

Am29C327

Double-Precision Floating-Point Processor

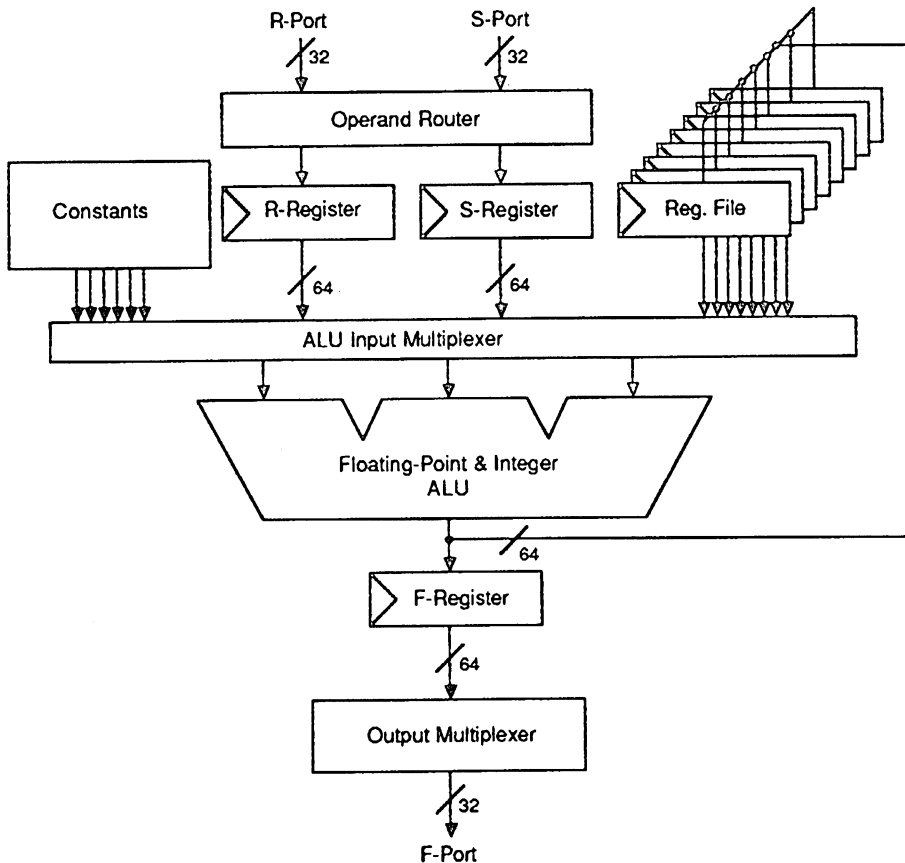


ADVANCE INFORMATION

DISTINCTIVE CHARACTERISTICS

- High-performance double-precision floating-point processor
- Comprehensive floating-point and integer instruction sets
- Single VLSI device performs single-, double-, and mixed-precision operations
- Performs conversions between precisions and between data formats
- Compatible with industry-standard floating-point formats
 - IEEE 754 format
 - DEC F, DEC D, and DEC G formats
 - IBM system/370 format
- Exact IEEE compliance for denormalized numbers with no speed penalty
- Eight-deep register file for intermediate results and on-chip 64-bit data path facilitates compound operations; e.g., Newton-Raphson division, sum-of-products, and transcendentals
- Supports pipelined or flow-through operation
- Fabricated with Advanced Micro Devices' 1.2 micron CMOS process

SIMPLIFIED SYSTEM DIAGRAM



BD007470

CONTENTS

DISTINCTIVE CHARACTERISTICS	1
SIMPLIFIED BLOCK DIAGRAM	1
GENERAL DESCRIPTION	2
RELATED AMD PRODUCTS	3
CONNECTION DIAGRAM	3
PIN DESIGNATIONS	4
LOGIC SYMBOL	5
ORDERING INFORMATION	5
PIN DESCRIPTION	6
FUNCTIONAL DESCRIPTION	6
Overview	6
Block Diagram Description	9
Mode Register Description	10
Input Modes	11
Pipelining of Operations	20
Instruction Set	24
Master/Slave Operation	32
APPENDICES	33
Appendix A — Data Formats	33
Appendix B — Rounding Modes	37
Appendix C — Additional Operation Details	40
ABSOLUTE MAXIMUM RATINGS	41
OPERATING RANGES	41
DC CHARACTERISTICS	41
SWITCHING CHARACTERISTICS	42
SWITCHING TEST CIRCUITS	43
SWITCHING TEST WAVEFORMS	44
SWITCHING WAVEFORMS	45
PHYSICAL DIMENSIONS	50

GENERAL DESCRIPTION

The Am29C327 double-precision floating-point processor is a single VLSI device that implements an extensive floating-point and integer instruction set, and can perform single-, double- or mixed-precision operations. The three most popular floating-point formats — IEEE, DEC, and IBM — are supported. IEEE operations comply with Standard 754, with direct implementation of special features such as gradual underflow and trap handling.

The Am29C327 consists of a 64-bit ALU, a 64-bit datapath, and a control unit. The ALU has three data input ports, and can perform compound operations of the form $(A * B) + C$. The data path comprises two 64-bit input operand registers, an 8-by-64-bit register file for storage of intermediate results, three operand-selection multiplexers that provide for orthogo-

nal selection of input operands, a 64-bit output register, and an output multiplexer that allows access to the 32 MSBs or 32 LSBs of the result data. Control signals determine the operation to be performed, the source of operands, operand precision, rounding mode, and other aspects of device operation.

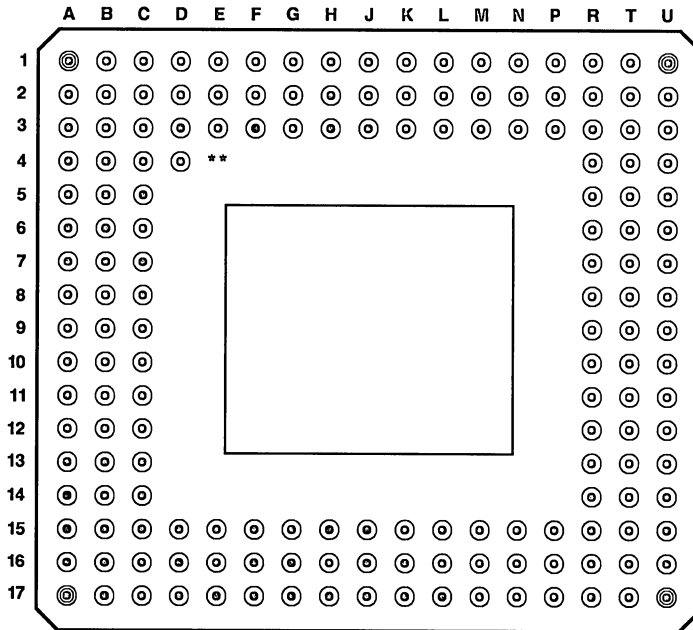
Operations can be performed in either of two modes: flow-through or pipelined. In the flow-through mode, the ALU is completely combinatorial; this mode is best suited for scalar operations. Pipelined mode divides the ALU into one or two pipelined stages, for use in vector operations, as often found in graphics or signal processing.

Fabricated with AMD's 1.2 micron technology, the Am29C327 is housed in a 169-lead pin-grid-array (PGA) package.

RELATED AMD PRODUCTS

Part No.	Description
Am29C10A	CMOS Microprogram Controller
Am29C116	CMOS Minimum Power 16-Bit Microprocessor
Am29C117	CMOS Two-Port 16-Bit Microprocessor
Am29PL141	Field-Programmable Controller (FPC)
Am29C323	CMOS 32-Bit Parallel Multiplier
Am29C325	CMOS 32-Bit Floating-Point Processor
Am29C331	CMOS 16-Bit Microprogram Sequencer
Am29C332	CMOS 32-Bit Arithmetic Logic Unit
Am29C334	CMOS Four-Port Dual-Access Register File

CONNECTION DIAGRAM 169-Lead PGA* Bottom View



CD009761

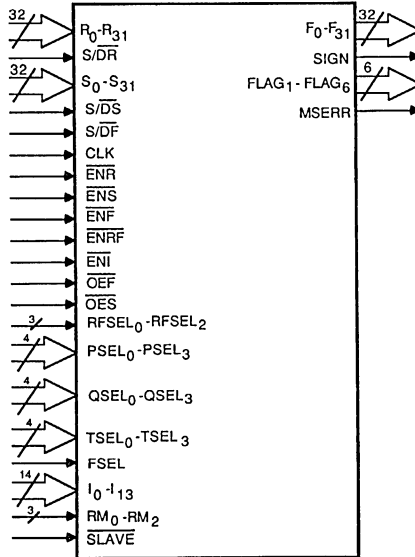
*Pinout observed from pin side of package.

**Alignment pin (not connected internally).

PIN DESIGNATIONS
(Sorted by Pin No.)

PIN NO.	PIN NAME	PIN NO.	PIN NAME	PIN NO.	PIN NAME	PIN NO.	PIN NAME
A-1		C-9		J-15		R-10	
A-2		C-10		J-16		R-11	
A-3		C-11		J-17		R-12	
A-4		C-12		K-1		R-13	
A-5		C-13		K-2		R-14	
A-6		C-14		K-3		R-15	
A-7		C-15		K-15		R-16	
A-8		C-16		K-16		R-17	
A-9		C-17		K-17		T-1	
A-10		D-1		L-1		T-2	
A-11		D-2		L-2		T-3	
A-12		D-3		L-3		T-4	
A-13		D-15		L-15		T-5	
A-14		D-16		L-16		T-6	
A-15		D-17		L-17		T-7	
A-16		E-1		M-1		T-8	
A-17		E-2		M-2		T-9	
B-1		E-3		M-3		T-10	
B-2		E-15		M-15		T-11	
B-3		E-16		M-16		T-12	
B-4		E-17		M-17		T-13	
B-5		F-1		N-1		T-14	
B-6		F-2		N-2		T-15	
B-7		F-3		N-3		T-16	
B-8		F-15		N-15		T-17	
B-9		F-16		N-16		U-1	
B-10		F-17		N-17		U-2	
B-11		G-1		P-1		U-3	
B-12		G-2		P-2		U-4	
B-13		G-3		P-3		U-5	
B-14		G-15		P-15		U-6	
B-15		G-16		P-16		U-7	
B-16		G-17		P-17		U-8	
B-17		H-1		R-1		U-9	
C-1		H-2		R-2		U-10	
C-2		H-3		R-3		U-11	
C-3		H-15		R-4		U-12	
C-4		H-16		R-5		U-13	
C-5		H-17		R-6		U-14	
C-6		J-1		R-7		U-15	
C-7		J-2		R-8		U-16	
C-8		J-3		R-9		U-17	

LOGIC SYMBOL



LS003080

ORDERING INFORMATION

Standard Products

AMD standard products are available in several packages and operating ranges. The order number (Valid Combination) is formed by a combination of:

- a. Device Number
- b. Speed Option (if applicable)
- c. Package Type
- d. Temperature Range
- e. Optional Processing

AM29C327

G

C

B

e. OPTIONAL PROCESSING

Blank = Standard processing
B = Burn-in

d. TEMPERATURE RANGE

C = Commercial (0 to +70°C)

c. PACKAGE TYPE

G = 169-Terminal Pin Grid Array (CGX169)

b. SPEED OPTION

Not Applicable

a. DEVICE NUMBER/DESCRIPTION

Am29C327
Double-Precision Floating-Point Processor

Valid Combinations	
AM29C327	GC, GCB

Valid Combinations

Valid Combinations list configurations planned to be supported in volume for this device. Consult the local AMD sales office to confirm availability of specific valid combinations, to check on newly released combinations, and to obtain additional data on AMD's standard military grade products.

PIN DESCRIPTION

CLK Clock (Input)

Clock input to all registers.

ENF F Register Enable (Input; Active LOW)

When $\overline{\text{ENF}}$ is HIGH, the contents of the F register are static. When $\overline{\text{ENF}}$ is LOW, the ALU output data is clocked into the F register on the next LOW-to-HIGH transition of CLK. Note that the F register can be made transparent by setting the mode register bit M17 HIGH (as described in the Mode Register Description section); when the F register is transparent, $\overline{\text{ENF}}$ has no effect.

ENI Instruction Register Enable (Input; Active LOW)

When $\overline{\text{ENI}}$ is LOW, an instruction word is clocked into the instruction register on the next LOW-to-HIGH transition of CLK. The instruction word comprises the following fields: P, Q, and T-multiplexer control inputs, rounding modes, ALU instruction inputs, and the precision of the output operand.

ENR R Register Enable (Input; Active LOW)

When $\overline{\text{ENR}}$ is HIGH, the contents of the R register are static. When $\overline{\text{ENR}}$ is LOW, new data is loaded into the R register on the next LOW-to-HIGH transition of CLK.

ENRF Register File Enable (Input; Active LOW)

When $\overline{\text{ENRF}}$ is HIGH, the contents of the register file are static. When $\overline{\text{ENRF}}$ is LOW, the ALU output operand is clocked into the register file on the next LOW-to-HIGH transition of CLK.

ENS S Register Enable (Input; Active LOW)

When $\overline{\text{ENS}}$ is HIGH, the contents of the S register are static. When $\overline{\text{ENS}}$ is LOW, new data is loaded into the S register on the next LOW-to-HIGH transition of CLK.

F₀ – F₃₁ F Output Bus (Output)

FLAG₁ – FLAG₆ Flag Outputs (Output)

The six flag outputs report the status of the last operation executed.

FSEL Output Multiplexer Control (Input)

When FSEL is HIGH, the most significant 32 bits of the output register are connected to the output driver. When FSEL is LOW, the least significant 32 bits of the output register are connected to the output driver.

I₀ – I₁₃ ALU Instruction Inputs (Input)

I₀ – I₁₃ select the operation to be performed by the ALU.

MSERR Master/Slave Error Flag (Output)

A HIGH level indicates a master/slave error on the current output.

$\overline{\text{OEF}}$ F Output Bus Enable (Input; Active LOW)

When $\overline{\text{OEF}}$ is HIGH, signals F₀ – F₃₁ assume a high-impedance state. When $\overline{\text{OEF}}$ is LOW (and $\overline{\text{SLAVE}}$ is HIGH), the output of the F multiplexer is placed on F₀ – F₃₁.

$\overline{\text{OES}}$ Flag Output Enable (Input)

When $\overline{\text{OES}}$ is HIGH, outputs SIGN and FLAG₁ through FLAG₆ assume a high-impedance state. When $\overline{\text{OES}}$ is LOW (and $\overline{\text{SLAVE}}$ is HIGH), these signals are enabled.

PSEL₀ – PSEL₃ P-Multiplexer Control Inputs (Input)

PSEL₀ – PSEL₃ select the data input to the ALU P-port.

QSEL₀ – QSEL₃ Q-Multiplexer Control Inputs (Input)

QSEL₀ – QSEL₃ select the data input to the ALU Q-port.

R₀ – R₃₁ R Input Bus (Input)

RFSEL₀ – RFSEL₂ Register File Select (Input)

RFSEL₀ – RFSEL₂ select the register file location (RF₀ – RF₇) to which the ALU result is to be written. Data is written to the register file if $\overline{\text{ENRF}}$ is LOW.

RM₀ – RM₂ Round Mode Control Inputs (Input)

The Am29C327 supports six rounding modes. RM₀ – RM₂ select the rounding mode to be applied to the current operation.

S₀ – S₃₁ S Input Bus (Input)

S/ $\overline{\text{DF}}$ F Output Single/Double Control (Input)

When S/ $\overline{\text{DF}}$ is HIGH, the ALU generates a single-precision result. When S/ $\overline{\text{DF}}$ is LOW, the ALU generates a double-precision result.

S/ $\overline{\text{DR}}$ R Input Single/Double Control (Input)

When S/ $\overline{\text{DR}}$ is HIGH, the data loaded into the R-port is treated as single precision. When S/ $\overline{\text{DR}}$ is LOW, the data loaded into the R register is treated as double precision.

S/ $\overline{\text{DS}}$ S Input Single/Double Control (Input)

When S/ $\overline{\text{DS}}$ is HIGH, the data loaded into the S-port is treated as single precision. When S/ $\overline{\text{DS}}$ is LOW, the data loaded into the S register is treated as double precision.

SIGN Sign Flag (Output)

If the final result of the last operation was negative, SIGN is HIGH. If the final result of the last operation was not negative, SIGN is LOW.

SLAVE Master/Slave Mode Select (Input)

When SLAVE is LOW, SLAVE mode is selected. In this mode, all outputs except MSERR are disabled. When SLAVE is HIGH, MASTER mode is selected.

TSEL₀ – TSEL₃ T-Multiplexer Control Inputs (Input)

TSEL₀ – TSEL₃ select the data input to the ALU T-port.

FUNCTIONAL DESCRIPTION

Overview

The Am29C327 is a high-performance, single-chip, double-precision floating-point processor.

Architecture

The Am29C327 comprises a high-speed ALU, a 64-bit data path, and control circuitry.

The core of the Am29C327 is a 64-bit floating-point/integer ALU. This ALU takes operands from three 64-bit input ports and performs the selected operation, placing the result on a 64-bit output port. Thirteen ALU flags report operation status via the 7-bit Flag port. The ALU is completely combinatorial for

reduced latency; optional pipelining is available to boost throughput for array operations.

The data path consists of the 32-bit input buses R and S; two 64-bit input operand registers; an 8-by-64-bit register file for storage of intermediate results; three operand-selection multiplexers that provide for orthogonal selection of input operands; a 64-bit output register; and an output multiplexer that permits the selection of 32 MSBs, or 32 LSBs of data. Input operands enter the processor through the R and S buses, and are then demultiplexed and buffered for subsequent storage in registers R and S. The operand selection multiplexers route the operands to the ALU. Operation results are stored in register F, and leave the device on the 32-bit output bus F. The results can also be stored in the register file for use in subsequent operations.

Instruction Set

The Am29C327 implements 58 arithmetic and logical instructions. Thirty-five instructions operate on floating-point numbers; these instructions fall into the following categories:

- Addition/subtraction
- Multiplication
- Multiplication-accumulation
- Comparison
- Selecting the larger or smaller of two numbers
- Rounding to integral value
- Absolute value, negation
- Reciprocal seed generation
- Conversion between any of the supported floating-point formats
- Conversion of a floating-point number to an integer format, with or without a scale factor
- Pass operand

By concatenating these operations, the user can also perform division, square-root extraction, polynomial evaluation, and other functions not implemented directly.

Twenty-two instructions operate on integers, and belong to the following general categories:

- Addition/subtraction
- Multiplication
- Comparison
- Selecting the smaller or larger of two numbers
- Absolute value, negation, pass operand

- Logical operations; e.g., AND, OR, XOR, NOT
- Arithmetic, logical, and funnel shifts
- Conversion between single- and double-precision integer formats
- Conversion of an integer number to a floating-point format, with or without a scale factor

One special instruction is provided to move data.

Mixed-Precision Operations

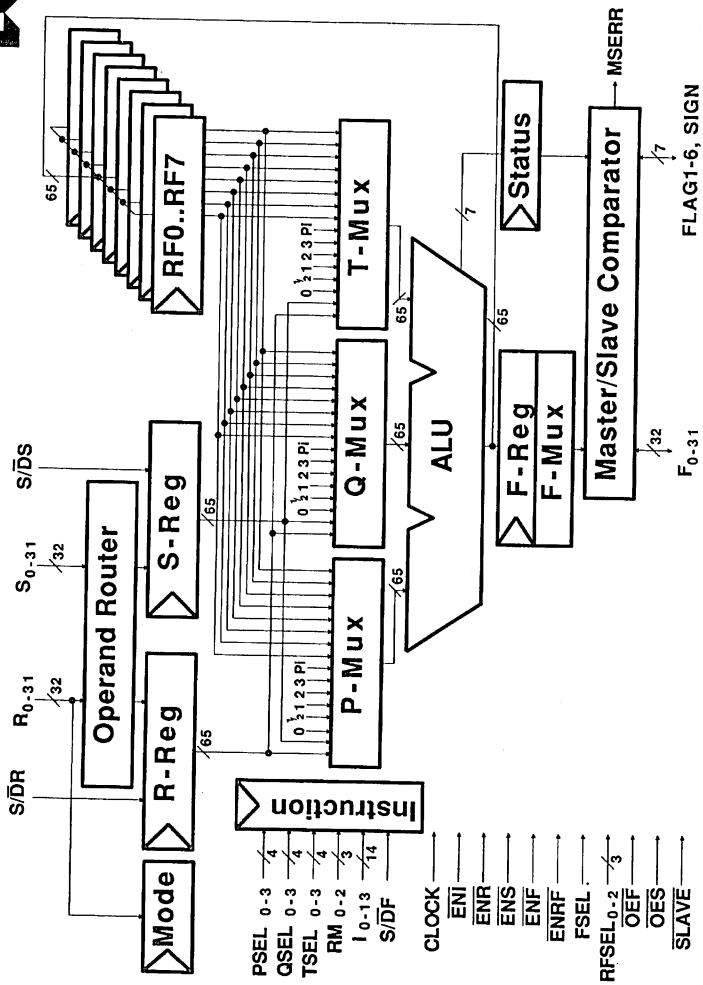
All Am29C327 instructions, floating-point or integer, can be performed with either single- or double-precision operands. In addition, the user can elect to mix precisions within an operation. All operations are performed in double-precision internally; the user specifies the precisions of the input operands and the required precision for the output operand. The necessary precision conversions are made in concert with the selected operation, with no additional cycle-time overhead.

I/O Modes

The Am29C327 supports eight I/O modes that afford flexible interface to a variety of 32- and 64-bit systems.

Fault Detection Features

The Am29C327 contains special comparison hardware to allow the operation of two processors in parallel, with one processor (the slave) checking the results produced by the other (the master). This feature is of particular importance in the design of high-reliability systems.



BD011050

Figure 1. Am29C327 Double-Precision Floating-Point Processor

Block Diagram Description

A block diagram of the Am29C327 is shown in Figure 1. The Am29C327 comprises input registers, operand selection multiplexers, instruction register, ALU, output register/register file, status register, output selection multiplexer, mode register, and the master/slave comparator.

Input Registers/Input Modes

Operands enter the processor through the R and S buses, and are then demultiplexed and buffered for subsequent storage in the 65-bit registers R and S. Input operands may be either single-precision (32-bit) or double-precision (64-bit) as specified by S/DR and S/DS. Accompanying the input registers are two 32-bit temporary registers, R-Temp and S-Temp, that allow for the overlapping of operand transfers and ALU operations. This arrangement of temporary registers and demultiplexers permits data and corresponding precision bit S/DR or S/DS to be loaded into the 65-bit R register and 65-bit S register via one of the eight input modes:

1. 32-bit-bus, double-cycle, LSWs first
2. 32-bit-bus, double-cycle, MSWs first
3. 32-bit-bus, single-cycle, LSWs first
4. 32-bit-bus, single-cycle, MSWs first
5. 64-bit-bus, double-cycle, R first
6. 64-bit-bus, double-cycle, S first
7. 64-bit-bus, single-cycle, R first
8. 64-bit-bus, single-cycle, S first

These modes are described in detail in the Input Modes Description section.

Operand Selection Multiplexers

The operand selection multiplexers route operands to the ALU. These multiplexers, as well as selecting operands from input registers R and S and register file locations RF0 – RF7, also have access to a set of constants (0, 0.5, 1, 2, 3, Pi). These constants are double-precision preprogrammed numbers for use in ALU operations, and are automatically provided in the appropriate floating-point or integer format.

Instruction Register

The instruction register stores a 32-bit word specifying the current processor operation. Included in the instruction word are fields that specify the P, Q, and T multiplexer selects, the rounding modes; the core operation to be performed by the ALU; sign-change controls for ALU input and result operands; and the single/double-precision control for the output operand. The multiplexer selects and the instruction word are described in detail in the Instruction Set section; Rounding modes are described in Appendix B.

ALU

The ALU is a combinatorial arithmetic/logic unit that performs a large repertoire of floating-point and integer operations. The

ALU has three operand inputs, and performs operations of the form $(P*Q) + T$. Most ALU operations require only one or two input operands; for example, addition requires only operands P and T, multiplication only operands P and Q, and precision conversion only operand P. Many ALU arithmetic operations allow for the independent control of operand signs, thus greatly increasing the number of arithmetic expressions that can be evaluated in a single ALU pass.

The ALU can be configured in either a flow-through mode, for which the ALU is completely combinatorial, or a pipelined mode, for which ALU operations incur one or two pipeline delays, but which results in a higher throughput than flow-through mode.

A detailed description of ALU operations appears in the Instruction Set section.

Output Register/Register File

The results of the operations performed by the ALU are stored in the 64-bit output register F. Results can also be stored in the 8-by-64-bit register file for use in subsequent operations. Each register file location contains a 65th bit indicating the precision of the operand stored in that location, thus permitting the ALU to correctly process the operand in subsequent operations.

Status Register

The status register is a 7-bit register that stores flags pertaining to the most recently performed operation. A detailed description is provided in the Instruction Set section.

Output Multiplexer

The output multiplexer routes operation results to the F bus. This multiplexer selects the 32 MSBs of the output register or the 32 LSBs.

Mode Register

The mode register contains processor parameters that are changed infrequently. The 32-bit mode word is loaded into the register via the R bus. A detailed description of the mode register is provided in the Mode Register Description section.

Master/Slave Comparator

Each Am29C327 output signal has associated logic that compares that signal with the signal that the processor is providing internally to the output driver; any discrepancies are indicated by assertion of signal MSERR.

For a single processor, this output comparison detects short circuits in output signals or defective output drivers, but does not detect open circuits. It is possible to connect a second processor in parallel with the first, with the second processor's outputs disabled by assertion of signal SLAVE. The second processor detects open-circuit signals, as well as providing a check of the outputs of the first.

Mode Register Description

The "Load Mode Register" instruction loads a 32-bit word appearing on the R port into the mode register. Data is clocked into the register on the LOW-to-HIGH transition of CLK. The register is organized as described below:

M0 – M3 — Floating-Point Format Select:

M1	M0	Primary Format
0	0	IEEE
0	1	DEC F (SINGLE), DEC D (DOUBLE)
1	0	DEC F (SINGLE), DEC G (DOUBLE)
1	1	IBM
M3	M2	Alternate Format
0	0	IEEE
0	1	DEC F (SINGLE), DEC D (DOUBLE)
1	0	DEC F (SINGLE), DEC G (DOUBLE)
1	1	IBM

Floating-point formats are discussed in further detail in Appendix A.

M4 — Saturate Enable: If M4 is HIGH, overflowed results are replaced by the largest representable value in the selected format of the same sign as the overflowed result. If M4 is LOW, the result is not changed. If M6 is HIGH, saturation is disabled.

M5 — IEEE Affine/Projective Select: If M5 is HIGH, affine mode is selected. If M5 is LOW, projective mode is selected. The interpretation of infinities is determined by M5. The only differences between the modes occur during the addition and subtraction of infinities.

Operation	Affine Mode	Projective Mode
$(+\infty) + (+\infty)$	Output $+\infty$	Output Quiet NAN, set invalid and reserved operand flags
$(-\infty) + (-\infty)$	Output $-\infty$	Output Quiet NAN, set invalid and reserved operand flags
$(+\infty) - (-\infty)$	Output $+\infty$	Output Quiet NAN, set invalid and reserved operand flags
$(-\infty) - (+\infty)$	Output $-\infty$	Output Quiet NAN, set invalid and reserved operand flags

If the current floating-point format is not IEEE, this bit has no effect.

M6 — IEEE Trap Enable: If M6 is HIGH, IEEE trapped operation is enabled; the saturate (M4) and sudden underflow (M7) bits are ignored. For an underflowed result, the exponent is replaced by $e = e + 192$ (SP), or $e = e + 1536$ (DP), with the significand unchanged. For an overflowed result, the exponent is replaced by $e = e - 192$ (SP), or $e = e - 1536$ (DP), with the significand unchanged. If M6 is LOW, IEEE trapped operation is disabled.

M7 — IEEE Sudden Underflow Enable: If M7 is HIGH and IEEE traps are disabled (M6 LOW), all IEEE denormalized results are replaced by a zero of the same sign. If M7 is LOW, a valid denormalized number will be produced. This bit has no effect for formats other than IEEE.

M8 — IBM Significance Mask Enable: If M8 is HIGH, certain IBM operations having intermediate results of 0 will produce a final result of 0 with the biased exponent unchanged. If M8 is

LOW, these operations will produce a final result of true-zero. This bit has no effect for formats other than IBM.

M9 — IBM Underflow Mask Enable: If M9 is HIGH, certain underflowed IBM operations will produce a normalized result with the exponent replaced by $e + 128$. If M9 is LOW, these operations will produce a final result of true-zero. This bit has no effect for formats other than IBM.

M10: Reserved for future use (must be set to Logic 0)

M11 — Integer Multiplication Signed/Unsigned Select: If M11 is HIGH, the input operands are treated as two's-complement numbers. If M11 is LOW, the input operands are treated as unsigned numbers. This bit has no effect for operations other than integer multiplication.

M12, M13 — Integer Multiplication Format Adjust: Selects the output format for integer multiplications. The user may select either the MSBs or the LSBs of the result of an integer multiplication:

M13	M12	Output Format
0	0	LSBs
0	1	LSBs, format-adjusted
1	0	MSBs
1	1	MSBs, format adjusted

"Format-adjusted" indicates that the product is shifted left one place before the MSBs or LSBs are selected.

M14 – M16 — Input Mode: Selects the input bus mode:

M16	M15	M14	Input Mode
0	0	0	32-bit-bus, single-cycle, LSW first
0	0	1	32-bit-bus, single-cycle, MSW first
0	1	0	32-bit-bus, double-cycle, LSW first
0	1	1	32-bit-bus, double-cycle, MSW first
1	0	0	64-bit-bus, single-cycle, R first
1	0	1	64-bit-bus, single-cycle, S first
1	1	0	64-bit-bus, double-cycle, R first
1	1	1	64-bit-bus, double-cycle, S first

Additional information on input modes can be found in the Input Modes section.

M17 – F Register Feedthrough Enable: When M17 is HIGH, register F is made transparent. When M17 is LOW, the ALU output data is clocked into the F register on the next LOW-to-HIGH transition of CLK.

M18 – Status Register Feedthrough Enable: When M18 is HIGH, the status register is made transparent. When M18 is LOW, the output flags are clocked into the status register on the next LOW-to-HIGH transition on CLK.

M19, M20 – Pipeline Mode Select:

M20	M19	Pipeline Mode
0	X	Flow-through mode
1	0	Single-pipeline mode for all operations
1	1	Double-pipeline mode for multiply/accumulate Single-pipeline mode for other operations

M21 – M31 – Reserved for factory test (must be set to Logic 0)

Input Modes

The Am29C327 supports a total of eight input modes for loading data into the R and S registers.

The 32-bit bus modes allow the user to connect each input port ($R_0 - R_{31}$ and $S_0 - S_{31}$) to separate 32-bit buses. 64-bit operands can then be loaded by placing the MSBs and LSBs alternately on the appropriate ports. In the 64-bit bus modes, the two input ports are configured internally as a single 64-bit port. The Am29C327 may then be connected directly to a 64-bit bus, and 64-bit operands may be loaded in single operation. Either the 32-bit bus modes or the 64-bit bus modes may be used regardless of the precision of the operands being transferred — the choice of input modes will in practice be determined by the system into which the Am29C327 is to be integrated.

Single-cycle input modes allow two 64-bit operands to be loaded in a single clock cycle. This necessitates driving the input buses at twice the speed of the Am29C327. For systems when this is not practical, the double-cycle modes allow the loading of one 64-bit operand (or two 32-bit operands) per clock cycle.

Data may be loaded from the input buses to the R register and S register using one of the eight input modes:

1. 32-Bit Bus, Single-Cycle, LSWs First
2. 32-Bit Bus, Single-Cycle, MSWs First
3. 32-Bit Bus, Double-Cycle, LSWs First
4. 32-Bit Bus, Double-Cycle, MSWs First
5. 64-Bit Bus, Single-Cycle, R First
6. 64-Bit Bus, Single-Cycle, S First
7. 64-Bit Bus, Double-Cycle, R First
8. 64-Bit Bus, Double-Cycle, S First

The choice of the input modes is determined by mode register bits M14 – M16.

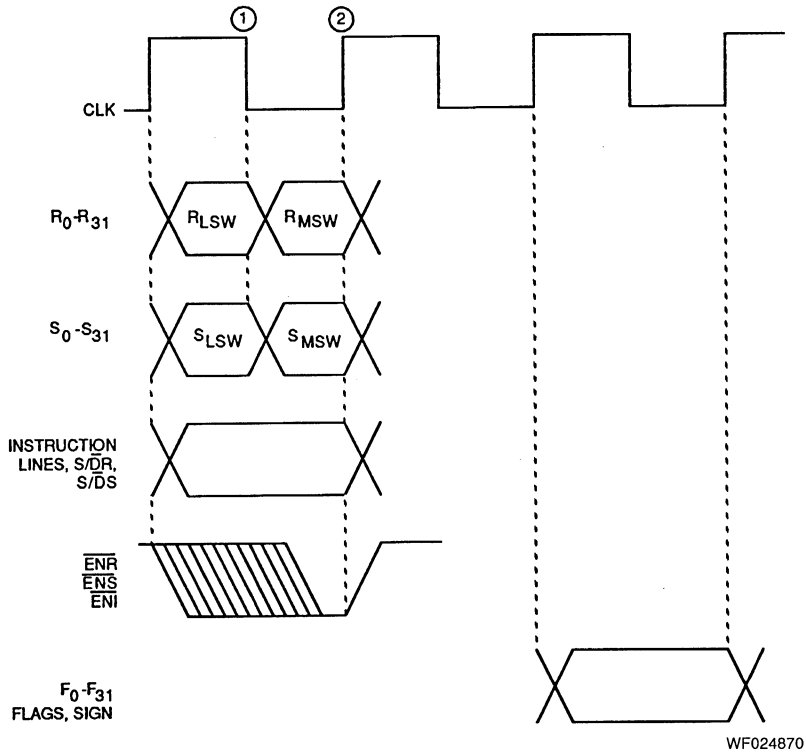
In order to permit the loading of new operands to be overlapped with the execution of a current operation, temporary registers are provided within the "operand router" block (shown in Figure 1). The operation of these temporary registers is transparent to the user. The conditions under which they are loaded depends on the input mode selected.

The eight input modes are described on the following pages.

32-Bit Bus, Single-Cycle, LSW First (M16 = 0, M15 = 0, M14 = 0)

In this mode, the two halves of the 64-bit R operand are placed on the R-input bus in successive half-cycles, with the S

operand similarly placed on the S-input port. After one complete cycle, the R and S registers contain the R and S operands, respectively.



Timing of Operations with Input Mode 1 (32-Bit Bus, Single-Cycle, LSW First)*

*Assumes flow-through operation, F register, and S register clocked.

In this mode, the temporary registers are clocked on every HIGH-to-LOW clock transition.

At 1, the least-significant 32 bits of the R operand are loaded from the R-input port into the R-temp register, and the least-significant 32 bits of the S operand are loaded from the S-input port into the S-temp register. Both words are loaded on the HIGH-to-LOW transition of the clock.

At 2, the most-significant 32 bits of the R operand are loaded from the R-input port into the most-significant half of the R

register, and the most-significant 32 bits of the S operand are loaded from the S-input port into the most-significant half of the S register.

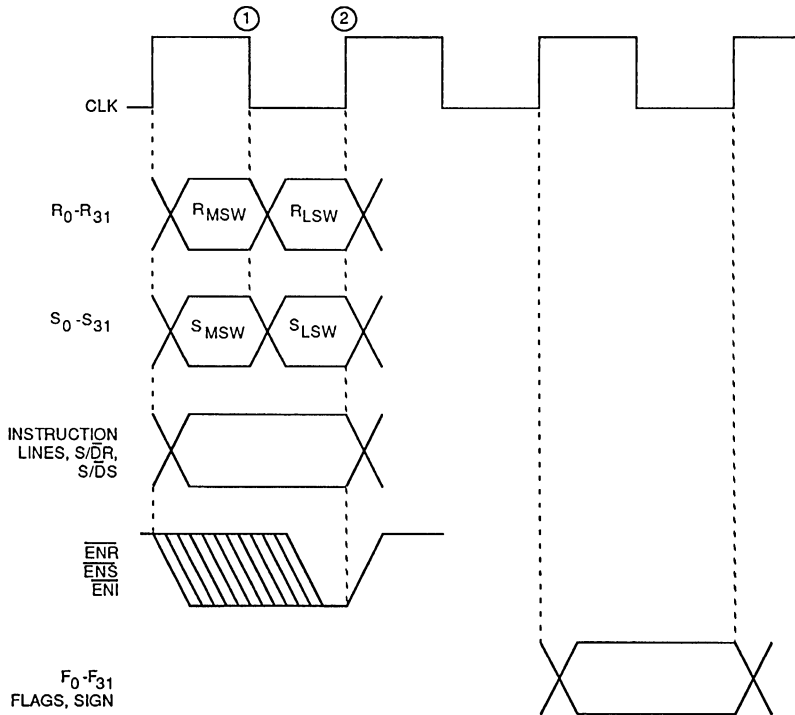
At the same time, at 2, the output of the R-temp register is loaded into the least-significant half of the R register, and the output of the S-temp register is loaded into the least-significant half of the S register.

If an input operand is single-precision, the 32-bit data is kept on the input bus for the full cycle.

32-Bit Bus, Single-Cycle, MSW First (M16 = 0, M15 = 0, M14 = 1)

In this mode, the two halves of the 64-bit R operand are placed on the R-input bus in successive half-cycles, with the S

operand similarly placed on the S-input port. After one complete cycle, the R and S registers contain the R and S operands, respectively.



WF024890

**Timing of Operations with Input Mode 2
(32-Bit Bus, Single-Cycle, MSW First)***

*Assumes flow-through operation, F register, and S register clocked.

In this mode, the temporary registers are clocked on every HIGH-to-LOW clock transition.

At 1, the most-significant 32 bits of the R operand are loaded from the R-input port into the R-temp register, and the most-significant 32 bits of the S operand are loaded from the S-input port into the S-temp register. Both words are loaded on the HIGH-to-LOW transition of the clock.

At 2, the least-significant 32 bits of the R operand are loaded from the R-input port into the least-significant half of the R

register, and the least-significant 32 bits of the S operand are loaded from the S-input port into the least-significant half of the S register.

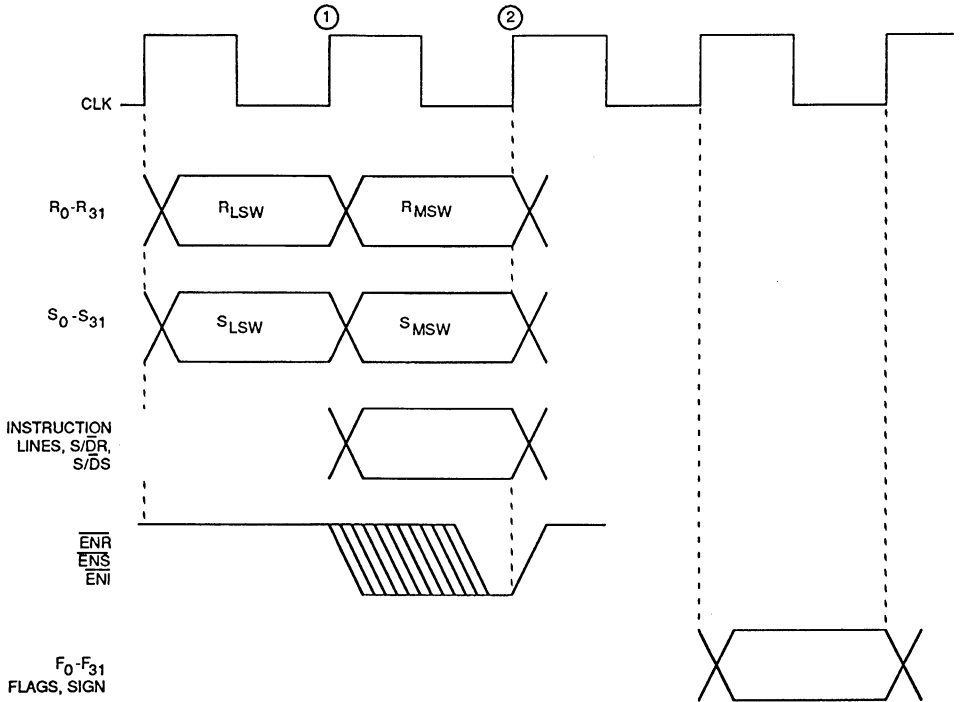
At the same time, at 2, the output of the R-temp register is loaded into the most-significant half of the R register, and the output of the S-temp register is loaded into the most-significant half of the S register.

If an input operand is single-precision, the 32-bit data is kept on the input bus for the full cycle.

32-Bit Bus, Double-Cycle, LSW First (M16 = 0, M15 = 1, M14 = 0)

In this mode, the two halves of the 64-bit R operand are placed on the R-input bus in successive cycles, with the S

operand similarly placed on the S-input port. After two cycles, the R and S registers contain the R and S operands, respectively.



WF024900

Timing of Operations with Input Mode 3 (32-Bit Bus, Double-Cycle, LSW First)*

*Assumes flow-through operation, F register, and S register clocked.

In this mode, the temporary registers are clocked on every LOW-to-HIGH clock transition.

At 1, the least-significant 32 bits of the R operand are loaded from the R-input port into the R-temp register, and the least-significant 32 bits of the S operand are loaded from the S-input port into the S-temp register.

At 2, the most-significant 32 bits of the R operand are loaded from the R-input port into the most-significant half of the R

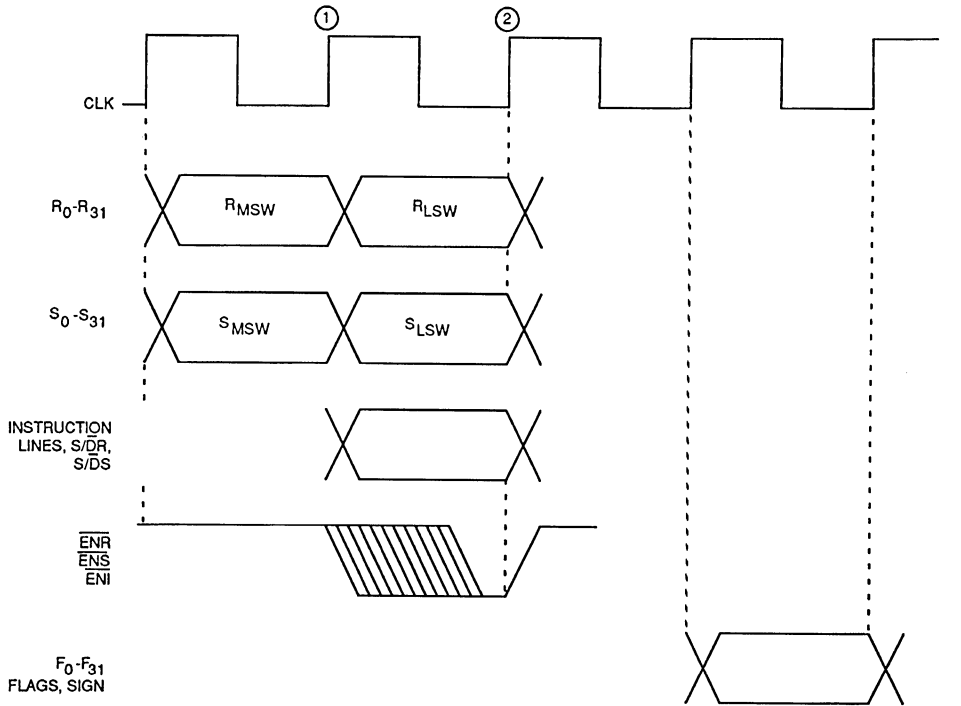
register, and the most-significant 32 bits of the S operand are loaded from the S-input port into the most-significant half of the S register.

At the same time, at 2, the output of the R-temp register is loaded into the least-significant half of the R register, and the output of the S-temp register is loaded into the least-significant half of the S register.

32-Bit Bus, Double-Cycle, MSW First (M16 = 0, M15 = 1, M14 = 1)

In this mode, the two halves of the 64-bit R operand are placed on the R-input bus in successive cycles, with the S

operand similarly placed on the S-input port. After two cycles, the R and S registers contain the R and S operands, respectively.



WF024910

**Timing of Operations with Input Mode 4
(32-Bit Bus, Double-Cycle, MSW First)***

*Assumes flow-through operation, F register, and S register clocked.

In this mode, the temporary registers are clocked on every LOW-to-HIGH clock transition.

At 1, the most-significant 32 bits of the R operand are loaded from the R-input port into the R-temp register, and the most-significant 32 bits of the S operand are loaded from the S-input port into the S-temp register.

At 2, the least-significant 32 bits of the R operand are loaded from the R-input port into the least-significant half of the R

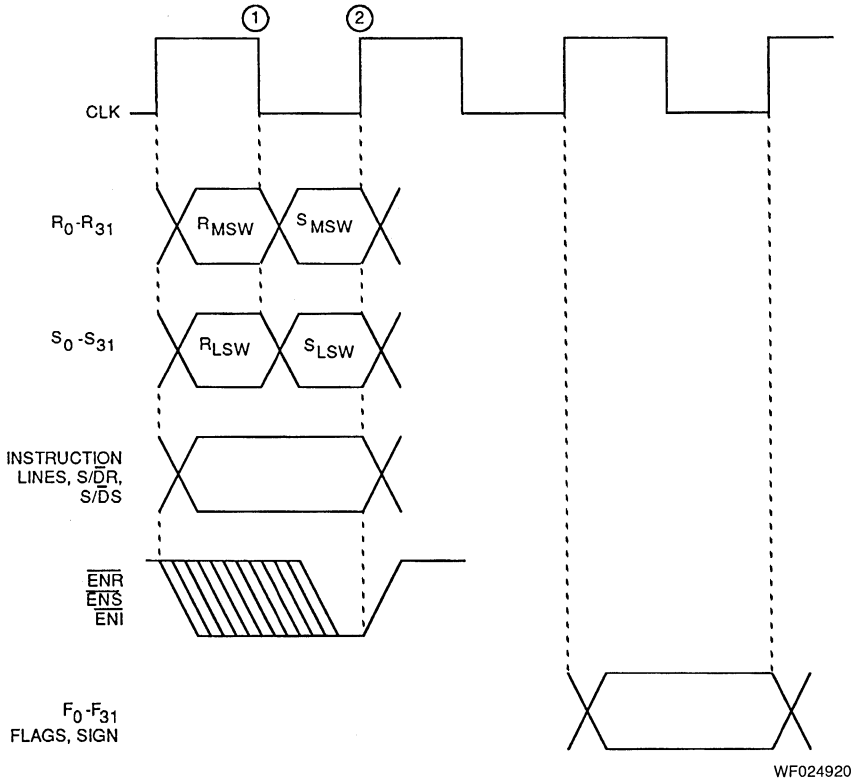
register, and the least-significant 32 bits of the S operand are loaded from the S-input port into the least-significant half of the S register.

At the same time, at 2, the output of the R-temp register is loaded into the most-significant half of the R register, and the output of the S-temp register is loaded into the most-significant half of the S register.

64-Bit Bus, Single-Cycle, R First (M16 = 1, M15 = 0, M14 = 0)

In this mode, the MSW of the 64-bit R operand is placed on the R-input bus and the LSW of the S-input bus. Both

halfwords are loaded in the first half cycle. Similarly, the two halves of the S operand are loaded in the second half cycle. After one full cycle, the R and S registers contain the R and S operands, respectively.



**Timing of Operations with Input Mode 5
(64-Bit Bus, Single-Cycle, R First)***

*Assumes flow-through operation, F register, and S register clocked.

In this mode, the temporary registers are clocked on every HIGH-to-LOW clock transition.

At 1, the most-significant 32 bits of the R operand are loaded from the R-input port into the R-temp register, and the least-significant 32 bits of the R operand are loaded from the S-input port into the S-temp register.

At 2, the most-significant 32 bits of the S operand are loaded from the R-input port into the most-significant half of the S

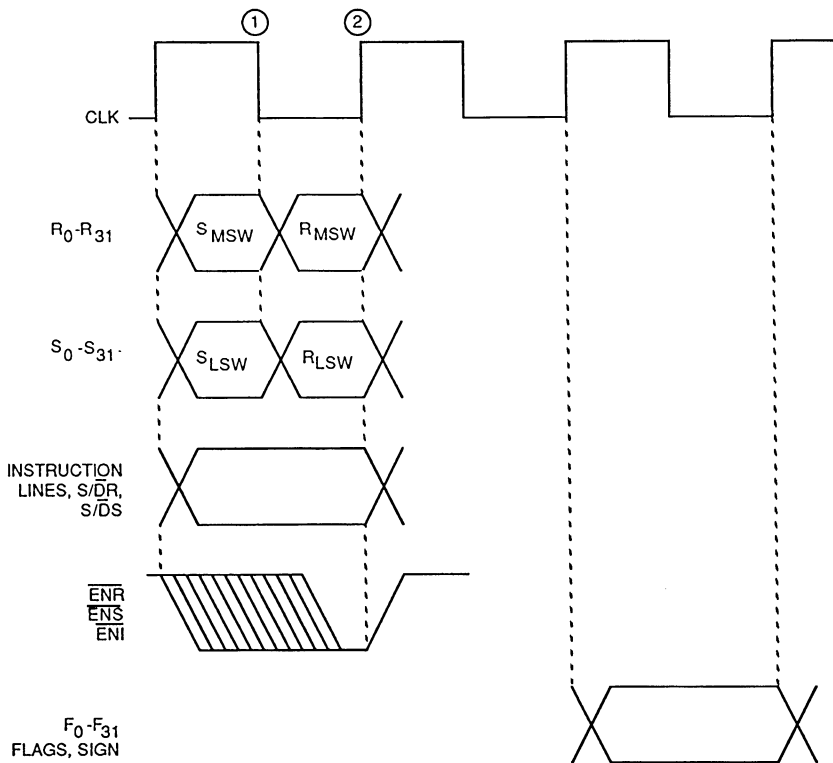
register, and the least-significant 32 bits of the S operand are loaded from the S-input port into the least-significant half of the S register.

At the same time, at 2, the output of the R-temp register is loaded into the most-significant half of the R register, and the output of the S-temp register is loaded into the least-significant half of the R register.

64-Bit Bus, Double-Cycle, S First (M16 = 1, M15 = 0, M14 = 1)

In this mode, the MSW of the 64-bit S operand is placed on the R-input bus and the LSW on the S-input bus. Both halfwords

are loaded in the first half cycle. Similarly, the two halves of the R operand are loaded in the second half cycle. After one full cycle, the R and S registers contain the R and S operands, respectively.



WF024930

Timing of Operations with Input Mode 6 (64-Bit Bus, Single-Cycle, S First)*

*Assumes flow-through operation, F register, and S register clocked.

In this mode, the temporary registers are clocked on every HIGH-to-LOW clock transition.

At 1, the most-significant 32 bits of the S operand are loaded from the R-input port into the R-temp register, and the least-significant 32 bits of the S operand are loaded from the S-input port into the S-temp register.

At 2, the most-significant 32 bits of the R operand are loaded from the R-input port into the most-significant half of the R

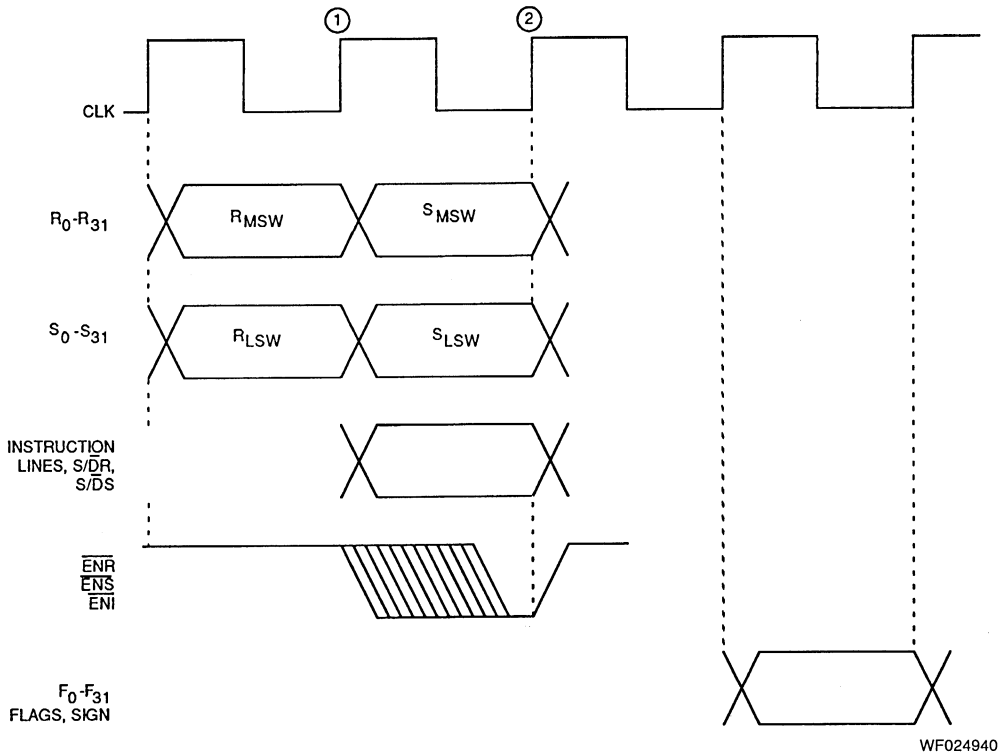
register, and the least-significant 32 bits of the R operand are loaded from the S-input port into the least-significant half of the R register.

At the same time, at 2, the output of the R-temp register is loaded into the most-significant half of the S register, and the output of the S-temp register is loaded into the least-significant half of the S register.

64-Bit Bus, Double-Cycle, R First (M16 = 1, M15 = 1, M14 = 0)

In this mode, the MSW of the 64-bit R operand is placed on the R-input bus and the LSW of the S-input bus. Both

halfwords are loaded in the first cycle. Similarly, the two halves of the S operand are loaded in the second cycle. After the two cycles, the R and S registers contain the R and S operands, respectively.



**Timing of Operations with Input Mode 7
(64-Bit Bus, Double-Cycle, R First)***

*Assumes flow-through operation, F register, and S register clocked.

In this mode, the temporary registers are clocked on every LOW-to-HIGH clock transition.

At 1, the most-significant 32 bits of the R operand are loaded from the R-input port into the R-temp register, and the least-significant 32 bits of the R operand are loaded from the S-input port into the S-temp register.

At 2, the most-significant 32 bits of the S operand are loaded from the R-input port into the most-significant half of the S

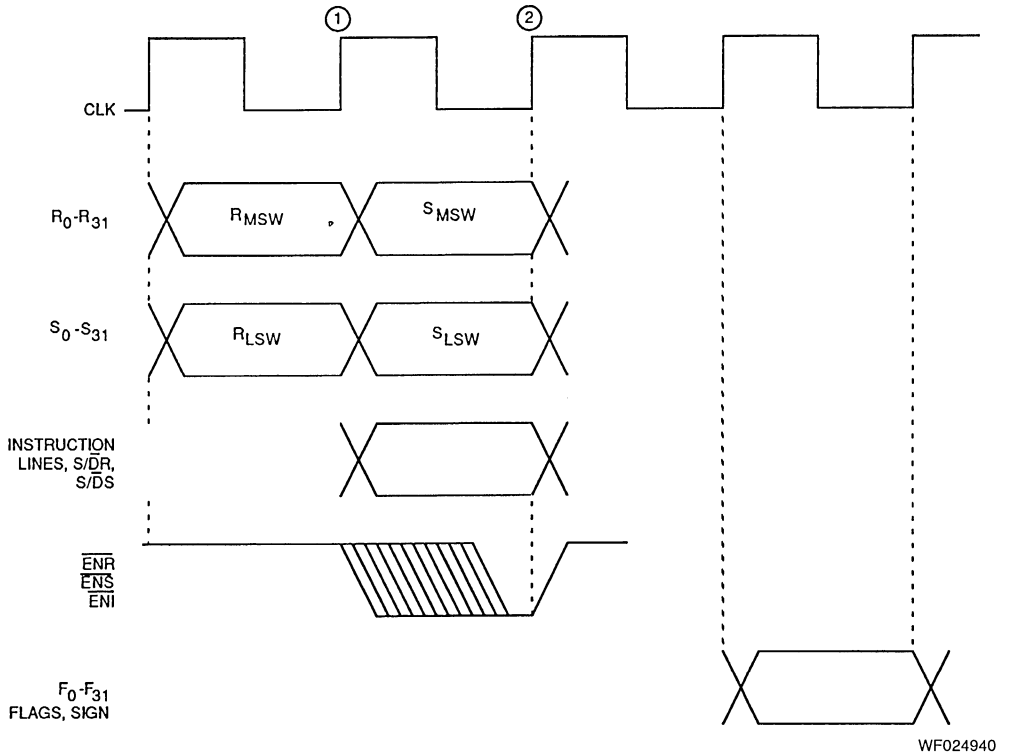
register, and the least-significant 32 bits of the S operand are loaded from the S-input port into the least-significant half of the S register.

At the same time, at 2, the output of the R-temp register is loaded into the most-significant half of the R register, and the output of the S-temp register is loaded into the least-significant half of the R register.

64-Bit Bus, Double-Cycle, S First (M16 = 1, M15 = 1, M14 = 1)

In this mode, the MSW of the 64-bit S operand is placed on the R-input bus and the LSW of the S-input bus. Both halfwords

are loaded in the first cycle. Similarly, the two halves of the R operand are loaded in the second cycle. After the two cycles, the R and S registers contain the R and S operands, respectively.



**Timing of Operations with Input Mode 8
(64-Bit Bus, Double-Cycle, S First)***

*Assumes flow-through operation, F register, and S register clocked.

In this mode, the temporary registers are clocked on every LOW-to-HIGH clock transition.

At 1, the most-significant 32 bits of the S operand are loaded from the R-input port into the R-temp register, and the least-significant 32 bits of the S operand are loaded from the S-input port into the S-temp register.

At 2, the most-significant 32 bits of the R operand are loaded from the R-input port into the most-significant half of the R

register, and the least-significant 32 bits of the R operand are loaded from the S-input port into the least-significant half of the R register.

At the same time, at 2, the output of the R-temp register is loaded into the most-significant half of the S register, and the output of the S-temp register is loaded into the least-significant half of the S register.

Pipelining of Operations

The floating-point ALU of the Am29C327 may be operated in one of three pipeline modes:

1. Flow-Through Mode
2. Single-Pipelined Mode
3. Double-Pipelined Mode

Flow-Through Mode

In this mode the floating-point ALU acts as a purely combinatorial device.

Single-Pipelined Mode

In this mode the floating-point ALU contains a single pipeline delay for all operations; throughput is roughly double that for unpipelined mode. Simplified diagrams for the ALU configuration for single-pipelined mode are shown in Figure 2.

Double-Pipelined Mode

In this mode, which applies only to the multiplication-accumulation operation, the ALU contains two pipeline delays; throughput is roughly triple that for the unpipelined multiplication-accumulation operation. Simplified block diagrams are shown in Figure 3.

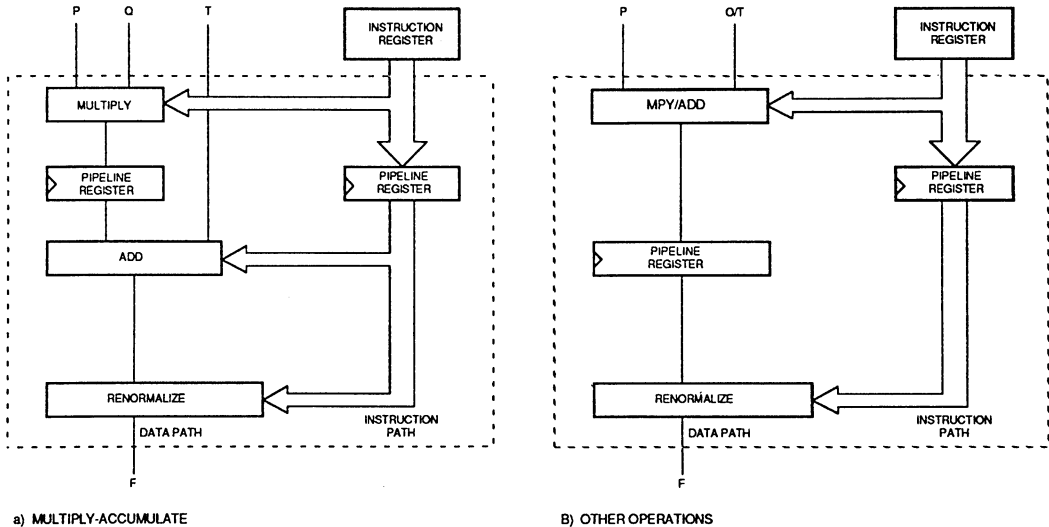
Figures 4 to 8 provide timing diagrams for all operations except multiply-accumulate illustrating flow-through mode, single-pipelined mode, and double-pipelined mode, respectively.

The choice of pipelining mode affects only the floating-point ALU. Operations of other parts of the Am29C327, such as the input registers, the output register, the mode register, and the instruction register are not affected by the choice of pipelining mode. However, the instruction bits are pipelined as they pass through the ALU. This permits instructions to be interleaved in pipelined mode.

The desired pipeline mode or modes can be invoked by setting mode register bits M19 and M20 to the appropriate values.

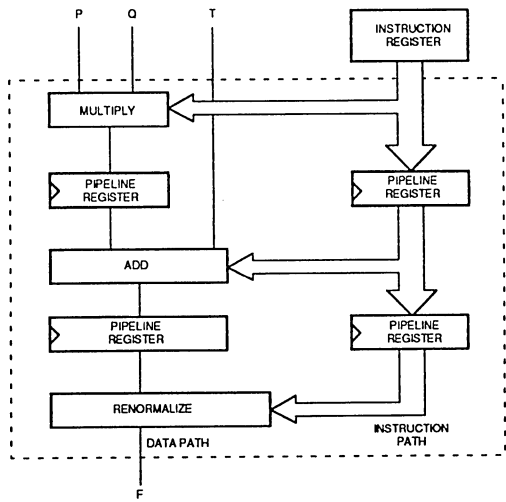
When using the Am29C327 in either single-pipelined or double-pipelined mode, two conditions must be observed:

1. The "load mode register" instruction is not pipelined, nor are any of the mode register bits. When the mode register is loaded, any differences between the current mode and the previous mode take effect immediately. In single-pipelined mode, the user should separate the last valid ALU instruction and the "load mode register" instruction with one "NO-OP" instruction. In double-pipelined mode, the user should separate them with two "NO-OP" instructions. A NO-OP instruction is any instruction whose result is not stored in register F, or the register file.
2. A multiplication-accumulation instruction cannot be immediately followed by any other type of instruction. This problem can be avoided by inserting a "dummy" multiplication-accumulation instruction at the end of a multiplication-accumulation instruction. This "dummy" is any instruction whose results are not stored in register F or the register file.

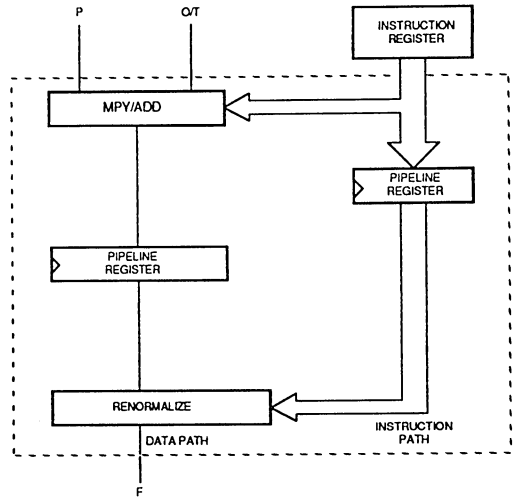


DF006260

Figure 2. ALU Configuration for Single-Pipelined Mode



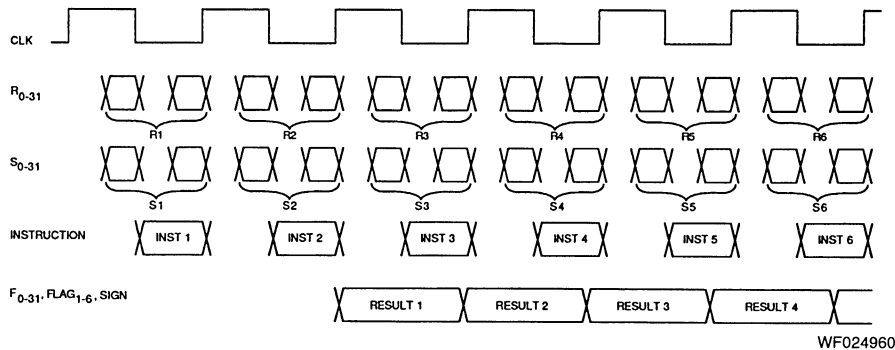
a) MULTIPLY-ACCUMULATE



b) OTHER OPERATIONS

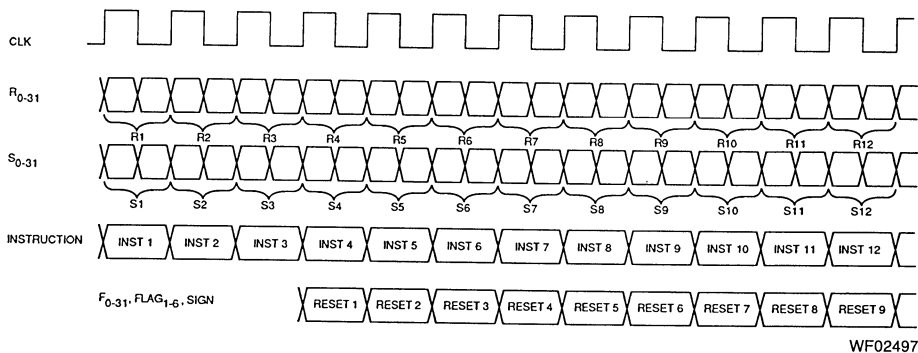
DF006270

Figure 3. ALU Configuration for Double-Pipelined Mode



WF024960

Figure 4. Timing for All Operations EXCEPT Multiply-Accumulate, Flow-Through Mode



WF024970

Figure 5. Timing for All Operations EXCEPT Multiply-Accumulate, Pipelined Mode

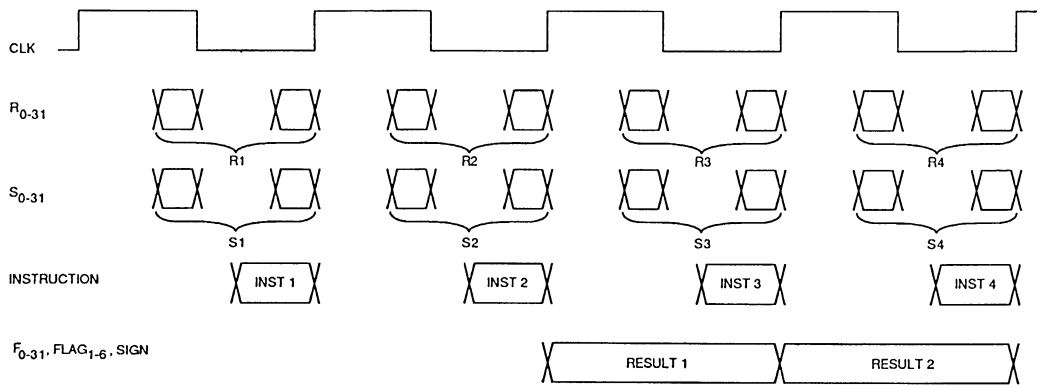


Figure 6. Timing for Multiply-Accumulate, Flow-Through Mode

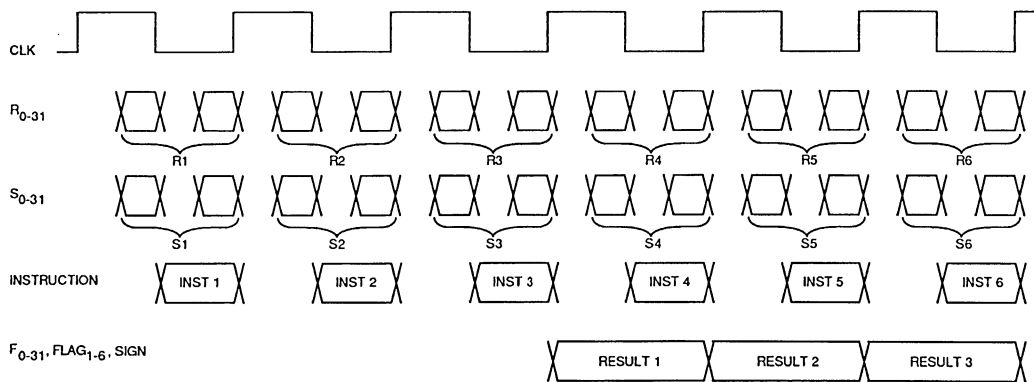


Figure 7. Timing for Multiply-Accumulate, Single-Pipelined Mode

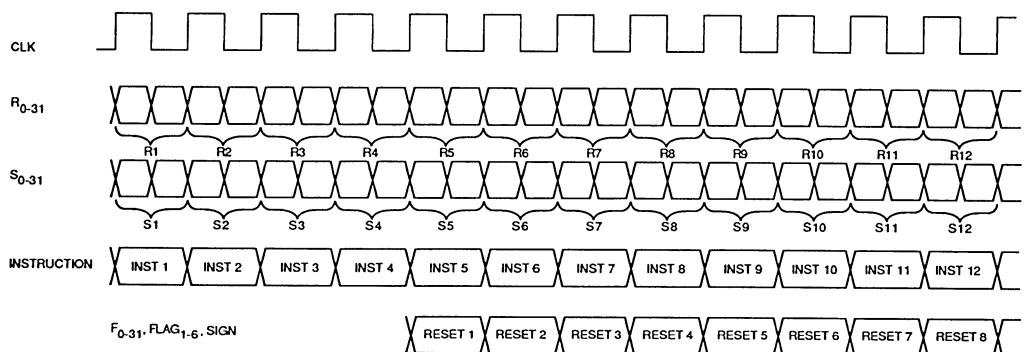


Figure 8. Timing for Multiply-Accumulate, Double-Pipelined Mode

Instruction Set

Instruction Register Format

The 14-bit instruction word $I_0 - I_{13}$ comprises sign-change controls, integer/floating-point select bit, and the opcode.

I_{13}	I_{12} I_{11}	I_{10} I_9	I_8 I_7	I_6	I_5	I_4 I_3 I_2 I_1 I_0
SIGN (P)	SIGN (Q)	SIGN (T)	SIGN (F)	INT/FP	OPCODE	

The opcode field, $I_4 - I_0$, specifies the core operation to be performed by the ALU; instruction bit I_5 selects between

floating-point and integer formats. The core operations and their corresponding opcodes are listed in Table 1.

TABLE 1. CORE OPERATIONS/OPCODES

I_5	I_4	I_3	I_2	I_1	I_0	Operation (Floating-Point)
0	0	0	0	0	0	P
0	0	0	0	0	1	P + T
0	0	0	0	1	0	P * Q
0	0	0	0	1	1	COMPARE P, T
0	0	0	1	0	0	MAX P, T
0	0	0	1	0	1	MIN P, T
0	0	0	1	1	0	CONVERT T TO INTEGER
0	0	0	1	1	1	SCALE T TO INTEGER BY Q
0	0	1	0	0	0	(P * Q) + T
0	0	1	0	0	1	ROUND T TO INTEGRAL VALUE
0	0	1	0	1	0	RECIPROCAL SEED OF P
0	0	1	0	1	1	CONVERT T TO ALTERNATE F.P. FORMAT
0	0	1	1	0	0	CONVERT T FROM ALTERNATE F.P. FORMAT
I_5	I_4	I_3	I_2	I_1	I_0	Operation (Integer)
1	0	0	0	0	0	P
1	0	0	0	0	1	P + T
1	0	0	0	1	0	P * Q
1	0	0	0	1	1	COMPARE P, T
1	0	0	1	0	0	MAX P, T
1	0	0	1	0	1	MIN P, T
1	0	0	1	1	0	CONVERT T TO FLOATING-POINT
1	0	0	1	1	1	SCALE T TO FLOATING-POINT BY Q
1	1	0	0	0	0	P OR T
1	1	0	0	0	1	P AND T
1	1	0	0	1	0	P XOR T
1	1	0	0	1	1	SHIFT P LOGICAL Q PLACES
1	1	0	1	0	0	SHIFT P ARITHMETIC Q PLACES
1	1	0	1	0	1	FUNNEL SHIFT PT LOGICAL Q PLACES

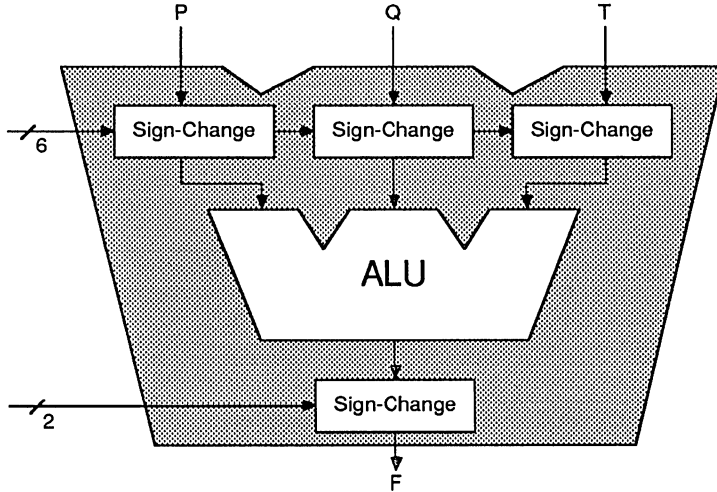
Core operations MOVE P and LOAD MODE REGISTER can both be performed in either floating-point or integer format:

I_5	I_4	I_3	I_2	I_1	I_0	Operation
X	1	1	0	0	0	MOVE P
X	1	1	1	1	1	LOAD MODE REGISTER

Sign-Change Selects

Each ALU input and output operand has associated hardware that can be used to modify operand signs (see Figure 9). These sign-change blocks, when applied to core operations, greatly increase the number of available operations. A core operation of $P + T$, for example, can be used to perform operations such as $P - T$, $ABS(P + T)$, $ABS(P) + ABS(T)$, and others, simply by modifying the signs of the input and output operands.

Using the sign-change blocks, the sign of an input operand may be left unchanged, inverted, set to zero, or set to one; the sign of the output operand may be left unchanged, set to zero, set to one, set to the sign of the P input operand, or set to the sign of the T input operand. Select decodes for the P, Q, T, and F operand sign-change blocks are shown in Table 2-1, 2-2, 2-3, and 2-4, respectively.



BD011060

Figure 9. ALU Sign-Change Blocks

TABLE 2-1. SELECT DECODE FOR P OPERAND SIGN-CHANGE BLOCK

I ₁₃	I ₁₂	Sign (P')
0	0	SIGN (P)
0	1	$\overline{\text{SIGN}} (P)$
1	0	0
1	1	1

TABLE 2-2. SELECT DECODE FOR Q OPERAND SIGN-CHANGE BLOCK

I ₁₁	I ₁₀	Sign (Q')
0	0	SIGN (Q)
0	1	$\overline{\text{SIGN}} (Q)$
1	0	0
1	1	1

TABLE 2-3. SELECT DECODE FOR T OPERAND SIGN-CHANGE BLOCK

I ₉	I ₈	Sign (T')
0	0	SIGN T
0	1	$\overline{\text{SIGN}} T$
1	0	0
1	1	1

TABLE 2-4. SELECT DECODE FOR F OPERAND SIGN-CHANGE BLOCK

Core Operation	I ₁₁	I ₁₀	I ₇	I ₆	Sign (F)
P,	0	x	0	0	SIGN (F')
Max P, T	0	x	0	1	$\overline{\text{SIGN}} (F')$
or	0	x	1	0	0
Min P, T	0	x	1	1	1
	1	0	x	x	SIGN (P)
	1	1	x	x	SIGN (T)
Other	x	x	0	0	SIGN (F')
	x	x	0	1	$\overline{\text{SIGN}} (F')$
	x	x	1	0	0
	x	x	1	1	1

Operand Multiplexer Selects

Instruction fields PSEL₀–PSEL₃, QSEL₀–QSEL₃, and TSEL₀–TSEL₃ specify the select codes for the P, Q, and T

operand multiplexers, respectively; the codes are summarized in Table 3.

TABLE 3. OPERAND MULTIPLEXER SELECT CODES

PSEL ₃ QSEL ₃ TSEL ₃	PSEL ₂ QSEL ₂ TSEL ₂	PSEL ₁ QSEL ₁ TSEL ₁	PSEL ₀ QSEL ₀ TSEL ₀	P Q T
0	0	0	0	R
0	0	0	1	S
0	0	1	0	O
0	0	1	1	0.5 (Floating Point) -1 (Integer)
0	1	0	0	1
0	1	0	1	2
0	1	1	0	3
0	1	1	1	Pi (Floating Point) Max Neg. Two's-Comp. Value (Integer)
1	0	0	0	Register File Location 0 (RF0)
1	0	0	1	Register File Location 1 (RF1)
1	0	1	0	Register File Location 2 (RF2)
1	0	1	1	Register File Location 3 (RF3)
1	1	0	0	Register File Location 4 (RF4)
1	1	0	1	Register File Location 5 (RF5)
1	1	1	0	Register File Location 6 (RF6)
1	1	1	1	Register File Location 7 (RF7)

Operand Precisions

The Am29C327 supports mixed-precision operations, so that it is possible, for example, for an operation to have single-precision inputs and a double-precision output, or one single- and one double-precision input, or any other combination.

Precision of the operands in registers R and S is specified by signals S/ \overline{DR} and S/ \overline{DS} . A logic HIGH indicates a single-precision operand or operands; a LOW, double precision.

Precision of an operation result is specified by signal S/ \overline{DF} . A logic HIGH indicates a single-precision operand; a logic LOW, double-precision.

Operands stored in the register file are each accompanied by a bit indicating that operand's precision; this precision informa-

tion is automatically supplied to the ALU when a register file location is used as an input operand to an operation.

Processor Operations

Table 4 illustrates a number of possible ALU instructions comprising the opcode, integer/floating-point select, and sign-change fields. Note that the remaining instruction bits — P, Q, and T operand multiplexer selects; the rounding modes; and the output operand precision — can be specified independently.

The user may create instructions using instruction words other than those listed in Table 4. For some core operations, sign-change control settings are completely arbitrary; for others, only the sign-change field values shown in Table 4 are valid. Table 5 summarizes permissible sign-change field values for each core operation.

TABLE 4. INSTRUCTION WORDS

Operation	Sign				I/ \bar{F}	Opcode
	P	Q	T	F		
FP P	00	00	xx	00	0	00000
FP -P	00	00	xx	01	0	00000
FP ABS (P)	00	00	xx	10	0	00000
FP Sign (T)*ABS (P)	00	11	xx	xx	0	00000
FP P + T	00	xx	00	00	0	00001
FP P - T	00	xx	01	00	0	00001
FP T - P	01	xx	00	00	0	00001
FP -P - T	01	xx	01	00	0	00001
FP ABS (P + T)	00	xx	00	10	0	00001
FP ABS (P - T)	00	xx	01	10	0	00001
FP ABS (P) + ABS (T)	10	xx	10	00	0	00001
FP ABS (P) - ABS (T)	10	xx	11	00	0	00001
FP ABS (ABS (P) - ABS (T))	10	xx	11	10	0	00001
FP P * Q	00	00	xx	00	0	00010
FP (-P) * Q	01	00	xx	00	0	00010
FP ABS (P * Q)	00	00	xx	10	0	00010
FP Compare P, T	00	xx	01	00	0	00011
FP Max P, T	00	00	01	00	0	00100
FP Max ABS (P), ABS (T)	10	00	11	00	0	00100
FP Min P, T	01	00	00	00	0	00101
FP Min ABS (P), ABS (T)	11	00	10	00	0	00101
FP Limit P to Magnitude T	11	10	10	xx	0	00101
FP Convert T to Integer	xx	xx	00	00	0	00110
FP Scale T to Integer by Q	xx	00	00	00	0	00111
FP T + P*Q	00	00	00	00	0	01000
FP T - P*Q	01	00	00	00	0	01000
FP -T + P*Q	00	00	01	00	0	01000
FP -T - P*Q	01	00	01	00	0	01000
FP ABS (T) + ABS (P*Q)	10	10	10	00	0	01000
FP ABS (T) - ABS (P*Q)	11	10	10	00	0	01000
FP ABS (P*Q) - ABS (T)	10	10	11	00	0	01000
FP Round T to Integral Value	xx	xx	00	00	0	01001
FP Reciprocal Seed (P)	00	xx	xx	00	0	01010
FP Convert T to Alternate Floating-point Format	xx	xx	00	00	0	01011
FP Convert T from Alternate Floating-point Format	xx	xx	00	00	0	01100

TABLE 4. INSTRUCTION WORDS (Cont'd.)

Operation	Sign				I/ \bar{F}	Opcode
	P	Q	T	F		
Int P	00	00	00	00	1	00000
Int -P	00	00	00	01	1	00000
Int ABS (P)	00	00	00	10	1	00000
Int sign (T)*ABS (P)	00	11	00	xx	1	00000
Int P + T	00	xx	00	00	1	00001
Int P - T	00	xx	01	00	1	00001
Int T - P	01	xx	00	00	1	00001
Int ABS (P + T)	00	xx	00	10	1	00001
Int ABS (P - T)	00	xx	01	10	1	00001
Int P * Q	00	00	xx	00	1	00010
Int Compare P, T	00	xx	01	00	1	00011
Int Max P, T	00	00	01	00	1	00100
Int Min P, T	01	00	00	00	1	00101
Int Convert T to Float	xx	xx	00	00	1	00110
Int Scale T to Float by Q	xx	00	00	00	1	00111
Int P OR T	xx	xx	xx	xx	1	10000
Int P AND T	xx	xx	xx	xx	1	10001
Int P XOR T	xx	xx	xx	xx	1	10010
Int NOT T (see Note 1)	xx	xx	xx	xx	1	10010
Int Shift P Logical Q Places	00	00	xx	00	1	10011
Int Shift P Arithmetic Q Places	00	00	xx	00	1	10100
Int Funnel Shift PT Q Places	00	00	00	00	1	10101
Int Move P	xx	xx	xx	xx	x	11000
Int Load Mode Register	xx	xx	xx	xx	x	11111

Notes: 1. NOT T is performed by XORing T with a word containing all 1's (integer -1). When invoking NOT T the user must set PSEL₃-PSEL₀ to 0011₂, thus selecting integer constant -1.

TABLE 5. ALLOWABLE SIGN-CHANGE/CORE-OPERATION COMBINATIONS

I I I I I I 5 43210	Core Operation	Sign-Change Fields			
		Sign (P)	Sign (Q)	Sign (T)	Sign (F)
0 00000	FP P	V	V	x	V
0 00001	FP P + T	V	x	V	V
0 00010	FP P*Q	V	V	x	V
0 00011	FP Compare P, T	F	x	F	F
0 00100	FP Max P, T	F	F	F	F
0 00101	FP Min P, T	F	F	F	F
0 00110	FP Cvt T to Int	x	x	F	F
0 00111	FP Scale T to Int	x	F	F	F
0 01000	FP P*Q + T	V	V	V	V
0 01001	FP Round T	x	x	F	F
0 01010	FP Recip Seed P	F	x	x	F
0 01011	FP Cvt T to Alt Fmt	x	x	F	F
0 01100	FP Cvt T fm Alt Fmt	x	x	F	F
1 00000	Int P	F	F	F	F
1 00001	Int P + T	F	x	F	F
1 00010	Int P*Q	F	F	x	F
1 00011	Int Compare P, T	F	x	F	F
1 00100	Int Max P, T	F	F	F	F
1 00101	Int Min P, T	F	F	F	F
1 00110	Int Cvt T to f.p.	x	x	F	F
1 00111	Int Scale T to f.p.	x	F	F	F
1 10000	Int P OR T	x	x	x	x
1 10001	Int P AND T	x	x	x	x
1 10010	Int P XOR T	x	x	x	x
1 10011	Int Shift P Logical	F	F	x	F
1 10100	Int Shift P Arith	F	F	x	F
1 10101	Int Funnel Shift PT	F	F	F	F
x 11000	Move P	x	x	x	x
x 11111	Load Mode Reg	x	x	x	x

Key: V = Variable; user can specify arbitrary sign change.
 F = Fixed; user is restricted to sign change combinations shown in Table 4.
 x = Don't care; this field does not affect the operation or its result.

Descriptions of Operations

P (Floating-Point or Integer): The operand on port P is passed through the ALU to port F. This operation may be used to change the precision of an operand, negate an operand, extract the absolute value of an operand, or transfer the sign of operand T to operand P.

P + T (Floating-Point or Integer): The addition operation (P + T) adds the operands on ports P and T, and places the result on port F.

P*Q (Floating-Point or Integer): The multiplication operation (P*Q) multiplies the operands on ports P and Q, and places the result on port F.

COMPARE P, T (Floating-Point or Integer): This operation compares the operands on ports P and T, and places (P - T) on port F. One of four comparison flags (=, >, <, #) is set according to the result of the comparison. Note that the unordered flag (#) can be set only when the format selected is IEEE or DEC.

MAX P, T (Floating-Point or Integer): This operation selects the most positive of the two operands on ports P and T, and places the result on port F.

MIN P, T (Floating-Point or Integer): This operation selects the most negative of the two operands on ports P and T, and places the result on port F.

LIMIT P TO MAGNITUDE T (Floating-Point): This operation imposes a clipping or saturation level on operand P by

comparing the magnitudes of the operands on ports P and T. If operand P has the smaller magnitude, it is placed on port F; if operand T has the smaller magnitude, it is placed on port F, but with its sign modified to agree with that of operand P. This operation is equivalent to operation $SIGN(P) * MIN(ABS(P), ABS(T))$.

CONVERT T TO INTEGER (Floating-Point): The floating-point-to-integer conversion operation takes a floating-point operand on port T and places the equivalent two's-complement integer value on port F.

CONVERT T TO FLOATING-POINT (Integer): The integer-to-floating-point conversion operation takes a two's-complement integer operand on port T and places the equivalent floating-point value on port F.

SCALE T TO INTEGER BY Q (Floating-Point): This operation converts the floating-point operand T to integer format using the floating-point operand Q as a scale factor. The true exponent of Q is added to the true exponent of T before the new value T is converted to integer format. The operation therefore permits T to be scaled by any multiple of two when the source format is IEEE or DEC, and by any multiple of 16 when the source format is IBM.

SCALE T TO FLOATING-POINT BY Q (Integer): This operation converts the integer operand T to floating-point format using the operand Q as a scale factor, where Q is a floating-point operand in the destination format. The true exponent of Q is added to the true exponent of T after T has been converted from integer to floating-point. The operation

therefore permits T to be scaled by any multiple of two when the destination format is IEEE or DEC, and by any multiple of 16 when the destination format is IBM.

(P*Q) + T (Floating-Point): This operation multiplies the operands on port P and Q, adds the product to the operand on port T, and places the result on port F.

ROUND T TO INTEGRAL VALUE (Floating-Point): This operation rounds a floating-point operand to an integer-valued floating-point operand of the same format. A value of 3.5, for example, would be rounded to either 3.0 or 4.0, the choice depending on the rounding mode.

RECIPROCAL SEED OF P (Floating-Point): The reciprocal seed of the floating-point operand on port P is placed on port F; the result obtained is a crude estimate of the input operand's reciprocal. This operation can be used as the initial step in performing Newton-Raphson division. A single-precision result is obtained after five iterations, and a double-precision result after six iterations. Alternately, an external seed look-up table can be used for faster convergence.

CONVERT T TO ALTERNATE FLOATING-POINT FORMAT (Floating-Point): This operation converts operand T from the primary floating-point format to the alternate floating-point format, thus allowing conversions among the IEEE, DEC, and IBM floating-point formats. The result obtained through iteration is approximate.

CONVERT T FROM ALTERNATE FLOATING-POINT FORMAT (Floating-Point): This operation converts operand T from the alternate floating-point format to the primary floating-point format, in a manner similar to that of CONVERT T TO ALTERNATE FLOATING-POINT FORMAT above.

P OR T, P AND T, P XOR T, NOT T (Integer): The logical operations (OR, AND, EXCLUSIVE OR) are performed on the operands on ports P and T, and the result is placed on port F. NOT T is performed by XORing T with a word containing all ones (integer -1). When invoking NOT T, instruction bits PSEL₃ - PSEL₀ must be set to 0011, thus selecting integer constant -1.

SHIFT P LOGICAL Q PLACES (Integer): This operation logically shifts operand P by Q places. If the shift is Q places to the right, Q zeros are filled from the left. If the shift is Q places to the left, Q zeros are filled from the right.

SHIFT P ARITHMETIC Q PLACES (Integer): This operation arithmetically shifts operand P by Q places. With a right shift, the result is sign extended Q places. With a left shift, Q zeros are filled from the right.

FUNNEL SHIFT PT LOGICAL Q PLACES (Integer): The operands on ports P and T are concatenated to form a double-width operand PT, which is then shifted to the right or left by Q places; the 32- or 64-bit result is placed on port F.

MOVE P (Floating-Point or Integer): The operand on port P is moved to port F. The operand is left unchanged, and only the sign flag is set.

Primary and Alternate Floating-Point Formats

All floating-point operations except format conversions are performed in the primary format selected by mode register bits M0 and M1. Conversions to the alternate format convert floating-point numbers from the primary format to the alternate format selected by mode register bits M2 and M3. Conversions from the alternate format convert floating-point data from the alternate format to the primary format. Conversions and scale operations to and from integer format operate on floating-point numbers in the primary format.

Operation Flags

For each operation, the ALU produces thirteen flags that indicate operation status. Of the flags produced, a maximum of seven are relevant to any given operation. The relevant flags are placed in the status register, and the other flags are discarded.

The ALU flags are:

C — CARRY: Carry-out bit produced by integer addition, subtraction, or comparison.

I — INVALID OPERATION: Input operands are unsuitable for the operation specified (e.g., $\infty * 0$).

R — RESERVED OPERAND: Reserved operand detected/generated.

S — SIGN: Result sign.

U — UNDERFLOW: Result underflowed the destination format.

V — OVERFLOW: Result overflowed the destination format.

W — WINNER: Indicates which of the two operands selected when performing Max/Min operations.

X — INEXACT RESULT: Result had to be rounded to fit the destination format.

Z — ZERO: Zero result.

>, =, <, # — GREATER THAN, EQUAL, LESS THAN, UNORDERED: Used to report the result of a comparison operation.

Table 6 lists the flags reported for each operation.

TABLE 6. ORGANIZATION OF FLAGS

Operations		Opcode I ₄ -I ₀	Flag Register						
			LSB						MSB 6
			0	1	2	3	4	5	
IEEE	Non-arithmetic single-operand	00000	I	R	V	U	X	Z	S
IEEE	Operations using add	00001	I	R	V	U	X	Z	S
IEEE	Operations using multiply	00010	I	R	V	U	X	Z	S
IEEE	Compare	00011	I	R	#	<	>	=	S
IEEE	Maximum, minimum, limit	0010x	I	R		W		Z	S
IEEE	Convert/scale to integer	0011x	I	R	V		X	Z	S
IEEE	Multiply/accumulate	01000	I	R	V	U		Z	S
IEEE	Round to integral value	01001	I	R	V		X	Z	S
IEEE	Reciprocal seed	01010	I	R	V	U		Z	S
IEEE	Convert to alt. f.p. format	01011	I	R	V	U	X	Z	S
IEEE	Convert from alt. f.p. format	01100	I	R	V	U	X	Z	S
DEC D	Non-arithmetic single-operand	00000		R	V		X	Z	S
DEC D	Operations using add	00001		R	V	U	X	Z	S
DEC D	Operations using multiply	00010		R	V	U	X	Z	S
DEC D	Compare	00011		R	#	<	>	=	S
DEC D	Maximum, minimum, limit	0010x		R		W		Z	S
DEC D	Convert/scale to integer	0011x	I	R	V		X	Z	S
DEC D	Multiply/accumulate	01000		R	V	U		Z	S
DEC D	Round to integral value	01001		R	V		X	Z	S
DEC D	Reciprocal seed	01010	I	R	V	U		Z	S
DEC D	Convert to alt. f.p. format	01011	I	R	V	U	X	Z	S
DEC D	Convert from alt. f.p. format	01100	I	R	V	U	X	Z	S
DEC G	Non-arithmetic single-operand	00000		R	V	U	X	Z	S
DEC G	Operations using add	00001		R	V	U	X	Z	S
DEC G	Operations using multiply	00010		R	V	U	X	Z	S
DEC G	Compare	00011		R	#	<	>	=	S
DEC G	Maximum, minimum, limit	0010x		R		W		Z	S
DEC G	Convert/scale to integer	0011x	I	R	V		X	Z	S
DEC G	Multiply/accumulate	01000		R	V	U		Z	S
DEC G	Round to integral value	01001		R	V		X	Z	S
DEC G	Reciprocal seed	01010	I	R	V	U		Z	S
DEC G	Convert to alt. f.p. format	01011	I	R	V	U	X	Z	S
DEC G	Convert from alt. f.p. format	01100	I	R	V	U	X	Z	S
IBM	Non-arithmetic single-operand	00000			V		X	Z	S
IBM	Operations using add	00001			V	U	X	Z	S
IBM	Operations using multiply	00010			V	U	X	Z	S
IBM	Compare	00011				<	>	=	S
IBM	Maximum, minimum, limit	0010x				W		Z	S
IBM	Convert/scale to integer	0011x			V		X	Z	S
IBM	Multiply/accumulate	01000			V	U		Z	S
IBM	Round to integral value	01001			V		X	Z	S
IBM	Reciprocal seed	01010	I		V			Z	S
IBM	Convert to alt. f.p. format	01011		R	V	U	X	Z	S
IBM	Convert from alt. f.p. format	01100	I	R	V	U	X	Z	S
Integer	Non-arithmetic single-operand	00000			V			Z	S
Integer	Sign transfer	00000			V			Z	S
Integer	Operations using add	00001	C		V			Z	S
Integer	Operations using multiply	00010			V			Z	S
Integer	Compare operations	00011	C		V	<	>	=	S
Integer	Maximum, minimum, limit	0010x				W		Z	S
Integer	Convert to float	00110					X	Z	S
Integer	Scale to float	00111		R	V	U	X	Z	S
Integer	Logical operations	100xx						Z	S
Integer	Arithmetic shift	10100			V			Z	S
Integer	Funnel shift	10101						Z	S
	Move operand	11000							S
	Load mode register	11111							S

Note: Unused flags assume the LOW state.

Master/Slave Operation

Two Am29C327 processors can be tied together in master/slave configuration, with the slave checking the results produced by the master. All input and output signals of the slave, with the exception of $\overline{\text{SLAVE}}$ and MSERR, are tied to the corresponding signals of the master. The master is selected by asserting signal $\overline{\text{SLAVE}}$ LOW; the slave, by asserting signal $\overline{\text{SLAVE}}$ HIGH.

The slave processor, by comparing its outputs to the outputs of the master processor, performs a comprehensive check of the operation of the master processor. In addition, the slave processor may detect open circuits and other faults in the electrical path between the master processor and the system. Note that the master processor still performs the comparison between its outputs and its own internally generated results, and is therefore able to detect faults in its output drivers.

APPENDICES

APPENDIX A — DATA FORMATS

The following data formats are supported: 32-bit integer, 64-bit integer, IEEE single-precision, IEEE double-precision, DEC F, DEC D, DEC G, IBM single-precision, and IBM double-precision.

The primary and alternate floating-point formats are selected by mode register bits M0 to M3. The user may select between floating-point operations and integer operations by means of instruction bit 15.

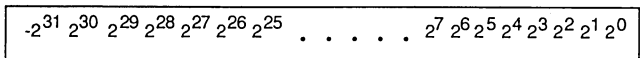
The nine supported formats are described below:

Integer Formats

32-Bit Integer

The 32-bit integer word is arranged as follows:

Bit 31 30 29 28 27 26 25 7 6 5 4 3 2 1 0



TB001030

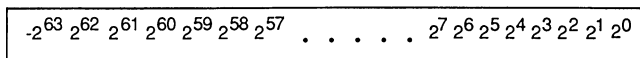
The 32-bit word is interpreted as a two's-complement integer. For integer multiplications, the user has the option of interpreting integers as unsigned. An unsigned single-precision integer

has a format similar to that of the two's-complement integer, but with an MSB weight of 2^{31} .

64-Bit Integer

The 64-bit integer word is arranged as follows:

Bit 63 62 61 60 59 58 57 7 6 5 4 3 2 1 0



TB001040

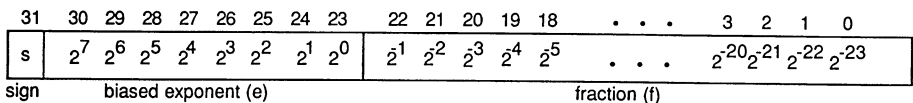
The 64-bit word is interpreted as a two's-complement integer. For integer multiplications, the user has the option of interpreting integers as unsigned. An unsigned double-precision inte-

ger has a format similar to that of the two's-complement integer, but with an MSB weight of 2^{63} .

IEEE Formats

IEEE Single-Precision

The IEEE single-precision word is 32 bits wide and is arranged in the format as follows:



TB001050

The floating-point word is divided into three fields: a single-bit sign, an 8-bit biased exponent, and a 23-bit fraction.

The sign bit is 0 for positive numbers and 1 for negative numbers. Zero may have either sign.

The biased exponent is an 8-bit unsigned integer representing a multiplicative factor of some power of two. The bias value is 127. If, for example, the multiplicative value for a floating-point

number is to be 2^a , the value of the biased exponent is $a + 127$, where "a" is the true exponent.

The fraction is a 23-bit unsigned fractional field containing the 23 least-significant bits of the floating-point number's 24-bit mantissa. The weight of the fraction's most-significant bit is 2^{-1} . The weight of the least-significant bit is 2^{-23} .

An IEEE floating-point number is evaluated or interpreted as follows:

- | | | |
|-----------------------------------|----------------------------------|---------------------|
| If $e = 255$ and $f \neq 0$ | value = NaN | Not-a-Number |
| If $e = 255$ and $f = 0$ | value = $(-1)^s \infty$ | Infinity |
| If $0 < e < 255$ | value = $(-1)^s 2^{e-127} (1.f)$ | Normalized number |
| If $e = 0$ and $f \neq 0$ | value = $(-1)^s 2^{-126} (0.f)$ | Denormalized number |
| If $e = 0$ and $f = 0$ | value = $(-1)^s 0$ | Zero |

Infinity: Infinity can have either a positive or negative sign. The interpretation of infinities is determined by the Affine/Projective select input AFF/PROJ.

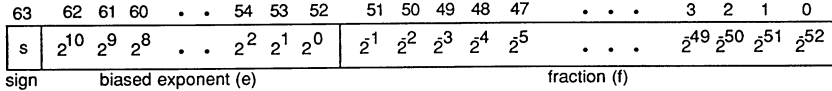
NaN: A NaN is interpreted as a signal or symbol. NaNs are used to indicate invalid operations, and as a means of passing process status through a series of calculations. They arise in

two ways: either generated by the Am29C327 to indicate an invalid operation, or provided by the user as an input. A signaling NaN has the MSB of its fraction set to 0 and at least one of the remaining fraction bits set to 1. A quiet NaN has the MSB of its fraction set to 1.

The IEEE format is fully described in IEEE Standard 754.

IEEE Double-Precision

The IEEE double-precision word is 64 bits wide and is arranged in the format shown below:



TB001060

The floating-point word is divided into three fields: a single-bit sign, an 11-bit biased exponent, and a 52-bit fraction.

The sign bit is 0 for positive numbers and 1 for negative numbers; zero may have either sign.

The biased exponent is an 11-bit unsigned integer representing a multiplicative factor of some power of two. The bias value is 1023. If, for example, the multiplicative value for a

floating-point number is to be 2^a , the value of the biased exponent is $a + 1023$, where "a" is the true exponent.

The fraction is a 52-bit unsigned fractional field containing the 52 least-significant bits of the floating-point number's 53-bit mantissa. The weight of the fraction's most-significant bit is 2^{-1} . The weight of the least-significant bit is 2^{-52} .

An IEEE floating-point number is evaluated or interpreted as follows:

- If $e = 2047$ and $f \neq 0$ value = Reserved operand Not-a-Number
- If $e = 2047$ and $f = 0$ value = $(-1)^s \infty$ Infinity
- If $0 < e < 2047$ value = $(-1)^s 2^{e-1023}(1.f)$ Normalized number
- If $e = 0$ and $f \neq 0$ value = $(-1)^s 2^{-1022}(0.f)$ Denormalized number
- If $e = 0$ and $f = 0$ value = $(-1)^s 0$ Zero

Infinity: Infinity can have either a positive or negative sign. The interpretation of infinities is determined by the Affine/Projective select input AFF/PROJ.

NaN: A NaN is interpreted as a signal or symbol. NaNs are used to indicate invalid operations, and as a means of passing process status through a series of calculations. They arise in

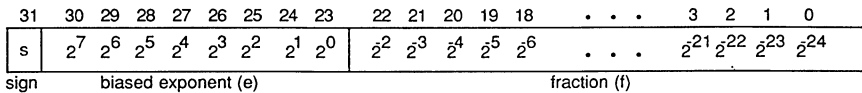
two ways: either generated by the Am29C327 to indicate an invalid operation, or provided by the user as an input. A signaling NaN has the MSB of its fraction set to 0 and at least one of the remaining fraction bits set to 1. A quiet NaN has the MSB of its fraction set to 1.

The IEEE format is fully described in IEEE Standard 754.

DEC Formats

DEC F

The DEC F word is 32 bits wide and is arranged in the format shown below:



TB001070

The floating-point word is divided into three fields: a single-bit sign, an 8-bit biased exponent, and a 23-bit fraction.

The sign bit is 0 for positive numbers and 1 for negative numbers; zero has a positive sign.

The biased exponent is an 8-bit unsigned integer representing a multiplicative factor of some power of two. The bias value is 128. If, for example, the multiplicative value for a floating-point number is to be 2^a , the value of the biased exponent is $a + 128$, where "a" is the true exponent.

The fraction is a 23-bit unsigned fractional field containing the 23 least-significant bits of the floating-point number's 24-bit mantissa. The weight of the fraction's most-significant bit is 2^{-2} . The weight of the least-significant bit is 2^{-24} .

A DEC F floating-point number is evaluated or interpreted as follows:

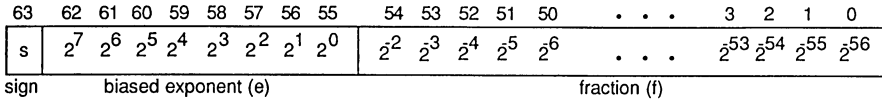
- If $e \neq 0$ value $\neq (-1)^s 2^{e-128}(0.1f)$
- If $s = 0$ and $e = 0$ value = 0
- If $s = 1$ and $e = 0$ value = DEC-Reserved Operand

DEC-Reserved Operand: A DEC-Reserved Operand is interpreted as a signal or symbol. DEC-Reserved Operands are used to indicate invalid operations and operations whose results have overflowed the destination format. They may also be used to pass symbolic information from one calculation to another.

The DEC formats are fully described in the VAX Architecture Manual.

DEC D

The DEC D word is 64 bits wide and is arranged in the format shown below:



TB001080

The floating-point word is divided into three fields: a single-bit sign, an 8-bit biased exponent, and a 55-bit fraction.

The sign bit is 0 for positive numbers and 1 for negative numbers; zero has a positive sign.

The biased exponent is an 8-bit unsigned integer representing a multiplicative factor of some power of two. The bias value is 128. If, for example, the multiplicative value for a floating-point number is to be 2^a , the value of the biased exponent is $a + 128$, where "a" is the true exponent.

The fraction is a 55-bit unsigned fractional field containing the 55 least-significant bits of the floating-point number's 56-bit mantissa. The weight of the fraction's most-significant bit is 2^{-2} . The weight of the least-significant bit is 2^{-56} .

A DEC D floating-point number is evaluated or interpreted as follows:

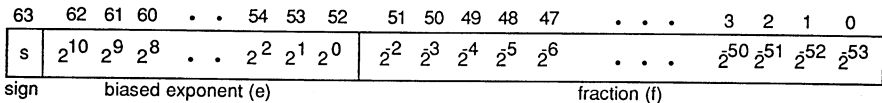
- If $e \neq 0$ value = $(-1)^s 2^{e-128} (0.1f)$
- If $s = 0$ and $e = 0$ value = 0
- If $s = 1$ and $e = 0$ value = DEC-Reserved Operand

DEC-Reserved Operand: A DEC-Reserved Operand is interpreted as a signal or symbol. DEC-Reserved Operands are used to indicate invalid operations and operations whose results have overflowed the destination format. They may also be used to pass symbolic information from one calculation to another.

The DEC formats are fully described in the VAX Architecture Manual.

DEC G

The DEC G word is 64 bits wide and is arranged in the format shown below:



TB001090

The floating-point word is divided into three fields: a single-bit sign, an 11-bit biased exponent, and a 52-bit fraction.

The sign bit is 0 for positive numbers and 1 for negative numbers; zero has a positive sign.

The biased exponent is an 11-bit unsigned integer representing a multiplicative factor of some power of two. The bias value is 1024. If, for example, the multiplicative value for a floating-point number is to be 2^a , the value of the biased exponent is $a + 1024$, where "a" is the true exponent.

The fraction is a 52-bit unsigned fractional field containing the 52 least-significant bits of the floating-point number's 53-bit mantissa. The weight of the fraction's most-significant bit is 2^{-2} . The weight of the least-significant bit is 2^{-53} .

A DEC G floating-point number is evaluated or interpreted as follows:

- If $e \neq 0$ value = $(-1)^s 2^{e-1024} (0.1f)$
- If $s = 0$ and $e = 0$ value = 0
- If $s = 1$ and $e = 0$ value = DEC-Reserved Operand

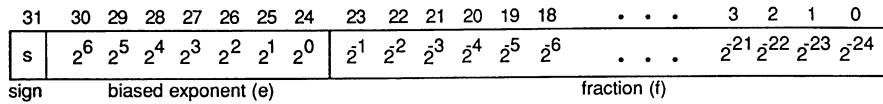
DEC-Reserved Operand: A DEC-Reserved Operand is interpreted as a signal or symbol. DEC-Reserved Operands are used to indicate invalid operations and operations whose results have overflowed the destination format. They may also be used to pass symbolic information from one calculation to another.

The DEC formats are fully described in the VAX Architecture Manual.

IBM Formats

IBM Single-Precision

The IBM single-precision word is 32 bits wide and is arranged in the format shown below:



TB001100

The floating-point word is divided into three fields: a single-bit sign, a 7-bit biased exponent, and a 24-bit fraction.

The sign bit is 0 for positive numbers and 1 for negative numbers; a True-zero has a positive sign.

The biased exponent is a 7-bit unsigned integer representing a multiplicative factor of some power of 16. The bias value is 64. If, for example, the multiplicative value for a floating-point number is to be 16^a , the value of the biased exponent is $a + 64$, where "a" is the true exponent.

The fraction is a 24-bit unsigned fractional field containing the 24 least-significant bits of the floating-point number's 25-bit mantissa. The weight of the fraction's most-significant bit is 2^{-1} . The weight of the least-significant bit is 2^{-24} .

An IBM floating-point number is evaluated or interpreted as follows:

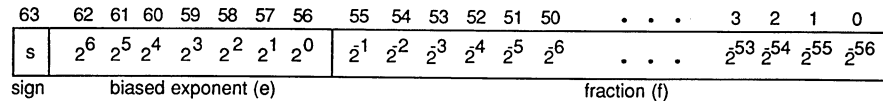
$$\text{value} = (-1)^s 16^{e-64} (0.f)$$

Zero: There are two possible classes of representations for zero. Since there is no leading bit in the IBM format, the range of the IBM fraction is equal to or greater than zero and less than one. If an operation causes the fraction of the result to cancel exactly, then the result is a floating-point zero. A True-zero has a positive sign, a biased exponent of zero, and a fraction of zero.

The IBM format is fully described in the IBM System/370 Principles of Operation Manual.

IBM Double-Precision

The IBM double-precision word is 64 bits wide and is arranged in the format shown below:



TB001110

The floating-point word is divided into three fields: a single-bit sign, a 7-bit biased exponent, and a 56-bit fraction.

The sign bit is 0 for positive numbers and 1 for negative numbers; a True-zero has a positive sign.

The biased exponent is a 7-bit unsigned integer representing a multiplicative factor of some power of 16. The bias value is 64. If, for example, the multiplicative value for a floating-point number is to be 16^a , the value of the biased exponent is $a + 64$, where "a" is the true exponent.

The fraction is a 56-bit unsigned fractional field containing the 56 least-significant bits of the floating-point number's 57-bit mantissa. The weight of the fraction's most-significant bit is 2^{-1} . The weight of the least-significant bit is 2^{-56} .

An IBM floating-point number is evaluated or interpreted as follows:

$$\text{value} = (-1)^s 16^{e-64} (0.f)$$

Zero: There are two possible classes of representations for zero. Since there is no leading bit in the IBM format, the range of the IBM fraction is equal to or greater than zero and less than one. If an operation causes the fraction of the result to cancel exactly, then the result is a floating-point zero. A True-zero has a positive sign, a biased exponent of zero, and a fraction of zero.

The IBM format is fully described in the IBM System/370 Principles of Operation Manual.

APPENDIX B — ROUNDING MODES

The Am29C327 provides six rounding modes for floating-point operations, and for integer multiplication:

Round to Nearest (Unbiased)

The infinitely precise result of an operation is rounded to the closest representable value in the destination format. If the infinitely precise result is exactly halfway between two representations, it is rounded to the representation having a least-significant bit of zero. This rounding mode conforms to the "round to nearest" mode described in the IEEE Floating-Point Standard.

Round to Minus Infinity

The infinitely precise result of an operation is rounded to the closest representable value in the destination format that is less than or equal to the infinitely precise result. This rounding mode conforms to the "round to minus infinity" mode described in the IEEE Floating-Point Standard.

Round to Plus Infinity

The infinitely precise result of an operation is rounded to the closest representable value in the destination format that is greater than or equal to the infinitely precise result. This round

mode conforms to the "round to plus infinity" mode described in the IEEE Floating-Point Standard.

Round to Zero

The infinitely precise result of an operation is rounded to the closest representable value in the destination format whose magnitude is less than or equal to the infinitely precise result. This rounding mode conforms to the "round to zero" mode described in the IEEE Floating-Point Standard.

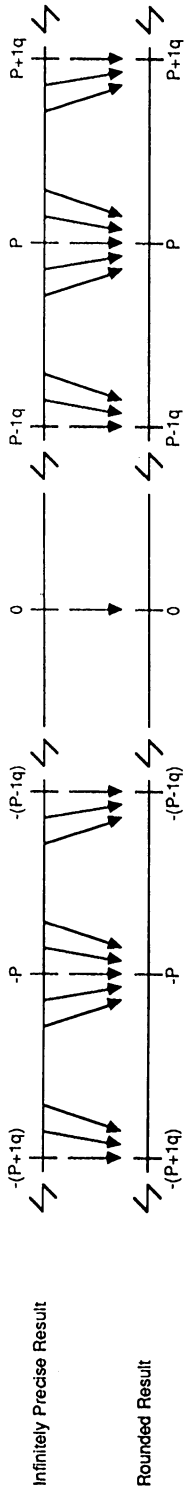
Round to Nearest (Biased)

The infinitely precise result of an operation is rounded to the closest representable value in the destination format. If the infinitely precise result is exactly halfway between two representations, it is rounded to the representation having the greater magnitude. This rounding mode is used by DEC VAX computers.

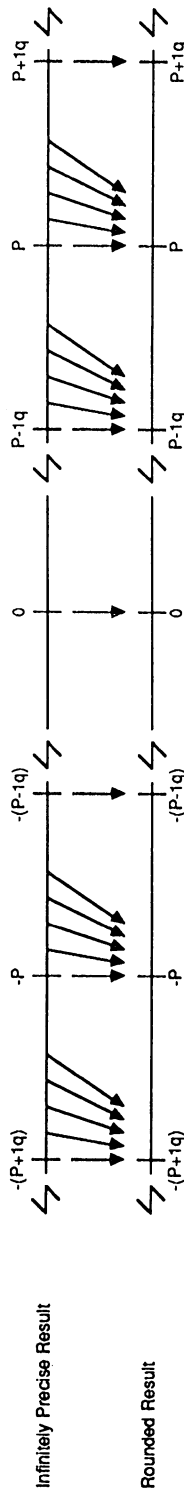
Round Away from Zero

The infinitely precise result of an operation is rounded to the closest representable value in the destination format whose magnitude is greater than or equal to the infinitely precise result.

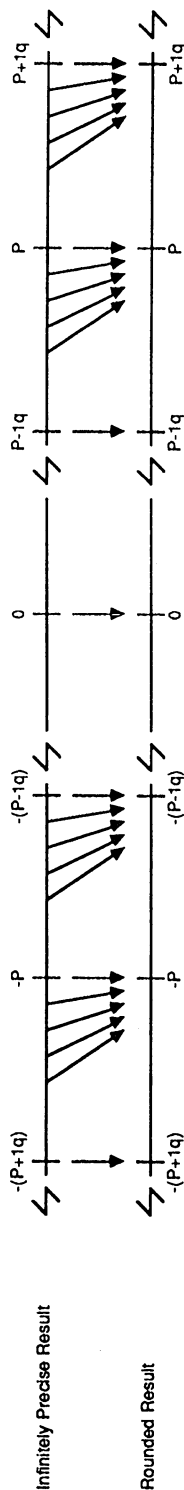
A graphical representation of these rounding modes is shown in Figures B1-1 and B1-2.



ROUND TO NEAREST (UNBIASED)

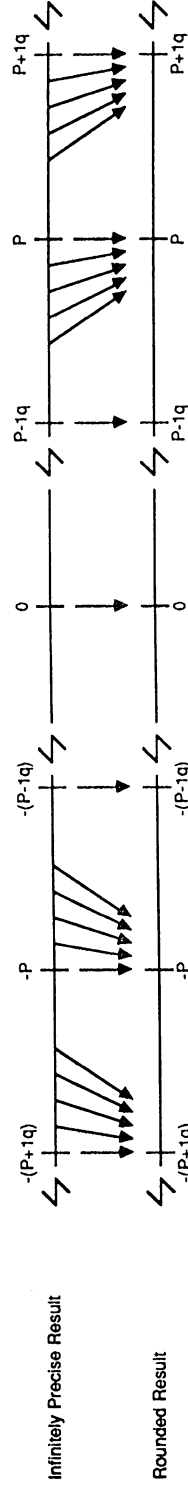
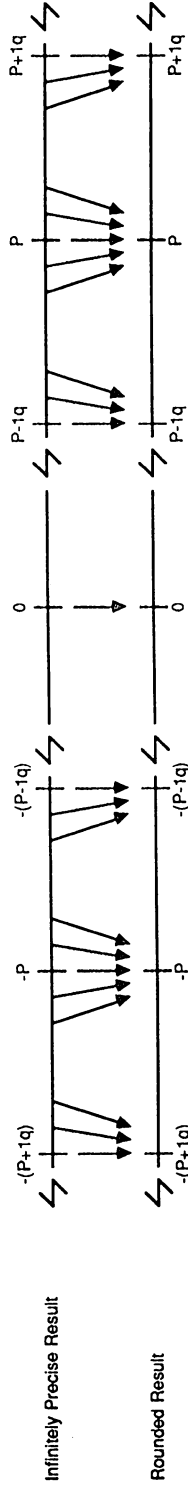
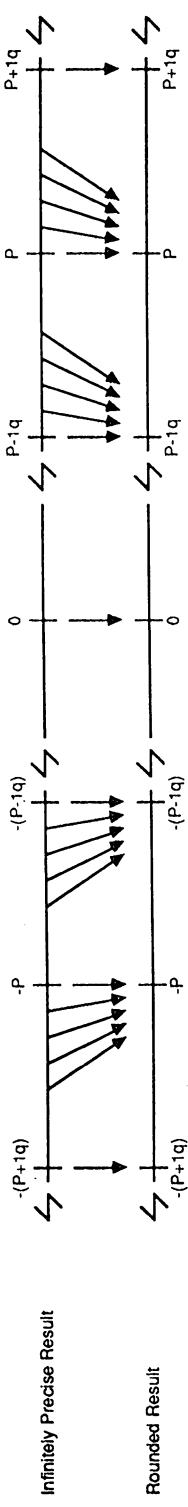


ROUND TO MINUS INFINITY



ROUND TO PLUS INFINITY

Figure B1-1. Graphical Interpretation of Round-to-Nearest (Unbiased), Round-to-Minus-Infinity, and Round-to-Plus-Infinity Rounding Modes



PF002480

Figure B1-2. Graphical Interpretation of Round-to-Zero, Round-to-Nearest, and Round-Away-from-Zero Rounding Modes

APPENDIX C — ADDITIONAL OPERATION DETAILS

Differences Between IEEE Floating-Point Standard and Am29C327 IEEE Operation

The IEEE floating-point standard recommends that a trapped overflow on conversion from a binary format return a result in that or a wider format, rounded to the destination format. The Am29C327 returns an operand in the destination format,

rounded to that format. Note that trapped operation is an optional aspect of the IEEE floating-point standard, and as such, is not necessary for compliance.

Differences Between IBM 370 Floating-Point Arithmetic and Am29C327 IBM Operation

For all arithmetic operations, the Am29C327 in general will produce a more precise result than the IBM 370.

Differences Between DEC Floating-Point Arithmetic and Am29C327 DEC Operation

The Am29C327 and DEC VAX floating-point formats contain identical information, but the sub-fields of the floating-point words are arranged differently:

The Am29C327 DEC F format is:

sign - bit 31
exponent - bits 30 - 23
mantissa - bits 22 - 0

The Am29C327 DEC D format is:

sign - bit 63
exponent - bits 62 - 55
mantissa - bits 54 - 0

The VAX format is:

sign - bit 15
exponent - bits 14 - 7
mantissa - bits 6 - 0,
bits 31 - 16

The VAX format is:

sign - bit 15
exponent - bits 14 - 7
mantissa - bits 6 - 0,
bits 31 - 16,
bits 47 - 32,
bits 63 - 48
bit 6 = MSB,
bit 48 = LSB

ABSOLUTE MAXIMUM RATINGS

Storage Temperature	-65 to +150°C
Ambient Temperature (T _A)	
Under Bias	-55 to +125°C
Supply Voltage to	
Ground Potential Continuous	-0.5 to +7.0 V
DC Voltage Applied to	
Outputs for HIGH State	-0.5 V to +V _{CC} Max.
DC Input Voltage	-0.5 to +5.5 V
DC Output Current, Into Outputs	30 mA
DC Input Current	-10 to +10 mA

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

Commercial (C) Devices	
Temperature (T _A)	0 to +70°C
Supply Voltage (V _{CC})	+5 V ± 5%
Min.	+4.75 V
Max.	+5.25 V
Military (M) Devices	
Temperature (T _A)	-55 to +125°C
Supply Voltage (V _{CC})	+5 V ± 10%
Min.	+4.5 V
Max.	+5.5 V

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

DC CHARACTERISTICS over operating range unless otherwise specified

Parameter Symbol	Parameter Description	Test Conditions (Note 1)	Min.	Max.	Unit
V _{OH}	Output HIGH Voltage	V _{CC} = Min. V _{IN} = V _{IL} or V _{IH} V _{IL} = 0.8 V V _{IH} = 2.0 V I _{OH} = -0.4 mA	2.4		V
V _{OL}	Output LOW Voltage	V _{CC} = Min. V _{IN} = V _{IL} or V _{IH} V _{IL} = 0.8 V V _{IH} = 2.0 V I _{OL} = 4.0 mA		0.5	V
V _{IH}	Input HIGH Level	Guaranteed Input Logical-HIGH Voltage for All Inputs	2.0		V
V _{IL}	Input LOW Level	Guaranteed Input Logical-LOW Voltage for All Inputs		0.8	V
V _I	Input Clamp Voltage	V _{CC} = Min. I _{IN} = -18 mA		-1.5	V
I _{IL}	Input LOW Current	V _{CC} = Max. V _{IN} = 0.4 V		-0.4	mA
I _{IH}	Input HIGH Current	V _{CC} = Max. V _{IN} = 2.4 V		75	μA
I _I	Input HIGH Current	V _{CC} = Max. V _{IN} = 5.5 V		1	mA
I _{OZH}	Off-State (High-Impedance) Output Current	V _{CC} = Max. V _O = 2.4 V		25	μA
I _{OZL}				-25	
I _{SC} (Note 2)	Output Short-Circuit Current	V _{CC} = Max. V _O = 0 V All Outputs	-3	-30	mA
I _{CC} (Note 3)	Power Supply Current	COM'L		300	mA
			MIL		
I _{CCQ1} (Note 4)	Quiescent Power Supply Current	COM'L			mA
			MIL		
I _{CCQ2} (Note 5)	Quiescent Power Supply Current	COM'L			mA
			MIL		

- Notes: 1. For conditions shown as Min. or Max., use the appropriate value specified under Electrical Characteristics for the applicable device type.
 2. Not more than one output should be shorted at a time. Duration of the short-circuit test should not exceed one second.
 3. I_{CC} is measured with clock frequency = 8 MHz and with outputs disabled. Inputs should be presented with random logic-HIGHs and LOWs to assure the toggling of internal nodes.
 4. V_{IN} ≥ V_{IH}, V_{IN} ≤ V_{IL}
 5. V_{IN} ≥ V_{CC} - 0.2 V, V_{IN} ≤ 0.2 V

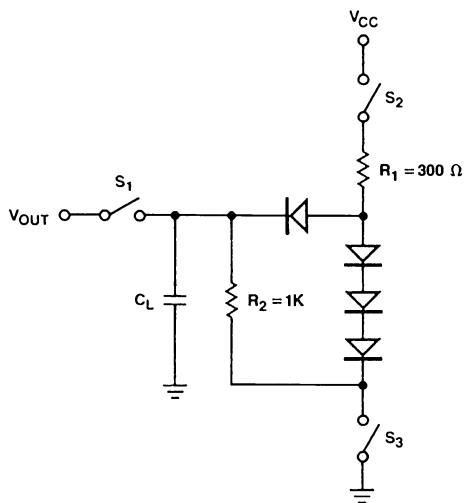
SWITCHING CHARACTERISTICS over operating range unless otherwise specified

No.	Parameter Description	Test Conditions	Min.	Max.	Unit
1	CLK Period Flow-Through Mode Multiply-Accumulate All Other Operations Single-Pipelined Mode Multiply-Accumulate All Other Operations Double-Pipelined Mode Multiply-Accumulate	(Note 1)	360 240	DC DC	ns ns
2	CLK LOW Time				ns
3	CLK HIGH Time				ns
4	CLK Rise Time	(Note 2)			ns
5	CLK Fall Time	(Note 2)			ns
6	Data/Instruction Setup Time	(Note 3)	15		ns
7	Data/Instruction Hold Time	(Note 3)	0		ns
8	Control Lines Setup Time	(Note 4)	15		ns
9	Control Lines Hold Time	(Note 4)	0		ns
10	F ₀₋₃₁ CLK-to-Output-Valid F Register Clocked			20	ns
11	FLAG ₁₋₆ SIGN CLK-to-Output-Valid Register Clocked			20	ns
12	F ₀₋₃₁ CLK-to-Output-Valid F Register Transparent Flow-Through Mode Multiply-Accumulate All Other Operations Single-Pipelined Mode Multiply-Accumulate All Other Operations Double-Pipelined Mode Multiply-Accumulate			380 260 260 140 140	ns ns ns ns ns
13	FLAG ₁₋₆ SIGN CLK-to-Output-Valid S Register Transparent Flow-Through Mode Multiply-Accumulate All Other Operations Single-Pipelined Mode Multiply-Accumulate All Other Operations Double-Pipelined Mode Multiply-Accumulate			380 260 260 140 140	ns ns ns ns ns
14	OE _F , OE _S , Disable Time HIGH to Z			15	ns
15	OE _F , OE _S , Disable Time LOW to Z			15	ns
16	OE _F , OE _S , Enable Time Z to HIGH			20	ns
17	OE _F , OE _S , Disable Time Z to LOW			20	ns
18	FSEL to F ₀₋₃₁			20	ns
19	MSERR Data-to-Valid Delay			20	ns

- Notes:** 1. CLK switching characteristics are made relative to 2.5 V.
 2. CLK rise time and fall time measured between 0.8 V and (V_{CC}-1.0 V).
 3. Data/Instruction signals include R₀₋₃₁, S₀₋₃₁, S/DR, S/DS, S/DF, RM₀₋₂, PSEL₀₋₃, QSEL₀₋₃, TSEL₀₋₃ and I₀₋₁₃.
 4. Control signals include ENR, ENS, ENF, ENRF, RFSEL₀₋₂, FSEL, ENI, OEF, and OES.

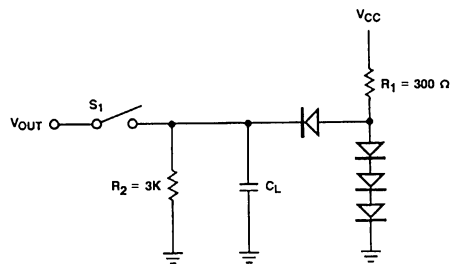
- Conditions:** A. All inputs/outputs except CLK are TTL-compatible for V_{IH}, V_{IL}, and V_{OL}.
 B. All outputs are driving 80 pF unless otherwise noted.
 C. All setup, hold, and delay times are measured relative to CLK at V_{CC}/2 volts unless otherwise noted.

SWITCHING TEST CIRCUITS



TCR01331

A. Three-State Outputs

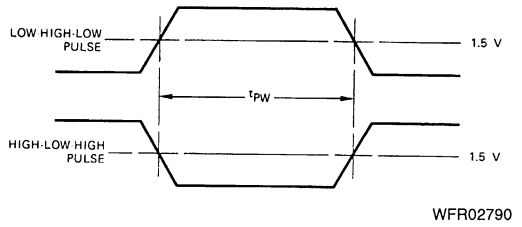
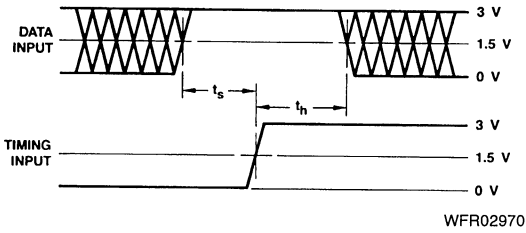


TC000421

B. Normal Outputs

- Notes:
1. $C_L = 50$ pF includes scope probe, wiring, and stray capacitances without device in test fixture.
 2. S_1, S_2, S_3 are closed during function tests and all AC tests except output enable tests.
 3. S_1 and S_3 are closed while S_2 is open for t_{pZH} test.
 4. $C_L = 5.0$ pF for output disable tests.

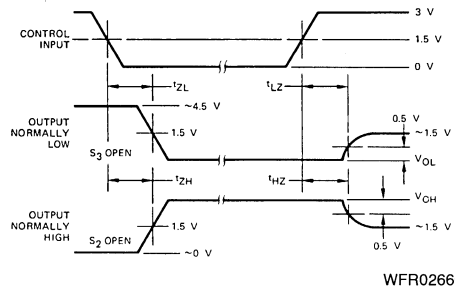
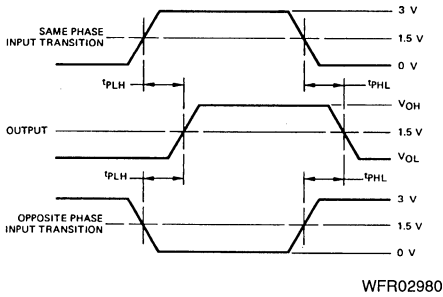
SWITCHING TEST WAVEFORMS



- Notes: 1. Diagram shown for HIGH data only. Output transition may be opposite sense.
 2. Cross-hatched area is don't care condition.

Setup, Hold, and Release Times

Pulse Width








- Notes: 1. Diagram shown for Input Control Enable-LOW and Input Control Disable-HIGH.
 2. S_1 , S_2 and S_3 of Load Circuit are closed except where shown.

Propagation Delay

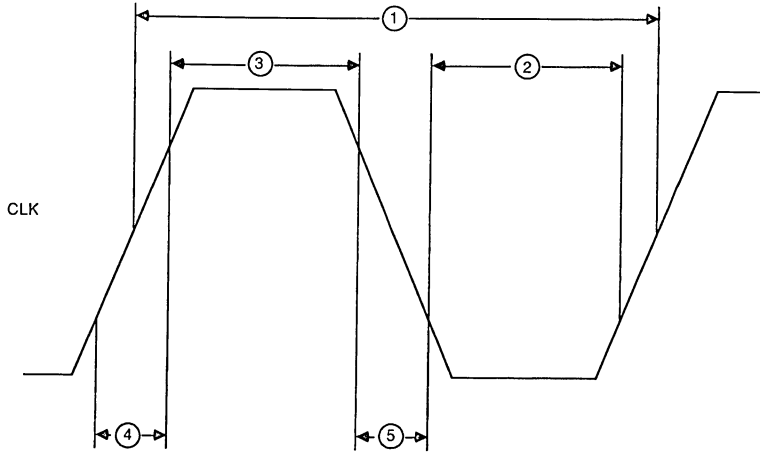
Enable and Disable Times

SWITCHING WAVEFORMS

KEY TO SWITCHING WAVEFORMS

WAVEFORM	INPUTS	OUTPUTS
	MUST BE STEADY	WILL BE STEADY
	MAY CHANGE FROM H TO L	WILL BE CHANGING FROM H TO L
	MAY CHANGE FROM L TO H	WILL BE CHANGING FROM L TO H
	DON'T CARE; ANY CHANGE PERMITTED	CHANGING; STATE UNKNOWN
	DOES NOT APPLY	CENTER LINE IS HIGH IMPEDANCE "OFF" STATE

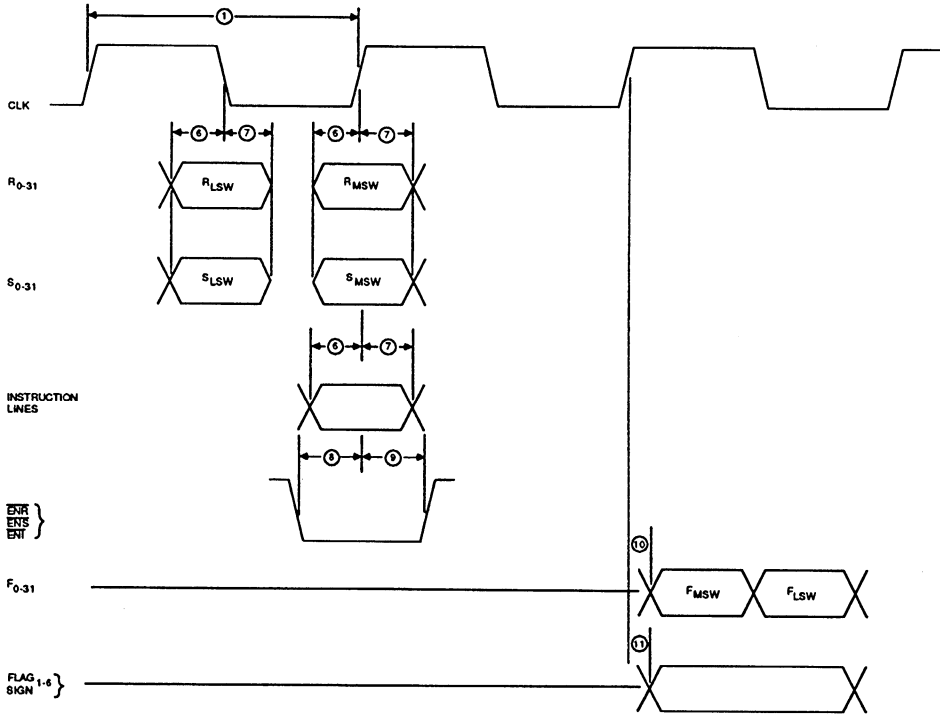
KS000010



WF025010

Input Clock Timing

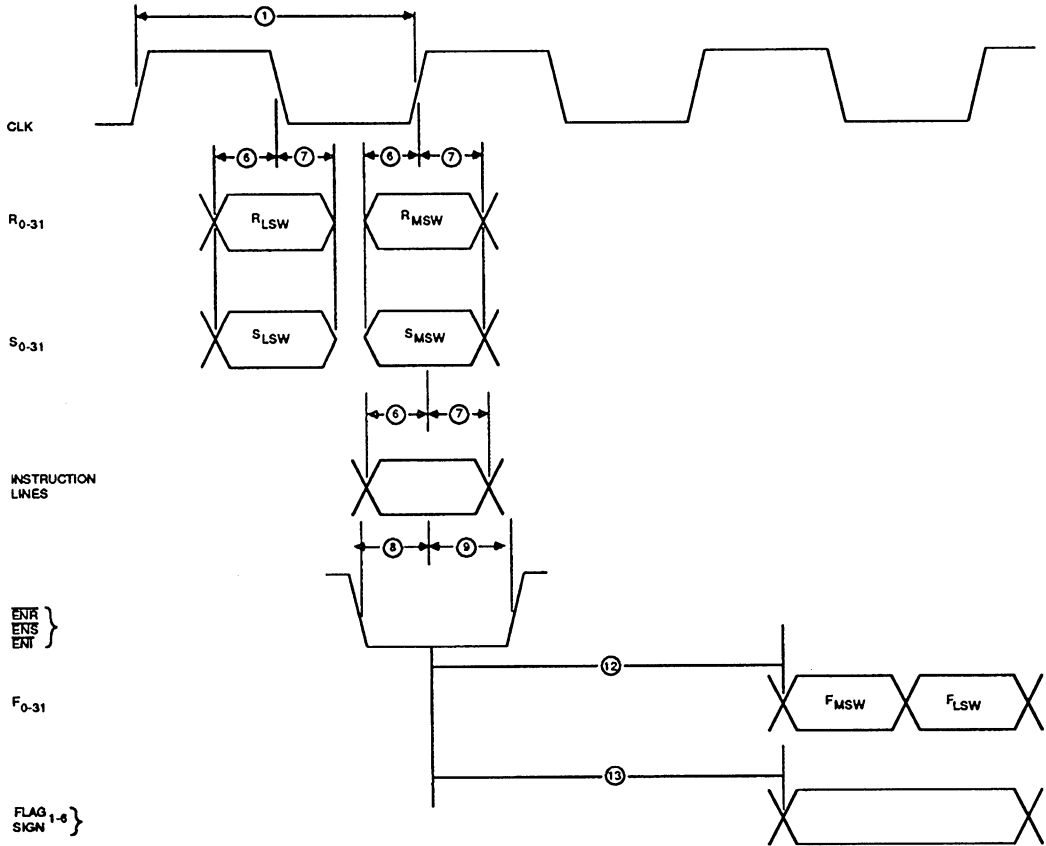
SWITCHING WAVEFORMS (Cont'd.)



WF025020

Timing of Operations with F Register and Status Clocked. Assumes 32-Bit Bus, Single-Cycle, LSW-First Input Mode and Flow-Through Operation

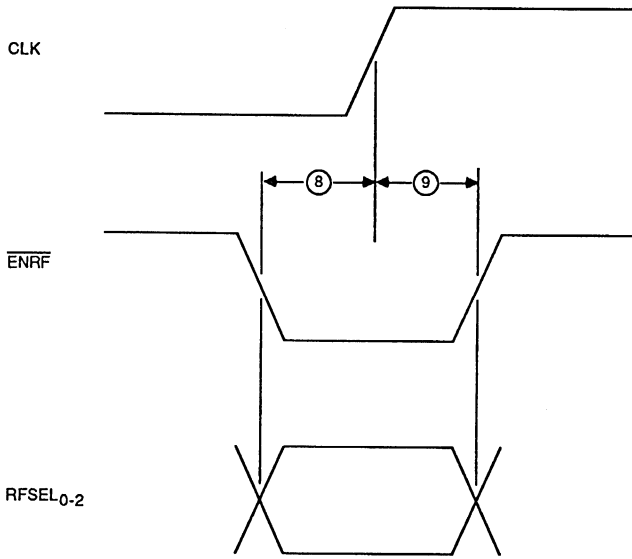
SWITCHING WAVEFORMS (Cont'd.)



WF025030

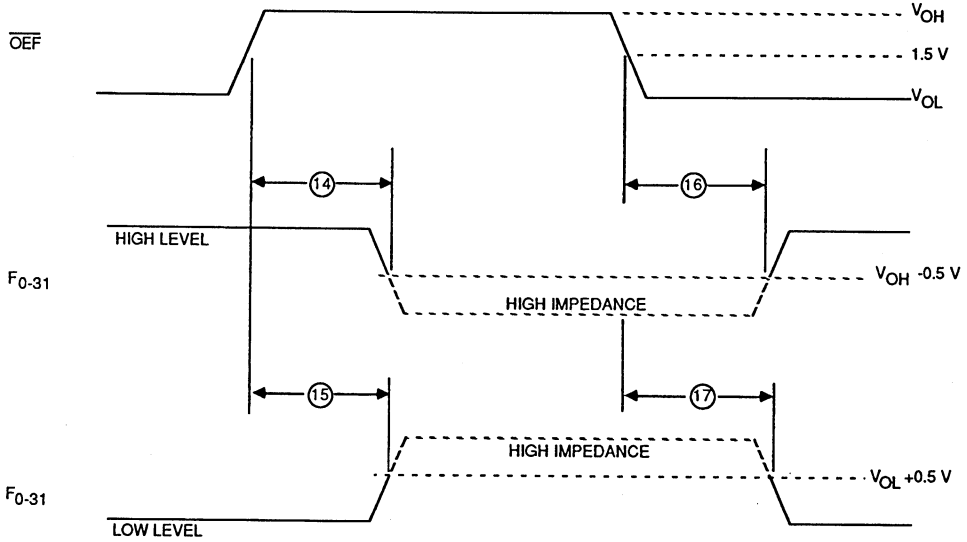
Timing of Operations with F-Register and Status Register in Feedthrough Mode. Assumes 32-Bit Bus, Single-Cycle, LSW-First Input Mode and Flow-Through Operation.

SWITCHING WAVEFORMS (Cont'd.)



WF025040

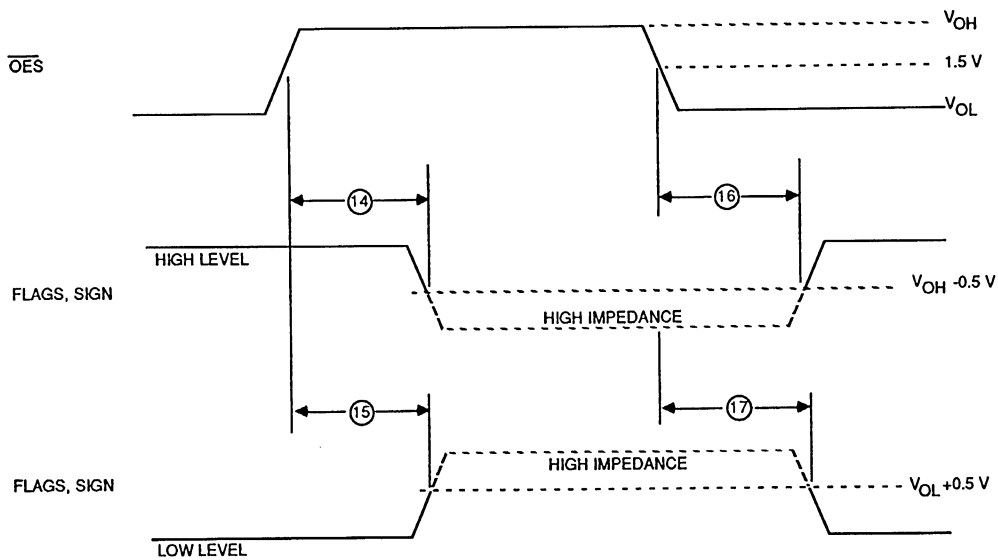
Register File Control Timing



WF025050

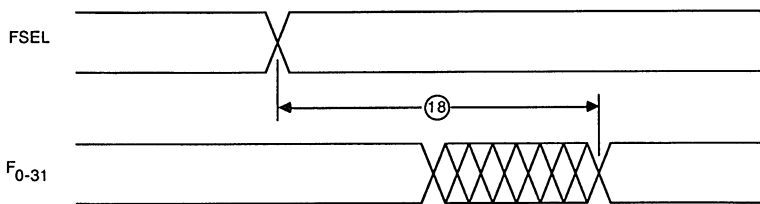
Enable/Disable Timing for F₀₋₃₁

SWITCHING WAVEFORMS (Cont'd.)



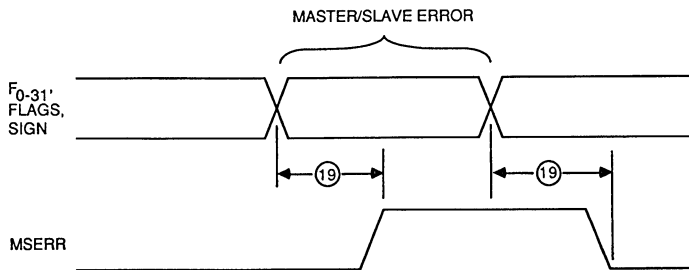
WF025060

Enable/Disable Timing for FLAG₁₋₆ and SIGN



WF025070

Output Selection Timing



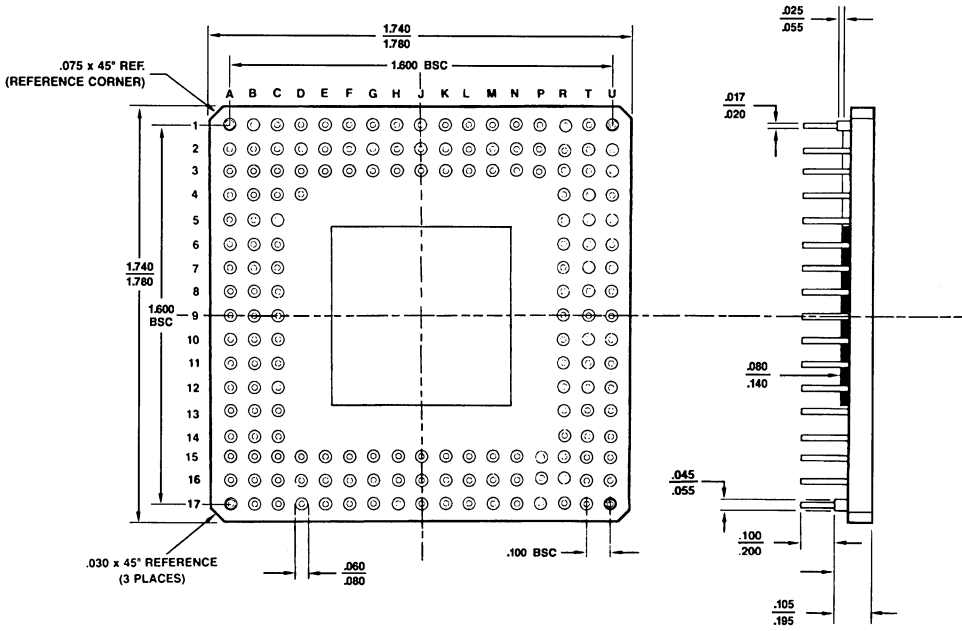
WF025080

Master/Slave Timing (Assumes SLAVE Mode)

PHYSICAL DIMENSIONS*

CGX169

BOTTOM VIEW



PID # 07322B

*For reference only.

ADVANCED MICRO DEVICES' U.S. SALES OFFICES

ALABAMA	(205) 882-9122	KANSAS	(913) 451-3115
ARIZONA, Tempe	(602) 242-4400	MARYLAND	(301) 796-9310
CALIFORNIA, Culver City	(213) 645-1524	MASSACHUSETTS	(617) 273-3970
Newport Beach	(714) 752-6262	MINNESOTA	(612) 938-0001
San Diego	(619) 560-7030	MISSOURI	(314) 275-4415
Santa Clara	(408) 727-3270	NEW JERSEY	(201) 299-0002
Woodland Hills	(818) 992-4155	NEW YORK, Liverpool	(315) 457-5400
COLORADO	(303) 741-2900	Poughkeepsie	(914) 471-8180
CONNECTICUT	(203) 264-7800	Woodbury	(516) 364-8020
FLORIDA, Clearwater	(813) 530-9971	NORTH CAROLINA	(919) 847-8471
Ft Lauderdale	(305) 484-8600	OREGON	(503) 245-0080
Melbourne	(305) 729-0496	OHIO	(614) 891-6455
Orlando	(305) 859-0831	PENNSYLVANIA, Allentown	(215) 398-8006
GEORGIA	(404) 449-7920	Willow Grove	(215) 657-3101
ILLINOIS, Chicago	(312) 773-4422	TEXAS, Austin	(512) 346-7830
Naperville	(312) 505-9517	Dallas	(214) 934-9099
INDIANA	(317) 244-7207	Houston	(713) 785-9001
		WASHINGTON	(206) 455-3600
		WISCONSIN	(414) 792-0590

ADVANCED MICRO DEVICES' INTERNATIONAL SALES OFFICES

BELGIUM, Bruxelles	TEL .. (02) 771 91 42	JAPAN, Tokyo	TEL	(03) 345-8241
	FAX .. (02) 762 37 12		FAX	3425196
	TLX		TLX	J24064AMDTKOJ
CANADA, Ontario, Kanata	TEL .. (613) 592-0060	Osaka	TEL	06-243-3250
Willowdale	TEL .. (416) 224-5193		FAX	06-243-3253
	FAX .. (416) 224-5193			
FRANCE, Paris	TEL (01) 49-75-10-10	KOREA, Seoul	TEL	82-2-784-7598
	FAX (01) 49-75-10-13		FAX	82-2-784-8014
	TLX			
GERMANY, Hannover area	TEL ... (05143) 50 55	LATIN AMERICA, Ft. Lauderdale	TEL	(305) 484-8600
	FAX ... (05143) 55 53		FAX	(305) 485-9736
	TLX		TLX ..	5109554261 AMDFTL
München	TEL ... (089) 41 14-0	SWEDEN, Stockholm	TEL	(08) 733 03 50
	FAX ... (089) 406490		FAX	(08) 733 22 85
	TLX		TLX	11602
Stuttgart	TEL .. (0711) 62 33 77	TAIWAN	TLX	886-2-7122066
	FAX .. (0711) 625187		FAX	886-2-7122017
	TLX			
HONG KONG, Kowloon	TEL	UNITED KINGDOM, Manchester area	TEL	(0925) 828008
	FAX		FAX	(0925) 827693
	TLX 504260AMDAPHX		TLX	628524
		London area	TEL	(04862) 22121
			FAX	(04862) 22179
			TLX	859103
ITALY, Milano	TEL ... (02) 3390541		TLX	886-2-7122066
	FAX ... (02) 3498000		FAX	886-2-7122017
	TLX			

NORTH AMERICAN REPRESENTATIVES

CALIFORNIA I ² INC	OEM (408) 988-3400	MICHIGAN SAI MARKETING CORP	(313) 750-1922
	DISTI (408) 498-6868	MISSOURI LORENZ SALES	(314) 997-4558
CANADA Calgary, Alberta VITEL ELECTRONICS	(403) 278-5833	NEBRASKA LORENZ SALES	(402) 475-4660
Kanata, Ontario VITEL ELECTRONICS	(613) 592-0090	NEW MEXICO THORSON DESERT STATES	(505) 293-8555
Mississauga, Ontario VITAL ELECTRONICS	(416) 676-9720	NEW YORK NYCOM, INC	(315) 437-8343
Quebec VITEL ELECTRONICS	(514) 636-5951	OHIO Columbus DOLFUSS ROOT & CO	(614) 885-4844
IDAHO INTERMOUNTAIN TECH MKGT	(208) 888-6071	Dayton DOLFUSS ROOT & CO	(513) 433-6776
INDIANA SAI MARKETING CORP	(317) 253-1668	Strongsville DOLFUSS ROOT & CO	(216) 238-0300
IOWA LORENZ SALES	(319) 377-4666	PENNSYLVANIA DOLFUSS ROOT & CO	(412) 221-4420
KANSAS LORENZ SALES	(913) 384-6556	UTAH R ² MARKETING	(801) 595-0631

Advanced Micro Devices reserves the right to make changes in its product without notice in order to improve design or performance characteristics. The performance characteristics listed in this document are guaranteed by specific tests, guard banding, design and other practices common to the industry. For specific testing details, contact your local AMD sales representative. The company assumes no responsibility for the use of any circuits described herein.