A
MIPS-Style Alpha Instruction Set

This appendix provides a summary of the Alpha instruction set and describes the 21264 IEEE floating-point conformance. It is organized as follows:

- Alpha instruction summary
- Reserved opcodes
- IEEE floating-point instructions
- VAX floating-point instructions
- Independent floating-point instructions
- Opcode summary
- Required PALcode function codes
- IEEE floating-point conformance

A.1 Alpha Instruction Summary

This section contains a summary of all Alpha architecture instructions. All values are in hexadecimal radix. Table A–1 describes the contents of the Format and Opcode columns that are in Table A–2.

Table A–1 Instruction Format and Opcode Notation

Instruction Format	Format Symbol	Opcode Notation	Meaning
Branch	Bra	oo	*oo* is the 6-bit opcode field.
Floating-point	F-P	oo.fff	*oo* is the 6-bit opcode field . *fff* is the 11-bit function code field.
Memory	Mem	oo	*oo* is the 6-bit opcode field.
Memory/function code	Mfc	oo.ffff	*oo* is the 6-bit opcode field. *ffff* is the 16-bit function code in the displacement field.
Memory/branch	Mbr	oo.h	*oo* is the 6-bit opcode field. *h* is the high-order 2 bits of the displacement field.

Table A–1 Instruction Format and Opcode Notation

Instruction Format	Format Symbol	Opcode Notation	Meaning
Operate	Opr	oo.ff	*oo* is the 6-bit opcode field. *ff* is the 7-bit function code field.
PALcode	Pcd	oo	*oo* is the 6-bit opcode field; the particular PALcode instruction is specified in the 26-bit function code field.

Qualifiers for operate instructions are shown in Table A–2. Qualifiers for IEEE and VAX floating-point instructions are shown in Tables A–5 and A–6, respectively.

Table A–2 Architecture Instructions

Mnemonic	Format	Opcode	Description
ADDF	F-P	15.080	Add F_floating
ADDG	F-P	15.0A0	Add G_floating
ADDL	Opr	10.00	Add longword
ADDL/V	—	10.40	—
ADDQ	Opr	10.20	Add quadword
ADDQ/V	—	10.60	—
ADDS	F-P	16.080	Add S_floating
ADDT	F-P	16.0A0	Add T_floating
AMASK	Opr	11.61	Architecture mask
AND	Opr	11.00	Logical product
BEQ	Bra	39	Branch if = zero
BGE	Bra	3E	Branch if ≥ zero
BGT	Bra	3F	Branch if > zero
BIC	Opr	11.08	Bit clear
BIS	Opr	11.20	Logical sum
BLBC	Bra	38	Branch if low bit clear
BLBS	Bra	3C	Branch if low bit set
BLE	Bra	3B	Branch if ≤ zero
BLT	Bra	3A	Branch if < zero
BNE	Bra	3D	Branch if ≠ zero
BR	Bra	30	Unconditional branch
BSR	Mbr	34	Branch to subroutine

Table A–2 Architecture Instructions (Continued)

Mnemonic	Format	Opcode	Description
CALL_PAL	Pcd	00	Trap to PALcode
CMOVEQ	Opr	11.24	CMOVE if = zero
CMOVGE	Opr	11.46	CMOVE if ≥ zero
CMOVGT	Opr	11.66	CMOVE if > zero
CMOVLBC	Opr	11.16	CMOVE if low bit clear
CMOVLBS	Opr	11.14	CMOVE if low bit set
CMOVLE	Opr	11.64	CMOVE if ≤ zero
CMOVLT	Opr	11.44	CMOVE if < zero
CMOVNE	Opr	11.26	CMOVE if ≠ zero
CMPBGE	Opr	10.0F	Compare byte
CMPEQ	Opr	10.2D	Compare signed quadword equal
CMPGEQ	F-P	15.0A5	Compare G_floating equal
CMPGLE	F-P	15.0A7	Compare G_floating less than or equal
CMPGLT	F-P	15.0A6	Compare G_floating less than
CMPLE	Opr	10.6D	Compare signed quadword less than or equal
CMPLT	Opr	10.4D	Compare signed quadword less than
CMPTEQ	F-P	16.0A5	Compare T_floating equal
CMPTLE	F-P	16.0A7	Compare T_floating less than or equal
CMPTLT	F-P	16.0A6	Compare T_floating less than
CMPTUN	F-P	16.0A4	Compare T_floating unordered
CMPULE	Opr	10.3D	Compare unsigned quadword less than or equal
CMPULT	Opr	10.1D	Compare unsigned quadword less than
CPYS	F-P	17.020	Copy sign
CPYSE	F-P	17.022	Copy sign and exponent
CPYSN	F-P	17.021	Copy sign negate
CVTDG	F-P	15.09E	Convert D_floating to G_floating
CVTGD	F-P	15.0AD	Convert G_floating to D_floating
CVTGF	F-P	15.0AC	Convert G_floating to F_floating
CVTGQ	F-P	15.0AF	Convert G_floating to quadword
CVTLQ	F-P	17.010	Convert longword to quadword
CVTQF	F-P	15.0BC	Convert quadword to F_floating

Table A–2 Architecture Instructions (Continued)

Mnemonic	Format	Opcode	Description
CVTQG	F-P	15.0BE	Convert quadword to G_floating
CVTQL	F-P	17.030	Convert quadword to longword
CVTQS	F-P	16.0BC	Convert quadword to S_floating
CVTQT	F-P	16.0BE	Convert quadword to T_floating
CVTST	F-P	16.2AC	Convert S_floating to T_floating
CVTTQ	F-P	16.0AF	Convert T_floating to quadword
CVTTS	F-P	16.0AC	Convert T_floating to S_floating
DIVF	F-P	15.083	Divide F_floating
DIVG	F-P	15.0A3	Divide G_floating
DIVS	F-P	16.083	Divide S_floating
DIVT	F-P	16.0A3	Divide T_floating
ECB	Mfc	18.E800	Evict cache block
EQV	Opr	11.48	Logical equivalence
EXCB	Mfc	18.0400	Exception barrier
EXTBL	Opr	12.06	Extract byte low
EXTLH	Opr	12.6A	Extract longword high
EXTLL	Opr	12.26	Extract longword low
EXTQH	Opr	12.7A	Extract quadword high
EXTQL	Opr	12.36	Extract quadword low
EXTWH	Opr	12.5A	Extract word high
EXTWL	Opr	12.16	Extract word low
FBEQ	Bra	31	Floating branch if = zero
FBGE	Bra	36	Floating branch if ≥ zero
FBGT	Bra	37	Floating branch if > zero
FBLE	Bra	33	Floating branch if ≤ zero
FBLT	Bra	32	Floating branch if < zero
FBNE	Bra	35	Floating branch if ≠ zero
FCMOVEQ	F-P	17.02A	FCMOVE if = zero
FCMOVGE	F-P	17.02D	FCMOVE if ≥ zero
FCMOVGT	F-P	17.02F	FCMOVE if > zero
FCMOVLE	F-P	17.02E	FCMOVE if ≤ zero
FCMOVLT	F-P	17.02C	FCMOVE if < zero

Table A–2 Architecture Instructions (Continued)

Mnemonic	Format	Opcode	Description
FCMOVNE	F-P	17.02B	FCMOVE if ≠ zero
FETCH	Mfc	18.8000	Prefetch data
FETCH_M	Mfc	18.A000	Prefetch data, modify intent
FTOIS	F-P	1C.78	Floating to integer move, S_floating
FTOIT	F-P	1C.70	Floating to integer move, T_floating
IMPLVER	Opr	11.6C	Implementation version
INSBL	Opr	12.0B	Insert byte low
INSLH	Opr	12.67	Insert longword high
INSLL	Opr	12.2B	Insert longword low
INSQH	Opr	12.77	Insert quadword high
INSQL	Opr	12.3B	Insert quadword low
INSWH	Opr	12.57	Insert word high
INSWL	Opr	12.1B	Insert word low
ITOFF	F-P	14.014	Integer to floating move, F_floating
ITOFS	F-P	14.004	Integer to floating move, S_floating
ITOFT	F-P	14.024	Integer to floating move, T_floating
JMP	Mbr	1A.0	Jump
JSR	Mbr	1A.1	Jump to subroutine
JSR_COROUTINE	Mbr	1A.3	Jump to subroutine return
LDA	Mem	08	Load address
LDAH	Mem	09	Load address high
LDBU	Mem	0A	Load zero-extended byte
LDF	Mem	20	Load F_floating
LDG	Mem	21	Load G_floating
LDL	Mem	28	Load sign-extended longword
LDL_L	Mem	2A	Load sign-extended longword locked
LDQ	Mem	29	Load quadword
LDQ_L	Mem	2B	Load quadword locked
LDQ_U	Mem	0B	Load unaligned quadword
LDS	Mem	22	Load S_floating
LDT	Mem	23	Load T_floating
LDWU	Mem	0C	Load zero-extended word

Table A–2 Architecture Instructions (Continued)

Mnemonic	Format	Opcode	Description
MAXSB8	Opr	1C.3E	Vector signed byte maximum
MAXSW4	Opr	1C.3F	Vector signed word maximum
MAXUB8	Opr	1C.3C	Vector unsigned byte maximum
MAXUW4	Opr	1C.3D	Vector unsigned word maximum
MB	Mfc	18.4000	Memory barrier
MF_FPCR	F-P	17.025	Move from FPCR
MINSB8	Opr	1C.38	Vector signed byte minimum
MINSW4	Opr	1C.39	Vector signed word minimum
MINUB8	Opr	1C.3A	Vector unsigned byte minimum
MINUW4	Opr	1C.3B	Vector unsigned word minimum
MSKBL	Opr	12.02	Mask byte low
MSKLH	Opr	12.62	Mask longword high
MSKLL	Opr	12.22	Mask longword low
MSKQH	Opr	12.72	Mask quadword high
MSKQL	Opr	12.32	Mask quadword low
MSKWH	Opr	12.52	Mask word high
MSKWL	Opr	12.12	Mask word low
MT_FPCR	F-P	17.024	Move to FPCR
MULF	F-P	15.082	Multiply F_floating
MULG	F-P	15.0A2	Multiply G_floating
MULL	Opr	13.00	Multiply longword
MULL/V		13.40	
MULQ	Opr	13.20	Multiply quadword
MULQ/V		13.60	
MULS	F-P	16.082	Multiply S_floating
MULT	F-P	16.0A2	Multiply T_floating
ORNOT	Opr	11.28	Logical sum with complement
PERR	Opr	1C.31	Pixel error
PKLB	Opr	1C.37	Pack longwords to bytes
PKWB	Opr	1C.36	Pack words to bytes
RC	Mfc	18.E000	Read and clear
RET	Mbr	1A.2	Return from subroutine

Table A–2 Architecture Instructions (Continued)

Mnemonic	Format	Opcode	Description
RPCC	Mfc	18.C000	Read process cycle counter
RS	Mfc	18.F000	Read and set
S4ADDL	Opr	10.02	Scaled add longword by 4
S4ADDQ	Opr	10.22	Scaled add quadword by 4
S4SUBL	Opr	10.0B	Scaled subtract longword by 4
S4SUBQ	Opr	10.2B	Scaled subtract quadword by 4
S8ADDL	Opr	10.12	Scaled add longword by 8
S8ADDQ	Opr	10.32	Scaled add quadword by 8
S8SUBL	Opr	10.1B	Scaled subtract longword by 8
S8SUBQ	Opr	10.3B	Scaled subtract quadword by 8
SEXTB	Opr	1C.00	Sign extend byte
SEXTW	Opr	1C.01	Sign extend word
SLL	Opr	12.39	Shift left logical
SQRTF	F-P	14.08A	Square root F_floating
SQRTG	F-P	14.0AA	Square root G_floating
SQRTS	F-P	14.08B	Square root S_floating
SQRTT	F-P	14.0AB	Square root T_floating
SRA	Opr	12.3C	Shift right arithmetic
SRL	Opr	12.34	Shift right logical
STB	Mem	0E	Store byte
STF	Mem	24	Store F_floating
STG	Mem	25	Store G_floating
STL	Mem	2C	Store longword
STL_C	Mem	2E	Store longword conditional
STQ	Mem	2D	Store quadword
STQ_C	Mem	2F	Store quadword conditional
STQ_U	Mem	0F	Store unaligned quadword
STS	Mem	26	Store S_floating
STT	Mem	27	Store T_floating
STW	Mem	0D	Store word
SUBF	F-P	15.081	Subtract F_floating
SUBG	F-P	15.0A1	Subtract G_floating

Table A–2 Architecture Instructions (Continued)

Mnemonic	Format	Opcode	Description
SUBL	Opr	10.09	Subtract longword
SUBL/V		10.49	
SUBQ	Opr	10.29	Subtract quadword
SUBQ/V		10.69	
SUBS	F-P	16.081	Subtract S_floating
SUBT	F-P	16.0A1	Subtract T_floating
TRAPB	Mfc	18.0000	Trap barrier
UMULH	Opr	13.30	Unsigned multiply quadword high
UNPKBL	Opr	1C.35	Unpack bytes to longwords
UNPKBW	Opr	1C.34	Unpack bytes to words
WH64	Mfc	18.F800	Write hint — 64 bytes
WMB	Mfc	18.4400	Write memory barrier
XOR	Opr	11.40	Logical difference
ZAP	Opr	12.30	Zero bytes
ZAPNOT	Opr	12.31	Zero bytes not

A.2 Reserved Opcodes

This section describes the opcodes that are reserved in the Alpha architecture. They can be reserved for Compaq or for PALcode.

A.2.1 Opcodes Reserved for Compaq

Table A–3 lists opcodes reserved for Compaq.

Table A–3 Opcodes Reserved for Compaq

Mnemonic	Opcode	Mnemonic	Opcode
OPC01	01	OPC05	05
OPC02	02	OPC06	06
OPC03	03	OPC07	07
OPC04	04	—	—

A.2.2 Opcodes Reserved for PALcode

Table A–4 lists the 21264-specific instructions. See Chapter 2 for more information.

Table A–4 Opcodes Reserved for PALcode

21264 Mnemonic	Opcode	Architecture Mnemonic	Function
HW_LD	1B	PAL1B	Performs Dstream load instructions.
HW_ST	1F	PAL1F	Performs Dstream store instructions.
HW_REI	1E	PAL1E	Returns instruction flow to the program counter (PC) pointed to by EXC_ADDR internal processor register (IPR).
HW_MFPR	19	PAL19	Accesses the Ibox, Mbox, and Dcache IPRs.
HW_MTPR	1D	PAL1D	Accesses the Ibox, Mbox, and Dcache IPRs.

A.3 IEEE Floating-Point Instructions

Table A–5 lists the hexadecimal value of the 11-bit function code field for the IEEE floating-point instructions, with and without qualifiers. The opcode for these instructions is 16_{16}.

Table A–5 IEEE Floating-Point Instruction Function Codes

Mnemonic	None	/C	/M	/D	/U	/UC	/UM	/UD
ADDS	080	000	040	0C0	180	100	140	1C0
ADDT	0A0	020	060	0E0	1A0	120	160	1E0
CMPTEQ	0A5	—	—	—	—	—	—	—
CMPTLT	0A6	—	—	—	—	—	—	—
CMPTLE	0A7	—	—	—	—	—	—	—
CMPTUN	0A4	—	—	—	—	—	—	—
CVTQS	0BC	03C	07C	0FC	—	—	—	—
CVTQT	0BE	03E	07E	0FE	—	—	—	—
CVTST	See below	—	—	—	—	—	—	—
CVTTQ	See below	—	—	—	—	—	—	—
CVTTS	0AC	02C	06C	0EC	1AC	12C	16C	1EC
DIVS	083	003	043	0C3	183	103	143	1C3
DIVT	0A3	023	063	0E3	1A3	123	163	1E3
MULS	082	002	042	0C2	182	102	142	1C2

Table A–5 IEEE Floating-Point Instruction Function Codes (Continued)

MULT	0A2	022	062	0E2	1A2	122	162	1E2
SQRTS	08B	00B	04B	0CB	18B	10B	14B	1CB
SQRTT	0AB	02B	06B	0EB	1AB	12B	16B	1EB
SUBS	081	001	041	0C1	181	101	141	1C1
SUBT	0A1	021	061	0E1	1A1	121	161	1E1

Mnemonic	/SU	/SUC	/SUM	/SUD	/SUI	/SUIC	/SUIM	/SUID
ADDS	580	500	540	5C0	780	700	740	7C0
ADDT	5A0	520	560	5E0	7A0	720	760	7E0
CMPTEQ	5A5							
CMPTLT	5A6							
CMPTLE	5A7							
CMPTUN	5A4							
CVTQS					7BC	73C	77C	7FC
CVTQT					7BE	73E	77E	7FE
CVTTS	5AC	52C	56C	5EC	7AC	72C	76C	7EC
DIVS	583	503	543	5C3	783	703	743	7C3
DIVT	5A3	523	563	5E3	7A3	723	763	7E3
MULS	582	502	542	5C2	782	702	742	7C2
MULT	5A2	522	562	5E2	7A2	722	762	7E2
SQRTS	58B	50B	54B	5CB	78B	70B	74B	7CB
SQRTT	5AB	52B	56B	5EB	7AB	72B	76B	7EB
SUBS	581	501	541	5C1	781	701	741	7C1
SUBT	5A1	521	561	5E1	7A1	721	761	7E1

Mnemonic	None	/S
CVTST	2AC	6AC

Mnemonic	None	/C	/V	/VC	/SV	/SVC	/SVI	/SVIC
CVTTQ	0AF	02F	1AF	12F	5AF	52F	7AF	72F

Mnemonic	D	/VD	/SVD	/SVID	/M	/VM	/SVM	/SVIM
CVTTQ	0EF	1EF	5EF	7EF	06F	16F	56F	76F

Programming Note:

In order to use CMPTxx with software completion trap handling, it is necessary to specify the /SU IEEE trap mode, even though an underflow trap is not possible. In order to use CVTQS or CVTQT with software completion trap handling, it is necessary to specify the /SUI IEEE trap mode, even though an underflow trap is not possible.

A.4 VAX Floating-Point Instructions

Table A–6 lists the hexadecimal value of the 11-bit function code field for the VAX floating-point instructions. The opcode for these instructions is 15_{16}.

Table A–6 VAX Floating-Point Instruction Function Codes

Mnemonic	None	/C	/U	/UC	/S	/SC	/SU	/SUC
ADDF	080	000	180	100	480	400	580	500
ADDG	0A0	020	1A0	120	4A0	420	5A0	520
CMPGEQ	0A5				4A5			
CMPGLE	0A7				4A7			
CMPGLT	0A6				4A6			
CVTDG	09E	01E	19E	11E	49E	41E	59E	51E
CVTGD	0AD	02D	1AD	12D	4AD	42D	5AD	52D
CVTGF	0AC	02C	1AC	12C	4AC	42C	5AC	52C
CVTGQ	See below							
CVTQF	0BC	03C						
CVTQG	0BE	03E						
DIVF	083	003	183	103	483	403	583	503
DIVG	0A3	023	1A3	123	4A3	423	5A3	523
MULF	082	002	182	102	482	402	582	502
MULG	0A2	022	1A2	122	4A2	422	5A2	522
SQRTF	08A	00A	18A	10A	48A	40A	58A	50A
SQRTG	0AA	02A	1AA	12A	4AA	42A	5AA	52A
SUBF	081	001	181	101	481	401	581	501
SUBG	0A1	021	1A1	121	4A1	421	5A1	521
Mnemonic	**None**	**/C**	**/V**	**/VC**	**/S**	**/SC**	**/SV**	**/SVC**
CVTGQ	0AF	02F	1AF	12F	4AF	42F	5AF	52F

A.5 Independent Floating-Point Instructions

Table A–7 lists the hexadecimal value of the 11-bit function code field for the floating-point instructions that are not directly tied to IEEE or VAX floating point. The opcode for the following instructions is 17_{16}.

Table A–7 Independent Floating-Point Instruction Function Codes

Mnemonic	None	/V	/SV
CPYS	020	—	—
CPYSE	022	—	—
CPYSN	021	—	—
CVTLQ	010	—	—
CVTQL	030	130	530
FCMOVEQ	02A	—	—
FCMOVGE	02D	—	—
FCMOVGT	02F	—	—
FCMOVLE	02E	—	—
FCMOVLT	02C	—	—
MF_FPCR	025	—	—
MT_FPCR	024	—	—

A.6 Opcode Summary

Table A–8 lists all Alpha opcodes from 00 (CALL_PAL) through 3F (BGT). In the table, the column headings that appear over the instructions have a granularity of 8_{16}. The rows beneath the Offset column supply the individual hexadecimal number to resolve that granularity.

If an instruction column has a 0 in the right (low) hexadecimal digit, replace that 0 with the number to the left of the backslash (\) in the Offset column on the instruction's row. If an instruction column has an 8 in the right (low) hexadecimal digit, replace that 8 with the number to the right of the backslash in the Offset column.

For example, the third row (2/A) under the 10_{16} column contains the symbol INTS*, representing the all-integer shift instructions. The opcode for those instructions would then be 12_{16} because the 0 in 10 is replaced by the 2 in the Offset column. Likewise, the third row under the 18_{16} column contains the symbol JSR*, representing all jump instructions. The opcode for those instructions is 1A because the 8 in the heading is replaced by the number to the right of the backslash in the Offset column. The instruction format is listed under the instruction symbol..

Table A–8 Opcode Summary

Offset	00	08	10	18	20	28	30	38
0/8	PAL* (pal)	LDA (mem)	INTA* (op)	MISC* (mem)	LDF (mem)	LDL (mem)	BR (br)	BLBC (br)
1/9	Res	LDAH (mem)	INTL* (op)	\PAL\	LDG (mem)	LDQ (mem)	FBEQ (br)	BEQ (br)
2/A	LDBU	Res	INTS* (op)	JSR* (mem)	LDS (mem)	LDL_L (mem)	FBLT (br)	BLT (br)

Table A–8 Opcode Summary (Continued)

Offset	00	08	10	18	20	28	30	38
3/B	Res	LDQ_U (mem)	INTM* (op)	\PAL\	LDT (mem)	LDQ_L (mem)	FBLE (br)	BLE (br)
4/C	LDWU	Res	ITFP*	FPTI*	STF (mem)	STL (mem)	BSR (br)	BLBS (br)
5/D	Res	STW	FLTV* (op)	\PAL\	STG (mem)	STQ (mem)	FBNE (br)	BNE (br)
6/E	Res	STB	FLTI* (op)	\PAL\	STS (mem)	STL_C (mem)	FBGE (br)	BGE (br)
7/F	Res	STQ_U (mem)	FLTL* (op)	\PAL\	STT (mem)	STQ_C (mem)	FBGT (br)	BGT (br)

Table A–9 explains the symbols used in Table A–8.

Table A–9 Key to Opcode Summary Used in Table A–8

Symbol	Meaning
FLTI*	IEEE floating-point instruction opcodes
FLTL*	Floating-point Operate instruction opcodes
FLTV*	VAX floating-point instruction opcodes
FPTI*	Floating-point to integer register move opcodes
INTA*	Integer arithmetic instruction opcodes
INTL*	Integer logical instruction opcodes
INTM*	Integer multiply instruction opcodes
INTS*	Integer shift instruction opcodes
ITFP*	Integer to floating-point register move opcodes
JSR*	Jump instruction opcodes
MISC*	Miscellaneous instruction opcodes
PAL*	PALcode instruction (CALL_PAL) opcodes
\PAL\	Reserved for PALcode
Res	Reserved for Compaq

A.7 Required PALcode Function Codes

Table A–10 lists opcodes required for all Alpha implementations. The notation used is *oo.ffff*, where *oo* is the hexadecimal 6-bit opcode and *ffff* is the hexadecimal 26-bit function code.

Table A–10 Required PALcode Function Codes

Mnemonic	Type	Function Code
DRAINA	Privileged	00.0002
HALT	Privileged	00.0000
IMB	Unprivileged	00.0086

A.8 IEEE Floating-Point Conformance

The 21264 supports the IEEE floating-point operations defined in the *Alpha System Reference Manual, Revision* 7 and therefore also from the *Alpha Architecture Handbook, Version 4*. Support for a complete implementation of the IEEE Standard for Binary Floating-Point Arithmetic (ANSI/IEEE Standard 754 1985) is provided by a combination of hardware and software. The 21264 provides several hardware features to facilitate complete support of the IEEE standard.

The 21264 provides the following hardware features to facilitate complete support of the IEEE standard:

- The 21264 implements precise exception handling in hardware, as denoted by the AMASK instruction returning bit 9 set. TRAPB instructions are treated as NOPs and are not issued.
- The 21264 accepts both Signaling and Quiet NaNs as input operands and propagates them as specified by the Alpha architecture. In addition, the 21264 delivers a canonical Quiet NaN when an operation is required to produce a NaN value and none of its inputs are NaNs. Encodings for Signaling NaN and Quiet NaN are defined by the *Alpha Architecture Handbook, Version 4*.
- The 21264 accepts infinity operands and implements infinity arithmetic as defined by the IEEE standard and the *Alpha Architecture Handbook, Version 4*.
- The 21264 implements SQRT for single (SQRTS) and double (SQRTT) precision in hardware.

Note: In addition, the 21264 also implements the VAX SQRTF and SQRTG instructions.

- The 21264 implements the FPCR[DNZ] bit. When FPCR[DNZ] is set, denormal input operand traps can be avoided for arithmetic operations that include the /S qualifier. When FPCR[DNZ] is clear, denormal input operands for arithmetic operations produce an unmaskable denormal trap. CPYSE/CPYSN, FCMOVxx, and MF_FPCR/MT_FPCR are not arithmetic operations, and pass denormal values without initiating arithmetic traps.
- The 21264 implements the following disable bits in the floating-point control register (FPCR):
  - Underflow disable (UNFD)
  - Overflow disable (OVFD)
  - Inexact result disable (INED)
  - Division by zero disable (DZED)
  - Invalid operation disable (INVD)

  If one of these bits is set, and an instruction with the /S qualifier set generates the associated exception, the 21264 produces the IEEE nontrapping result and suppresses the trap. These nontrapping responses include correctly signed infinity, largest finite number, and Quiet NaNs as specified by the IEEE standard.

The 21264 will not produce a Denormal result for the underflow exception. Instead, a true zero (+0) is written to the destination register. In the 21264 the FPCR underflow to zero (UNDZ) bit must be set if underflow disable (UNFD) bit is set. If desired, trapping on underflow can be enabled by the instruction and the FPCR, and software may compute the Denormal value as defined in the IEEE Standard.

The 21264 records floating-point exception information in two places:

- The FPCR status bits record the occurence of all exceptions that are detected, whether or not the corresponding trap is enabled. The status bits are cleared only through an explicit clear command (MT_FPCR); hence, the exception information they record is a summary of all exceptions that have occurred since the last time they were cleared.
- If an exception is detected and the corresponding trap is enabled by the instruction, and is not disabled by the FPCR control bits, the 21264 will record the condition in the EXC_SUM register and initiate an arithmetic trap.

The following items apply to Table A–11:

- The 21264 traps on a Denormal input operand for all arithmetic operations unless FPCR[DNZ] = 1.
- Input operand traps take precedence over arithmetic result traps.
- The following abbreviations are used:

  Inf: Infinity

  QNaN: Quiet NaN

  SNaN: Signalling NaN

  CQNaN: Canonical Quiet NaN

  For IEEE instructions with /S, Table A–11 lists all exceptional input and output conditions recognized by the 21264, along with the result and exception generated for each condition.

Table A–11 Exceptional Input and Output Conditions

Alpha Instructions	21264 Hardware Supplied Result	Exception
ADDx SUBx INPUT		
Inf operand	±Inf	(none)
QNaN operand	QNaN	(none)
SNaN operand	QNaN	Invalid Op
Effective subtract of two Inf operands	CQNaN	Invalid Op
ADDx SUBx OUTPUT		
Exponent overflow	±Inf or ±MAX	Overflow
Exponent underflow	+0	Underflow
Inexact result	Result	Inexact
MULx INPUT		

Table A–11 Exceptional Input and Output Conditions (Continued)

Alpha Instructions	21264 Hardware Supplied Result	Exception
Inf operand	±Inf	(none)
QNaN operand	QNaN	(none)
SNaN operand	QNaN	Invalid Op
0 * Inf	CQNaN	Invalid Op
MULx OUTPUT (same as ADDx)		
DIVx INPUT		
QNaN operand	QNaN	(none)
SNaN operand	QNaN	Invalid Op
0/0 or Inf/Inf	CQNaN	Invalid Op
A/0 (A not 0)	±Inf	Div Zero
A/Inf	±0	(none)
Inf/A	±Inf	(none)
DIVx OUTPUT (same as ADDx)		
SQRTx INPUT		
+Inf operand	+Inf	(none)
QNaN operand	QNaN	(none)
SNaN operand	QNaN	Invalid Op
-A (A not 0)	CQNaN	Invalid Op
-0	-0	(none)
SQRTx OUTPUT		
Inexact result	root	Inexact
CMPTEQ CMPTUN INPUT		
Inf operand	True or False	(none)
QNaN operand	False for EQ, True for UN	(none)
SNaN operand	False for EQ,True for UN	Invalid Op
CMPTLT CMPTLE INPUT		
Inf operand	True or False	(none)
QNaN operand	False	Invalid Op
SNaN operand	False	Invalid Op
CVTfi INPUT		
Inf operand	0	Invalid Op
QNaN operand	0	Invalid Op

Table A–11 Exceptional Input and Output Conditions (Continued)

Alpha Instructions	21264 Hardware Supplied Result	Exception
SNaN operand	0	Invalid Op
CVTfi OUTPUT		
Inexact result	Result	Inexact
Integer overflow	Truncated result	Invalid Op
CVTif OUTPUT		
Inexact result	Result	Inexact
CVTff INPUT		
Inf operand	±Inf	(none)
QNaN operand	QNaN	(none)
SNaN operand	QNaN	Invalid Op
CVTff OUTPUT (same as ADDx)		
FBEQ FBNE FBLT FBLE FBGT FBGE LDS LDT STS STT CPYS CPYSN FCMOVx		

See Section 2.3 for information about the floating-point control register (FPCR).