Appendix A

Intrinsic Functions

This appendix contains the following subsections:

- "Generic and Specific Names"
- "Operational Conventions and Restrictions"
- "List of Functions"

This appendix describes the intrinsic functions provided with Fortran. Fortran intrinsic functions are identified by two categories of names: specific and generic. An **IMPLICIT** statement does not change the data type of an intrinsic function.

Generic and Specific Names

A *generic name* is the name given to a class of objects. Intrinsic functions that perform the same mathematical function, such as square root, are given a single name. For example, the generic name of the square root function is **SQRT**; this function has four specific names for different data types: **SQRT**, **DSQRT**, **CSQRT**, and **ZSRT** (see Table A−1). However, you can use the generic name **SQRT** regardless of the data type of the arguments.

An intrinsic function preceded by the letters **CD** is equivalent to the generic function with the same base name, except that the arguments must be of type **DOUBLE COMPLEX**.

Intrinsic functions starting with **II** are equivalent to generic functions with the same base name, except that the arguments must of type **INTEGER*2**. Similarly, arguments to intrinsic functions starting with **JI** must be type **INTEGER*4**: for example, **IIAND**, **IIQINT**, **IIQNNT**, **JIQINT**, **JIQNNT**.

When a generic name is referenced, the processor substitutes a function call for a specific name, depending on the data type of the arguments. In this way, the same name can be used for different types of arguments.

When an intrinsic function is to be used as the actual argument to another function, you must always use the specific name, never the generic name.

If a generic name is referenced, the type of the result is the same as the type of the argument, except for functions performing type conversion, nearest integer, and absolute value with a complex argument. Some intrinsic functions allow more than one argument, in which case all the arguments must be of the same type so that the function can decide which specific name function it should use.

If the specific name or generic name appears as the dummy argument of a function or subroutine, that symbolic name cannot identify an intrinsic function in that program unit.

A name in an **INTRINSIC** statement must be the specific name or generic name of an intrinsic function, as given in Table A−1

Referencing an Intrinsic Function

Reference an intrinsic function in the form

fun (*a*[,*a*]...)

where

fun is the generic or specific name of the intrinsic function.

a is an actual argument.

The actual arguments (*a*) constitute the argument list and must agree in order, number, and type with the specification described in this appendix and with each other. Each argument can be any expression. The expression cannot contain an concatenation in which one or more of the operand lengths are specified with an asterisk.

A function reference can be used as a primary in an expression. The following example involves referencing an intrinsic function:

```
X = SQRT(B**2-4*A*C)
```

The result of a function becomes undefined when its arguments are not mathematically defined or exceed the numeric range of the processor.

Operational Conventions and Restrictions

For most intrinsic functions, the data type of the result of the intrinsic function is the same as the arguments. If two or more arguments are required or permitted, then all arguments must be of the same type. An **IMPLICIT** statement does not change the data type of a specific or generic name of an intrinsic function.

If an intrinsic function name is used as an actual argument in an external procedure reference, the name must be one of the specific names and must appear in an **INTRINSIC** statement. However, names of intrinsic functions for type conversion, for lexical relationship, and for choosing the smallest or largest value cannot be used as actual arguments.

List of Functions

Table A−1lists the available intrinsic functions. Operational conventions and restrictions (other than those already given) are listed at the end of the table.

Note: REAL*16 intrinsic functions are not supported. The compiler issues a warning message when the name of a **REAL*16** intrinsic function is encountered; the equivalent double precision (**REAL*8**) function is used instead.

Table A−1 Intrinsic Functions

Function	Number of Arguments	Generic Name	Specific Name	Type of Argument	Type of Result
Conversion to INTEGER	1	INT[a]		INTEGER*1	INTEGER*2
				INTEGER*1	INTEGER*4
				INTEGER*1	INTEGER*8
				INTEGER*2	INTEGER*4

				INTEGER*2	INTEGER*8
				INTEGER*4	INTEGER*4
				INTEGER*4	INTEGER*8
				INTEGER*8	INTEGER*8
			IINT	REAL*4	INTEGER*2
			JINT	REAL*4	INTEGER*4
			KINT	REAL*4	INTEGER*8
			IIDINT	REAL*8	INTEGER*2
			JIDINT	REAL*8	INTEGER*4
			KIDINT	REAL*8	INTEGER*8
				COMPLEX*8	INTEGER*2
				COMPLEX*8	INTEGER*4
				COMPLEX*8	INTEGER*8
				COMPLEX*16	INTEGER*2
				COMPLEX*16	INTEGER*4
				COMPLEX*16	INTEGER*8
		SHORT		INTEGER*1	INTEGER*2
				INTEGER*2	INTEGER*2
				INTEGER*4	INTEGER*2
				REAL*4	INTEGER*2
				REAL*8	INTEGER*2
				COMPLEX*8	INTEGER*2
				COMPLEX*16	INTEGER*2
		LONG		INTEGER*1	INTEGER*4
				INTEGER*2	INTEGER*4
				INTEGER*4	INTEGER*4
				REAL*4	INTEGER*4
				REAL*8	INTEGER*4
				COMPLEX*8	INTEGER*4
				COMPLEX*16	INTEGER*4
	1	IFIX	IIFIX	REAL*4	INTEGER*2
			JIFIX	REAL*4	INTEGER*4
			KIFIX	REAL*4	INTEGER*8
		IDINT	IIDINT	REAL*8	INTEGER*2
			JIDINT	REAL*8	INTEGER*4
			KIDINT	REAL*8	INTEGER*8
Truncation		AINT	AINT	REAL*4	REAL*4
			DINT	REAL*8	REAL*8
Conversion to REAL	1	REAL		INTEGER*1	REAL*4
			FLOATI	INTEGER*2	REAL*4
			FLOATJ	INTEGER*4	REAL*4
			FLOATK	INTEGER*8	REAL*4
				REAL*4	REAL*4
			SNGL	REAL*8	REAL*4
				COMPLEX*8	REAL*4
				COMPLEX*16	REAL*4
	1	FLOAT		INTEGER*1	REAL*4
			FLOATI	INTEGER*2	REAL*4
			FLOATJ	INTEGER*4	REAL*4
			FLOATK	INTEGER*8	REAL*4
	1	SNGL		INTEGER*1	REAL*4
			FLOATI	INTEGER*2	REAL*4

			FLOATJ	INTEGER*4	REAL*4
			FLOATK	INTEGER*8	REAL*4
			REAL	REAL*4	REAL*4
				REAL*8	REAL*4
Conversion to DOUBLE PRECISION	1	DBLE		INTEGER*1	REAL*8
				INTEGER*2	REAL*8
				INTEGER*4	REAL*8
				INTEGER*8	REAL*8
			DBLE	REAL*4	REAL*8
				REAL*8	REAL*8
				COMPLEX*8	REAL*8
				COMPLEX*16	REAK*8
		DFLOAT		INTEGER*1	REAL*8
			DFLOTI	INTEGER*2	REAL*8
			DFLOTJ	INTEGER*4	REAL*8
			DFLOTK	INTEGER*8	REAL*8
			DFLOATK	INTEGER*8	REAL*8
Conversion to COMPLEX	1, 2	CMPLX		INTEGER*1	COMPLEX*8
	1,2			INTEGER*2	COMPLEX*8
	1,2			INTEGER*4	COMPLEX*8
	1,2			INTEGER*8	COMPLEX*8
	1,2			REAL*4	COMPLEX*8
	1,2			REAL*8	COMPLEX*8
	1			COMPLEX*8	COMPLEX*8
	1			COMPLEX*16	COMPLEX*8
Complex Conjugate	1	CONJG	CONJG	COMPLEX*8	COMPLEX*8
			DCONJG	COMPLEX*8	COMPLEX*16
Conversion to DOUBLE COMPLEX	1, 2	DCMPLX		INTEGER*1	COMPLEX*16
	1,2			INTEGER*2	COMPLEX*16
	1,2			INTEGER*4	COMPLEX*16
	1,2			INTEGER*8	COMPLEX*16
	1,2			REAL*4	COMPLEX*16
	1,2			REAL*8	COMPLEX*16
	1			COMPLEX*8	COMPLEX*16
	1			COMPLEX*16	COMPLEX*16
Conversion to character	1		CHAR	LOGICAL*1	CHARACTER
				INTEGER*1	CHARACTER
				INTEGER*2	CHARACTER
				INTEGER*4	CHARACTER
				INTEGER*8	CHARACTER
Maximum value	2 or more	MAX		INTEGER*1	INTEGER*1
			IMAX0	INTEGER*2	INTEGER*2
			JMAX0	INTEGER*4	INTEGER*4
			KMAX0	INTEGER*8	INTEGER*8
			AMAX1	REAL*4	REAL*4
			DMAX1	REAL*8	REAL*8
		MAX0		INTEGER*1	INTEGER*1
			IMAX0	INTEGER*2	INTEGER*2
			JMAX0	INTEGER*4	INTEGER*4
			KMAX0	INTEGER*8	INTEGER*8
		MAX1	IMAX1	REAL*4	INTEGER*2
			JMAX1	REAL*4	INTEGER*4

			KMAX1	REAL*4	INTEGER*8
		AMAX0		INTEGER*1	REAL*4
			AIMAX0	INTEGER*2	REAL*4
			AJMAX0	INTEGER*4	REAL*4
			AKMAX0	INTEGER*8	REAL*4
Minimum value	2 or more	MIN		INTEGER*1	INTEGER*1
			IMIN0	INTEGER*2	INTEGER*2
			JMIN0	INTEGER*4	INTEGER*4
			KMIN0	INTEGER*8	INTEGER*8
			AMIN1	REAL*4	REAL*4
			DMIN1	REAL*8	REAL*8
		MIN0		INTEGER*1	INTEGER*1
			IMIN0	INTEGER*2	INTEGER*2
			JMIN0	INTEGER*4	INTEGER*4
			KMIN0	INTEGER*8	INTEGER*8
		MIN1	IMIN1	REAL*4	INTEGER*2
			JMIN1	REAL*4	INTEGER*4
			KMIN1	REAL*4	INTEGER*8
		AMIN0		INTEGER*1	REAL*4
			AIMIN0	INTEGER*2	REAL*4
			AJMIN0	INTEGER*4	REAL*4
			AKMIN0	INTEGER*8	REAL*4
Nearest integer	1	NINT[b]	ININT	REAL*4	INTEGER*2
			JNINT	REAL*4	INTEGER*4
			KNINT	REAL*4	INTEGER*8
			IIDNNT	REAL*8	INTEGER*2
			JIDNNT	REAL*8	INTEGER*4
			KIDNNT	REAL*8	INTEGER*8
		ANINT	ANINT	REAL*4	REAL*4
			DNINT	REAL*8	REAL*8
		IDNINT	IIDNNT	REAL*8	INTEGER*2
			JIDNNT	REAL*8	INTEGER*4
			KIDNNT	REAL*8	INTEGER*8
Zero−Extend functions	1	ZEXT	IZEXT	LOGICAL*1	INTEGER*2
				LOGICAL*2	INTEGER*2
				INTEGER*1	INTEGER*2
				INTEGER*2	INTEGER*2
			JZEXT	LOGICAL*1	INTEGER*4
				LOGICAL*2	INTEGER*4
				LOGICAL*4	INTEGER*4
				INTEGER*1	INTEGER*4
				INTEGER*2	INTEGER*4
				INTEGER*4	INTEGER*4
			KZEXT	LOGICAL*1	INTEGER*8
				LOGICAL*2	INTEGER*8
				LOGICAL*4	INTEGER*8
				LOGICAL*8	INTEGER*8
				INTEGER*1	INTEGER*8
				INTEGER*2	INTEGER*8
				INTEGER*4	INTEGER*8
				INTEGER*8	INTEGER*8

Absolute value	1	ABS		INTEGER*1	INTEGER*1
			IIABS	INTEGER*2	INTEGER*2
			JIABS	INTEGER*4	INTEGER*4
			KIABS	INTEGER*8	INTEGER*8
			ABS	REAL*4	REAL*4
			DABS	REAL*8	REAL*8
			CABS	COMPLEX*8	COMPLEX*8
			CDABS	COMPLEX*16	COMPLEX*16
			ZABS	COMPLEX*16	COMPLEX*16
		IABS[c]		INTEGER*1	INTEGER*1
			IIABS	INTEGER*2	INTEGER*2
			JIABS	INTEGER*4	INTEGER*4
			KIABS	INTEGER*8	INTEGER*8
Remaindering	2	MOD[d]		INTEGER*1	INTEGER*1
			IMOD	INTEGER*2	INTEGER*2
			JMOD	INTEGER*4	INTEGER*4
			KMOD	INTEGER*8	INTEGER*8
			AMOD	REAL*4	REAL*4
			DMOD	REAL*8	REAL*8
Transfer of sign	2	SIGN		INTEGER*1	INTEGER*1
			IISIGN	INTEGER*2	INTEGER*2
			JISIGN	INTEGER*4	INTEGER*4
			KISIGN	INTEGER*8	INTEGER*8
			SIGN	REAL*4	REAL*4
			DSIGN	REAL*8	REAL*8
		ISIGN[e]		INTEGER*1	INTEGER*1
			IISIGN	INTEGER*2	INTEGER*2
			JISIGN	INTEGER*4	INTEGER*4
			KISIGN	INTEGER*8	INTEGER*8
Positive difference	2	DIM		INTEGER*1	INTEGER*1
			IIDIM	INTEGER*2	INTEGER*2
			JIDIM	INTEGER*4	INTEGER*4
			KIDIM	INTEGER*8	INTEGER*8
			DIM	REAL*4	REAL*4
			DDIM	REAL*8	REAL*8
		IDIM		INTEGER*1	INTEGER*1
			IIDIM	INTEGER*2	INTEGER*2
			JIDIM	INTEGER*4	INTEGER*4
			KIDIM	INTEGER*8	INTEGER*8
DOUBLE PRECISION product of REALs	2		DPROD	REAL*4	REAL*4
Length of character entry	1		LEN	CHARACTER	INTEGER*4
Index of a substring	2		INDEX[f]	CHARACTER	INTEGER*4
Character (ASCII value of 1−byte character argument)	1		ICHAR	CHARACTER	INTEGER*2
				CHARACTER	INTEGER*4
				CHARACTER	INTEGER*8
Logically greater than or equal	2		LGE	CHARACTER	LOGICAL*4

Logically greater than	2		LGT	CHARACTER	LOGICAL*4
Logically less than or equal	2		LLE	CHARACTER	LOGICAL*4
Logically less than	2		LLT[g]	CHARACTER	LOGICAL*4
Imaginary part of complex number	1	IMAG	AIMAG DIMAG	COMPLEX*8 COMPLEX*16	REAL*4 REAL*8
Real part of complex number	1		REAL DREAL	COMPLEX*8 COMPLEX*16	REAL*4 REAL*8
Square root	1	SQRT	SQRT[h] DSQRT CSQRT CDSQRT ZSQRT	REAL*4 REAL*8 COMPLEX*8 COMPLEX*16 COMPLEX*16	REAL*4 REAL*8 COMPLEX*8 COMPLEX*16 COMPLEX*16
Exponential	1	EXP	EXP DEXP CEXP CDEXP ZEXP	REAL*4 REAL*8 COMPLEX*8 COMPLEX*16 COMPLEX*16	REAL*4 REAL*8 COMPLEX*8 COMPLEX*16 COMPLEX*16
Natural logarithm	1	LOG	ALOG[i] DLOG CLOG CDLOG ZLOG	REAL*4 REAL*8 COMPLEX*8 COMPLEX*16 COMPLEX*16	REAL*4 REAL*8 COMPLEX*8 COMPLEX*16 COMPLEX*16
Common logarithm	1	LOG10	ALOG10 DLOG10	REAL*4 REAL*8	REAL*4 REAL*8
Sine	1	SIN	SIN DSIN CSIN CDSIN ZSIN	REAL*4 REAL*8 COMPLEX*8 COMPLEX*16 COMPLEX*16	REAL*4 REAL*8 COMPLEX*8 COMPLEX*16 COMPLEX*16
Sine (degree)	1	SIND[j]	SIND DSIND	REAL*4 REAL*8	REAL*4 REAL*8
Cosine	1	COS	COS DCOS CCOS CDCOS ZCOS	REAL*4 REAL*8 COMPLEX*8 COMPLEX*16 COMPLEX*16	REAL*4 REAL*8 COMPLEX*8 COMPLEX*16 COMPLEX*16
Cosine (degree)	1	COSD	COSD DCOSD	REAL*4 REAL*8	REAL*4 REAL*8
Tangent	1	TAN	TAN DTAN	REAL*4 REAL*8	REAL*4 REAL*8
Tangent (degree)	1	TAND	TAND DTAND	REAL*4 REAL*8	REAL*4 REAL*8
Arcsine	1	ASIN[k,l,m]	ASIN DASIN	REAL*4 REAL*8	REAL*4 REAL*8
Arcsine (degree)	1	ASIND[n]	ASIND DASIND	REAL*4 REAL*8	REAL*4 REAL*8

Arccosine	1	ACOS	ACOS	REAL*4	REAL*4
			DACOS	REAL*8	REAL*8
Arccsine (degree)	1	ACOSD	ACOSD	REAL*4	REAL*4
			DACOSD	REAL*8	REAL*8
Arctangent	1	ATAN[o]	ATAN	REAL*4	REAL*4
			DATAN	REAL*8	REAL*8
Arctangent (degree)	1	ATAND[p]	ATAND	REAL*4	REAL*4
			DATAND	REAL*8	REAL*8
Arctangent	2	ATAN2[q,r]	ATAN2	REAL*4	REAL*4
			DATAN2	REAL*8	REAL*8
Arctangent (degree)	2	ATAN2D	ATAN2D	REAL*4	REAL*4
			DATAN2D	REAL*8	REAL*8
Hyperbolic sine	1	SINH	SINH	REAL*4	REAL*4
			DSINH	REAL*8	REAL*8
Hyperbolic cosine	1	COSH	COSH	REAL*4	REAL*4
			DCOSH	REAL*8	REAL*8
Hyperbolic tangent	1	TANH	TANH	REAL*4	REAL*4
			DTANH	REAL*8	REAL*8
Bitwise AND	2	IAND[1]		INTEGER*1	INTEGER*1
			IIAND	INTEGER*2	INTEGER*2
			JIAND	INTEGER*4	INTEGER*4
			KIAND	INTEGER*8	INTEGER*8
Bitwise inclusive OR	2	IOR[1]		INTEGER*1	INTEGER*1
			IIOR	INTEGER*2	INTEGER*2
			JIOR	INTEGER*4	INTEGER*4
			KIOR	INTEGER*8	INTEGER*8
Bitwise complement	1	NOT[1]	INOT	INTEGER*1	INTEGER*1
			JNOT	INTEGER*2	INTEGER*2
			KNOT	INTEGER*4	INTEGER*4
				INTEGER*8	INTEGER*8
Bitwise exclusive OR	2	IEOR[1]		INTEGER*1	INTEGER*1
			IIEOR	INTEGER*2	INTEGER*2
			JIEOR	INTEGER*4	INTEGER*4
			KIEOR	INTEGER*8	INTEGER*8
Bitwise logical shift	2	ISHFT		INTEGER*1	INTEGER*1
			IISHFT	INTEGER*2	INTEGER*2
			JISHFT	INTEGER*4	INTEGER*4
			KISHFT	INTEGER*8	INTEGER*8
Bitwise circular shift	2	ISHFTC		INTEGER*1	INTEGER*1
			IISHFTC	INTEGER*2	INTEGER*2
			JISHFTC	INTEGER*4	INTEGER*4
			KISHFTC	INTEGER*8	INTEGER*8
Bit extraction	3	IBITS		INTEGER*1	INTEGER*1
			IIBITS	INTEGER*2	INTEGER*2
			JIBITS	INTEGER*4	INTEGER*4
			KIBITS	INTEGER*8	INTEGER*8
Bit set	2	IBITSET		INTEGER*1	INTEGER*1
			IIBSET	INTEGER*2	INTEGER*2
			JIBSET	INTEGER*4	INTEGER*4
			KIBSET	INTEGER*8	INTEGER*8

Bit test	2	BTEST		INTEGER*1	LOGICAL*4
			BITEST	INTEGER*2	LOGICAL*2
			BJTEST	INTEGER*4	LOGICAL*4
			BKTEST	INTEGER*8	LOGICAL*8
Bit clear	2	IBCLR		INTEGER*1	INTEGER*1
			IIBCLR	INTEGER*2	INTEGER*2
			JIBCLR	INTEGER*4	INTEGER*4
			KIBCLR	INTEGER*8	INTEGER*8

[a] INT and IFIX return type INTEGER*2 if the **-i2** compile option is in effect; otherwise, the result type is INTEGER*4.

[b] When NINT or IDNINT is specified as an *argument* in a subroutine call or function reference, the compiler supplies either an INTEGER*2 or an INTEGER*4 function depending on the **-i2** command line option (see Chapter 1of the *Fortran 77 Programmer's Guide*).

[c] The IABS, ISIGN, IDIM, and integer MOD intrinsics accept either INTEGER*2 arguments or INTEGER*4 arguments, and the result is the same type.

[d] The result for MOD, AMOD, and DMOD is undefined when the value of the second argument is zero.

[e] If the value of the first argument of ISIGN, SIGN, or DSIGN is zero, the result is zero.

[f] The result of INDEX is an integer value indicating the position in the first argument of the first substring which is identical to the second argument. The result of INDEX('ABCDEF', 'CD'), for example, would be 3. If no substring of the first argument matches the second argument, the result is zero. INDEX and ICHAR return the result type INTEGER*2 if the **-i2** compile option is in effect; otherwise, the result type is INTEGER*4.

[g] The character relational intrinsics (LLT, LGT, LEE, and LGE) return result type LOGICAL*2 if the **$log2** (see Chapter 11) compile option is in effect; otherwise, the result type is LOGICAL*4.

[h] The value of the argument of SQRT and DSQRT must be greater than or equal to zero. The result of CSQRT is the principal value with the real part greater than or equal to zero. When the real part is zero, the imaginary part is greater than or equal to zero.

[i] The argument of ALOG and DLOG must be greater than zero. The argument of CLOG must not be (0.,0.). The range of the imaginary part of the result of CLOG is: −p <imaginary part <p.

[j] The argument for SIND, COSD, or TAND must be in degrees and is treated as modulo 360.

[k] The absolute value of the arguments of ASIN, DASIN, ASIND, DASIND, ACOS, DACOS, ACOSD, and DACSOD must be less than or equal to 1.

[l] The range of the result for ASIN and DASIN is -π/2 <result < π/2; the range of the result for DASIN is 0 < result < π; and the range of the result of ACOS and DACOS is less than or equal to one.

[m] The result of ASIN, DASIN, ACOS, and DACOS is in radians.

[n] The result of ASIND, DASIND, ACOS, DACOSD is in degrees.

[o] The result of ATAN, DATAN, ATAN2, and DTAN2 is in radians.

[p] The result of ATAND, DATAND, ATAN2D, and DATAN2D is in degrees.

[q] If the value of the first argument of ATAN2 or DATAN2 is positive, the result is positive. When the value of the first argument is zero, the result is zero if the second argumemt is positive and P if the second argument is negative. If the value of the first argument is negative, the result is negative. If the value of the second argument is zero, the aboslute value of the result is P/2. Both arguments must not have the value zero.

[r] Note 3 on this page also applies to ATAN2 and DTAN2D, except for the range of the result, which is:
−180 degrees << result << 180 degrees.