



## NS32081-10/NS32081-15 Floating-Point Units

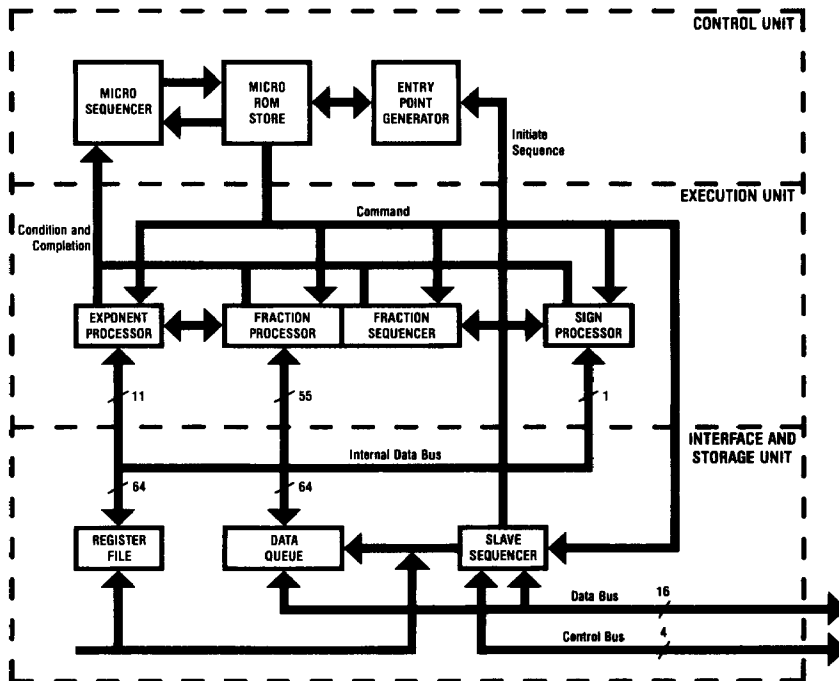
### General Description

The NS32081 Floating-Point Unit functions as a slave processor in National Semiconductor's Series 32000® micro-processor family. It provides a high-speed floating-point instruction set for any Series 32000 family CPU, while remaining architecturally consistent with the full two-address architecture and powerful addressing modes of the Series 32000 micro-processor family.

### Features

- Eight on-chip data registers
- 32-bit and 64-bit operations
- Supports proposed IEEE standard for binary floating-point arithmetic, Task P754
- Directly compatible with NS32016, NS32008 and NS32032 CPUs
- High-speed XMOSTM technology
- Single 5V supply
- 24-pin dual in-line package

### Block Diagram



TL/EE/5234-1

## Table of Contents

### 1.0 PRODUCT INTRODUCTION

- 1.1 Operand Formats
  - 1.1.1 Normalized Numbers
  - 1.1.2 Zero
  - 1.1.3 Reserved Operands
  - 1.1.4 Integers
  - 1.1.5 Memory Representations

### 2.0 ARCHITECTURAL DESCRIPTION

- 2.1 Programming Model
  - 2.1.1 Floating-Point Registers
  - 2.1.2 Floating-Point Status Register (FSR)
    - 2.1.2.1 FSR Mode Control Fields
    - 2.1.2.2 FSR Status Fields
    - 2.1.2.3 FSR Software Field (SWF)
- 2.2 Instruction Set
  - 2.2.1 General Instruction Format
  - 2.2.2 Addressing Modes
  - 2.2.3 Floating-Point Instruction Set
- 2.3 Traps

### 3.0 FUNCTIONAL DESCRIPTION

- 3.1 Power and Grounding
- 3.2 Clocking
- 3.3 Resetting

### 3.0 FUNCTIONAL DESCRIPTION (Continued)

- 3.4 Bus Operation
  - 3.4.1 Bus Cycles
  - 3.4.2 Operand Transfer Sequences
- 3.5 Instruction Protocols
  - 3.5.1 General Protocol Sequence
  - 3.5.2 Floating-Point Protocols

### 4.0 DEVICE SPECIFICATIONS

- 4.1 Pin Descriptions
  - 4.1.1 Supplies
  - 4.1.2 Input Signals
  - 4.1.3 Input/Output Signals
- 4.2 Absolute Maximum Ratings
- 4.3 Electrical Characteristics
- 4.4 Switching Characteristics
  - 4.4.1 Definitions
  - 4.4.2 Timing Tables
    - 4.4.2.1 Output Signals: Internal Propagation Delays
    - 4.4.2.2 Input Signals Requirements
    - 4.4.2.3 Clocking Requirements
  - 4.4.3 Timing Diagrams

## List of Illustrations

|                                                                      |      |
|----------------------------------------------------------------------|------|
| Floating-Point Operand Formats .....                                 | 1-1  |
| Register Set .....                                                   | 2-1  |
| The Floating-Point Status Register .....                             | 2-2  |
| General Instruction Format .....                                     | 2-3  |
| Index Byte Format .....                                              | 2-4  |
| Displacement Encodings .....                                         | 2-5  |
| Floating-Point Instruction Formats .....                             | 2-6  |
| Recommended Supply Connections .....                                 | 3-1  |
| Power-On Reset Requirements .....                                    | 3-2  |
| General Reset Timing .....                                           | 3-3  |
| System Connection Diagram .....                                      | 3-4  |
| Slave Processor Read Cycle .....                                     | 3-5  |
| Slave Processor Write Cycle .....                                    | 3-6  |
| FPU Protocol Status Word Format .....                                | 3-7  |
| Dual-In-Line Package .....                                           | 4-1  |
| Timing Specification Standard (Signal Valid After Clock Edge) .....  | 4-2  |
| Timing Specification Standard (Signal Valid Before Clock Edge) ..... | 4-3  |
| Clock Timing .....                                                   | 4-4  |
| Power-On-Reset .....                                                 | 4-5  |
| Non-Power-On-Reset .....                                             | 4-6  |
| Read Cycle From FPU .....                                            | 4-7  |
| Write Cycle To FPU .....                                             | 4-8  |
| $\overline{\text{SPC}}$ Pulse from FPU .....                         | 4-9  |
| $\overline{\text{RST}}$ Release Timing .....                         | 4-10 |

## List of Tables

|                                            |     |
|--------------------------------------------|-----|
| Sample F Fields .....                      | 1-1 |
| Sample E Fields .....                      | 1-2 |
| Normalized Number Ranges .....             | 1-3 |
| Series 32000 Family Addressing Modes ..... | 2-1 |
| General Instruction Protocol .....         | 3-1 |
| Floating-Point Instruction Protocols ..... | 3-2 |

# 1.0 Product Introduction

The NS32081 Floating-Point Unit (FPU) provides high speed floating-point operations for the Series 32000 family, and is fabricated using National high-speed CMOS technology. It operates as a slave processor for transparent expansion of the Series 32000 CPU's basic instruction set. The FPU can also be used with other microprocessors as a peripheral device by using additional TTL interface logic. The NS32081 is compatible with the IEEE Floating-Point Formats by means of its hardware and software features.

## 1.1 OPERAND FORMATS

The NS32081 FPU operates on two floating-point data types—single precision (32 bits) and double precision (64 bits). Floating-point instruction mnemonics use the suffix F (Floating) to select the single precision data type, and the suffix L (Long Floating) to select the double precision data type.

A floating-point number is divided into three fields, as shown in Figure 1-1.

The F field is the fractional portion of the represented number. In Normalized numbers (Section 1.1.1), the binary point is assumed to be immediately to the left of the most significant bit of the F field, with an implied 1 bit to the left of the binary point. Thus, the F field represents values in the range  $1.0 \leq x \leq 2.0$ .

TABLE 1-1. Sample F Fields

| F Field | Binary Value | Decimal Value |
|---------|--------------|---------------|
| 000...0 | 1.000...0    | 1.000...0     |
| 010...0 | 1.010...0    | 1.250...0     |
| 100...0 | 1.100...0    | 1.500...0     |
| 110...0 | 1.110...0    | 1.750...0     |

↑  
Implied Bit

The E field contains an unsigned number that gives the binary exponent of the represented number. The value in the E field is biased; that is, a constant bias value must be subtracted from the E field value in order to obtain the true exponent. The bias value is 011...11<sub>2</sub>, which is either 127 (single precision) or 1023 (double precision). Thus, the true exponent can be either positive or negative, as shown in Table 1-2.

TABLE 1-2. Sample E Fields

| E Field   | F Field | Represented Value          |
|-----------|---------|----------------------------|
| 011...110 | 100...0 | $1.5 \times 2^{-1} = 0.75$ |
| 011...111 | 100...0 | $1.5 \times 2^0 = 1.50$    |
| 100...000 | 100...0 | $1.5 \times 2^1 = 3.00$    |

Two values of the E field are not exponents. 11...11 signals a reserved operand (Section 2.1.3). 00...00 represents the number zero if the F field is also all zeroes, otherwise it signals a reserved operand.

The S bit indicates the sign of the operand. It is 0 for positive and 1 for negative. Floating-point numbers are in sign-magnitude form, that is, only the S bit is complemented in order to change the sign of the represented number.

### 1.1.1 Normalized Numbers

Normalized numbers are numbers which can be expressed as floating-point operands, as described above, where the E field is neither all zeroes nor all ones.

The value of a Normalized number can be derived by the formula:

$$(-1)^S \times 2^{(E-Bias)} \times (1 + F)$$

The range of Normalized numbers is given in Table 1-3.

### 1.1.2 Zero

There are two representations for zero—positive and negative. Positive zero has all-zero F and E fields, and the S bit is zero. Negative zero also has all-zero F and E fields, but its S bit is one.

### 1.1.3 Reserved Operands

The proposed IEEE Standard for Binary Floating-Point Arithmetic (Task P754) provides for certain exceptional forms of floating-point operands. The NS32081 FPU treats these forms as reserved operands. The reserved operands are:

- Positive and negative infinity
- Not-a-Number (NaN) values
- Denormalized numbers

Both Infinity and NaN values have all ones in their E fields. Denormalized numbers have all zeroes in their E fields and non-zero values in their F fields.

The NS32081 FPU causes an Invalid Operation trap (Section 2.1.2.2) if it receives a reserved operand, unless the operation is simply a move (without conversion). The FPU does not generate reserved operands as results.

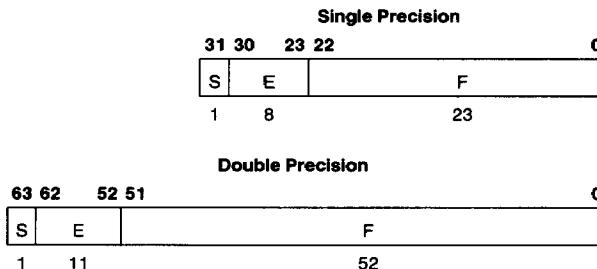


FIGURE 1-1. Floating-Point Operand Formats

## 1.0 Product Introduction (Continued)

TABLE 1-3. Normalized Number Ranges

|                | Single Precision                                                  | Double Precision                                                            |
|----------------|-------------------------------------------------------------------|-----------------------------------------------------------------------------|
| Most Positive  | $2^{127} \times (2 - 2^{-23})$<br>$= 3.40282346 \times 10^{38}$   | $2^{1023} \times (2 - 2^{-52})$<br>$= 1.7976931348623157 \times 10^{308}$   |
| Least Positive | $2^{-126}$<br>$= 1.17549436 \times 10^{-38}$                      | $2^{-1022}$<br>$= 2.2250738585072014 \times 10^{-308}$                      |
| Least Negative | $-(2^{-126})$<br>$= -1.17549436 \times 10^{-38}$                  | $-(2^{-1022})$<br>$= -2.2250738585072014 \times 10^{-308}$                  |
| Most Negative  | $-2^{127} \times (2 - 2^{-23})$<br>$= -3.40282346 \times 10^{38}$ | $-2^{1023} \times (2 - 2^{-52})$<br>$= -1.7976931348623157 \times 10^{308}$ |

**Note:** The values given are extended one full digit beyond their represented accuracy to help in generating rounding and conversion algorithms.

### 1.1.4 Integers

In addition to performing floating-point arithmetic, the NS32081 FPU performs conversions between integer and floating-point data types. Integers are accepted or generated by the FPU as two's complement values of byte (8 bits), word (16 bits) or double word (32 bits) length.

### 1.1.5 Memory Representations

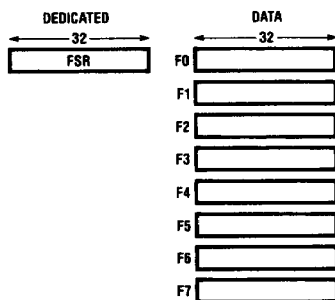
The NS32081 FPU does not directly access memory. However, it is cooperatively involved in the execution of a set of two-address instructions with its Series 32000 Family CPU. The CPU determines the representation of operands in memory.

In the Series 32000 family of CPUs, operands are stored in memory with the least significant byte at the lowest byte address. The only exception to this rule is the Immediate addressing mode, where the operand is held (within the instruction format) with the most significant byte at the lowest address.

## 2.0 Architectural Description

### 2.1 PROGRAMMING MODEL

The Series 32000 architecture includes nine registers that are implemented on the NS32081 Floating-Point Unit (FPU).



TL/EE/5234-4

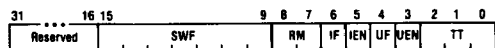
FIGURE 2-1. Register Set

### 2.1.1 Floating-Point Registers

There are eight registers (F0-F7) on the NS32081 FPU for providing high-speed access to floating-point operands. Each is 32 bits long. A floating-point register is referenced whenever a floating-point instruction uses the Register addressing mode (Section 2.2.2) for a floating-point operand. All other Register mode usages (i.e., integer operands) refer to the General Purpose Registers (R0-R7) of the CPU, and the FPU transfers the operand as if it were in memory. When the Register addressing mode is specified for a double precision (64-bit) operand, a pair of registers holds the operand. The programmer must specify the even register of the pair. The even register contains the least significant half of the operand and the next consecutive register contains the most significant half.

### 2.1.2 Floating-Point Status Register (FSR)

The Floating-Point Status Register (FSR) selects operating modes and records any exceptional conditions encountered during execution of a floating-point operation. Figure 2-2 shows the format of the FSR.



TL/EE/5234-5

FIGURE 2-2. The Floating-Point Status Register

#### 2.1.2.1 FSR Mode Control Fields

The FSR mode control fields select FPU operation modes. The meanings of the FSR mode control bits are given below.

**Rounding Mode (RM):** Bits 7 and 8. This field selects the rounding method. Floating-point results are rounded whenever they cannot be exactly represented. The rounding modes are:

- 00 Round to nearest value. The value which is nearest to the exact result is returned. If the result is exactly halfway between the two nearest values the even value (LSB = 0) is returned.
- 01 Round toward zero. The nearest value which is closer to zero or equal to the exact result is returned.

## 2.0 Architectural Description (Continued)

10 Round toward positive infinity. The nearest value which is greater than or equal to the exact result is returned.

11 Round toward negative infinity. The nearest value which is less than or equal to the exact result is returned.

**Underflow Trap Enable (UEN):** Bit 3. If this bit is set, the FPU requests a trap whenever a result is too small in absolute value to be represented as a normalized number. If it is not set, any underflow condition returns a result of exactly zero.

**Inexact Result Trap Enable (IEN):** Bit 5. If this bit is set, the FPU requests a trap whenever the result of an operation cannot be represented exactly in the operand format of the destination. If it is not set, the result is rounded according to the selected rounding mode.

### 2.1.2.2 FSR Status Fields

The FSR Status Fields record exceptional conditions encountered during floating-point data processing. The meanings of the FSR status bits are given below:

**Trap Type (TT):** bits 0-2. This 3-bit field records any exceptional condition detected by a floating-point instruction. The TT field is loaded with zero whenever any floating-point instruction except LFSR or SFSR completes without encountering an exceptional condition. It is also set to zero by a hardware reset or by writing zero into it with the Load FSR (LFSR) instruction. Underflow and Inexact Result are always reported in the TT field, regardless of the settings of the UEN and IEN bits.

000 No exceptional condition occurred.

001 Underflow. A non-zero floating-point result is too small in magnitude to be represented as a normalized floating-point number in the format of the destination operand. This condition is always reported in the TT field and UF bit, but causes a trap only if the UEN bit is set. If the UEN bit is not set, a result of Positive Zero is produced, and no trap occurs.

010 Overflow. A result (either floating-point or integer) of a floating-point instruction is too great in magnitude to be held in the format of the destination operand. Note that rounding, as well as calculations, can cause this condition.

011 Divide by zero. An attempt has been made to divide a non-zero floating-point number by zero. Dividing zero by zero is considered an Invalid Operation instead (below).

100 Illegal Instruction. Two undefined floating-point instruction forms are detected by the FPU as being illegal. The binary formats causing this trap are:

```
xxxxxxxxxx0011xx10111110
```

```
xxxxxxxxxx1001xx10111110
```

101 Invalid Operation. One of the floating-point operands of a floating-point instruction is a Reserved operand, or an attempt has been made to divide zero by zero using the DIVf instruction.

110 Inexact Result. The result (either floating-point or integer) of a floating-point instruction cannot be represented exactly in the format of the destination operand, and a rounding step must alter it to fit. This condition is always reported in the TT field and IF bit unless any other exceptional condition has occurred in the same instruction. In this case, the TT field always contains the code for the other exception and the IF bit is not altered. A trap is caused by this condition only if the IEN bit is set; otherwise the result is rounded and delivered, and no trap occurs.

111 (Reserved for future use.)

**Underflow Flag (UF):** Bit 4. This bit is set by the FPU whenever a result is too small in absolute value to be represented as a normalized number. Its function is not affected by the state of the UEN bit. The UF bit is cleared only by writing a zero into it with the Load FSR instruction or by a hardware reset.

**Inexact Result Flag (IF):** Bit 6. This bit is set by the FPU whenever the result of an operation must be rounded to fit within the destination format. The IF bit is set only if no other error has occurred. It is cleared only by writing a zero into it with the Load FSR instruction or by a hardware reset.

### 2.1.2.3 FSR Software Field (SWF)

Bits 9-15 of the FSR hold and display any information written to them (using the LFSR and SFSR instructions), but are not otherwise used by FPU hardware. They are reserved for use with NSC floating-point extension software.

## 2.2 INSTRUCTION SET

### 2.2.1 General Instruction Format

Figure 2-3 shows the general format of an Series 32000 instruction. The Basic Instruction is one to three bytes long

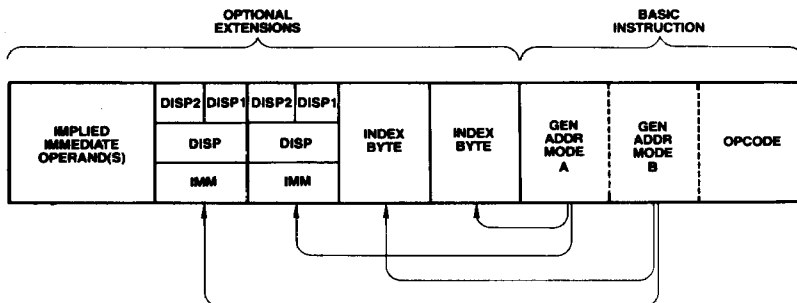


FIGURE 2-3. General Instruction Format

TL/EE/5234-6

## 2.0 Architectural Description (Continued)

and contains the opcode and up to two 5-bit General Addressing Mode (Gen) fields. Following the Basic Instruction field is a set of optional extensions, which may appear depending on the instruction and the addressing modes selected.

The only form of extension issued to the NS32081 FPU is an Immediate operand. Other extensions are used only by the CPU to reference memory operands needed by the FPU.

Index Bytes appear when either or both Gen fields specify Scaled Index. In this case, the Gen field specifies only the Scale Factor (1, 2, 4 or 8), and the Index Byte specifies which General Purpose Register to use as the index, and which addressing mode calculation to perform before indexing. See Figure 2-4.

Following Index Bytes come any displacements (addressing constants) or immediate values associated with the selected addressing modes. Each Disp/Imm field may contain one or two displacements, or one immediate value. The size of a Displacement field is encoded within the top bits of that field, as shown in Figure 2-5, with the remaining bits interpreted as a signed (two's complement) value. The size of an immediate value is determined from the Opcode field. Both Displacement and Immediate fields are stored most significant byte first.

Some non-FPU instructions require additional, "implied" immediates and/or displacements, apart from those associated with addressing modes. Any such extensions appear at the end of the instruction, in the order that they appear within the list of operands in the instruction definition.

### 2.2.2 Addressing Modes

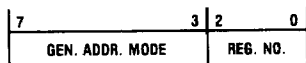
The Series 32000 Family CPUs generally access an operand by calculating its Effective Address based on information available when the operand is to be accessed. The method to be used in performing this calculation is specified by the programmer as an "addressing mode."

Addressing modes in the Series 32000 family are designed to optimally support high-level language accesses to variables. In nearly all cases, a variable access requires only one addressing mode within the instruction which acts upon that variable. Extraneous data movement is therefore minimized.

Series 32000 Addressing Modes fall into nine basic types:

**Register:** In floating-point instructions, these addressing modes refer to a Floating-Point Register (F0-F7) if the operand is of a floating-point type. Otherwise, a CPU General Purpose Register (R0-R7) is referenced. See Section 2.1.1.

**Register Relative:** A CPU General Purpose Register contains an address to which is added a displacement value from the instruction, yielding the Effective Address of the operand in memory.



TL/EE/5234-7

FIGURE 2-4. Index Byte Format

**Memory Space:** Identical to Register Relative above, except that the register used is one of the dedicated CPU registers PC, SP, SB or FP. These registers point to data areas generally needed by high-level languages.

**Memory Relative:** A pointer variable is found within the memory space pointed to by the CPU SP, SB or FP register. A displacement is added to that pointer to generate the Effective Address of the operand.

**Immediate:** The operand is encoded within the instruction. This addressing mode is not allowed if the operand is to be written. Floating-point operands as well as integer operands may be specified using Immediate mode.

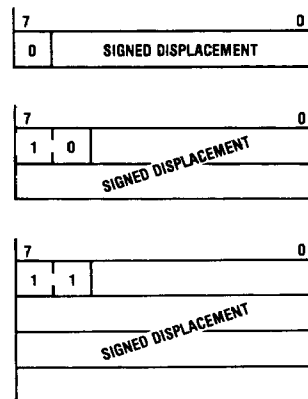
**Absolute:** The address of the operand is specified by a Displacement field in the instruction.

**External:** A pointer value is read from a specified entry of the current Link Table. To this pointer value is added a displacement, yielding the Effective Address of the operand.

**Top of Stack:** The currently-selected CPU Stack Pointer (SP0 or SP1) specifies the location of the operand. The operand is pushed or popped, depending on whether it is written or read.

**Scaled Index:** Although encoded as an addressing mode, Scaled Indexing is an option on any addressing mode except Immediate or another Scaled Index. It has the effect of calculating an Effective Address, then multiplying any General Purpose Register by 1, 2, 4 or 8 and adding it into the total, yielding the final Effective Address of the operand.

The following table, Table 2-1, is a brief summary of the addressing modes. For a complete description of their actions, see the Series 32000 Instruction Set Reference Manual.



TL/EE/5234-10

FIGURE 2-5. Displacement Encodings

## 2.0 Architectural Description (Continued)

TABLE 2-1. Series 32000 Family Addressing Modes

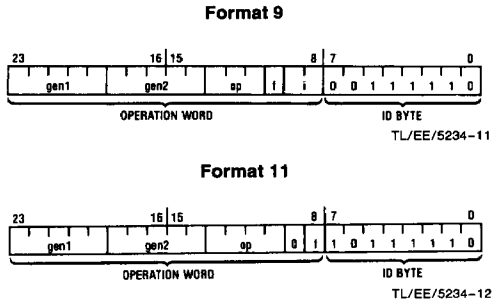
| Encoding                 | Mode                      | Assembler Syntax    | Effective Address                                                                                                    |
|--------------------------|---------------------------|---------------------|----------------------------------------------------------------------------------------------------------------------|
| <b>REGISTER</b>          |                           |                     |                                                                                                                      |
| 00000                    | Register 0                | R0 or F0            | None: Operand is in the specified register.                                                                          |
| 00001                    | Register 1                | R1 or F1            |                                                                                                                      |
| 00010                    | Register 2                | R2 or F2            |                                                                                                                      |
| 00011                    | Register 3                | R3 or F3            |                                                                                                                      |
| 00100                    | Register 4                | R4 or F4            |                                                                                                                      |
| 00101                    | Register 5                | R5 or F5            |                                                                                                                      |
| 00110                    | Register 6                | R6 or F6            |                                                                                                                      |
| 00111                    | Register 7                | R7 or F7            |                                                                                                                      |
| <b>REGISTER RELATIVE</b> |                           |                     |                                                                                                                      |
| 01000                    | Register 0 relative       | disp(R0)            | Disp + Register.                                                                                                     |
| 01001                    | Register 1 relative       | disp(R1)            |                                                                                                                      |
| 01010                    | Register 2 relative       | disp(R2)            |                                                                                                                      |
| 01011                    | Register 3 relative       | disp(R3)            |                                                                                                                      |
| 01100                    | Register 4 relative       | disp(R4)            |                                                                                                                      |
| 01101                    | Register 5 relative       | disp(R5)            |                                                                                                                      |
| 01110                    | Register 6 relative       | disp(R6)            |                                                                                                                      |
| 01111                    | Register 7 relative       | disp(R7)            |                                                                                                                      |
| <b>MEMORY SPACE</b>      |                           |                     |                                                                                                                      |
| 11000                    | Frame memory              | disp(FP)            | Disp + Register; "SP" is either SP0 or SP1, as selected in PSR.                                                      |
| 11001                    | Stack memory              | disp(SP)            |                                                                                                                      |
| 11010                    | Static memory             | disp(SB)            |                                                                                                                      |
| 11011                    | Program memory            | * + disp            |                                                                                                                      |
| <b>MEMORY RELATIVE</b>   |                           |                     |                                                                                                                      |
| 10000                    | Frame memory relative     | disp2(disp1(FP))    | Disp2 + Pointer; Pointer found at address Disp1 + Register. "SP" is either SP0 or SP1, as selected in PSR.           |
| 10001                    | Stack memory relative     | disp2(disp1(SP))    |                                                                                                                      |
| 10010                    | Static memory relative    | disp2(disp1(SB))    |                                                                                                                      |
| <b>IMMEDIATE</b>         |                           |                     |                                                                                                                      |
| 10100                    | Immediate                 | value               | None: Operand is issued from CPU instruction queue.                                                                  |
| <b>ABSOLUTE</b>          |                           |                     |                                                                                                                      |
| 10101                    | Absolute                  | @disp               | Disp.                                                                                                                |
| <b>EXTERNAL</b>          |                           |                     |                                                                                                                      |
| 10110                    | External                  | EXT (disp1) + disp2 | Disp2 + Pointer; Pointer is found at Link Table Entry number Disp1.                                                  |
| <b>TOP OF STACK</b>      |                           |                     |                                                                                                                      |
| 10111                    | Top of Stack              | TOS                 | Top of current stack, using either User or Interrupt Stack Pointer, as selected in PSR. Automatic Push/Pop included. |
| <b>SCALED INDEX</b>      |                           |                     |                                                                                                                      |
| 11100                    | Index, bytes              | mode[Rn:B]          | Mode + Rn.                                                                                                           |
| 11101                    | Index, words              | mode[Rn:W]          | Mode + 2 × Rn.                                                                                                       |
| 11110                    | Index, double words       | mode[Rn:D]          | Mode + 4 × Rn.                                                                                                       |
| 11111                    | Index, quad words         | mode[Rn:Q]          | Mode + 8 × Rn.                                                                                                       |
|                          |                           |                     | "Mode" and "n" are contained within the Index Byte.                                                                  |
| 10011                    | (Reserved for Future Use) |                     |                                                                                                                      |

## 2.0 Architectural Description (Continued)

### 2.2.3 Floating-Point Instruction Set

The NS32081 FPU instructions occupy formats 9 and 11 of the Series 32000 Family instruction set (*Figure 2-6*). A list of all Series 32000 family instruction formats is found in the applicable CPU data sheet.

Certain notations in the following instruction description tables serve to relate the assembly language form of each instruction to its binary format in *Figure 2-6*.



**FIGURE 2-6. Floating-Point Instruction Formats**

The Format column indicates which of the two formats in *Figure 2-6* represents each instruction.

The Op column indicates the binary pattern for the field called "op" in the applicable format.

The Instruction column gives the form of each instruction as it appears in assembly language. The form consists of an instruction mnemonic in upper case, with one or more suffixes (i or f) indicating data types, followed by a list of operands (gen1, gen2).

An i suffix on an instruction mnemonic indicates a choice of integer data types. This choice affects the binary pattern in the i field of the corresponding instruction format (*Figure 2-6*) as follows:

| Suffix i | Data Type   | i Field |
|----------|-------------|---------|
| B        | Byte        | 00      |
| W        | Word        | 01      |
| D        | Double Word | 11      |

An f suffix on an instruction mnemonic indicates a choice of floating-point data types. This choice affects the setting of the f bit of the corresponding instruction format (*Figure 2-6*) as follows:

| Suffix f | Data Type               | f Bit |
|----------|-------------------------|-------|
| F        | Single Precision        | 1     |
| L        | Double Precision (Long) | 0     |

An operand designation (gen1, gen2) indicates a choice of addressing mode expressions. This choice affects the binary pattern in the corresponding gen1 or gen2 field of the instruction format (*Figure 2-6*). Refer to Table 2-1 for the options available and their patterns.

Further details of the exact operations performed by each instruction are found in the Series 32000 Instruction Set Reference Manual.

### Movement and Conversion

The following instructions move the gen1 operand to the gen2 operand, leaving the gen1 operand intact.

| Format | Op   | Instruction        | Description                                                                                  |
|--------|------|--------------------|----------------------------------------------------------------------------------------------|
| 11     | 0001 | MOVf gen1, gen2    | Move without conversion                                                                      |
| 9      | 010  | MOVLF gen1, gen2   | Move, converting from double precision to single precision.                                  |
| 9      | 011  | MOVFL gen1, gen2   | Move, converting from single precision to double precision.                                  |
| 9      | 000  | MOVif gen1, gen2   | Move, converting from any integer type to any floating-point type.                           |
| 9      | 100  | ROUNDfi gen1, gen2 | Move, converting from floating-point to the nearest integer.                                 |
| 9      | 101  | TRUNCfi gen1, gen2 | Move, converting from floating-point to the nearest integer closer to zero.                  |
| 9      | 111  | FLOORfi gen1, gen2 | Move, converting from floating-point to the largest integer less than or equal to its value. |

**Note:** The MOVLF instruction f bit must be 1 and the i field must be 10.  
The MOVFL instruction f bit must be 0 and the i field must be 11.

### Arithmetic Operations

The following instructions perform floating-point arithmetic operations on the gen1 and gen2 operands, leaving the result in the gen2 operand.

| Format | Op   | Instruction     | Description                          |
|--------|------|-----------------|--------------------------------------|
| 11     | 0000 | ADDf gen1, gen2 | Add gen1 to gen2.                    |
| 11     | 0100 | SUBf gen1, gen2 | Subtract gen1 from gen2.             |
| 11     | 1100 | MULf gen1, gen2 | Multiply gen2 by gen1.               |
| 11     | 1000 | DIVf gen1, gen2 | Divide gen2 by gen1.                 |
| 11     | 0101 | NEGf gen1, gen2 | Move negative of gen1 to gen2.       |
| 11     | 1101 | ABSf gen1, gen2 | Move absolute value of gen1 to gen2. |

## 2.0 Architectural Description (Continued)

### Comparison

The Compare instruction compares two floating-point values, sending the result to the CPU PSR Z and N bits for use as condition codes. See Figure 3-7. The Z bit is set if the gen1 and gen2 operands are equal; it is cleared otherwise. The N bit is set if the gen1 operand is greater than the gen2 operand; it is cleared otherwise. The CPU PSR L bit is unconditionally cleared. Positive and negative zero are considered equal.

| Format | Op   | Instruction     | Description           |
|--------|------|-----------------|-----------------------|
| 11     | 0010 | CMPf gen1, gen2 | Compare gen1 to gen2. |

### Floating-Point Status Register Access

The following instructions load and store the FSR as a 32-bit integer.

| Format | Op  | Instruction | Description |
|--------|-----|-------------|-------------|
| 9      | 001 | LFSR gen1   | Load FSR    |
| 9      | 110 | SFSR gen2   | Store FSR   |

### 2.3 TRAPS

Upon detecting an exceptional condition in executing a floating-point instruction, the NS32081 FPU requests a trap by setting the Q bit of the status word transferred during the slave protocol (Section 3.5). The CPU responds by performing a trap using a default vector value of 3. See the Series 32000 Instruction Set Reference Manual and the applicable CPU data sheet for trap service details.

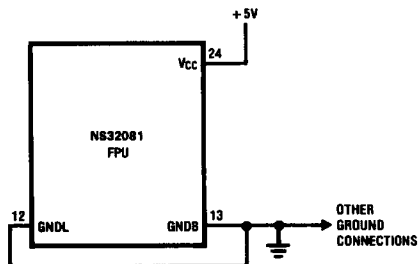
A trapped floating-point instruction returns no result, and does not affect the CPU Processor Status Register (PSR). The FPU displays the reason for the trap in the Trap Type (TT) field of the FSR (Section 2.1.2.2).

## 3.0 Functional Description

### 3.1 POWER AND GROUNDING

The NS32081 requires a single 5V power supply, applied on pin 24 ( $V_{CC}$ ). See DC Electrical Characteristics table.

Grounding connections are made on two pins. Logic Ground (GNDL, pin 12) is the common pin for on-chip logic, and Buffer Ground (GNDB, pin 13) is the common pin for the output drivers. For optimal noise immunity, it is recommended that GNDL be attached through a single conductor directly to GNDB, and that all other grounding connections be made only to GNDB, as shown below (Figure 3-1).



TL/EE/5234-13

FIGURE 3-1. Recommended Supply Connections

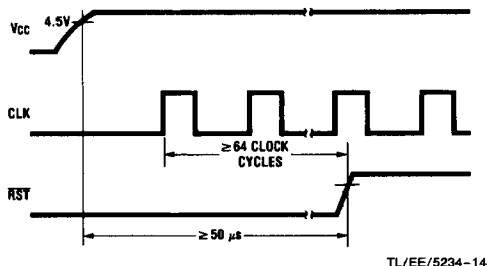
### 3.2 CLOCKING

The NS32081 FPU requires a single-phase TTL clock input on its CLK pin (pin 14). When the FPU is connected to a Series 32000 CPU, the CLK signal is provided from the CCTL pin of the NS32201 Timing Control Unit.

### 3.3 RESETTING

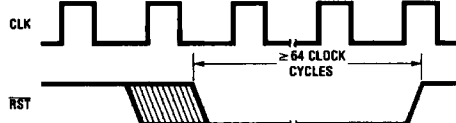
The RST pin serves as a reset for on-chip logic. The FPU may be reset at any time by pulling the RST pin low for at least 64 clock cycles. Upon detecting a reset, the FPU terminates instruction processing, resets its internal logic, and clears the FSR to all zeroes.

On application of power,  $\overline{RST}$  must be held low for at least 50  $\mu s$  after  $V_{CC}$  is stable. This ensures that all on-chip voltages are completely stable before operation. See Figures 3-2 and 3-3.



TL/EE/5234-14

FIGURE 3-2. Power-On Reset Requirements

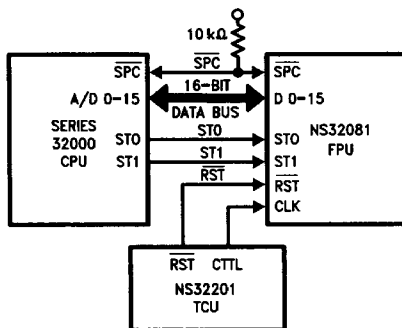


TL/EE/5234-15

FIGURE 3-3. General Reset Timing

### 3.4 BUS OPERATION

Instructions and operands are passed to the NS32081 FPU with slave processor bus cycles. Each bus cycle transfers either one byte (8 bits) or one word (16 bits) to or from the FPU. During all bus cycles, the SPC line is driven by the CPU as an active low data strobe, and the FPU monitors



TL/EE/5234-2

FIGURE 3-4. System Connection Diagram

### 3.0 Functional Description (Continued)

pins ST0 and ST1 to keep track of the sequence (protocol) established for the instruction being executed. This is necessary in a virtual memory environment, allowing the FPU to retry an aborted instruction.

#### 3.4.1 Bus Cycles

A bus cycle is initiated by the CPU, which asserts the proper status on ST0 and ST1 and pulses SPC low. ST0 and ST1 are sampled by the FPU on the leading (falling) edge of the SPC pulse. If the transfer is from the FPU (a slave processor read cycle), the FPU asserts data on the data bus for the duration of the SPC pulse. If the transfer is to the FPU (a slave processor write cycle), the FPU latches data from the data bus on the trailing (rising) edge of the SPC pulse. *Figures 3-5 and 3-6* illustrate these sequences.

The direction of the transfer and the role of the bidirectional SPC line are determined by the instruction protocol being performed. SPC is always driven by the CPU during slave processor bus cycles. Protocol sequences for each instruction are given in Section 3.5.

#### 3.4.2 Operand Transfer Sequences

An operand is transferred in one or more bus cycles. A 1-byte operand is transferred on the least significant byte of the data bus (D0-D7). A 2-byte operand is transferred on the entire bus. A 4-byte or 8-byte operand is transferred in consecutive bus cycles, least significant word first.

### 3.5 INSTRUCTION PROTOCOLS

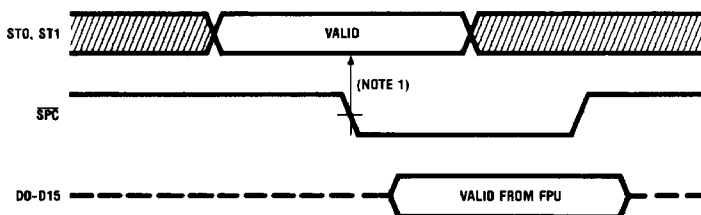
#### 3.5.1 General Protocol Sequence

Slave Processor instructions have a three-byte Basic Instruction field, consisting of an ID byte followed by an Operation Word. See Section 2.2.3 for FPU instruction encodings. The ID Byte has three functions:

- 1) It identifies the instruction to the CPU as being a Slave Processor instruction.
- 2) It specifies which Slave Processor will execute it.
- 3) It determines the format of the following Operation Word of the instruction.

Upon receiving a Slave Processor instruction, the CPU initiates the sequence outlined in Table 3-2. While applying Status Code 11 (Broadcast ID, Table 3-1), the CPU transfers the ID Byte on the least significant half of the Data Bus (D0-D7). All Slave Processors input this byte and decode it. The Slave Processor selected by the ID Byte is activated, and from this point the CPU is communicating only with it. If any other slave protocol was in progress (e.g., an aborted Slave instruction), this transfer cancels it.

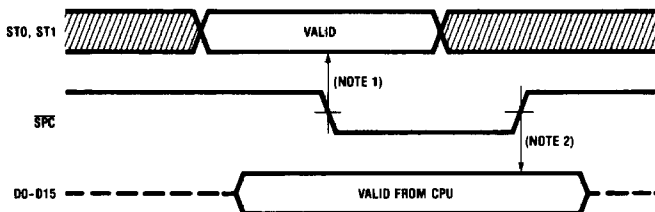
The CPU next sends the Operation Word while applying Status Code 01 (Transfer Slave Operand, Table 3-1). Upon receiving it, the FPU decodes it, and at this point both the CPU and the FPU are aware of the number of operands to be transferred and their sizes. The Operation Word is swapped on the Data Bus; that is, bits 0-7 appear on pins D8-D15, and bits 8-15 appear on pins D0-D7.



Note 1: FPU samples CPU status here.

TL/EE/5234-16

FIGURE 3-5. Slave Processor Read Cycle



Note 1: FPU samples CPU status here.

Note 2: FPU samples data bus here.

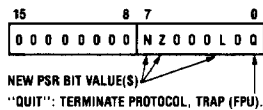
TL/EE/5234-17

FIGURE 3-6. Slave Processor Write Cycle

### 3.0 Functional Description (Continued)

Using the Addressing Mode fields within the Operation Word, the CPU starts fetching operands and issuing them to the FPU. To do so, it references any Addressing Mode extensions appended to the FPU instruction. Since the CPU is solely responsible for memory accesses, these extensions are not sent to the Slave Processor. The Status Code applied is 01 (Transfer Slave Processor Operand, Table 3-1). After the CPU has issued the last operand, the FPU starts the actual execution of the instruction. Upon completion, it will signal the CPU by pulsing  $\overline{SPC}$  low. To allow for this, the CPU releases the SPC signal, causing it to float. SPC must be held high by an external pull-up resistor.

Upon receiving the pulse on  $\overline{SPC}$ , the CPU uses  $\overline{SPC}$  to read a Status Word from the FPU, applying Status Code 10. This word has the format shown in Figure 3-7. If the Q bit ("Quit", Bit 0) is set, this indicates that an error has been detected by the FPU. The CPU will not continue the protocol, but will immediately trap through the Slave vector in the Interrupt Table. If the instruction being performed is CMPf (Section 2.2.3) and the Q bit is not set, the CPU loads Processor Status Register (PSR) bits N, Z and L from the corresponding bits in the Status Word. The NS32081 FPU always sets the L bit to zero.



TL/EE/5234-18

FIGURE 3-7. FPU Protocol Status Word Format

The last step in the protocol is for the CPU to read a result, if any, and transfer it to the destination. The Read cycles from the FPU are performed by the CPU while applying Status Code 01 (Section 4.1.2).

TABLE 3-1. General Instruction Protocol

| Step | Status | Action                           |
|------|--------|----------------------------------|
| 1    | 11     | CPU sends ID Byte.               |
| 2    | 01     | CPU sends Operation Word.        |
| 3    | 01     | CPU sends required operands.     |
| 4    | XX     | FPU starts execution.            |
| 5    | XX     | FPU pulses $\overline{SPC}$ low. |
| 6    | 10     | CPU reads Status Word.           |
| 7    | 01     | CPU reads result (if any).       |

#### 3.5.2 Floating-Point Protocols

Table 3-2 gives the protocols followed for each floating-point instruction. The instructions are referenced by their mnemonics. For the bit encodings of each instruction, see Section 2.2.3.

The Operand Class columns give the Access Classes for each general operand, defining how the addressing modes are interpreted by the CPU (see Series 32000 Instruction Set Reference Manual).

The Operand Issued columns show the sizes of the operands issued to the Floating-Point Unit by the CPU. "D" indicates a 32-bit Double Word. "I" indicates that the instruction specifies an integer size for the operand (B = Byte, W = Word, D = Double Word). "F" indicates that the instruction specifies a floating-point size for the operand (F = 32-bit Standard Floating, L = 64-bit Long Floating).

The Returned Value Type and Destination column gives the size of any returned value and where the CPU places it. The PSR Bits Affected column indicates which PSR bits, if any, are updated from the Slave Processor Status Word (Figure 3-7).

Any operand indicated as being of type "F" will not cause a transfer if the Register addressing mode is specified, because the Floating-Point Registers are physically on the Floating-Point Unit and are therefore available without CPU assistance.

TABLE 3-2. Floating Point Instruction Protocols

| Mnemonic | Operand 1<br>Class | Operand 2<br>Class | Operand 1<br>Issued | Operand 2<br>Issued | Returned Value<br>Type and Dest. | PSR Bits<br>Affected |
|----------|--------------------|--------------------|---------------------|---------------------|----------------------------------|----------------------|
| ADDf     | read.f             | rmw.f              | f                   | f                   | f to Op. 2                       | none                 |
| SUBf     | read.f             | rmw.f              | f                   | f                   | f to Op. 2                       | none                 |
| MULf     | read.f             | rmw.f              | f                   | f                   | f to Op. 2                       | none                 |
| DIVf     | read.f             | rmw.f              | f                   | f                   | f to Op. 2                       | none                 |
| MOVf     | read.f             | write.f            | f                   | N/A                 | f to Op. 2                       | none                 |
| ABSF     | read.f             | write.f            | f                   | N/A                 | f to Op. 2                       | none                 |
| NEGF     | read.f             | write.f            | f                   | N/A                 | f to Op. 2                       | none                 |
| CMPf     | read.f             | read.f             | f                   | f                   | N/A                              | N,Z,L                |
| FLOORfi  | read.f             | write.i            | f                   | N/A                 | i to Op. 2                       | none                 |
| TRUNCfi  | read.f             | write.i            | f                   | N/A                 | i to Op. 2                       | none                 |
| ROUNDfi  | read.f             | write.i            | f                   | N/A                 | i to Op. 2                       | none                 |
| MOVFL    | read.F             | write.L            | F                   | N/A                 | L to Op. 2                       | none                 |
| MOVLF    | read.L             | write.F            | L                   | N/A                 | F to Op. 2                       | none                 |
| MOVif    | read.i             | write.f            | i                   | N/A                 | f to Op. 2                       | none                 |
| LFSR     | read.D             | N/A                | D                   | N/A                 | N/A                              | none                 |
| SFSR     | N/A                | write.D            | N/A                 | N/A                 | D to Op. 2                       | none                 |

D = Double Word

i = Integer size (B, W, D) specified in mnemonic.

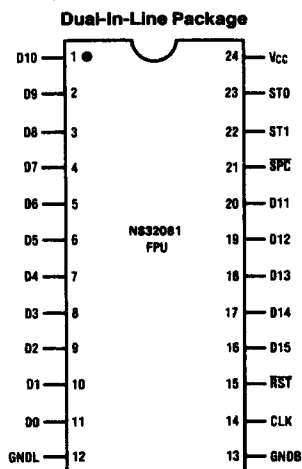
F = Floating-Point type (F, L) specified in mnemonic.

N/A = Not Applicable to this instruction.

## 4.0 Device Specifications

### 4.1 PIN DESCRIPTIONS

The following are brief descriptions of all NS32081 FPU pins. The descriptions reference the relevant portions of the Functional Description, Section 3.



TL/EE/5234-3

Top View  
FIGURE 4-1. Connection Diagram

Order Number NS32081D-10 or NS32081D-15  
See NS Package Number D24C

Order Number NS32081N-10 or NS32081N-15  
See NS Package Number N24A

### 4.2 ABSOLUTE MAXIMUM RATINGS

|                                                     |                 |
|-----------------------------------------------------|-----------------|
| Temperature Under Bias                              | 0°C to +70°C    |
| Storage Temperature                                 | -65°C to +150°C |
| All Input or Output Voltages<br>with Respect to GND | -0.5V to +7.0V  |
| Power Dissipation                                   | 1.5W            |

### 4.1.1 Supplies

**Power (V<sub>CC</sub>):** +5V positive supply. Section 3.1.

**Logic Ground (GNDL):** Ground reference for on-chip logic. Section 3.1.

**Buffer Ground (GNDB):** Ground reference for on-chip drivers connected to output pins. Section 3.1.

### 4.1.2 Input Signals

**Clock (CLK):** TTL-level clock signal.

**Reset (RST):** Active low. Initiates a Reset, Section 3.3.

**Status (ST0, ST1):** Input from CPU. ST0 is the least significant bit. Section 3.4 encodings are:

- 00—(Reserved)
- 01—Transferring Operation Word or Operand
- 10—Reading Status Word
- 11—Broadcasting Slave ID

### 4.1.3 Input/Output Signals

**Slave Processor Control (SPC):** Active low. Driven by the CPU as the data strobe for bus transfers to and from the NS32081 FPU, Section 3.4. Driven by the FPU to signal completion of an operation, Section 3.5.1. Must be held high with an external pull-up resistor while floating.

**Data Bus (D0–D15):** 16-bit bus for data transfer. D0 is the least significant bit. Section 3.4.

**If Military/Aerospace specified devices are required, please contact the National Semiconductor Sales Office/Distributors for availability and specifications.**

*Note: Absolute maximum ratings indicate limits beyond which permanent damage may occur. Continuous operation at these limits is not intended; operation should be limited to those conditions specified under Electrical Characteristics.*

### 4.3 ELECTRICAL CHARACTERISTICS $T_A = 0^\circ\text{C to } 70^\circ\text{C}$ , $V_{CC} = 5V \pm 5\%$ , $GND = 0V$

| Symbol   | Parameter                                                         | Conditions                               | Min   | Typ | Max            | Units         |
|----------|-------------------------------------------------------------------|------------------------------------------|-------|-----|----------------|---------------|
| $V_{IH}$ | HIGH Level Input Voltage                                          |                                          | 2.0   |     | $V_{CC} + 0.5$ | V             |
| $V_{IL}$ | LOW Level Input Voltage                                           |                                          | -0.5  |     | 0.8            | V             |
| $V_{OH}$ | HIGH Level Output Voltage                                         | $I_{OH} = -400 \mu\text{A}$              | 2.4   |     |                | V             |
| $V_{OL}$ | LOW Level Output Voltage                                          | $I_{OL} = 4 \text{ mA}$                  |       |     | 0.45           | V             |
| $I_I$    | Input Load Current                                                | $0 \leq V_{IN} \leq V_{CC}$              | -10.0 |     | 10.0           | $\mu\text{A}$ |
| $I_L$    | Leakage Current<br>Output and I/O Pins in<br>TRI-STATE/Input Mode | $0.45 \leq V_{IN} \leq 2.4V$             | -20.0 |     | 20.0           | $\mu\text{A}$ |
| $I_{CC}$ | Active Supply Current                                             | $I_{OUT} = 0$ , $T_A = 25^\circ\text{C}$ |       | 200 | 300            | mA            |

# 4.0 Device Specifications (Continued)

## 4.4 SWITCHING CHARACTERISTICS

### 4.4.1 Definitions

All the Timing Specifications given in this section refer to 0.8V and 2.0V on all the input and output signals as illustrated in *Figures 4.2 and 4.3*, unless specifically stated otherwise.

### ABBREVIATIONS

L.E. — Leading Edge  
T.E. — Trailing Edge  
R.E. — Rising Edge  
F.E. — Falling Edge

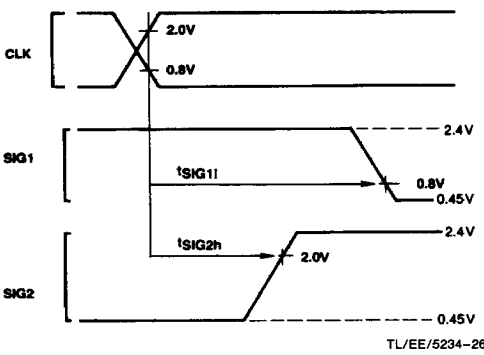


FIGURE 4-2. Timing Specification Standard  
(Signal Valid After Clock Edge)

TL/EE/5234-26

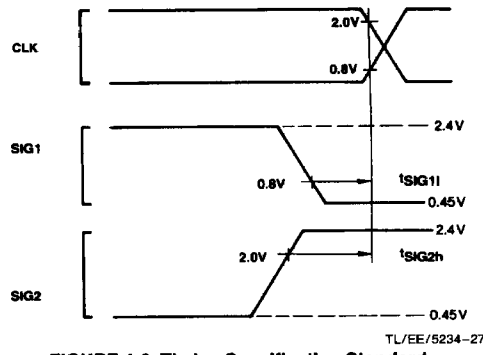


FIGURE 4-3. Timing Specification Standard  
(Signal Valid Before Clock Edge)

TL/EE/5234-27

## 4.0 Device Specifications (Continued)

### 4.4.2 Timing Tables

#### 4.4.2.1 Output Signal Propagation Delays

Maximum times assume capacitive loading of 100 pF.

| Name                | Figure | Description                              | Reference/<br>Conditions    | NS32081-10             |                        | NS32081-15             |                        | Units |
|---------------------|--------|------------------------------------------|-----------------------------|------------------------|------------------------|------------------------|------------------------|-------|
|                     |        |                                          |                             | Min                    | Max                    | Min                    | Max                    |       |
| t <sub>Dv</sub>     | 4-7    | Data Valid                               | After $\overline{SPC}$ L.E. |                        | 45                     |                        | 30                     | ns    |
| t <sub>Df</sub>     | 4-7    | D <sub>0</sub> –D <sub>15</sub> Floating | After $\overline{SPC}$ T.E. |                        | 50                     | 2                      | 35                     | ns    |
| t <sub>SPCFw</sub>  | 4-9    | $\overline{SPC}$ Pulse Width from FPU    | At 0.8V (Both Edges)        | t <sub>CLKp</sub> – 50 | t <sub>CLKp</sub> + 50 | t <sub>CLKp</sub> – 40 | t <sub>CLKp</sub> + 40 | ns    |
| t <sub>SPCFi</sub>  | 4-9    | $\overline{SPC}$ Output Active           | After CLK R.E.              |                        | 55                     |                        | 38                     | ns    |
| t <sub>SPCFh</sub>  | 4-9    | $\overline{SPC}$ Output Inactive         | After CLK R.E.              |                        | 55                     |                        | 38                     | ns    |
| t <sub>SPCFnf</sub> | 4-9    | $\overline{SPC}$ Output Nonforcing       | After CLK F.E.              |                        | 45                     |                        | 35                     | ns    |

#### 4.4.2.2 Input Signal Requirements

| Name              | Figure | Description                                | Reference/<br>Conditions           | Min | Max | Min | Max | Units             |
|-------------------|--------|--------------------------------------------|------------------------------------|-----|-----|-----|-----|-------------------|
| t <sub>PWR</sub>  | 4-5    | Power Stable to RST R.E.                   | After V <sub>CC</sub> Reaches 4.5V | 50  |     | 50  |     | μs                |
| t <sub>RSTw</sub> | 4-6    | RST Pulse Width                            | At 0.8V (Both Edges)               | 64  |     | 64  |     | t <sub>CLKp</sub> |
| t <sub>Ss</sub>   | 4-7    | Status (ST0–ST1) Setup                     | Before $\overline{SPC}$ L.E.       | 50  |     | 33  |     | ns                |
| t <sub>Sh</sub>   | 4-7    | Status (ST0–ST1) Hold                      | After $\overline{SPC}$ L.E.        | 40  |     | 35  |     | ns                |
| t <sub>Ds</sub>   | 4-8    | D <sub>0</sub> –D <sub>15</sub> Setup Time | Before $\overline{SPC}$ T.E.       | 40  |     | 30  |     | ns                |
| t <sub>Dh</sub>   | 4-8    | D <sub>0</sub> –D <sub>15</sub> Hold Time  | After $\overline{SPC}$ T.E.        | 50  |     | 35  |     | ns                |
| t <sub>SPCw</sub> | 4-7    | SPC Pulse Width from CPU                   | At 0.8V (Both Edges)               | 70  |     | 50  |     | ns                |
| t <sub>SPCs</sub> | 4-7    | $\overline{SPC}$ Input Active              | Before CLK R.E.                    | 40  |     | 35  |     | ns                |
| t <sub>SPCh</sub> | 4-7    | $\overline{SPC}$ Input Inactive            | After CLK R.E.                     | 0   |     | 0   |     | ns                |
| t <sub>RSTs</sub> | 4-10   | RST Setup                                  | Before CLK F.E.                    | 10  |     | 10  |     | ns                |
| t <sub>RSTh</sub> | 4-10   | RST R.E. Delay                             | After CLK R.E.                     | 0   |     | 0   |     | ns                |

#### 4.4.2.3 Clocking Requirements

| Name              | Figure | Description     | Reference/<br>Conditions  | Min | Max  | Min | Max  | Units |
|-------------------|--------|-----------------|---------------------------|-----|------|-----|------|-------|
| t <sub>CLKh</sub> | 4-4    | Clock High Time | At 2.0V (Both Edges)      | 42  | 1000 | 27  | 1000 | ns    |
| t <sub>CLKl</sub> | 4-4    | Clock Low Time  | At 0.8V (Both Edges)      | 42  | 1000 | 27  | 1000 | ns    |
| t <sub>CLKp</sub> | 4-4    | Clock Period    | CLK R.E. to Next CLK R.E. | 100 | 2000 | 66  |      | ns    |

## 4.0 Device Specifications (Continued)

### 4.4.3 Timing Diagrams

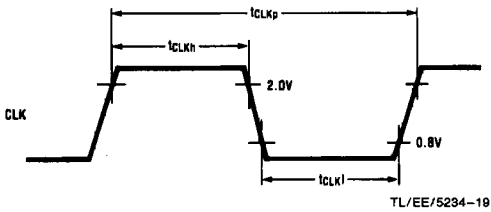


FIGURE 4-4. Clock Timing

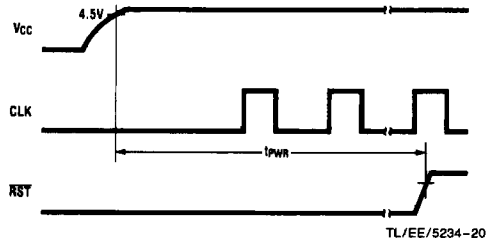


FIGURE 4-5. Power-On Reset

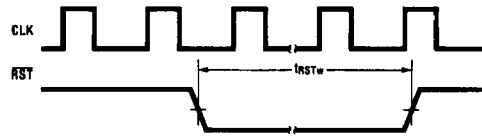


FIGURE 4-6. Non-Power-On Reset

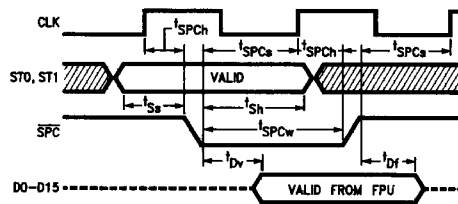


FIGURE 4-7. Read Cycle from FPU

Note: SPC pulse must be (nominally) 1 clock wide when writing into FPU.

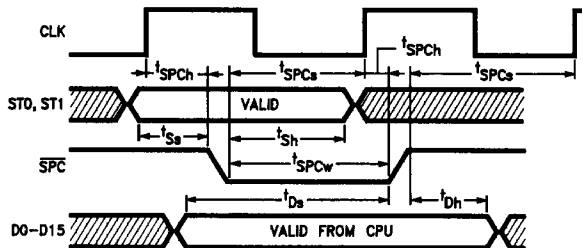


FIGURE 4-8. Write Cycle to FPU

Note: SPC pulse may also be 2 clocks wide, but its edges must meet the  $t_{SPCa}$  and  $t_{SPCh}$  requirements with respect to CLK.

## 4.0 Device Specifications (Continued)

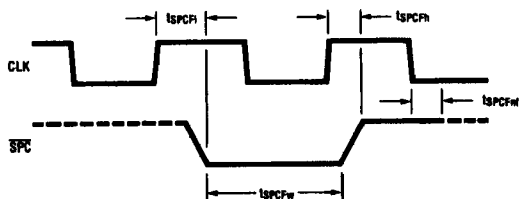


FIGURE 4-9. SPC Pulse from FPU

TL/EE/5234-24

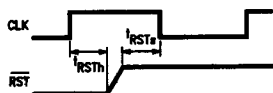


FIGURE 4-10.  $\overline{RST}$  Release Timing

TL/EE/5234-25

Note: The rising edge of  $\overline{RST}$  must occur while CLK is high, as shown.