80287
80-Bit HMOS
NUMERIC PROCESSOR EXTENSION

- **High Performance 80-Bit Internal Architecture**
- **Implements Proposed IEEE Floating Point Standard 754**
- **Expands iAPX 286/10 Datatypes to Include 32-, 64-, 80-Bit Floating Point, 32-, 64-Bit Integers and 18-Digit BCD Operands**
- **Object Code Compatible with 8087**
- **Built-in Exception Handling**
- **Operates in Both Real and Protected Mode iAPX 286 Systems**
- **Protected Mode Operation Completely Conforms to the iAPX 286 Memory Management and Protection Mechanisms**
- **Directly Extends iAPX 286/10 Instruction Set to Trigonometric, Logarithmic, Exponential and Arithmetic Instructions for All Datatypes**
- **8x80-Bit, Individually Addressable, Numeric Register Stack**
- **Available in EXPRESS—Standard Temperature Range**

The Intel® 80287 is a high performance numerics processor extension that extends the iAPX 286/10 architecture with floating point, extended integer and BCD data types. The iAPX 286/20 computing system (80286 with 80287) fully conforms to the proposed IEEE Floating Point Standard. Using a numerics oriented architecture, the 80287 adds over fifty mnemonics to the iAPX 286/20 instruction set, making the iAPX 286/20 a complete solution for high performance numeric processing. The 80287 is implemented in N-channel, depletion load, silicon gate technology (HMOS) and packaged in a 40-pin ceramic package. The iAPX 286/20 is object code compatible with the iAPX 86/20 and iAPX 88/20.

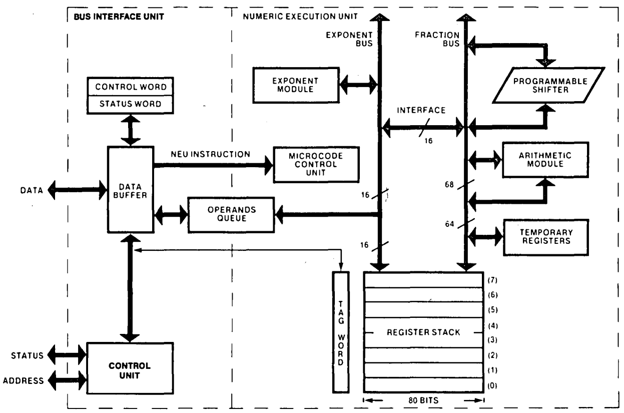
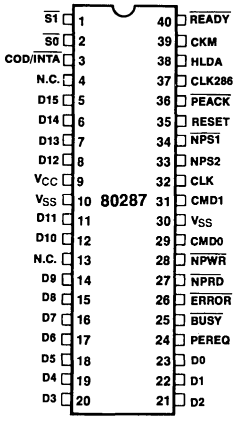


Figure 1. 80287 Block Diagram



NOTE:
N.C. PINS MUST NOT BE CONNECTED.

Figure 2. 80287 Pin Configuration

Intel Corporation Assumes No Responsibility for the Use of Any Circuitry Other Than Circuitry Embodied in an Intel Product. No Other Circuit Patent Licenses are Implied.

FEBRUARY 1983

ORDER NUMBER: 210920-001

Table 1. 80287 Pin Description

Symbols	Type	Name and Function
CLK	I	Clock input: this clock provides the basic timing for internal 80287 operations. Special MOS level inputs are required. The 82284 or 8284A CLK outputs are compatible to this input.
CKM	I	Clock Mode signal: indicates whether CLK input is to be divided by 3 or used directly. A HIGH input will select the latter option. This input may be connected to V_{CC} or V_{SS} as appropriate. This input must be either HIGH or LOW 20 CLK cycles before RESET goes LOW.
RESET	I	System Reset: causes the 80287 to immediately terminate its present activity and enter a dormant state. RESET is required to be HIGH for more than 4 80287 CLK cycles. For proper initialization the HIGH-LOW transition must occur no sooner than 50 μs after V_{CC} and CLK meet their D.C. and A.C. specifications.
D15-D0	I/O	Data: 16-bit bidirectional data bus. Inputs to these pins may be applied asynchronous to the 80287 clock.
$\overline{\text{BUSY}}$	O	Busy status: asserted by the 80287 to indicate that it is currently executing a command.
$\overline{\text{ERROR}}$	O	Error status: reflects the ES bit of the status word. This signal indicates that an unmasked error condition exists.
PEREQ	O	Processor Extension Data Channel operand transfer request: a HIGH on this output indicates that the 80287 is ready to transfer data. PEREQ will be disabled upon assertion of $\overline{\text{PEACK}}$ or upon actual data transfer, whichever occurs first, if no more transfers are required.
$\overline{\text{PEACK}}$	I	Processor Extension Data Channel operand transfer ACKnowledge: acknowledges that the request signal (PEREQ) has been recognized. Will cause the request (PEREQ) to be withdrawn in case there are no more transfers required. $\overline{\text{PEACK}}$ may be asynchronous to the 80287 clock.
$\overline{\text{NPRD}}$	I	Numeric Processor Read: Enables transfer of data from the 80287. This input may be asynchronous to the 80287 clock.
$\overline{\text{NPWR}}$	I	Numeric Processor Write: Enables transfer of data to the 80287. This input may be asynchronous to the 80287 clock.
$\overline{\text{NPS1}}$, NPS2	I	Numeric Processor Selects: indicate the CPU is performing an ESCAPE instruction. Concurrent assertion of these signals (i.e., $\overline{\text{NPS1}}$ is LOW and NPS2 is HIGH) enables the 80287 to perform floating point instructions. No data transfers involving the 80287 will occur unless the device is selected. These inputs may be asynchronous to the 80287 clock.
CMD1, CMD0	I	Command lines: These, along with select inputs, allow the CPU to direct the operation of the 80287. No actions will occur if these signals are both HIGH. These inputs may be asynchronous to the 80287 clock.

Table 1. 80287 Pin Description (cont.)

Symbols	Type	Name and Function
CLK286	I	CPU Clock: This input provides a sampling edge for the 80287 inputs $\overline{S1}$, $\overline{S0}$, COD/$\overline{INTA}$, $\overline{READY}$, and HLDA. It must be connected to the 80286 CLK input.
$\overline{S1}$, $\overline{S0}$ COD/$\overline{INTA}$	I	Status: These inputs allow the 80287 to monitor the execution of ESCAPE instructions by the 80286. They must be connected to the corresponding 80286 pins.
HLDA	I	Hold Acknowledge: This input informs the 80287 when the 80286 controls the local bus. It must be connected to the 80286 HLDA output.
$\overline{READY}$	I	Ready: The end of a bus cycle is signaled by this input. It must be connected to the 80286 $\overline{READY}$ input.
V_{SS}	I	System ground, both pins must be connected to ground.
V_{CC}	I	+5V supply

FUNCTIONAL DESCRIPTION

The 80287 Numeric Processor Extension (NPX) provides arithmetic instructions for a variety of numeric data types in iAPX 286/20 systems. It also executes numerous built-in transcendental functions (e.g., tangent and log functions). The 80287 executes instructions in parallel with a 80286. It effectively extends the register and instruction set of an iAPX 286/10 system for existing iAPX 286 data types and adds several new data types as well. Figure 3 presents the program visible register model of the iAPX 286/20. Essentially, the 80287 can be treated as an additional resource or an extension to the iAPX 286/10 that can be used as a single unified system, the iAPX 286/20.

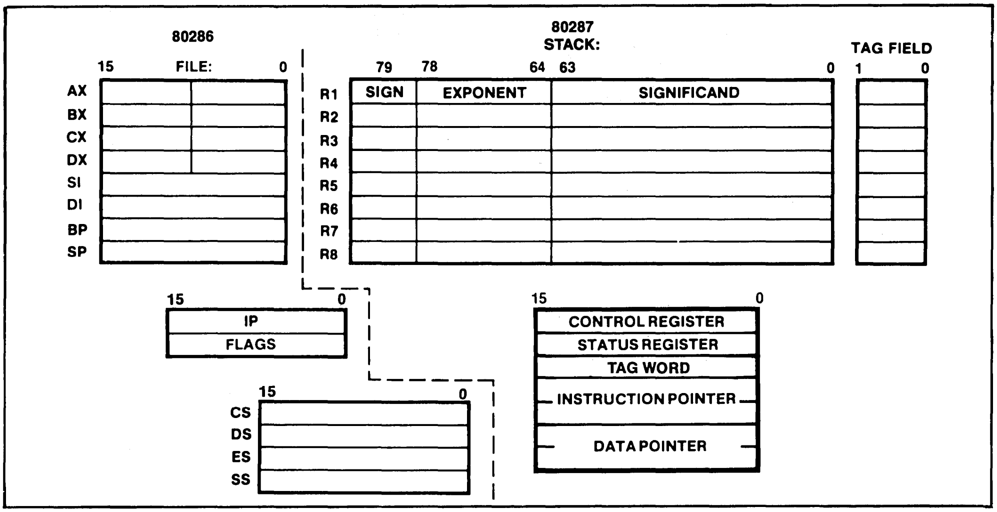


Figure 3. iAPX 286/20 Architecture

The 80287 has two operating modes similar to the two modes of the 80286. When reset, 80287 is in the real address mode. It can be placed in the protected virtual address mode by executing the SETPM ESC instruction. The 80287 cannot be switched back to the real address mode except by reset. In the real address mode, the iAPX 286/20 is completely software compatible with iAPX 86/20, 88/20.

Once in protected mode, all references to memory for numerics data or status information, obey the iAPX 286 memory management and protection rules giving a fully protected extension of the 80286 CPU. In the protected mode, iAPX 286/20 numerics software is also completely compatible with iAPX 86/20 and iAPX 88/20.

SYSTEM CONFIGURATION

As a processor extension to an 80286, the 80287 can be connected to the CPU as shown in Figure 4. The data channel control signals (PEREQ, PEACK), the BUSY signal and the NPRD, NPWR signals, allow the NPX to receive instructions and data from the CPU. When in the protected mode, all information received by the NPX is validated by the 80286 memory management and protection unit. Once started, the 80287 can process in parallel with and independent of the host CPU. When the NPX detects an error or exception, it will indicate this to the CPU by asserting the ERROR signal.

The NPX uses the processor extension request and acknowledge pins of the 80286 CPU to implement data transfers with memory under the protection model of the CPU. The full virtual and physical address space of the 80286 is available. Data for the 80287 in memory is addressed and represented in the same manner as for an 8087.

The 80287 can operate either directly from the CPU clock or with a dedicated clock. For operation with the CPU clock (CKM=0), the 80287 works at one-third the frequency of the system clock (i.e., for an 8 MHz 80286, the 16 MHz system clock is divided down to 5.3 MHz). The 80287 provides a capability to internally divide the CPU clock by three to produce the required internal clock (33% duty cycle). To use a higher performance 80287 (8 MHz), an 8284A clock driver and appropriate crystal may be used to directly drive the 80287 with a 1/3 duty cycle clock on the CLK input (CKM=1).

HARDWARE INTERFACE

Communication of instructions and data operands between the 80286 and 80287 is handled by the CMD0, CMD1, NPS1, NPS2, NPRD, and NPWR signals. I/O port addresses 00F8H, 00FAH, and 00FCH are used by the 80286 for this communication. When any of these addresses are used, the NPS1 input must be LOW and NPS2 input HIGH. The IORC and IOWC outputs of the 82288 identify I/O space transfers (see Figure 4). CMD0 should be connected to latched 80286 A1 and CMD1 should be connected to latched 80286 A2.

I/O ports 00F8H to 00FFH are reserved for the 80286/80287 interface. To guarantee correct operation of the 80287, programs must not perform any I/O operations to these ports.

The PEREQ, PEACK, BUSY, and ERROR signals of the 80287 are connected to the same-named 80286 input. The data pins of the 80287 should be directly connected to the 80286 data bus. Note that all bus drivers connected to the 80286 local bus must be inhibited when the 80286 reads from the 80287. The use of COD/INTA and M/IO in the decoder prevents INTA bus cycles from disabling the data transceivers.

The S1, S0 COD/INTA, READY, HLDA, and CLK pins of the 80286 are connected to the same named pins on the 80287. These signals allow the 80287 to monitor the execution of ESCAPE instructions by the 80826.

PROGRAMMING INTERFACE

Table 2 lists the seven data types the 80287 supports and presents the format for each type. These values are stored in memory with the least significant digits at the lowest memory address. Programs retrieve these values by generating the lowest address. All values should start at even addresses for maximum system performance.

Internally the 80287 holds all numbers in the temporary real format. Load instructions automatically convert operands represented in memory as 16-, 32-, or 64-bit integers, 32- or 64-bit floating point number or 18-digit packed BCD numbers into temporary real format. Store instructions perform the reverse type conversion.

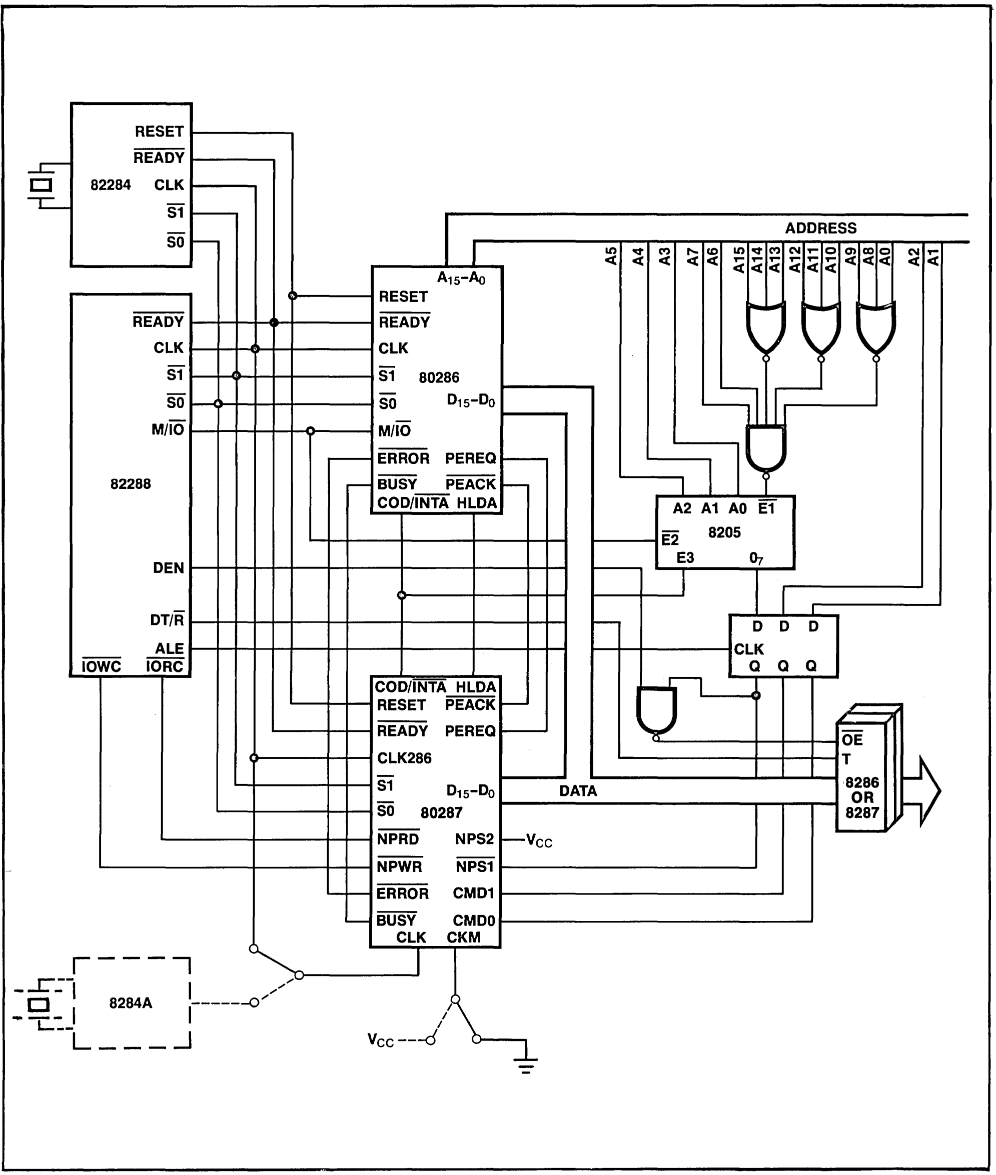


Figure 4. iAPX 286/20 System Configuration

Table 2. 80287 Datatype Representation in Memory

Data Formats	Range	Precision	Most Significant Byte 7 0 | 7 0 | 7 0 | 7 0 | 7 0 | 7 0 | 7 0 | 7 0 | 7 0 | 7 0	
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^{19}	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}	

NOTES:

(1) Integer: I

(2) Packed BCD $(-1)^S(D_{17}...D_0)$

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

(4) Bias =127 for Short Real
1023 for Long Real
16383 for Temp Real

80287 computations use the processor's register stack. These eight 80-bit registers provide the equivalent capacity of 40 16-bit registers. The 80287 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.

Table 6 lists the 80287's instructions by class. No special programming tools are necessary to use the 80287 since all new instructions and data types are directly supported by the iAPX 286 assembler and appropriate high level languages. All iAPX 86/88 development tools which support the 8087 can also be used to develop software for the iAPX 286/20 in real address mode.

Table 3 gives the execution times of some typical numeric instructions.

Table 3. Execution Time for Selected 80287 Instructions

Floating Point Instruction	Approximate Execution Time (μs)
	80287 (5 MHz Operation)
Add/Subtract	14/18
Multiply (single precision)	19
Multiply (extended precision)	27
Divide	39
Compare	9
Load (double precision)	10
Store (double precision)	21
Square Root	36
Tangent	90
Exponentiation	100

SOFTWARE INTERFACE

The iAPX 286/20 is programmed as a single processor. All communication between the 80286 and the 80287 is transparent to software. The CPU automatically controls the 80287 whenever a numeric instruction is executed. All memory addressing modes, physical memory, and virtual memory of the CPU are available for use by the NPX.

Since the NPX operates in parallel with the CPU, any errors detected by the NPX may be reported after the CPU has executed the ESCAPE instruction which caused it. To allow identification of the failing numeric instruction, the NPX contains two pointer registers which identify the address of the failing numeric instruction and the numeric memory operand if appropriate for the instruction encountering this error.

INTERRUPT DESCRIPTION

Several interrupts of the iAPX 286 are used to report exceptional conditions while executing numeric programs in either real or protected mode. The interrupts and their functions are shown in Table 4.

PROCESSOR ARCHITECTURE

As shown in Figure 1, the NPX is internally divided into two processing elements, the bus interface unit (BIU) and the numeric execution unit (NEU). The NEU executes all numeric instructions, while the BIU receives and decodes instructions, requests operand transfers to and from memory and executes processor control instructions. The two units are able to operate independently of one another allowing the BIU to maintain asynchronous communication with the CPU while the NEU is busy processing a numeric instruction.

BUS INTERFACE UNIT

The BIU decodes the ESC instruction executed by the CPU. If the ESC code defines a math instruction, the BIU transmits the formatted instruction to the NEU. If the ESC code defines an administrative instruction, the BIU executes it independently of the NEU. The parallel operation of the NPX with the CPU is normally transparant to the user. The BIU generates the $\overline{\text{BUSY}}$ and $\overline{\text{ERROR}}$ signals for 80826/80287 processor synchronization.

The 80287 executes a single numeric instruction at a time. When executing most ESC instructions, the

Table 4. Interrupt Vectors

Interrupt Number	Interrupt Function
7	An ESC instruction was encountered when EM or TS of the 80286 MSW was set. EM=1 indicates that software emulation of the instruction is required. When TS is set, either an ESC or WAIT instruction will cause interrupt 7. This indicates that the current NPX context may not belong to the current task.
9	The second or subsequent words of a numeric operand in memory exceeded a segment's limit. This interrupt occurs after executing an ESC instruction. The saved return address will not point at the numeric instruction causing this interrupt. After processing the addressing error, the iAPX 286 program can be restarted at the return address with IRET. The address of the failing numeric instruction and numeric operand are saved in the 80287. An interrupt handler for this interrupt *must* execute FNINIT before *any* other ESC or WAIT instruction.
13	The starting address of a numeric operand is not in the segment's limit. The return address will point at the ESC instruction (including prefixes) causing this error. The 80287 has not executed this instruction. The instruction and data address in 80287 refer to a previous, correctly executed, instruction.
16	The previous numeric instruction caused an unmasked numeric error. The address of the faulty numeric instruction or numeric data operand is stored in the 80287. Only ESC or WAIT instructions can cause this interrupt. The 80286 return address will point at a WAIT or ESC instruction, including prefixes, which may be restarted after clearing the error condition in the NPX.

80286 tests the $\overline{\text{BUSY}}$ pin and waits until the 80287 indicates that it is not busy before initiating the command. Once initiated, the 80286 continues program execution while the 80287 executes the ESC instruction. In iAPX 86/20 systems, this synchronization is achieved by placing a WAIT instruction before an ESC instruction. For most ESC instructions, the iAPX 286/20 does not require a WAIT instruction before the ESC opcode. However, the iAPX 286/20 will operate correctly with these WAIT instructions. In all cases, a WAIT or ESC instruction should be inserted after any 80287 store to memory (except FSTSW and FSTCW) or load from memory (except FLDENV or FRSTOR) before the 80286 reads or changes the value.

Data transfers between memory and the 80287, when needed, are controlled by the PEREQ $\overline{\text{PEACK}}$, $\overline{\text{NPRD}}$, $\overline{\text{NPWR}}$, $\overline{\text{NPS1}}$, NPS2 signals. The 80286 does the actual data transfer with memory through its processor extension data channel. Numeric data transfers with memory performed by the 80286 use the same timing as any other bus cycle. Control signals for the 80287 are generated by the 80826 as shown in Figure 4, and meet the timing requirements shown in the AC requirements section.

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 BIU $\overline{\text{BUSY}}$ signal. This signal is used in conjunction with the CPU WAIT instruction or automatically with most of the ESC instructions to synchronize both processors.

REGISTER SET

The 80287 register set is shown in Figure 5. Each of the eight data registers in the 8087's register stack

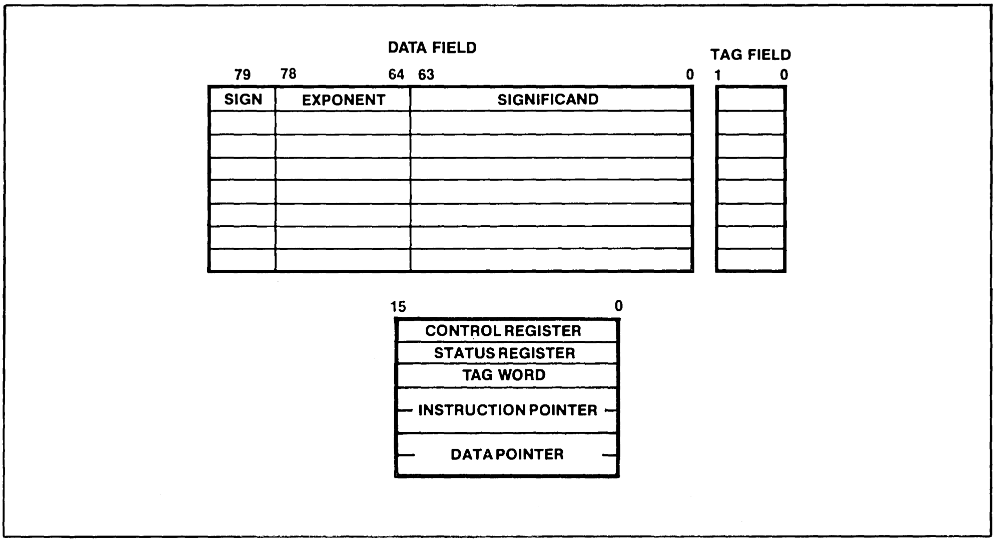


Figure 5. 80287 Register Set

is 80 bits wide and is divided into "fields" corresponding to the NPX's temporary real data type.

At a given point in time the TOP field in the status 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. Like 80286 stacks in memory, the 80287 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 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 16-bit status word (in the status register) shown in Figure 6 reflects the overall state of the 80287. It may be read and inspected by CPU code. The busy bit (bit 15) indicates whether the NEU is executing an instruction (B = 1) or is idle (B = 0).

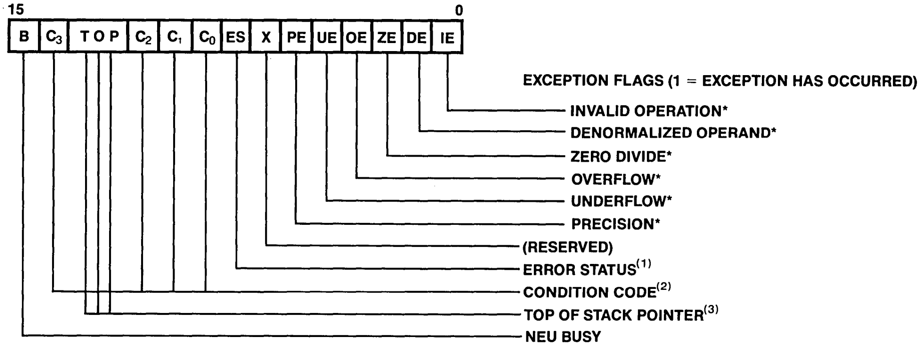
The instructions FSTSW, FSTENV, and FSAVE which store the status word are executed exclusively by the BIU and do not set the busy bit themselves or require the Busy bit be cleared in order to be executed.

The four numeric condition code bits (C_0–C_3) are similar to the flags in a CPU: instructions that perform arithmetic operations update these bits to reflect the outcome of NDP operations. The effect of these instructions on the condition code bits is summarized in Tables 5a and 5b.

Bits 14–12 of the status word point to the 80287 register that is the current top-of-stack (TOP) as described above. Figure 6 shows the six error flags in bits 7–0 of the status word. The section on exception handling explains how they are set and used.

Bit 7 is the error status bit. This bit is set if any unmasked exception bit is set and cleared otherwise. If this bit is set, the ERROR signal is asserted.

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



[1] ES IS SET IF ANY UNMASKED EXCEPTION BIT IS SET, CLEARED OTHERWISE.
[2] SEE TABLE 5 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

***For definitions, see the section on exception handling**

Figure 6. 80287 Status Word

TAG WORD

The tag word marks the content of each register as shown in Figure 7. The principal function of the tag word is to optimize the NPX's performance. The tag word can be used, however, to interpret the contents of 80287 registers.

INSTRUCTION AND DATA POINTERS

The instruction and data pointers (See Figures 8a and 8b) are provided for user-written error handlers. Whenever the 80287 executes a new instruction, the BIU saves the instruction address, the operand address (if present) and the instruction opcode. 80287 instructions can store this data into memory.

The instruction and data pointers appear in one of two formats depending on the operating mode of the 80287. In real mode, these values are the 20-bit physical address and 11-bit opcode formatted like the 8087. In protected mode, these values are the 32-bit virtual addresses used by the program which executed an ESC instruction. The same FLDENV/FSTENV/FSAVE/FRSTOR instructions as those of the 8087 are used to transfer these values between the 80287 registers and memory.

The saved instruction address in the 80287 will point at any prefixes which preceded the instruction. This is different than in the 8087 which only pointed at the ESCAPE instruction opcode.

CONTROL WORD

The NPX 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 fields in the control word.

The low order byte of this control word configures the 80287 error and exception masking. Bits 5–0 of the control word contain individual masks for each of the six exceptions that the 80287 recognizes. The high order byte of the control word configures the 80287 operating mode including precision,

Table 5a. Condition Code Interpretation

Instruction Type	C_3	C_2	C_1	C_0	Interpretation
Compare, Test	0	0	X	0	ST > Source or 0 (FTST)
	0	0	X	1	ST < Source or 0 (FTST)
	1	0	X	0	ST = Source or 0 (FTST)
	1	1	X	1	ST is not comparable
Remainder	Q_1	0	Q_0	Q_2	Complete reduction with three low bits of quotient (See Table 5b)
	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

NOTES:

1. ST = Top of stack
2. X = value is not affected by instruction
3. U = value is undefined following instruction
4. Q_n = Quotient bit n

Table 5b. Condition Code Interpretation after FPREM Instruction As a Function of Dividend Value

Dividend Range	Q_2	Q_1	Q_0
Dividend < 2 * Modulus	C_3	C_1	Q_0
Dividend < 4 * Modulus	C_3	Q_1	Q_0
Dividend ⩾ 4 * Modulus	Q_2	Q_1	Q_0

NOTE:

1. Previous value of indicated bit, not affected by FPREM instruction execution.

rounding, and infinity control. The precision control bits (bits 9–8) can be used to set the 80287 internal operating precision at less than the default of temporary real (80-bit) precision. This can be useful in providing compatibility with the early generation arithmetic processors of smaller precision than the 80287. The rounding control bits (bits 11–10) provide for directed rounding and true chop as well as the unbiased round to nearest even mode specified in the IEEE standard. Control over closure of the number space at infinity is also provided (either affine closure: ± ∞, or projective closure: ∞, is treated as unsigned, may be specified).

15							0
TAG (7)	TAG (6)	TAG (5)	TAG (4)	TAG (3)	TAG (2)	TAG (1)	TAG (0)

TAG VALUES:
00 = VALID
01 = ZERO
10 = INVALID or INFINITY
11 = EMPTY

Figure 7. 80287 Tag Word

15 0	MEMORY OFFSET
CONTROL WORD	+0
STATUS WORD	+2
TAG WORD	+4
IP OFFSET	+6
CS SELECTOR	+8
DATA OPERAND OFFSET	+10
DATA OPERAND SELECTOR	+12

Figure 8a. Protected Mode Instruction and Data Pointer Image in Memory

EXCEPTION HANDLING

The 80287 detects six different exception conditions that can occur during instruction execution. Any or all exceptions will cause the assertion of ERROR signal if the appropriate exception masks are not set.

The exceptions that the 80287 detects and the 'default' procedures that will be carried out if the exception is masked, are as follows:

Invalid Operation: Stack overflow, stack underflow, indeterminate form (0/0, ∞-∞, etc.) or the use of a Non-Number (NAN) as an operand. An exponent value of all ones and non-zero significand is reserved to identify NANs. If this exception is masked, the 80287 default response is to generate a specific NAN called INDEFINITE, or to propogate already existing NANs as the calculation result.

Overflow: The result is too large in magnitude to fit the specified format. The 80287 will generate an encoding for infinity if this exception is masked.

Zero Divisor: The divisor is zero while the dividend is a non-infinite, non-zero number. Again, the 80287 will generate an encoding for infinity if this exception is masked.

Underflow: The result is non-zero but too small in magnitude to fit in the specified format. If this exception is masked the 82087 will denormalize (shift right) the fraction until the exponent is in range. The process is called gradual underflow.

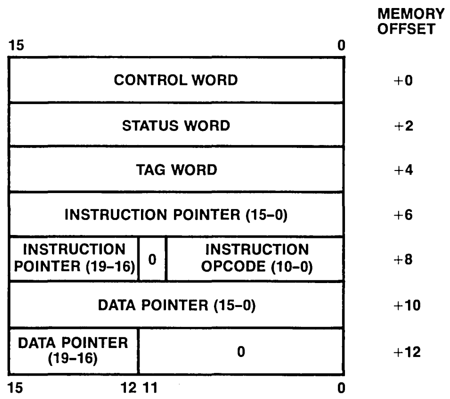
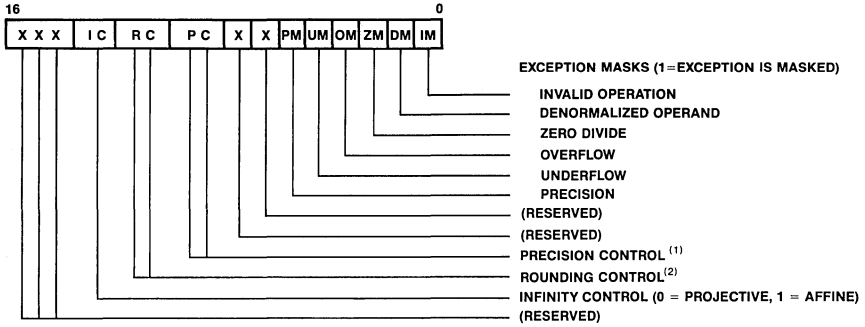


Figure 8b. Real Mode 80287 Instruction and Data Pointer Image in Memory



(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 −∞)
10 = ROUND UP (TOWARD +∞)
11 = CHOP (TRUNCATE TOWARD ZERO)

Figure 9. 80287 Control Word

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

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.

If the error is not masked, the corresponding error bit and the error status bit (ES) in the control word will be set, and the $\overline{\text{ERROR}}$ output signal will be asserted. If the CPU attempts to execute another ESC or WAIT instruction, exception 7 will occur.

The error condition must be resolved via an interrupt service routine. The 80287 saves the address of the floating point instruction causing the error as well as the address of the lowest memory location of any memory operand required by that instruction.

iAPX 86/20 COMPATIBILITY:

iAPX 286/20 supports portability of iAPX 86/20 programs when it is in the real address mode. However, because of differences in the numeric error handing techniques, error handling routines *may* need to be changed. The differences between an iAPX 286/20 and iAPX 86/20 are:

1. The NPX error signal does not pass through an interrupt controller (8087 INT signal does). Therefore, any interrupt controller oriented instructions for the iAPX 86/20 may have to be deleted.
2. Interrupt vector 16 must point at the numeric error handler routine.
3. The saved floating point instruction address in the 80287 includes any leading prefixes before the ESCAPE opcode. The corresponding saved address of the 8087 does not include leading prefixes.
4. In protected mode, the format of the saved instruction and operand pointers is different than for the 8087. The instruction opcode is not saved—it must be read from memory if needed.
5. Interrupt 7 will occur when executing ESC instructions with either TS or EM of MSW=1. If TS of MSW=1 then WAIT will also cause interrupt 7. An interrupt handler should be added to handle this situation.
6. Interrupt 9 will occur if the second or subsequent words of a floating point operand fall outside a segment's size. Interrupt 13 will occur if the starting address of a numeric operand falls outside a segment's size. An interrupt handler should be added to report these programming errors.

In the protected mode, iAPX 86/20 application code can be directly ported via recompilation if the 286 memory protection rules are not violated.

ABSOLUTE MAXIMUM RATINGS*

Ambient Temperature Under Bias .. 0°C to 70°C
Storage Temperature .... −65°C to +150°C
Voltage on Any Pin with
Respect to Ground .... −1.0 to +7V
Power Dissipation .... 3.0 Watt

**NOTICE: Stresses above those listed under Absolute Maximum Ratings may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any other conditions above those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.*

D.C. CHARACTERISTICS T_A = 0°C to 70°C, V_{CC} = 5V ± 10%

Symbol	Parameter	Min.	Max.	Units	Test Conditions
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		0.45	V	I_{OL} = 3.0 mA
V_{OH}	Output High Voltage	2.4		V	I_{OH} = .400 μA
I_{CC}	Power Supply Current		475	mA	T_A = 25°C
I_{LI}	Input Leakage Current		±10	μA	0V≤V_{IN}≤V_{CC}
I_{LO}	Output Leakage Current		±10	μA	0.45V≤V_{OUT}≤V_{CC}
V_{CL}	Clock Input Low Voltage CKM=1 CKM=0	 −0.5 −0.5	 +0.6 +0.8	 V V	
V_{CH}	Clock Input High Voltage CKM=1 CKM=0	 3.8 2.0	 V_{CC}+1.0 V_{CC}+1.0	 V V	
C_{IN}	Capacitance of Input & Output Buffers (all except I/O Buffer and CLK)		10	pF	fc = 1 Mhz
C_{IO}	Capacitance of I/O Buffer for D_{15}–D_0		20	pF	fc = 1 MHz
C_{CLK}	Capacitance of CLK Input		12	pF	fc = 1 MHz

A.C. CHARACTERISTICS T_A=0°C to 70°C, V_{CC}= 5V± 10%
TIMING REQUIREMENTS

Symbol	Parameter	Min.	Max.	Units	Test Conditions
TCLCL	CLK cycle period CKM=1: 5MHz 8MHz CKM=0: 16MHz	 200 125 62.5	 500 500 250	ns	at 1.5 V
TCLCH	CLK low time CKM=1 CKM=0	118 15	 230	ns	at 0.6 V at 0.8 V
TCHCL	CLK high time CKM=1 CKM=0	69 20	 235	ns	at 3.8 V at 2.0 V
TCH1CH2	CLK rise time		10	ns	from 1.0 to 3.5 V
TCL2CL1	CLK fall time		10	ns	from 3.5 to 1.0V
TDVWH	Data valid set up to $\overline{\text{NPWR}}$ inactive	75		ns	at 1.5 V
TWHDX	Data hold from $\overline{\text{NPWR}}$ inactive	0		ns	
TWLWH, TRLRH	$\overline{\text{NPWR}}$, $\overline{\text{NPRD}}$ active time	95		ns	
TAVRL, TAVWL	Command Valid to $\overline{\text{NPWR}}$ or $\overline{\text{NPRD}}$ active	0		ns	
TMHRL	Minimum response from PEREQ active set up to $\overline{\text{NPRD}}$ active	130		ns	
TKLKH	$\overline{\text{PEACK}}$ active time	95		ns	
TKHKL	$\overline{\text{PEACK}}$ inactive time	95		ns	
TKHCH	$\overline{\text{PEACK}}$ inactive to $\overline{\text{NPWR}}$, $\overline{\text{NPRD}}$ inactive	0		ns	
TCHKL	Data Channel $\overline{\text{NPRD}}$, $\overline{\text{NPWR}}$ inactive set up to $\overline{\text{PEACK}}$ active	−25		ns	
TWHAX, TRHAX	Command hold from $\overline{\text{NPWR}}$, $\overline{\text{NPRD}}$ inactive	0		ns	
TKLCL	$\overline{\text{PEACK}}$ active set up to command active	0		ns	
T2CLCL	80286 Clock period	62.5	250	ns	
T2CLCH	80286 Clock low time	15	230	ns	at 0.8V
T2CHCL	80286 Clock high time	20	235	ns	at 2.0V

A.C. CHARACTERISTICS T_A=0°C to 70°C, V_{CC}= 5V± 10%
TIMING REQUIREMENTS (cont.)

Symbol	Parameter	Min.	Max.	Units	Test Conditions
TCLSH	$\overline{S1}$, $\overline{S0}$ hold time	0		ns	at 1.5 V
TSVCL	$\overline{S1}$, $\overline{S0}$ valid setup time	22.5		ns	
TCIVCL	COD/$\overline{\text{INTA}}$ valid setup time	0		ns	
TCLCIH	COD/$\overline{\text{INTA}}$ hold time	0		ns	
TRVCL	$\overline{\text{READY}}$ valid setup time	38.5		ns	
TCLRH	$\overline{\text{READY}}$ hold time	25		ns	
THVCL	HLDA valid setup time	0		ns	
TCLHH	HLDA hold time	0		ns	
TCLIH	Input from CLK hold time	45		ns	at 1.5V (See Note 1)
TIVCH	Input to CLK setup time	70		ns	

NOTE:
1. To guarantee a predetermined response for testing purposes.

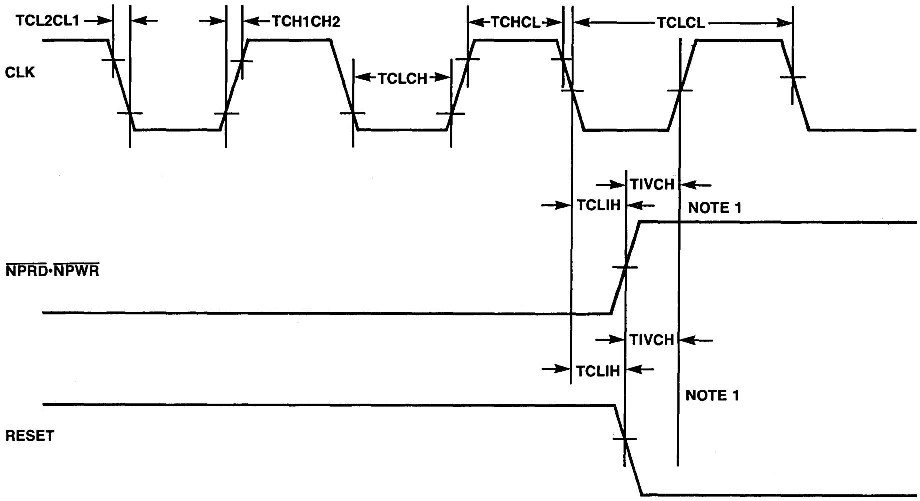
A.C. CHARACTERISTICS—TIMING RESPONSES

Symbol	Parameter	Min.	Max.	Units	Test Conditions
TRHQZ	$\overline{\text{NPRD}}$ inactive to data tri-state		37.5	ns	CL=20pF−100pF at 1.5V
TRLQV	$\overline{\text{NPRD}}$ active to data valid		60	ns	
TILBH	$\overline{\text{ERROR}}$ active to $\overline{\text{BUSY}}$ inactive	100		ns	CL=100pF at 1.5V
TWLBV	$\overline{\text{NPWR}}$ active to $\overline{\text{BUSY}}$ valid		100	ns	
TCLML	$\overline{\text{NPRD}}$, $\overline{\text{NPWR}}$ active to PEREQ inactive		120	ns	CL = 100pF at 1.5V (See Note 1)
TKLML	$\overline{\text{PEACK}}$ active to PEREQ inactive		127	ns	
TCMDI	Minimum command inactive time Write-Write Read-Read Write-Read Read-Write	 95 150 105 95		 ns ns ns ns	at 1.5V
TRHQH	Data valid hold from $\overline{\text{NPRD}}$ inactive	5		ns	CL=40pF at 1.5V

NOTE:
1. On last data transfer of numeric instruction.

WAVEFORMS

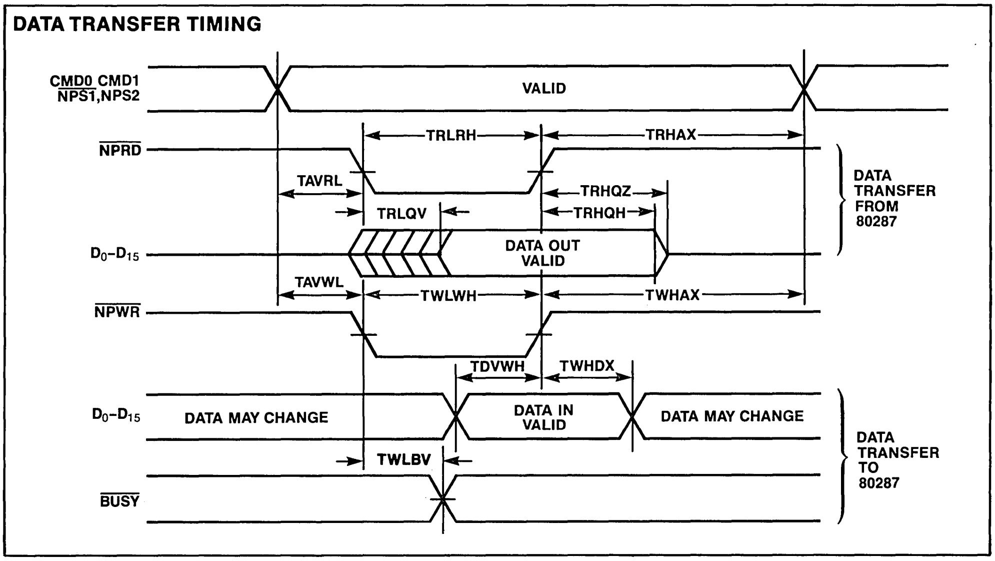
CLOCK TIMING

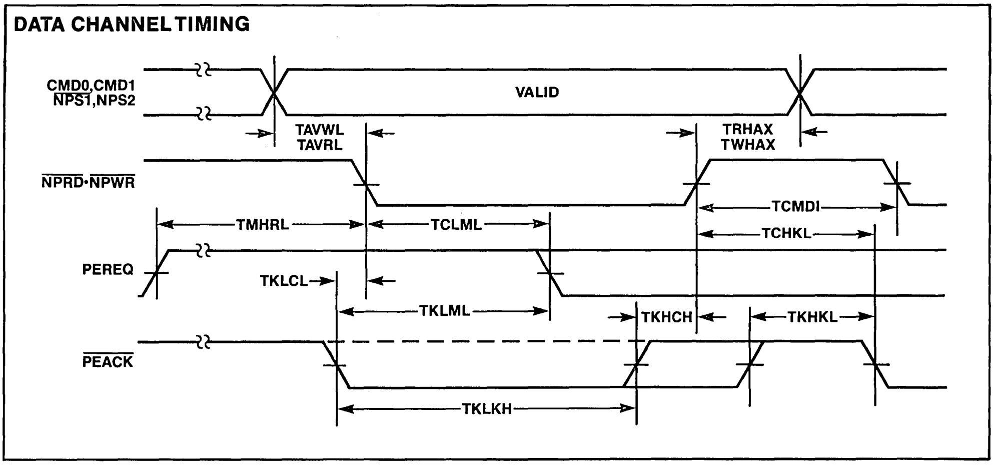


NOTE:

1. Shown for CKM=1, when CKM=0 TIVCH is measured from the second falling edge of CLK. These inputs may be asynchronous to CLK. This input timing guarantees a predetermined response.

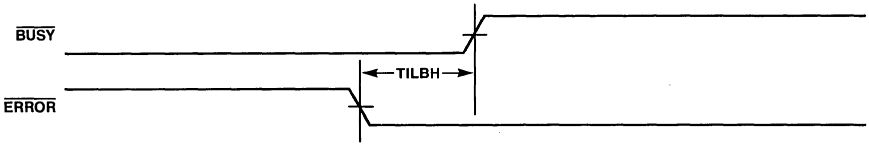
WAVEFORMS (cont.)



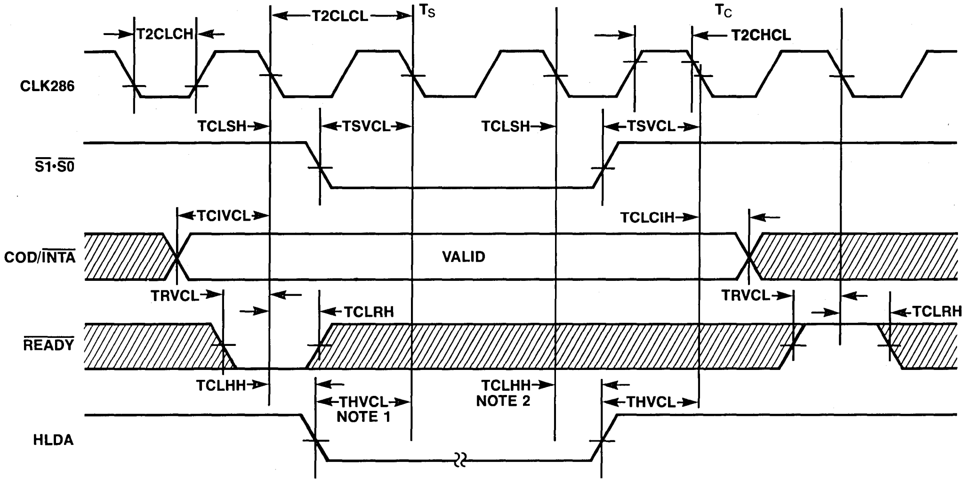


WAVEFORMS (cont.)

ERROR OUTPUT TIMING



80286 STATUS TIMING



NOTES:
1. This input transition occurs before T_S.
2. This input transition occurs after T_C.

Table 6. 80287 Extensions to the 80286 Instruction Set

Data Transfer			Optional 8,16 Bit Displacement	Clock Count Range			
				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	52-60	40-60	46-54
Long Integer Memory to ST(0)	ESCAPE 1 1 1	MOD 1 0 1 R/M	DISP	60-68			
Temporary Real Memory to ST(0)	ESCAPE 0 1 1	MOD 1 0 1 R/M	DISP	53-65			
BCD Memory to ST(0)	ESCAPE 1 1 1	MOD 1 0 0 R/M	DISP	290-310			
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	82-92	96-104	80-90
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	84-94	98-106	82-92
ST(0) to Long Integer Memory	ESCAPE 1 1 1	MOD 1 1 1 R/M	DISP	94-105			
ST(0) to Temporary Real Memory	ESCAPE 0 1 1	MOD 1 1 1 R/M	DISP	52-58			
ST(0) to BCD Memory	ESCAPE 1 1 1	MOD 1 1 0 R/M	DISP	520-540			
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	78-91	65-75	72-86
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	80-93	67-77	74-88
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			

Table 6. 80287 Extensions to the 80286 Instruction Set (cont.)

Constants			Optional 8,16 Bit Displacement	Clock Count Range			
				32 Bit Real	32 Bit Integer	64 Bit Real	16 Bit Integer
	MF	=		00	01	10	11
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 π 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	108–143	95–125	102–137
ST(i) and ST(0)	ESCAPE d P 0	1 1 0 0 0 ST(i)		70–100 (Note 1)			
FSUB = Subtraction							
Integer/Real Memory with ST(0)	ESCAPE MF 0	MOD 1 0 R R/M	DISP	90–120	108–143	95–125	102–137
ST(i) and ST(0)	ESCAPE d P 0	1 1 1 0 R 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	130–144	112–168	124–138
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 R/M	DISP	215–225	230–243	220–230	224–238
ST(i) and ST(0)	ESCAPE d P 0	1 1 1 1 R 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			

NOTE:
1. If P=1 then add 5 clocks.

Table 6. 80287 Extensions to the 80286 Instruction Set (cont.)

			Optional 8,16 Bit Displacement	Clock Count Range
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 = Initialize NPX	ESCAPE 0 1 1	1 1 1 0 0 0 1 1		2–8
FSETPM = Enter Protected Mode	ESCAPE 0 1 1	1 1 1 0 0 1 0 0		2–8
FSTSW AX = Store Control Word	ESCAPE 1 1 1	1 1 1 0 0 0 0 0		10–16
FLDCW = Load Control Word	ESCAPE 0 0 1	MOD 1 0 1 R/M	DISP	7–14
FSTCW = Store Control Word	ESCAPE 0 0 1	MOD 1 1 1 R/M	DISP	12–18
FSTSW = Store Status Word	ESCAPE 1 0 1	MOD 1 1 1 R/M	DISP	12–18
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
FLDENV = Load Environment	ESCAPE 0 0 1	MOD 1 0 0 R/M	DISP	35–45
FSAVE = Save State	ESCAPE 1 0 1	MOD 1 1 0 R/M	DISP	205–215
FRSTOR = Restore State	ESCAPE 1 0 1	MOD 1 0 0 R/M	DISP	205–215
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

Table 6. 80287 Extensions to the 80286 Instruction Set (cont.)

			Clock Count Range
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

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 mod=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**: $-0 \leq ST(0) \leq +\infty$
   For **FSCALE**: $-2^{15} \leq ST(1) < +2^{15}$ and ST(1) integer
   For **F2XM1**: $0 \leq ST(0) \leq 2^{-1}$
   For **FYL2X**: $0 < ST(0) < \infty$
   $-\infty < ST(1) < +\infty$
   For **FYL2XP1**: $0 \leq |ST(0)| < (2 - \sqrt{2})/2$
   $-\infty < ST(1) < \infty$
   For **FPTAN**: $0 \leq ST(0) \leq \pi/4$
   For **FPATAN**: $0 \leq ST(0) < ST(1) < +\infty$

intel

INTEL CORPORATION, 3065 Bowers Avenue, Santa Clara, CA 95051 (408) 987-8080

Intel International, Brussels, Belgium; Tel (02)661 07 11.

Intel Japan k.k., Ibaraki-ken; Tel. 029747-8511.

Printed in U.S.A./T-2402/25K/0383/JOS RM