will enter the "RQ/GT acknowledge" state at the next CLK. The NDP's control unit is disconnected logically from the local bus during "RQ/GT acknowledge."<br>3. A pulse 1 CLK wide from the requesting master indicates to the 8087 (pulse 3) that the "RQ/GT" request is about to end and that the 8087 can reclaim the local bus at the next CLK.<br>Each master-master exchange of the local bus is a sequence of 3 pulses. There must be one dead CLK cycle after each bus exchange. Pulses are active LOW. |
| 24, 25 | $QS_1$, $QS_0$ | I | $QS_1$, $QS_0$. $QS_1$ and $QS_0$ provide the 8087 with status to allow tracking of the CPU instruction queue.<br>$QS_1$ / $QS_0$<br>0 (LOW) 0 No Operation<br>0 1 First Byte of Op Code from Queue<br>1 (HIGH) 0 Empty the Queue<br>1 1 Subsequent Byte from Queue |
| 32 | INT | O | Interrupt. This line is used to indicate that an unmasked exception has occurred during numeric instruction execution when 8087 interrupts are enabled. This signal is typically routed to an 8259A. INT is active HIGH. |
| 23 | BUSY | O | Busy. This signal indicates that the 8087 NEU is executing a numeric instruction. It is connected to the CPU's $\overline{TEST}$ pin to provide CPU-NDP synchronization. In the case of an unmasked exception, BUSY remains active until the exception is cleared. BUSY is active HIGH. |
| 22 | READY | I | Ready. READY is the acknowledgment from the addressed memory device that it will complete the data transfer. The RDY signal from memory is synchronized by the 8284A Clock Generator to form READY. This signal is active HIGH. |
| 21 | RESET | I | Reset. RESET causes the processor to immediately terminate its present activity. The signal must be active HIGH for at least four clock cycles. RESET is internally synchronized. |
| 19 | CLK | I | Clock. The clock provides the basic timing for the processor and bus controller. It is asymmetric with a 33% duty cycle to provide optimized internal timing. |
| 40 | $V_{CC}$ | | Power. $V_{CC}$ is the +5V power supply pin. |
| 1, 20 | GND | | Ground. GND are the ground pins. |

## DETAILED DESCRIPTION

The 8087 is a numeric processor extension that provides arithmetic and logical instruction support for a variety of numeric data types. It also executes numerous built-in transcendental functions (e.g., tangent and log functions). The 8087 executes instructions as a coprocessor to a maximum mode 8086 or 8088. Figure 3 presents the registers of the 8087 plus CPU combination. Table 2 shows the range of data types supported by the NDP. The 8087 is treated as an extension to the CPU, providing register, data types, control, and instruction capabilities at the hardware level. At the programmer's level, the CPU and NDP is viewed as a single unified processor.

### System Configuration

As a coprocessor to an 8086 or 8088, the 8087 is wired in parallel with the CPU as shown in Figure 4. The CPU's status ($\overline{S}_0 - \overline{S}_2$) and queue status lines ($QS_0 - QS_1$) enable the 8087 to monitor and decode instructions in synchronization with the CPU and without any CPU overhead. Once started the 8087 can process in parallel with, and independent of, the host CPU. The NPX can interrupt the CPU when it detects an error or exception. The 8087's interrupt request line is typically routed to the CPU through an 8259A Programmable Interrupt Controller.

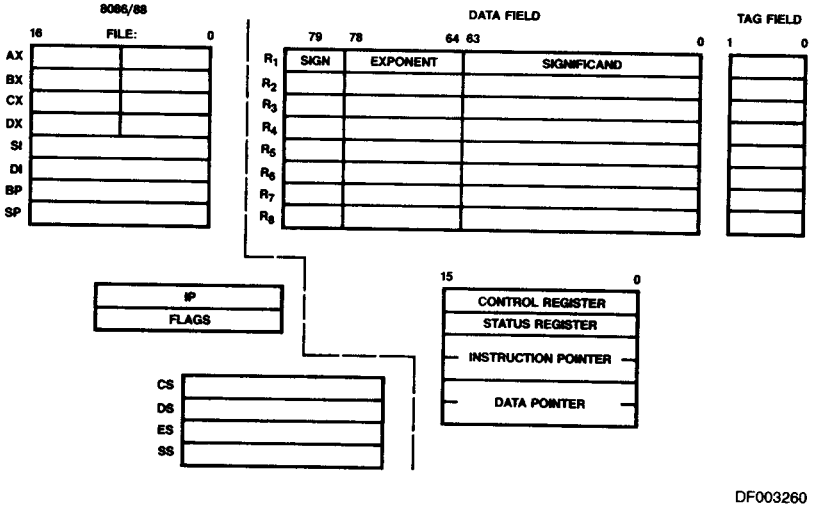



**Figure 3. 8087 Register Architecture**

**Table 2. 8087 Data Types**

| Data Formats | Range | Precision | Most Significant Byte |
|---|---|---|---|
| Byte Integer | $10^2$ | 8 Bits | $I_7$ $I_0$ Two's Complement |
| Word Integer | $10^4$ | 16 Bits | $I_{15}$ $I_0$ Two's Complement |
| Short Integer | $10^9$ | 32 Bits | $I_{31}$ $I_0$ Two's Complement |
| Long Integer | $10^{18}$ | 64 Bits | $I_{63}$ $I_0$ Two's Complement |
| Packed BCD | $10^{18}$ | 18 Digits | S — $D_{17}$ $D_{16}$ $D_1$ $D_0$ |
| Short Real | $10^{\pm 38}$ | 24 Bits | S $E_7$ $E_0$ $F_1$ $F_{23}$ $F_0$ Implicit |
| Long Real | $10^{\pm 308}$ | 53 Bits | S $E_{10}$ $E_0$ $F_1$ $F_{52}$ $F_0$ Implicit |
| Temporary Real | $10^{\pm 4932}$ | 64 Bits | S $E_{14}$ $E_0$ $F_0$ $F_{63}$ |

Integer: 1

Packed BCD: $(-1)^S$ $(D_{17} \ldots D_0)$

Real: $(-1)^S$ $(2^{E\text{-BIAS}})(F_0 \cdot F_1 \ldots)$

Bias = 127 for Short Real
1023 for Long Real
16383 for Temp Real

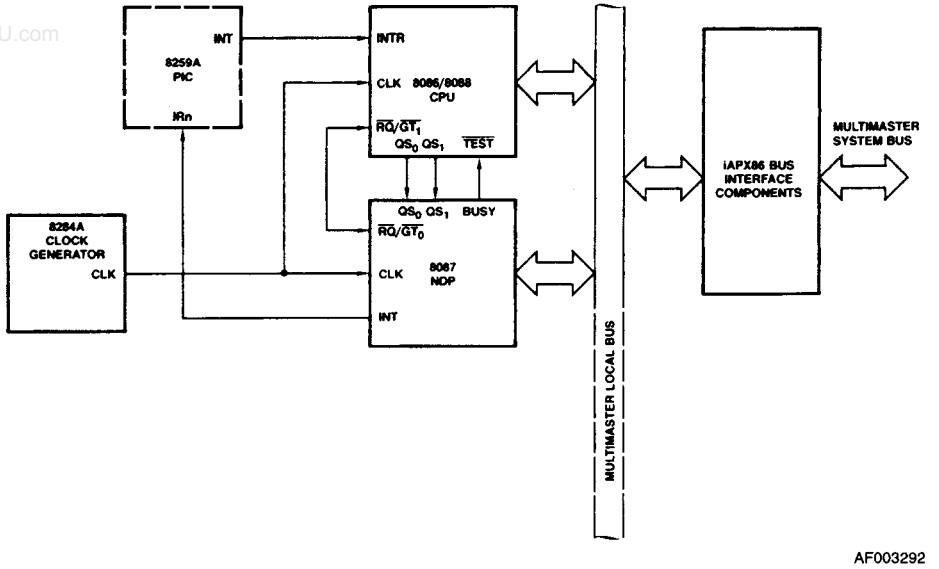



**Figure 4. NDP System Configuration**

The 8087 uses one of the request/grant lines (typically $\overline{RQ}$/$\overline{GT}_1$) to obtain control of the local bus for data transfers. The other request/grant line is available for general system use (for instance by an I/O processor in LOCAL mode). A bus master can also be connected to the 8087's RQ/$GT_1$ line. In this configuration the 8087 will pass the request/grant handshake signals between the CPU and the attached master when the 8087 is not in control of the bus and will relinquish the bus to the master directly when the 8087 is in control. In this way two additional masters can be configured; one will share the 8086 bus with the 8087 on a first come first served basis, and the second will be guaranteed to be higher in priority than the 8087.

As Figure 4 shows, all processors utilize the same clock generator and system bus interface components.

## Bus Operation

The 8087 bus structure, operation and timing are identical to all other processors in the 8086 family. The address is time multiplexed with the data on the first 16/8 lines of the address/data bus. $A_{16}$ through $A_{19}$ are time multiplexed with four status lines $S_3 - S_6$; $S_3$, $S_4$ and $S_6$ are always one (HIGH) for 8087 driven bus cycles while $S_5$ is always zero (LOW). When the 8087 is monitoring CPU bus cycles (passive mode), $S_6$ is also monitored by the 8087 to differentiate 8086/8088 activity from that of a local I/O processor or any other local bus master. (The 8086/8088 must be the only processor on the local bus to drive $S_6$ low.) $S_7$ is multiplexed with and has the same value as $\overline{BHE}$ for all 8087 bus cycles.

The first three status lines, $\overline{S}_0 - \overline{S}_2$, are used with an 8288 bus controller to determine the type of bus cycle being run:

| $\overline{S}_2$ | $\overline{S}_1$ | $\overline{S}_0$ | |
|---|---|---|---|
| 0 | X | X | Unused |
| 1 | 0 | 0 | Unused |
| 1 | 0 | 1 | Memory Data Read |
| 1 | 1 | 0 | Memory Data Write |
| 1 | 1 | 1 | Passive (no bus cycle) |

## Programming Interface

The NDP includes the standard 8086/88 instruction set for general data manipulation and program control. It also includes 68 numeric instructions for extended precision integer, floating point, trigonometric, logarithmic, and exponential functions. Sample execution times for several NDP functions are shown in Figure 4.

Any instruction executed by the NDP is the combined result of the CPU and NPX activity. The CPU and NPX have specialized functions and registers providing fast concurrent operation. The CPU controls overall program execution while the NPX uses the coprocessor interface to recognize and perform numeric operations.

Table 2 lists the eight data types the 8087 supports and presents the format for each type. Internally, the NPX holds all numbers in the temporary real format. Load and store instructions automatically convert operands represented in memory as 16-, 32-, or 64-bit integers, 32- or 64-bit floating point numbers, or 18-digit packed BCD numbers into temporary real format and vice versa. The NDP also provides the capability to control round off, underflow, and overflow errors in each calculation.

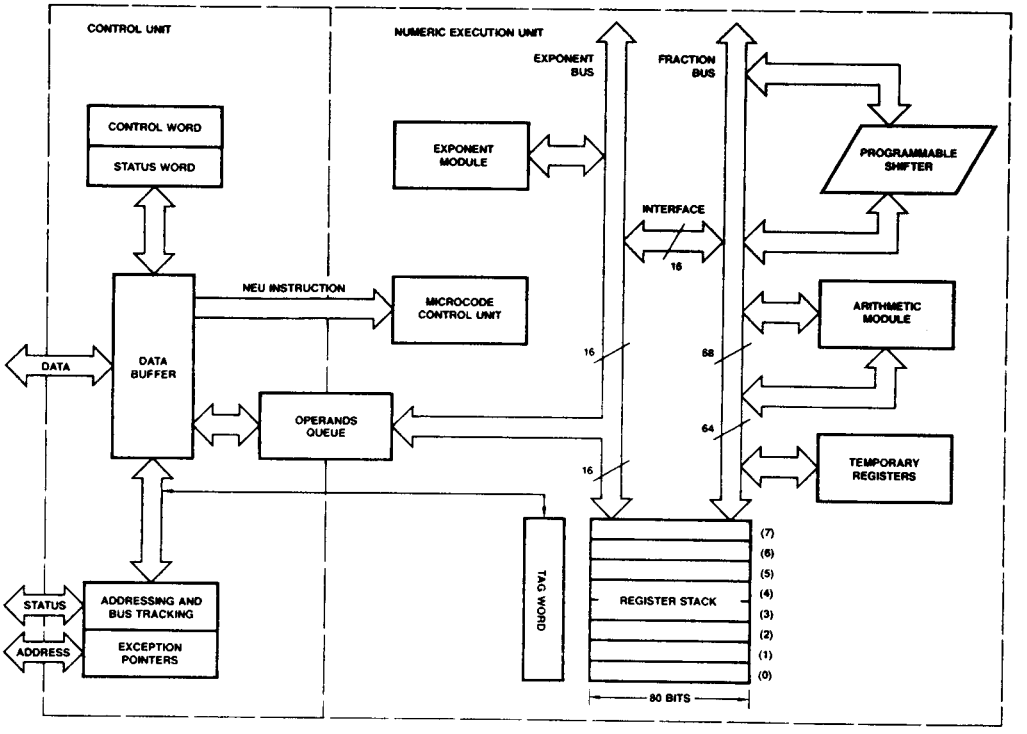
Computations in the NPX use the processor's register stack. These eight 80-bit registers provide the equivalent capacity of 20 32-bit registers. The NPX register set can be accessed as a stack, with instructions operating on the top one or two stack elements, or as a fixed register set, with instructions operating on explicitly designated registers.

All 8087 instructions appear as ESCAPE instructions to the host CPU. Assembly language programs are written in ASM-86, the 8086/88 assembly language. Table 3 gives the execution times of some typical numeric instructions.

## Numeric Processor Extension Architecture

As shown in Figure 5, the 8087 is internally divided into two processing elements, the control unit (CU) and the numeric

execution unit (NEU). The NEU executes all numeric instructions, while the CU receives and decodes instructions, reads and writes memory operands, and executes NPX control instructions. The two elements are able to operate independently of one another, allowing the CU to maintain synchronization with the CPU while the NEU is busy processing a numeric instruction.




BD003730

**Figure 5. 8087 Black Diagram**

**TABLE 3. EXECUTION TIME FOR SELECTED 8087 NUMERIC INSTRUCTIONS AND CORRESPONDING 8086 EMULATION**

| Floating Point Instruction | Approximate Execution Time ($\mu$s) | |
|---|---|---|
| | 8087 (5MHz Clock) | 8086 Emulation |
| Add/Subtract Magnitude | 14/18 | 1,600 |
| Multiply (single precision) | 19 | 1,600 |
| Multiply (extended precision) | 27 | 2,100 |
| Divide | 39 | 3,200 |
| Compare | 9 | 1,300 |
| Load (double percision) | 10 | 1,700 |
| Store (double percision) | 21 | 1,200 |
| Square Root | 36 | 19,600 |
| Tangent | 90 | 13,000 |
| Exponentiation | 100 | 17,000 |

## Control Unit

The CU keeps the 8087 operating in synchronization with its host CPU. 8087 instructions are intermixed with CPU instructions in a single instruction stream. The CPU fetches all instructions from memory; by monitoring the status signals ($\overline{S}_0 - \overline{S}_2$, $S_6$) emitted by the CPU, the NPX control unit determines when an 8086 instruction is being fetched. The CU monitors the Data bus in parallel with the CPU to obtain instructions that pertain to the 8087.

The CU maintains an instruction queue that is identical to the queue in the host CPU. The CU automatically determines if the CPU is an 8086 or an 8088 immediately after reset (by monitoring the $\overline{BHE}/S_7$ line) and matches its queue length accordingly. By monitoring the CPU's queue status lines ($QS_0$, $QS_1$), the CU obtains and decodes instructions from the queue in synchronization with the CPU.

A numeric instruction appears as an ESCAPE instruction to the 8086 or 8088 CPU. Both the CPU and NPX decode and execute the ESCAPE instruction together. The 8087 only recognizes the numeric instructions shown in Table 5. The start of a numeric operation is accomplished when the CPU executes the ESCAPE instruction. The instruction may or may not identify a memory operand.

The CPU does, however, distinguish between ESC instructions that reference memory and those that do not. If the instruction refers to a memory operand, the CPU calculates

the operand's address using any one of its available addressing modes, and then performs a "dummy read" of the word at that location. (Any location within the 1M byte address space is allowed.) This is a normal read cycle except that the CPU ignores the data it receives. If the ESC instruction does not contain a memory reference (e.g., an 8087 stack operation), the CPU simply proceeds to the next instruction.

An 8087 instruction can have one of three memory reference options: (1) not reference memory; (2) load an operand from memory into the 8087; or (3) store an operand from the 8087 into memory. If no memory reference is required, the 8087 simply executes its instruction. If a memory reference is required, the CU uses a "dummy read" cycle initiated by the CPU to capture and save the address that the CPU places on the bus. If the instruction is a load, the CU additionally captures the data word when it becomes available on the local data bus. If data required is longer than one word, the CU immediately obtains the bus from the CPU using the request/grant protocol and reads the rest of the information in consecutive bus cycles. In a store operation, the CU captures and saves the store address as in a load and ignores the data word that follows in the "dummy read" cycle. When the 8087 is ready to perform the store, the CU obtains the bus from the CPU and writes the operand starting at the specified address.

### Numeric Execution Unit

The NEU executes all instructions that involve the register stack; these include arithmetic, logical, transcendental, constant and data transfer instructions. The data path in the NEU is 84 bits wide (68 fraction bits, 15 exponent bits and a sign bit) which allows internal operand transfers to be performed at very high speeds.

When the NEU begins executing an instruction, it activates the 8087 BUSY signal. This signal can be used in conjunction with the CPU WAIT instruction to resynchronize both processors when the NEU has completed its current instruction.

### Register Set

The 8087 register set is shown in Figure 3. Each of the eight data registers in the 8087's register stack is 80 bits wide and is divided into "fields" corresponding to the NDP's temporary real data type.

At a given point in time, the TOP field in the control word identifies the current top-of-stack register. A "push" operation decrements TOP by 1 and loads a value into the new top register. A "pop" operation stores the value from the current top register and then increments TOP by 1. The 8087 register stack grows "down" toward lower-addressed registers.

Instructions may address the data registers either implicitly or explicitly. Many instructions operate on the register at the top of the stack. These instructions implicitly address the register pointed to by the TOP. Other instructions allow the programmer to explicitly specify the register which is to be used. Explicit register addressing is "top-relative."

### Status Word

The status word shown in Figure 6 reflects the overall state of the 8087; it may be stored in memory and then inspected by CPU code. The status word is a 16-bit register divided into fields as shown in Figure 6. The busy bit (bit 15) indicates whether the NEU is either executing an instruction or has an interrupt request pending (B = 1), or is idle (B = 0). Several instructions which store and manipulate the status word are executed exclusively by the CU, and these do not set the busy bit themselves.

The four numeric condition code bits ($C_0 - C_3$) are similar to the flags in a CPU: various instructions update these bits to reflect the outcome of NDP operations. The effect of these instructions on the condition code bits is summarized in Table 4.

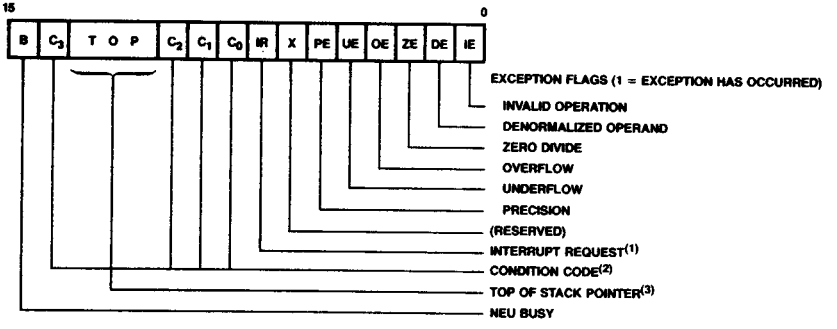
Bits 14-12 of the status word point to the 8087 register that is the current top-of-stack (TOP) as described above.

Bit 7 is the interrupt request bit. This bit is set if any unmasked exception bit is set and cleared otherwise.

Bits 5 – 0 are set to indicate that the NEU has detected an exception while executing an instruction.

### Tag Word

The tag word marks the contents of each register as shown in Figure 7. The principal function of the tag word is to optimize the NDP's performance. The tag word can be used, however, to interpret the contents of 8087 registers.



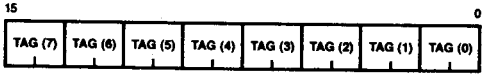

(1) R is set if any unmasked exception bit is set, cleared otherwise.
(2) See Table 3 for condition code interpretation.
(3) Top Values:
000 = Register 0 is Top of Stack.
001 = Register 1 is Top of Stack.
.
.
.
111 = Register 7 is Top of Stack.

**Figure 6. 8087 Status Word**

**TABLE 4. CONDITION CODE INTERPRETATION**

| Instruction | $C_3$ | $C_2$ | $C_1$ | $C_0$ | Interpretation |
|---|---|---|---|---|---|
| Compare, Test | 0 | X | X | 0 | A > B |
| | 0 | X | X | 1 | A < B |
| | 1 | X | X | 0 | A = B |
| | 1 | X | X | 1 | A ? B (not comparable) |
| Remainder | U | 0 | U | U | Complete reduction |
| | U | 1 | U | U | Incomplete reduction |
| Examine | 0 | 0 | 0 | 0 | Valid, positive, unnormalized |
| | 0 | 0 | 0 | 1 | Invalid, positive, exponent = 0 |
| | 0 | 0 | 1 | 0 | Valid, negative, unnormalized |
| | 0 | 0 | 1 | 1 | Invalid, negative, exponent = 0 |
| | 0 | 1 | 0 | 0 | Valid, positive, normalized |
| | 0 | 1 | 0 | 1 | Infinity, positive |
| | 0 | 1 | 1 | 0 | Valid, negative, normalized |
| | 0 | 1 | 1 | 1 | Infinity, negative |
| | 1 | 0 | 0 | 0 | Zero, positive |
| | 1 | 0 | 0 | 1 | Empty |
| | 1 | 0 | 1 | 0 | Zero, negative |
| | 1 | 0 | 1 | 1 | Empty |
| | 1 | 1 | 0 | 0 | Invalid, positive, exponent = 0 |
| | 1 | 1 | 0 | 1 | Empty |
| | 1 | 1 | 1 | 0 | Invalid, negative, exponent = 0 |
| | 1 | 1 | 1 | 1 | Empty |

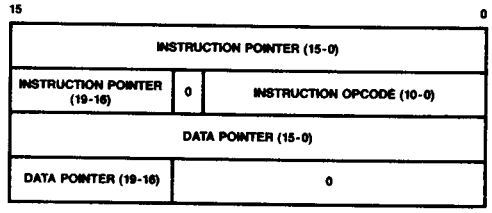
X = value is not affected by instruction.
U = value is undefined following instruction.




DF003280

TAG VALUES:
00 = VALID
01 = ZERO
10 = SPECIAL
11 = EMPTY

**Figure 7. 8087 Tag Word**




DF003290

**Figure 8. 8087 Instruction and Data Pointers**

## Instruction and Data Pointers

The instruction and data pointers (see Figure 8) are provided for user-written error handlers. Whenever the 8087 executes an NEU instruction, the CU saves the instruction address, the operand address (if present), and the instruction opcode. 8087 instructions can store this data into memory.

## Control Word

The 8087 provides several processing options which are selected by loading a word from memory into the control word. Figure 9 shows the format and encoding of the fields in the control word.

The low order byte of this control word configures 8087 interrupts and exception masking. Bits 5 – 0 of the control word contain individual masks for each of the six exceptions that the 8087 recognizes, and bit 7 contains a general mask bit for all 8087 interrupts. The high order byte of the control word configures the 8087 operating mode including precision, rounding, and infinity controls. The precision control bits (bits 9 – 8) can be used to set the 8087 internal operating precision at less than the default of temporary real precision. This can be useful in providing compatibility with earlier generation arithmetic processors of smaller precision than the 8087. The rounding control bits (bits 11 – 10) provide for directed rounding and true chop as well as the unbiased round to the nearest mode specified in the proposed IEEE standard. Control over closure of the number space at infinity is also provided (either affine closure, $\pm\infty$, or projective closure, $\infty$, is treated as unsigned, may be specified).

## Exception Handling

The 8087 detects six different conditions that can occur during instruction execution. Any or all exceptions will cause an interrupt if unmasked and interrupts are enabled.

If interrupts are disabled, the 8087 will simply continue execution regardless of whether the host clears the exception. If a specific exception class is masked and that exception occurs, however, the 8087 will post the exception in the status register and perform an on-chip default exception handling procedure, thereby allowing processing to continue. The exceptions that the 8087 detects are the following:

1. INVALID OPERATION: Stock overflow, stack underflow, indeterminate form (0/0, $\infty - \infty$, etc.) or the use of a Non-Number (NAN) as an operand. An exponent value is reserved and any bit pattern with this value in the exponent field is termed a Non-Number and causes this exception. If this exception is masked, the 8087's default response is to generate a specific NAN called







Refer to page 7-1 for Essential Information on Military Devices

INDEFINITE or to propagate already existing NANs as the calculation result.

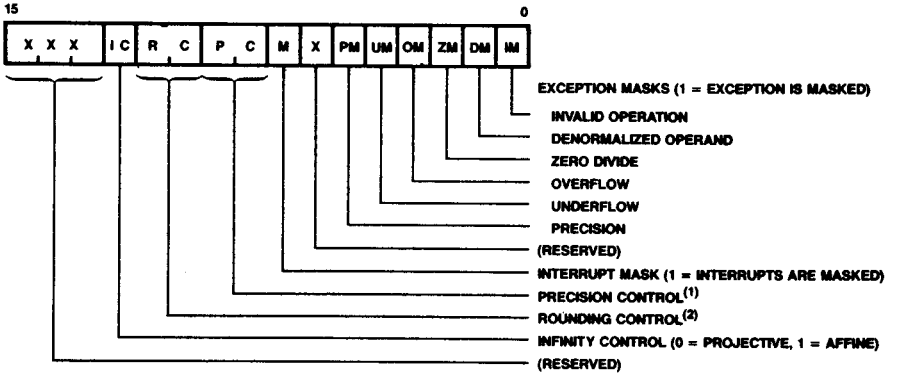
2. OVERFLOW: The result is too large in magnitude to fit the specified format. The 8087 will generate an encoding for infinity if this exception is masked.

3. ZERO DIVISOR: The divisor is zero while the dividend is a non-infinite, non-zero number. Again, the 8087 will generate an encoding for infinity if this exception is masked.

4. UNDERFLOW: The result is non-zero but too small in magnitude to fit in the specified format. If this exception is masked, the 8087 will denormalize (shift right) the fraction until the exponent is in range. This process is called gradual underflow.

5. DENORMALIZED OPERAND: At least one of the operands or the result is denormalized; it has the smallest exponent but a non-zero significand. Normal processing continues if this exception is masked off.

6. INEXACT RESULT: If the true result is not exactly representable in the specified format, the result is rounded according to the rounding mode, and this flag is set. If this exception is masked, processing will simply continue.




(1)Precision Control
00 = 24 bits
01 = Reserved
10 = 53 bits
11 = 64 bits

(2)Rounding Control
00 = Round to Nearest or Even
01 = Round Down (toward $-\infty$)
10 = Round Up (toward $+\infty$)
11 = Chop (truncate toward zero)

**Figure 9. 8087 Control Word**

## ABSOLUTE MAXIMUM RATINGS

Storage Temperature ..... −65 to +150°C
Voltage on Any Pin
with Respect to Ground ..... −1.0 to +7.0V
Power Dissipation ..... 3.0W

*Stresses above those listed under ABSOLUTE MAXIMUM RATINGS may cause permanent device failure. Functionality at or above these limits is not implied. Exposure to absolute maximum ratings for extended periods may affect device reliability.*

## OPERATING RANGES

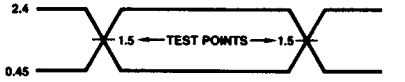
| Part Number | $T_A$ | $V_{CC}$ |
|---|---|---|
| 8087 | $T_A$ = 0°C to 70°C | 5V ±5% |
| 8087-2 | | |

*Operating ranges define those limits over which the functionality of the device is guaranteed.*

## DC CHARACTERISTICS (over Operating Ranges)

| Parameters | Description | Test Conditions | Min | Max | Units |
|---|---|---|---|---|---|
| $V_{IL}$ | Input Low Voltage | | −0.5 | +0.8 | V |
| $V_{IH}$ | Input High Voltage | | 2.0 | $V_{CC}$ + 0.5 | V |
| $V_{OL}$ | Output Low Voltage | $I_{OL}$ = 2.0μA | | 0.45 | V |
| $V_{OH}$ | Output High Voltage | $I_{OH}$ = −400μA | 2.4 | | V |
| $I_{CC}$ | Power Supply Current | $T_A$ = 25°C | | 475 | mA |
| $I_{LI}$ | Input Leakage Current | 0V ≤ $V_{IN}$ ≤ $V_{CC}$ | | ±10 | μA |
| $I_{LO}$ | Output Leakage Current | 0.45V ≤ $V_{OUT}$ ≤ $V_{CC}$ | | ±10 | μA |
| $V_{CL}$ | Clock Input Low Voltage | | −0.5 | +0.6 | V |
| $V_{CH}$ | Clock Input High Voltage | | 3.9 | $V_{CC}$ + 1.0 | V |
| $V_{IN}$ | Capacitance of Inputs | fc = 1MHz | | 10 | pF |
| $C_{IO}$ | Capacitance of I/O Buffer ($AD_{0-15}$, $A_{16}$-$A_{19}$, BHE, $S_2$-$S_0$, RQ/GT) and CLK | fc = 1MHz | | 15 | pF |
| $C_{OUT}$ | Capacitance of Outputs BUSY, INT | fc = 1 MHz | | 10 | pF |

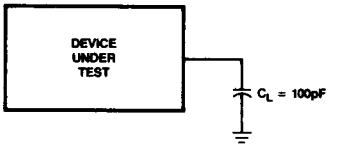
### SWITCHING TEST INPUT/OUTPUT WAVEFORM




WF006600

AC testing: inputs are driven at 2.4V for a logic "1" and 0.45V for a logic "0." The clock is driven at 4.3V and 0.25V timing measurements are made at 1.5V for both a logic "1" and "0."

### SWITCHING TEST LOAD CIRCUIT




TC001870

$C_L$ includes jig capacitance

## SWITCHING CHARACTERISTICS
### TIMING REQUIREMENTS

| Parameters | Description | Test Conditions | 8087 | | 8087-2 | | Units |
|---|---|---|---|---|---|---|---|
| | | | Min | Max | Min | Max | |
| TCLCL | CLK Cycle Period | | 200 | 500 | 125 | 500 | ns |
| TCLCH | CLK Low Time | | 118 | | 68 | | ns |
| TCHCL | CLK High Time | | 69 | | 44 | | ns |
| $TCH_1CH_2$ | CLK Rise Time | From 1.0 to 3.5V | | 10 | | 10 | ns |
| $TCL_2CL_1$ | CLK Fall Time | From 3.5 to 1.0V | | 10 | | 10 | ns |
| TDVCL | Data In Set-up Time | | 30 | | 20 | | ns |
| TCLDX | Data In Hold Time | | 10 | | 10 | | ns |
| TRYHCH | READY Set-up Time | | 118 | | 68 | | ns |
| TCHRYX | READY Hold Time | | 30 | | 20 | | ns |
| TRYLCL | READY Inactive to CLK (See Note 3) | | -8 | | -8 | | ns |
| TGVCH | RQ/GT Set-up Time | | 30 | | 15 | | ns |
| TCHGX | RQ/GT Hold Time | | 40 | | 30 | | ns |
| TQVCL | $QS_{0-1}$ Set-up Time | | 30 | | 30 | | ns |
| TCLQX | $QS_{0-1}$ Hold Time | | 10 | | 10 | | ns |
| TSACH | Status Active Set-up Time | | 30 | | 30 | | ns |
| TSNCL | Status Inactive Set-up Time | | 30 | | 30 | | ns |
| TILIH | Input Rise Time (Except CLK) | From 0.8 to 2.0V | | 20 | | 20 | ns |
| TIHIL | Input Fall Time (Except CLK) | From 2.0 to 0.8V | | 12 | | 12 | ns |

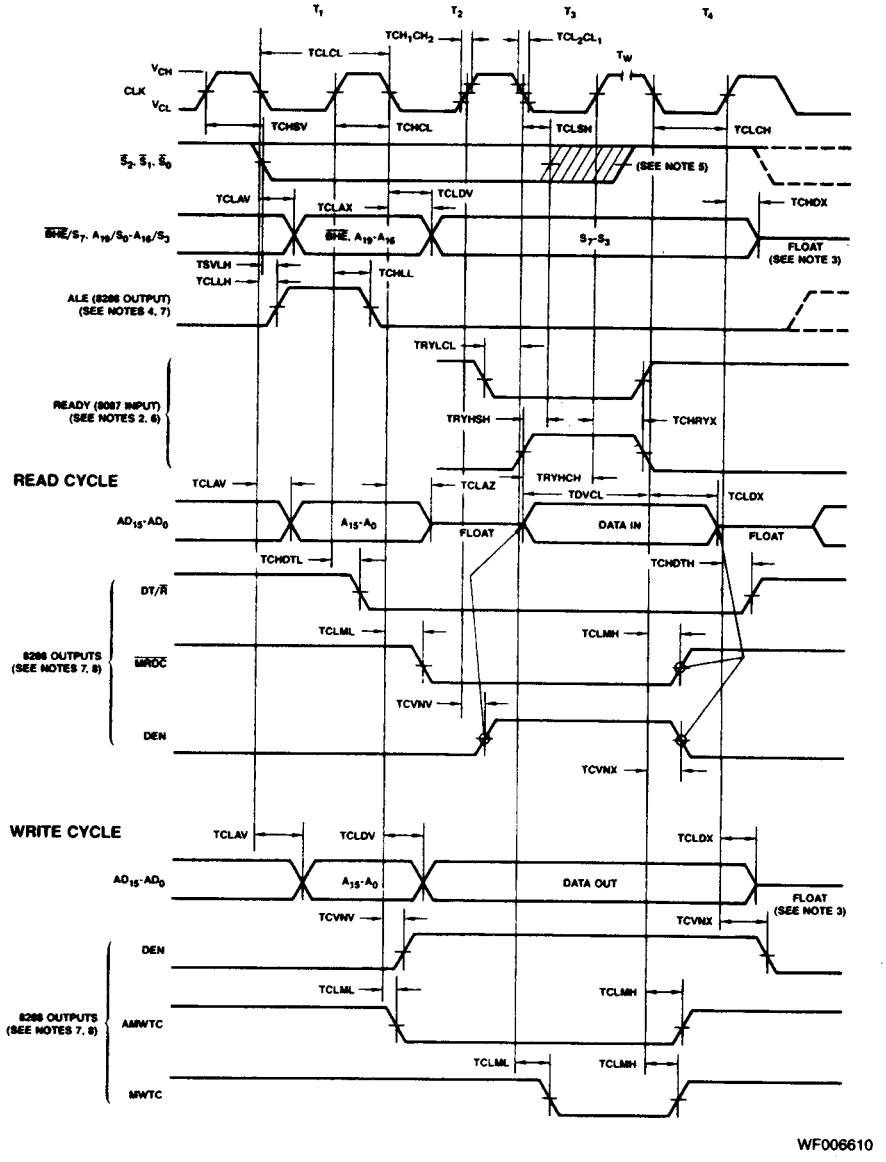
## SWITCHING CHARACTERISTICS
### TIMING RESPONSES

| Parameters | Description | Test Conditions | 8087 | | 8087-2 | | Units |
|---|---|---|---|---|---|---|---|
| | | | Min | Max | Min | Max | |
| TCLML | Command Active Delay (See Note 1) | $C_L$ = 20-100pF for all 8087 outputs (In addition to 8087 self-load) | 10 | 35 | 10 | 35 | ns |
| TCLMH | Command Inactive Delay (See Note 1) | | 10 | 35 | 10 | 35 | ns |
| TRYHSH | Ready Active to Status Passive (See Note 2) | | | 110 | | 65 | ns |
| TCHSV | Status Active Delay | | 10 | 110 | 10 | 60 | ns |
| TCLSH | Status Inactive Delay | | 10 | 130 | 10 | 70 | ns |
| TCLAV | Address Valid Delay | | 10 | 110 | 10 | 60 | ns |
| TCLAX | Address Hold Time | | 10 | | 10 | | ns |
| TCLAZ | Address Float Delay | | TCLAX | 80 | TCLAX | 50 | ns |
| TSVLH | Status Valid to ALE High (See Note 1) | | | 15 | | 15 | ns |
| TCLLH | CLK Low to ALE Valid (See Note 1) | | | 15 | | 15 | ns |
| TCHLL | ALE Inactive Delay (See Note 1) | | | 15 | | 15 | ns |
| TCLDV | Data Valid Delay | | 10 | 110 | 10 | 60 | ns |
| TCHDX | Data Hold Time | | 10 | | 10 | | ns |
| TCVNV | Control Active Delay (See Note 1) | | 5 | 45 | 5 | 45 | ns |
| TCVNX | Control Inactive Delay (See Note 1) | | 10 | 45 | 10 | 45 | ns |
| TCHBV | BUSY and INT Valid Delay | | 10 | 150 | 10 | 85 | ns |
| TCHDTL | Direction Control Active Delay (See Note 1) | | | 50 | | 50 | ns |
| TCHDTH | Direction Control Inactive Delay (See Note 1) | | | 30 | | 30 | ns |
| TCLGL | RQ/GT Active Delay | $C_L$ = 40pF (in addition to 8087 self-load) | 0 | 85 | 0 | 50 | ns |
| TCLGH | RQ/GT Inactive Delay | | 0 | 85 | 0 | 50 | ns |
| TOLOH | Output Rise Time | From 0.8 to 2.0V | | 20 | | 20 | ns |
| TOHOL | Output Fall Time | From 2.0 to 0.8V | | 12 | | 12 | ns |

Notes: 1. Signal at 8284A or 8288 shown for reference only.
2. Applies only to $T_3$ and wait states.
3. Applies only to $T_2$ state (8ns into $T_3$).

## SWITCHING WAVEFORMS

### MASTER MODE

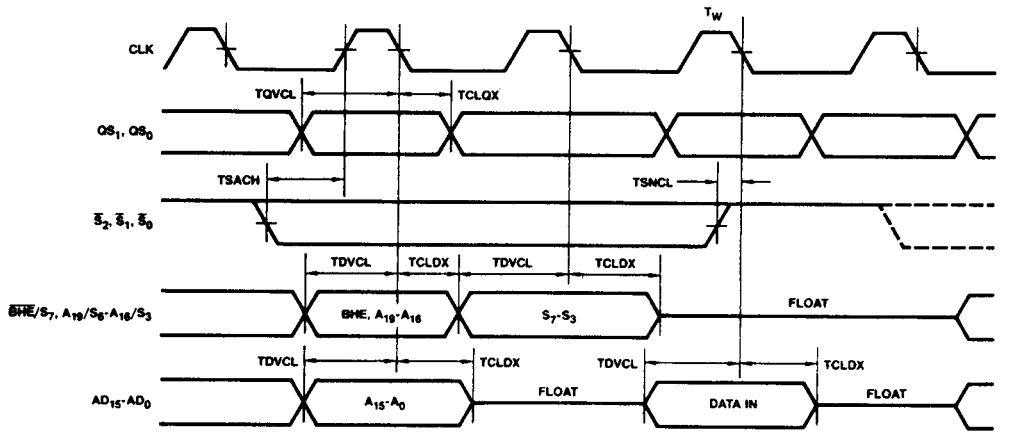



WF006610

Notes: 1. All signals switch between $V_{OL}$ and $V_{OH}$ unless otherwise specified.
2. RDY is sampled near the end of $T_2$, $T_3$ and $T_W$ to determine if $T_W$ machine states are to be inserted.
3. The local bus floats only if the 8087 is returning control to the 8086/8088.
4. ALE rises at later of (TSVLH, TCLLH).
5. Status inactive in state just prior to $T_4$.
6. Ready should remain active until $S_{0-2}$ become inactive.
7. Signals at 8284A or 8288 are shown for reference only.
8. The issuance of 8288 command and control signals $\overline{MRDC}$, $\overline{MWTC}$, $\overline{AMWC}$ and DEN lags the active high 8288 CEN.
9. All timing measurements are made at 1.5V unless otherwise noted.

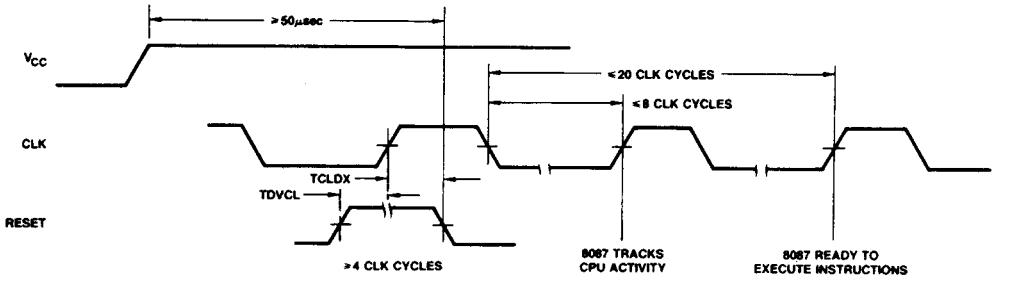
## SWITCHING WAVEFORMS (Cont.)

### PASSIVE MODE



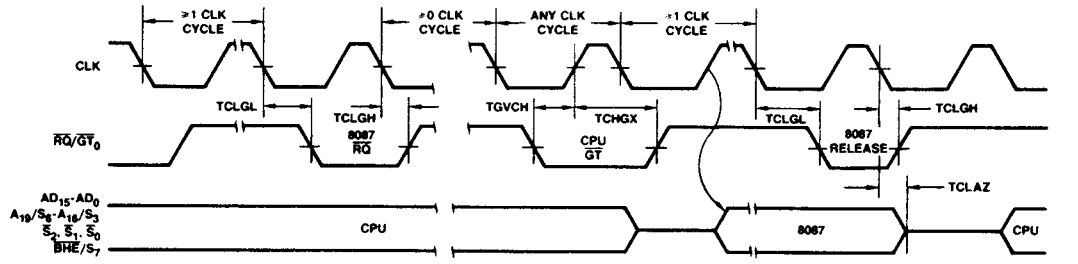

WF006620

### RESET




WF006630

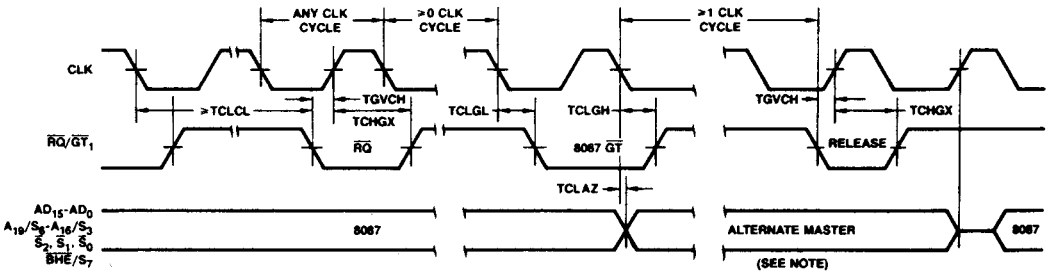
### REQUEST/GRANT$_0$




WF006640

## SWITCHING WAVEFORMS (Cont.)

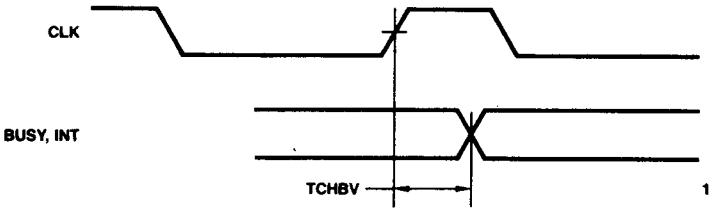
### REQUEST/GRANT$_1$




WF006580

Note: Alternate master may not drive the buses outside of the region shown without risking contention.

### BUSY AND INTERRUPT




WF006590

## Table 5. 8087 Extensions to the 86/186 Instructions Sets

| | Optional 8, 16 Bit Displacement | Clock Count Range | | | |
|---|---|---|---|---|---|
| **Data Transfer** | | 32 Bit Real | 32 Bit Integer | 64 Bit Real | 16 Bit Integer |
| **FLD = LOAD** | MF = | 00 | 01 | 10 | 11 |
| Integer/Real Memory to ST(0) | ESCAPE MF 1 \| MOD 0 0 0 R/M \| DISP | 38-56 +EA | 52-60 +EA | 40-60 +EA | 46-54 +EA |
| Long Integer Memory to ST(0) | ESCAPE 1 1 1 \| MOD 1 0 1 R/M \| DISP | 60-68+EA | | | |
| Temporary Real Memory to ST(0) | ESCAPE 0 1 1 \| MOD 1 0 1 R/M \| DISP | 53-65+EA | | | |
| BCD Memory to ST(0) | ESCAPE 1 1 1 \| MOD 1 0 0 R/M \| DISP | 290-310+EA | | | |
| ST(i) to ST(0) | ESCAPE 0 0 1 \| 1 1 0 0 0 ST(i) | 17-22 | | | |
| **FST** = STORE | | | | | |
| ST(0) to Integer/Real Memory | ESCAPE MF 1 \| MOD 0 1 0 R/M \| DISP | 84-90 +EA | 82-92 +EA | 96-104 +EA | 80-90 +EA |
| ST(0) to ST(i) | ESCAPE 1 0 1 \| 1 1 0 1 0 ST(i) | 15-22 | | | |
| **FSTP** = STORE AND POP | | | | | |
| ST(0) to Integer/Real Memory | ESCAPE MF 1 \| MOD 0 1 1 R/M \| DISP | 86-92 +EA | 84-94 +EA | 98-106 +EA | 82-92 +EA |
| ST(0) to Long Integer Memory | ESCAPE 1 1 1 \| MOD 1 1 1 R/M \| DISP | 95-105+EA | | | |
| ST(0) to Temporary Real Memory | ESCAPE 0 1 1 \| MOD 1 1 1 R/M \| DISP | 52-58+EA | | | |
| ST(0) to BCD Memory | ESCAPE 1 1 1 \| MOD 1 1 0 R/M \| DISP | 520-540+EA | | | |
| ST(0) to ST(i) | ESCAPE 1 0 1 \| 1 1 0 1 1 ST(i) | 17-24 | | | |
| **FXCH** = Exchange ST(i) and ST(0) | ESCAPE 0 0 1 \| 1 1 0 0 1 ST(i) | 10-15 | | | |
| **Comparison** | | | | | |
| **FCOM** = Compare | | | | | |
| Integer/Real Memory to ST(0) | ESCAPE MF 0 \| MOD 0 1 0 R/M \| DISP | 60-70 +EA | 78-91 +EA | 65-75 +EA | 72-86 +EA |
| ST(i) to ST(0) | ESCAPE 0 0 0 \| 1 1 0 1 0 ST(i) | 40-50 | | | |
| **FCOMP** = Compare and Pop | | | | | |
| Integer/Real Memory to ST(0) | ESCAPE MF 0 \| MOD 0 1 1 R/M \| DISP | 63-73 +EA | 80-93 +EA | 67-77 +EA | 74-88 +EA |
| ST(i) to ST(0) | ESCAPE 0 0 0 \| 1 1 0 1 1 ST(i) | 45-52 | | | |
| **FCOMPP** = Compare ST(1) to ST(0) and Pop Twice | ESCAPE 1 1 0 \| 1 1 0 1 1 0 0 1 | 45-55 | | | |
| **FTST** = Test ST(0) | ESCAPE 0 0 1 \| 1 1 1 0 0 1 0 0 | 38-48 | | | |
| **FXAM** = Examine ST(0) | ESCAPE 0 0 1 \| 1 1 1 0 0 1 0 1 | 12-23 | | | |
| **Constants** | | | | | |
| **FLDZ** = LOAD +0.0 into ST(0) | ESCAPE 0 0 1 \| 1 1 1 0 1 1 1 0 | 11-17 | | | |
| **FLD1** = LOAD +1.0 into ST(0) | ESCAPE 0 0 1 \| 1 1 1 0 1 0 0 0 | 15-21 | | | |
| **FLDPI** = LOAD $\pi$ into ST(0) | ESCAPE 0 0 1 \| 1 1 1 0 1 0 1 1 | 16-22 | | | |
| **FLDL2T** = LOAD $\log_2 10$ into ST(0) | ESCAPE 0 0 1 \| 1 1 1 0 1 0 0 1 | 16-22 | | | |
| **FLDL2E** = LOAD $\log_2 e$ into ST(0) | ESCAPE 0 0 1 \| 1 1 1 0 1 0 1 0 | 15-21 | | | |
| **FLDLG2** = LOAD $\log_{10} 2$ into ST(0) | ESCAPE 0 0 1 \| 1 1 1 0 1 1 0 0 | 18-24 | | | |
| **FLDLN2** = LOAD $\log_e 2$ into ST(0) | ESCAPE 0 0 1 \| 1 1 1 0 1 1 0 1 | 17-23 | | | |
| **Arithmetic** | | | | | |
| **FADD** = Addition | | | | | |
| Integer/Real Memory with ST(0) | ESCAPE MF 0 \| MOD 0 0 0 R/M \| DISP | 90-120 +EA | 108-143 +EA | 95-125 +EA | 102-137 +EA |
| ST(i) and ST(0) | ESCAPE d P 0 \| 1 1 0 0 0 ST(i) | 70-100 (Note 1) | | | |

**Note:** 1. If P = 1 then add 5 clocks.

## Table 5. 8087 Extensions to the 86/186 Instructions Sets (Cont.)

| Arithmetic (Cont.) | | Optional 8, 16 Bit Displacement | Clock Count Range | | | |
|---|---|---|---|---|---|---|
| | | | 32 Bit Real | 32 Bit Integer | 64 Bit Real | 16 Bit Integer |
| **FSUB = Subtraction** | **MF** = | | 00 | 01 | 10 | 11 |
| Integer/Real Memory with ST(0) | ESCAPE MF 0 \| MOD 1 0 R/M | DISP | 90-120 + EA | 108-143 + EA | 95-125 + EA | 102-137 + EA |
| ST(i) and ST(0) | ESCAPE d P 0 \| 1 1 1 0 R/M | | 70-100 (Note 1) | | | |
| **FMUL = Multiplication** | | | | | | |
| Integer/Real Memory with ST(0) | ESCAPE MF 0 \| MOD 0 0 1 R/M | DISP | 110-125 + EA | 130-144 + EA | 112-168 + EA | 124-138 + EA |
| ST(i) and ST(0) | ESCAPE d P 0 \| 1 1 0 0 1 R/M | | 90-145 (Note 1) | | | |
| **FDIV = Division** | | | | | | |
| Integer/Real Memory with ST(0) | ESCAPE MF 0 \| MOD 1 1 R/M | DISP | 215-225 + EA | 230-243 + EA | 220-230 + EA | 224-238 + EA> |
| ST(i) and ST(0) | ESCAPE d P 0 \| 1 1 1 1 R/M | | 193-203 (Note 1) | | | |
| **FSQRT** = Square Root of ST(0) | ESCAPE 0 0 1 \| 1 1 1 1 1 0 1 0 | | 180-186 | | | |
| **FSCALE** = Scale ST(0) by ST(1) | ESCAPE 0 0 1 \| 1 1 1 1 1 1 0 1 | | 32-38 | | | |
| **FPREM** = Partial Remainder of ST(0) ÷ ST(1) | ESCAPE 0 0 1 \| 1 1 1 1 1 0 0 0 | | 15-190 | | | |
| **FRNDINT** = Round ST(0) to Integer | ESCAPE 0 0 1 \| 1 1 1 1 1 1 0 0 | | 16-50 | | | |
| **FXTRACT** = Extract Components of ST(0) | ESCAPE 0 0 1 \| 1 1 1 1 0 1 0 0 | | 27-55 | | | |
| **FABS** = Absolute Value of ST(0) | ESCAPE 0 0 1 \| 1 1 1 0 0 0 0 1 | | 10-17 | | | |
| **FCHS** = Change Sign of ST(0) | ESCAPE 0 0 1 \| 1 1 1 0 0 0 0 0 | | 10-17 | | | |
| **Transcendental** | | | | | | |
| **FPTAN** = Partial Tangent of ST(0) | ESCAPE 0 0 1 \| 1 1 1 1 0 0 1 0 | | 30-540 | | | |
| **FPATAN** = Partial Arctangent of ST(0) ÷ ST(1) | ESCAPE 0 0 1 \| 1 1 1 1 0 0 1 1 | | 250-800 | | | |
| **F2XM1** = $2^{ST(0)} - 1$ | ESCAPE 0 0 1 \| 1 1 1 1 0 0 0 0 | | 310-630 | | | |
| **FYL2X** = ST(1) · $Log_2$ [ST(0)] | ESCAPE 0 0 1 \| 1 1 1 1 0 0 0 1 | | 900-1100 | | | |
| **FYL2XP1** = ST(1) · $Log_2$ [ST(0) + 1] | ESCAPE 0 0 1 \| 1 1 1 1 1 0 0 1 | | 700-1000 | | | |
| **Processor Control** | | | | | | |
| **FINIT** = Initialized 8087 | ESCAPE 0 1 1 \| 1 1 1 0 0 0 1 1 | | 2-8 | | | |
| **FENI** = Enable Interrupts | ESCAPE 0 1 1 \| 1 1 1 0 0 0 0 0 | | 2-8 | | | |
| **FDISI** = Disable Interrupts | ESCAPE 0 1 1 \| 1 1 1 0 0 0 0 1 | | 2-8 | | | |
| **FLDCW** = Load Control Word | ESCAPE 0 0 1 \| MOD 1 0 1 R/M | DISP | 7-14+ EA | | | |
| **FSTCW** = Store Control Word | ESCAPE 0 0 1 \| MOD 1 1 1 R/M | DISP | 12-18+ EA | | | |
| **FSTSW** = Store Status Word | ESCAPE 1 0 1 \| MOD 1 1 1 R/M | DISP | 12-18+ EA | | | |
| **FCLEX** = Clear Exceptions | ESCAPE 0 1 1 \| 1 1 1 0 0 0 1 0 | | 2-8 | | | |
| **FSTENV** = Store Environment | ESCAPE 0 0 1 \| MOD 1 1 0 R/M | DISP | 40-50+ EA | | | |
| **FLDENV** = Load Environment | ESCAPE 0 0 1 \| MOD 1 0 0 R/M | DISP | 35-45+ EA | | | |
| **FSAVE** = Save State | ESCAPE 1 0 1 \| MOD 1 1 0 R/M | DISP | 197-207+ EA | | | |
| **FRSTOR** = Restore State | ESCAPE 1 0 1 \| MOD 1 0 0 R/M | DISP | 197-207+ EA | | | |
| **FINCSTP** = Increment Stack Pointer | ESCAPE 0 0 1 \| 1 1 1 1 0 1 1 1 | | 6-12 | | | |
| **FDECSTP** = Decrement Stack Pointer | ESCAPE 0 0 1 \| 1 1 1 1 0 1 1 0 | | 6-12 | | | |
| **FFREE** = Free ST(i) | ESCAPE 1 0 1 \| 1 1 0 0 0 ST(i) | | 9-16 | | | |
| **FNOP** = No Operation | ESCAPE 0 0 1 \| 1 1 0 1 0 0 0 0 | | 10-16 | | | |
| **FWAIT** = CPU Wait for 8087 | 1 0 0 1 1 0 1 1 | | 3 + 5n* | | | |

*n = number of times CPU examines TEST line before 8087 lowers BUSY.

## NOTES:

1. if mod = 00 then DISP = 0*, disp-low and disp-high are absent
   if mod = 01 then DISP = disp-low sign-extended to 16-bits, disp-high is absent
   if mod = 10 then DISP = disp-high; disp-low
   if mod = 11 then r/m is treated as an ST(i) field
2. if r/m = 000 then EA = (BX) + (SI) + DISP
   if r/m = 001 then EA = (BX) + (DI) + DISP
   if r/m = 010 then EA = (BP) + (SI) + DISP
   if r/m = 011 then EA = (BP) + (DI) + DISP
   if r/m = 100 then EA = (SI) + DISP
   if r/m = 101 then EA = (DI) + DISP
   if r/m = 110 then EA = (BP) + DISP
   if r/m = 111 then EA = (BX) + DISP

   *except if mode = 000 and r/m = 110 then EA = disp-high; disp-low.
3. MF = Memory Format
   00 — 32-bit Real
   01 — 32-bit Integer
   10 — 64-bit Real
   11 — 16-bit Integer
4. ST(0) = Current stack top
   ST(i) = $i^{th}$ register below stack top
5. d = Destination
   0 — Destination is ST(0)
   1 — Destination is ST(i)
6. P = Pop
   0 — No pop
   1 — Pop ST(0)
7. R = Reverse: When d = 1 reverse the sense of R
   0 — Destination (op) Source
   1 — Source (op) Destination
8. For **FSQRT:** $-0 \leqslant ST(0) \leqslant +\infty$
   For **FSCALE:** $-2^{15} \leqslant ST(1) < +2^{15}$ and ST(1) integer
   For **F2XM1:** $0 \leqslant ST(0) \leqslant 2^{-1}$
   For **FYL2X:** $0 < ST(0) < \infty$
   $-\infty < ST(1) < +\infty$
   For **FYL2XP1:** $0 \leqslant |ST(0)| < (2-\sqrt{2})/2$
   $-\infty < ST(1) < \infty$
   For **FPTAN:** $0 \leqslant ST(0) \leqslant \pi/4$
   For **FPATAN:** $0 \leqslant ST(0) < ST(1) < +\infty$