Message	Meaning	# (hex)
Bad Packet Block check	Computed packet checksum doesn't match checksum in packet.	C01
Can't Edit Null Char.	Attempted to edit a string containing character ■ (character code 0).	102
Circular Reference	Attempted to store a variable name into itself.	129
Connecting	Indicates verifying IR or serial connection.	C0A
Constant?	HP Solve application or ROOT returned same value at every sample point of current equation.	A02
Copied to stack	[CST] copied selected equation to stack.	623
Current equation:	Identifies current equation.	608
Deleting Column	MatrixWriter application is deleting a column.	504
Deleting Row	MatrixWriter application is deleting a row.	503
Directory Not Allowed	Name of existing directory variable used as argument.	12A
Directory Recursion	Attempted to store a directory into itself.	002
Empty catalog	No data in current catalog (Equation, Statistics, Alarm).	60D
Empty stack	The stack contains no data.	C15

Message	Meaning	# (hex)
Enter alarm, press SET	Alarm entry prompt.	61A
Enter eqn, press NEW	Store new equation in *EQ*.	60A
Enter value (zoom out if >1), press ENTER	Zoom operations prompt.	622
EQ Invalid for MINIT	*EQ* must contain at least two equations (or programs) and two variables.	E403
Extremum	Result returned by HP Solve application or ROOT is an extremum rather than a root.	A06
HALT Not Allowed	A program containing HALT executed while MatrixWriter application, DRAW, or HP Solve application active.	126
I/O setup menu	Identifies I/O setup menu.	61C
Illegal During MROOT	Multiple-Equation Solver command attempted during MROOT execution.	E406
Implicit () off	Implicit parentheses off.	207
Implicit () on	Implicit parentheses on.	208
Incomplete Subexpression	▶, ▼, or [ENTER] pressed before all function arguments supplied.	206

Messages Listed Alphabetically (continued)

Message	Meaning	# (hex)
Inconsistent Units	Attempted unit conversion with incompatible units.	B02
Infinite Result	Math exception: Calculation such as 1/0 infinite result.	305
Inserting Column	MatrixWriter application is inserting a column.	506
Inserting Row	MatrixWriter application is inserting a row.	505
Insufficient Memory	Not enough free memory to execute operation.	001
Insufficient Σ Data	A Statistics command was executed when *ΣDAT* did not contain enough data points for calculation.	603
Interrupted	The HP Solve application or ROOT was interrupted by CANCEL.	A03
Invalid Array Element	ENTER returned object of wrong type for current matrix.	502
Invalid Card Data	HP 48 does not recognize data on plug-in card.	008
Invalid Date	Date argument not real number in correct format, or was out of range.	D01
Invalid Definition	Incorrect structure of equation argument for DEFINE.	12C
Invalid Dimension	Array argument had wrong dimensions.	501

Messages Listed Alphabetically (continued)

Message	Meaning	# (hex)
Invalid EQ	Attempted operation from PICTURE FCN menu when *EQ* did not contain algebraic, or, attempted DRAW with CONIC plot type when *EQ* did not contain algebraic.	607
Invalid IOPAR	*IOPAR* not a list, or one or more objects in list missing or invalid.	C12
Invalid Mpar	*Mpar* variable not created by MINIT.	E401
Invalid Name	Received illegal filename, or server asked to send illegal filename.	C17
Invalid PPAR	*PPAR* not a list, or one or more objects in list missing or invalid.	12E
Invalid PRTPAR	*PRTPAR* not a list, or one or more objects in list missing or invalid.	C13
Invalid PTYPE	Plot type invalid for current equation.	620
Invalid Repeat	Alarm repeat interval out of range.	D03
Invalid Server Cmd.	Invalid command received while in Server mode.	C08
Invalid Syntax	HP 48 unable execute ENTER, OBJ→, or STR→ due to invalid object syntax.	106

Messages Listed Alphabetically (continued)

Message	Meaning	# (hex)
Invalid Time	Time argument not real number in correct format, or out of range.	D02
Invalid Unit	Unit operation attempted with invalid or undefined user unit.	B01
Invalid User Function	Type or structure of object executed as user-defined function was incorrect.	103
Invalid Σ Data	Statistics command executed with invalid object stored in ΣDAT.	601
Invalid Σ Data LN(Neg)	Non-linear curve fit attempted when ΣDAT matrix contained a negative element.	605
Invalid Σ Data LN(0)	Non-linear curve fit attempted when ΣDAT matrix contained a 0 element.	606
Invalid ΣPAR	ΣPAR not list, or one or more objects in list missing or invalid.	604
Keyword Conflict	A plug-in card conflicts with an equation library variable. Remove the card to continue.	E303
LAST CMD Disabled	(→)(CMD) pressed while that recovery feature disabled.	125
LAST STACK Disabled	(→)(UNDO) pressed while that recovery feature disabled.	124
LASTARG Disabled	(→)(ARG) executed while that recovery feature disabled.	205

Messages Listed Alphabetically (continued)

Message	Meaning	# (hex)
Low Battery	System batteries too low to safely print or perform I/O.	C14
Memory Clear	HP 48 memory was cleared.	005
Name Conflict	Execution of | (where) attempted to assign value to variable of integration or summation index.	13C
Name the equation, press ENTER	Name equation and store it in EQ.	60B
Name the stat data, press ENTER	Name statistics data and store it in ΣDAT.	621
Negative Underflow	Math exception: Calculation returned negative, non-zero result greater than −MINR.	302
No Current Equation	[illegible]VAR, DRAW, or RCEQ executed with nonexistent EQ.	104
No current equation.	Plot and HP Solve application status message.	609
No Picture Available	No picture is included for the selected equation.	E304
No Room in Port	Insufficient free memory in specified RAM port.	00B
No Room to Save Stack	Not enough free memory to save copy of the stack. LAST STACK is automatically disabled.	101

Messages Listed Alphabetically (continued)

Message	Meaning	# (hex)
No Room to Show Stack	Stack objects displayed by type only due to low memory condition.	131
No stat data to plot	No data stored in ΣDAT.	60F
Non-Empty Directory	Attempted to purge non-empty directory.	12B
Non-Real Result	Execution of HP Solve application, ROOT, DRAW, or $\int$ returned result other than real number or unit.	12F
Nonexistent Alarm	Alarm list did not contain alarm specified by alarm command.	D04
Nonexistent ΣDAT	Statistics command executed when ΣDAT did not exist.	602
Object Discarded	Sender sent an EOF (Z) packet with a "D" in the data field.	C0F
Object In Use	Attempted PURGE or STO into a backup object when its stored object was in use.	009
Object Not in Port	Attempted to access a nonexistent backup object or library.	00C
(OFF SCREEN)	Function value, root, extremum, or intersection was not visible in current display.	61F

Messages Listed Alphabetically (continued)

Message	Meaning	# (hex)
Out of Memory	One or more objects must be purged to continue calculator operation.	135
Overflow	Math exception: Calculation returned result greater in absolute value than MAXR.	303
Packet #	Indicates packet number during send or receive.	C10
Parity Error	Received bytes' parity bit doesn't match current parity setting.	C05
Plot Type:	Label introducing current plot type.	61D
Port Closed	Possible I/R or serial hardware failure. Run self-test.	C09
Port Not Available	Used a port command on an empty port, or one containing ROM instead of RAM.	00A
	Attempted to execute a server command that itself uses the I/O port.	
Positive Underflow	Math exception: Calculation returned positive, non-zero result less than MINR.	301
Power Lost	Calculator turned on following a power loss. Memory may have been corrupted.	006

Message	Meaning	# (hex)
Processing Command	Indicates processing of host command packet.	C11
Protocol Error	Received a packet whose length was shorter than a null packet.	C07
	Maximum packet length parameter from other machine is illegal.	
Receive Buffer Overrun	Kermit: More than 255 bytes of retries sent before HP 48 received another packet.	C04
	SRECV: Incoming data overflowed the buffer.	
Receive Error	UART overrun or framing error.	C03
Receiving	Identifies object name while receiving.	C0E
Retry #	Indicates number of retries while retrying packet exchange.	C0B
Select a model	Select statistics curve fitting model.	614
Select plot type	Select plot type.	60C
Select repeat interval	Select alarm repeat interval.	61B
Sending	Identifies object name while sending.	C0D

Message	Meaning	# (hex)
Sign Reversal	HP Solve application or ROOT unable to find point at which current equation evaluates to zero, but did find two neighboring points at which equation changed sign.	A05
Single Equation	Only one equation supplied to Multiple-Equation Solver. Use HP Solve application.	E402
Timeout	Printing to serial port: Received XOFF and timed out waiting for XON.	C02
	Kermit: Timed out waiting for packet to arrive.	
Too Few Arguments	Command required more arguments than were available on stack.	201
Too Many Unknowns	Multiple Equation Solver can't calculate a value given the current knowns. Supply another value or add an equation.	E404
Transfer Failed	Ten successive attempts to receive a good packet were unsuccessful.	C06
Unable to find root	PROOT is unable to determine all roots of the polynomial.	C001
Unable to Isolate	ISOL failed because specified name absent or contained in argument a function with no inverse.	130

Messages Listed Alphabetically (continued)

Message	Meaning	# (hex)
Undefined Local Name	Executed or recalled local name for which corresponding local variable did not exist.	003
Undefined Name	Executed or recalled global name for which corresponding variable does not exist.	204
Undefined Result	Calculation such as 0/0 generated mathematically undefined result.	304
Undefined XLIB Name	Executed an XLIB name when specified library absent.	004
Warning:	Label introducing current status message.	007
Wrong Argument Count	User-defined function evaluated with an incorrect number of parenthetical arguments.	128
x and y-axis zoom.	Identifies zoom option.	627
x axis zoom.	Identifies zoom option.	625
x axis zoom w/AUTO.	Identifies zoom option.	624
y axis zoom.	Identifies zoom option.	626
ZERO	Result returned by the HP Solve application or ROOT is a root (a point at which current equation evaluates to zero).	A04
""	Identifies no execution action when [ENTER] pressed.	61E

Messages Listed Numerically

# (hex)	Message
	General Messages
001	Insufficient Memory
002	Directory Recursion
003	Undefined Local Name
004	Undefined XLIB Name
005	Memory Clear
006	Power Lost
008	Invalid Card Data
009	Object In use
00A	Port Not Available
00B	No Room in Port
00C	Object Not in Port
101	No Room to Save Stack
102	Can't Edit Null Char.
103	Invalid User Function
104	No Current Equation
106	Invalid Syntax
124	LAST STACK Disabled
125	LAST CMD Disabled
126	HALT Not Allowed
128	Wrong Argument Count
129	Circular Reference
12A	Directory Not Allowed
12B	Non-Empty Directory
12C	Invalid Definition
12E	Invalid PPAR
12F	Non-Real Result
130	Unable to Isolate
131	No Room to Show Stack
	Out-of-Memory Prompts
135	Out of Memory
13C	Name Conflict

# (hex)	Message
	Stack Errors
201	Too Few Arguments
202	Bad Argument Type
203	Bad Argument Value
204	Undefined Name
205	LASTARG Disabled
	EquationWriter Application Messages
206	Incomplete Subexpression
207	Implicit () off
208	Implicit () on
	Floating-Point Errors
301	Positive Underflow
302	Negative Underflow
303	Overflow
304	Undefined Result
305	Infinite Result
	Array Messages
501	Invalid Dimension
502	Invalid Array Element
503	Deleting Row
504	Deleting Column
505	Inserting Row
506	Inserting Column
	Statistics Messages
601	Invalid Σ Data
602	Nonexistent ΣDAT
603	Insufficient Σ Data
604	Invalid ΣPAR
605	Invalid Σ Data LN(Neg)
606	Invalid Σ Data LN(0)

# (hex)	Message
	Plot, I/O, Time and HP Solve Application Messages
607	Invalid EQ
608	Current equation:
609	No current equation.
60A	Enter eqn, press NEW
60B	Name the equation, press ENTER
60C	Select plot type
60D	Empty catalog
60F	No stat data to plot
610	Autoscaling
614	Select a model
619	Acknowledged
61A	Enter alarm, press SET
61B	Select repeat interval
61C	I/O setup menu
61D	Plot type:
61E	""
61F	(OFF SCREEN)
620	Invalid PTYPE
621	Name the stat data, press ENTER
622	Enter value (zoom out if >1), press ENTER
623	Copied to stack
624	x axis zoom w/AUTO.
625	x axis zoom.
626	y axis zoom.
627	x and y-axis zoom.
A01	Bad Guess(es)
A02	Constant?
A03	Interrupted
A04	Zero
A05	Sign Reversal
A06	Extremum

# (hex)	Message
	Unit Management
B01	Invalid Unit
B02	Inconsistent Units
	I/O and Printing
C01	Bad Packet Block check
C02	Timeout
C03	Receive Error
C04	Receive Buffer Overrun
C05	Parity Error
C06	Transfer Failed
C07	Protocol Error
C08	Invalid Server Cmd.
C09	Port Closed
C0A	Connecting
C0B	Retry #
C0C	Awaiting Server Cmd.
C0D	Sending
C0E	Receiving
C0F	Object Discarded
C10	Packet #
C11	Processing Command
C12	Invalid IOPAR
C13	Invalid PRTPAR
C14	Low Battery
C15	Empty Stack
C17	Invalid Name

# (hex)	Message
	Time Messages
D01	Invalid Date
D02	Invalid Time
D03	Invalid Repeat
D04	Nonexistent Alarm
	Equation Library Messages
E303	Keyword Conflict
E304	No Picture Available
	Multiple-Equation Solver Messages
E401	Invalid Mpar
E402	Single Equation
E403	EQ Invalid for MINIT
E404	Too Many Unknowns
E405	All Variables Known
E406	Illegal During MROOT
	Miscellaneous Messages
70000	(user-defined message created with DOERR)

B

Table of Units

HP 48 Units

Unit (Full Name)	Value in SI Units
a (are)	100 m^2
A (ampere)	1 A
acre (acre)	4046.87260987 m^2
arcmin (minute of arc)	2.90888208666 × 10^{-4} r
arcs (second of arc)	4.8481368111 × 10^{-6} r
atm (atmosphere)	101325 kg/m·s^2
au (astronomical unit)	1.495979 × 10^{11} m
Å (Angstrom)	1 × 10^{-10} m
b (barn)	1 × 10^{-28} m^2
bar (bar)	100000 kg/m·s^2
bbl (barrel)	.158987294928 m^3
Bq (becquerel)	1 1/s
Btu (international table Btu)	1055.05585262 kg·m^2/s^2
bu (bushel)	.03523907 m^3
°C (degree Celsius)	1 K or 274.15 K
c (speed of light)	299792458 m/s
C (coulomb)	1 A·s
cal (calorie)	4.1868 kg·m^2/s^2
cd (candela)	1 cd
chain (chain)	20.1168402337 m
Ci (curie)	3.7 × 10^{10} 1/s
ct (carat)	.0002 kg
cu (US cup)	2.365882365 × 10^{-4} m^3
° (degree)	1.74532925199 × 10^{-2} r
d (day)	86400 s

HP 48 Units (continued)

Unit (Full Name)	Value in SI Units
dyn (dyne)	.00001 kg·m/s^2
erg (erg)	.0000001 kg·m^2/s^2
eV (electron volt)	1.60217733 × 10^{-19} kg·m^2/s^2
F (farad)	1 A^2·s^4/kg·m^2
°F (degrees Fahrenheit)	0.555555555556 K or 255.927777778 K
fath (fathom)	1.82880365761 m
fbm (board foot)	.002359737216 m^3
fc (footcandle)	10.7639104167 cd·sr/m^2
Fdy (faraday)	96487 A·s
fermi (fermi)	1 × 10^{-15} m
flam (footlambert)	3.42625909964 cd/m^2
ft (international foot)	.3048 m
ftUS (survey foot)	.304800609601 m
g (gram)	.001 kg
ga (standard freefall)	9.80665 m/s^2
gal (US gallon)	.003785411784 m^3
galC (Canadian gallon)	.00454609m^3
galUK (UK gallon)	.004546092 m^3
gf (gram-force)	.00980665 kg·m/s^2
grad (gradient)	1.57079632679 × 10^{-2} r
grain (grain)	.00006479891 kg
Gy (gray)	1 m^2/s^2
H (henry)	1 kg·m^2/A^2·s^2
h (Hour)	3600 s
hp (horsepower)	745.699871582 kg·m^2/s^3
Hz (hertz)	1/s
in (inch)	.0254 m
inHg (inches of mercury, 0°C)	3386.38815789 kg/m·s^2
inH2O (inches of water, 60°F)	248.84 kg/m·s^2
J (joule)	1 kg·m^2/s^2
K (kelvins)	1 K
kg (kilogram)	1 kg

HP 48 Units (continued)

Unit (Full Name)	Value in SI Units
kip (kilopound-force)	4448.22161526 kg·m/s^2
knot (nautical miles per hour)	.514444444444 m/s
kph (kilometers per hour)	.277777777778 m/s
l (liter)	.001 m^3
lam (lambert)	3183.09886184 cd/m^2
lb (avoirdupois pound)	.45359237 kg
lbf (pound-force)	4.44822161526 kg·m/s^2
lbt (troy pound)	.3732417216 kg
lm (lumen)	1 cd·sr
lx (lux)	1 cd·sr/m^2
lyr (light year)	9.46052840488 × 10^{15} m
m (meter)	1 m
μ (micron)	1 × 10^{-6} m
mho (mho)	1 A^2·s^3/kg·m^2
mi (international mile)	1609.344 m
mil (mil)	.0000254 m
min (minute)	60 s
miUS (US statute mile)	1609.34721869 m
mmHg (millimeter of mercury (torr), 0°C)	133.322368421 kg/m·s^2
mol (mole)	1 mol
mph (miles per hour)	.44704 m/s
N (newton)	1 kg·m/s^2
nmi (nautical mile)	1852 m
Ω (ohm)	1 kg·m^2/A^2·s^3
oz (ounce)	.028349523125 kg
ozfl (US fluid ounce)	2.95735295625 × 10^{-5} m^3
ozt (troy ounce)	.0311034768 kg
ozUK (UK fluid ounce)	2.8413075 × 10^{-5} m^3
P (poise)	.1 kg/m·s
Pa (pascal)	1 kg/m·s^2
pc (parsec)	3.08567818585 × 10^{16} m
pdl (poundal)	.138254954376 kg·m/s^2
ph (phot)	10000 cd·sr/m^2
pk (peck)	.0088097675 m^3

HP 48 Units (continued)

Unit (Full Name)	Value in SI Units
psi (pounds per square inch)	6894.75729317 kg/m·s^2
pt (pint)	.000473176473 m^3
qt (quart)	.000946352946 m^3
r (radian)	1 r
R (roentgen)	.000258 A·s/kg
°R (degrees Rankine)	0.555555555556 K
rad (rad)	.01 m^2/s^2
rd (rod)	5.02921005842 m
rem (rem)	.01 m^2/s^2
s (second)	1 s
S (siemens)	1 A^2·s^3/kg·m^2
sb (stilb)	10000 cd/m^2
slug (slug)	14.5939029372 kg
sr (steradian)	1 sr
st (stere)	1 m^3
St (stokes)	.0001 m^2/s
Sv (sievert)	1 m^2/s^2
t (metric ton)	1000 kg
T (tesla)	1 kg/A·s^2
tbsp (tablespoon)	1.47867647813 × 10^{-5} m^3
therm (EEC therm)	105506000 kg·m^2/s^2
ton (short ton)	907.18474 kg
tonUK (long ton (UK))	1016.0469088 kg
torr (torr (mmHg))	133.322368421 kg/m·s^2
tsp (teaspoon)	4.92892159375 × 10^{-6} m^3
u (unified atomic mass)	1.6605402 × 10^{-27} kg
V (volt)	1 kg·m^2/A·s^3
W (watt)	1 kg·m^2/s^3
Wb (weber)	1 kg·m^2/A·s^2
yd (international yard)	.9144 m
yr (year)	31556925.9747 s

C

System Flags

This appendix lists the HP 48 system flags. You can set, clear, and test all flags. The default state of the flags is *clear*—except for the Binary Integer Wordsize flags (flags −5 through −10).

System Flags

Flag	Description
−1	Principal Solution. *Clear*: QUAD and ISOL return a result representing all possible solutions. *Set*: QUAD and ISOL return only the principal solution.
−2	Symbolic Constants. *Clear*: Symbolic constants (e, i, π, MAXR, and MINR) retain their symbolic form when evaluated, unless the Numerical Results flag −3 is set. *Set*: Symbolic constants evaluate to numbers, regardless of the state of the Numerical Results flag −3.
−3	Numerical Results. *Clear*: Functions with symbolic arguments, including symbolic constants, evaluate to symbolic results. *Set*: Functions with symbolic arguments, including symbolic constants, evaluate to numbers.
−4	Not used.
−5 thru −10	Binary Integer Wordsize. Combined states of flags −5 through −10 set the wordsize from 1 to 64 bits.

System Flags (continued)

Flag	Description
−11 and −12	Binary Integer Base. HEX: −11 *set*, −12 *set*. DEC. −11 *clear*, −12 *clear*. OCT: −11 *set*, −12 *clear*. BIN: −11 *clear*, −12 *set*.
−13	Not used.
−14	Financial Payment Mode. *Clear:* TVM calculations assume end-of-period payments. *Set:* TVM calculations assume beginning-of-period payments.
−15 and −16	Rectangular: −16 *clear*. Polar/Cylindrical: −15 *clear*, −16 *set*. Polar/Spherical: −15 *set*, −16 *set*.
−17 and −18	Degrees: −17 *clear*, −18 *clear*. Radians: −17 *set*. Grads: −17 *clear*, −18 *set*.
−19	*Clear*:→V2 and creates a 2-dimensional vector from 2 real numbers. *Set*:→V2 and creates a complex number from 2 real numbers.
−20	Underflow Exception. *Clear*: Underflow exception returns 0, sets flag −23 or −24. *Set*: Underflow exception treated as an error.
−21	Overflow Exception. *Clear*: Overflow exception returns ±9.99999999999E499 and sets flag −25. *Set*: Overflow exception treated as an error.
−22	Infinite Result Exception. *Clear*: Infinite result exception treated as an error. *Set*: Infinite result exception returns ±9.99999999999E499 and sets flag −26.
−23 −24 −25 −26	Negative Underflow Indicator. Positive Underflow Indicator. Overflow Indicator. Infinite Result Indicator. When an exception occurs, corresponding flag (−23 through −26) is set only if the exception is *not* treated as an error.

System Flags (continued)

Flag	Description
−27	Display of symbolic complex numbers. *Clear*: Displays symbolic complex numbers in coordinate form (i.e. '(x,y)'). *Set*: Displays symbolic complex numbers using 'i' (i.e. 'x+y*i').
−28	Simultaneous Plotting of Multiple Functions. *Clear*: Multiple equations are plotted serially. *Set*: Multiple equations are plotted simultaneously.
−29	Draw Axes. *Clear*: Axes are drawn for two-dimensional and statistical plots. *Set*: Axes are not drawn for two-dimensional and statistical plots.
−30	Not used.
−31	Curve Filling. *Clear*: Curve filling between plotted points enabled. *Set*: Curve filling between plotted points suppressed.
−32	Graphics Cursor. *Clear*: Graphics cursor always dark. *Set*: Graphics cursor dark on light background and light on dark background.
−33	I/O Device. *Clear*: I/O directed to serial port. *Set*: I/O directed to IR port.
−34	Printing Device. *Clear*: Printer output directed to IR printer. *Set*: Printer output directed to serial port if flag −33 is clear.
−35	I/O Data Format. *Clear*: Objects transmitted in ASCII form. *Set*: Objects transmitted in binary (memory image) form.

System Flags (continued)

Flag	Description
−36	I/O Receive Overwrite. *Clear*: If file name received by HP 48 matches existing HP 48 variable name, new variable name with number extension is created to prevent overwrite. *Set*: If file name received by HP 48 matches existing HP 48 variable name, existing variable is overwritten.
−37	Double-Spaced Printing. *Clear*: Single-spaced printing. *Set*: Double-spaced printing.
−38	Line Feed. *Clear*: Linefeed added at end of each print line. *Set*: No linefeed added at end of each print line.
−39	I/O Messages. *Clear*: I/O messages displayed. *Set*: I/O messages suppressed.
−40	Clock Display. *Clear*: Clock displayed only when TIME menu selected. *Set*: Ticking clock displayed at all times.
−41	Clock Format. *Clear*: 12-hour clock. *Set*: 24-hour clock.
−42	Date Format. *Clear*: MM/DD/YY (month/day/year) format. *Set*: DD.MM.YY (day.month.year) format.
−43	Repeat Alarms Not Rescheduled. *Clear*: Unacknowledged repeat appointment alarms automatically rescheduled. *Set*: Unacknowledged repeat appointment alarms not rescheduled.
−44	Acknowledged Alarms Saved. *Clear*: Acknowledged appointment alarms deleted from alarm list. *Set*: Acknowledged appointment alarms saved in alarm list.

System Flags (continued)

Flag	Description
−45 thru −48	Number of Decimal Digits. Combined states of flags −45 through −48 sets number of decimal digits in Fix, Scientific, and Engineering modes.
−49 and −50	Number Display Format. Standard: −49 *clear*, −50 *clear*. Fix: −49 *set*, −50 *clear*. Scientific: −49 *clear*, −50 *set*. Engineering: −49 *set*, −50 *set*.
−51	Fraction Mark. *Clear*: Fraction mark is . (period). *Set*: Fraction mark is , (comma).
−52	Single-Line Display. *Clear*: Display gives preference to object in level 1, using up to four lines of stack display. *Set*: Display of object in level 1 restricted to one line.
−53	Precedence. *Clear*: Certain parentheses in algebraic expressions suppressed to improve legibility. *Set*: All parentheses in algebraic expressions displayed.
−54	Tiny Array Elements. *Clear*: Singular values computed by RANK (and other commands that compute the rank of a matrix) that are more than 1×10^{-14} times smaller than the largest computed singular value in the matrix are converted to zero. Automatic rounding for DET is enabled. *Set*: Small computed singular values (see above) not converted. Automatic rounding for DET is disabled.
−55	Last Arguments. *Clear*: Command arguments saved. *Set*: Command arguments not saved.
−56	Error Beep. *Clear*: Error and BEEP-command beeps enabled. *Set*: Error and BEEP-command beeps suppressed.

System Flags (continued)

Flag	Description
−57	Alarm Beep. *Clear*: Alarm beep enabled. *Set*: Alarm beep suppressed.
−58	Verbose Messages. *Clear*: Parameter variable data automatically displayed. *Set*: Automatic display of parameter variable data is suppressed.
−59	Fast Browser Display. *Clear*: Variable Browser shows variable names and contents. *Set*: Variable Browser shows variable names only.
−60	Alpha Lock. *Clear*: Single-Alpha activated by pressing [α] once. Alpha lock activated by pressing [α] twice. *Set*: Alpha lock activated by pressing [α] once. (Single-Alpha not available.)
−61	User-Mode Lock. *Clear*: 1-User mode activated by pressing [←][USER] once. User mode activated by pressing [←][USER] twice. *Set*: User mode activated by pressing [←][USER] once. (1-User mode not available.)
−62	User Mode. *Clear*: User mode not active. *Set*: User mode active.
−63	Vectored [ENTER]. *Clear*: [ENTER] evaluates command line. *Set*: User-defined [ENTER] activated.
−64	Index Wrap Indicator. *Clear*: Last execution of GETI or PUTI did not increment index to first element. *Set*: Last execution of GETI or PUTI did increment index to first element.

D

Reserved Variables

The HP 48 uses the following *reserved variables*. These have specific purposes, and their names are used as implicit arguments for certain commands. Avoid using these variables' names for other purposes, or you may interfere with the execution of the commands that use these variables.

You can change some of the values in these variables with programmable commands, while others require you to store new values into the appropriate place.

Reserved Variable	What It Contains	Used By
ALRMDAT	Alarm parameters.	TIME ALRM operations
CST	List defining the CST (custom) menu.	MENU, [CST]
"*der*"-names	User-defined derivative.	∂
EQ	Current equation.	ROOT, DRAW
EXPR	Current expression.	SYMBOLIC
IOPAR	I/O parameters.	I/O commands
MHpar	Minehunt game status.	MINEHUNT
Mpar	Multiple-Equation Solver equations.	EQ LIB
n1, *n2*, ...	Arbitrary integers.	ISOL, QUAD
Nmines	Minehunt game data.	MINEHUNT

Reserved Variable	What It Contains	Used By
PPAR	Plotting parameters.	DRAW
PRTPAR	Printing parameters.	PRINT commands
s1, s2, ...	Arbitrary signs.	ISOL, QUAD
VPAR	Viewing parameters.	DRAW
ZPAR	Plot zoom factors.	DRAW
ΣDAT	Statistical data.	Statistics application, DRAW
ΣPAR	Statistical parameters.	Statistics application, DRAW

Contents of the Reserved Variables

Most reserved variables (except *ALRMDAT*, *IOPAR* and *PRTPAR*) can be stored with different contents in different directories. This allows you, for example, to save several sets of statistical data in different directories.

ALRMDAT

ALRMDAT does not reside in a particular directory. You cannot access the variable itself, but you can access its data from any directory using the RCLALARM and STOALARM commands, or through the Alarm Catalog.

ALRMDAT contains a list of these alarm parameters:

Parameter (Command)	Description	Default Value
date (→DATE)	A real number specifying the date of the alarm: *MM.DDYYYY* (or *DD.MMYYYY* if flag −42 is set). If *YYYY* is not included, the current year is used.	Current date.
time (→TIME)	A real number specifying the time of the alarm: *HH.MMSS*.	00.0000
action	A string or object: ■ a string creates an *appointment alarm*, which beeps and displays the string ■ any other object creates a *control alarm*, which executes the object	Empty string (appointment alarm).
repeat	A real number specifying the interval between automatic recurrences of the alarm, given in ticks (a tick is $^1/_{8192}$ of a second).	0

Parameters without commands can be modified with a program by storing new values in the list contained in *ALRMDAT* (use the PUT command).

CST

CST contains a list (or a name specifying a list) of the objects that define the CST (*custom*) menu. Objects in the custom menu behave as do objects in built-in menus. For example:

- Names behave like the VAR menu keys. Thus, if *ABC* is a variable name, A..C evaluates *ABC*, → A..C recalls its contents, and ← A..C stores new contents in *ABC*.
- The menu label for the name of a directory has a bar over the left side of the label; pressing the menu key switches to that directory.
- Unit objects act like unit catalog entries (and have left-shifted conversion capabilities, for example).

- String keys echo the string.
- You can include backup objects in the list defining a custom menu by tagging the name of the backup object with its port location (0 through 33).

You can specify menu labels and key actions independently by replacing a single object within the custom-menu list with a list of the form « "*label-object*" *action-object* ». (See "Customizing Menus" and "Enhancing Custom Menus" in chapter 30 of the *HP 48 User's Guide* for more information.)

To provide different shifted actions for custom menu keys, *action-object* can be a list containing three action objects in this order:

- The unshifted action (required if you want to specify the shifted actions).
- The left-shifted action.
- The right-shifted action.

See "Enhancing Custom Menus" in chapter 30 of the *HP 48 User's Guide*.

"der-" Names

If ∂ is applied to a function for which there is no built-in derivative, it returns a new function whose name is "der" followed by the original function name. These "der"-function names are reserved variable names.

For an example, refer to "Creating User-Defined Derivatives" in chapter 20 of the *HP 48 User's Guide*.

EQ

EQ contains the current equation or the name of the variable containing the current equation.

EQ supplies the equation for ROOT, as well as for the plotting command DRAW when the plot type is FUNCTION, CONIC, POLAR, PARAMETER, TRUTH, or DIFFEQ. (ΣDAT supplies the information when the plot type is HISTOGRAM, BAR, or SCATTER.)

The object in *EQ* can be an algebraic object, a number, a name, or a program. How DRAW interprets *EQ* depends on the plot type.

For graphics use, *EQ* can also be a list of equations or other objects. If *EQ* contains a list, then DRAW treats each object in turn as the current equation, and plots them successively. However, ROOT in the HP Solve application *cannot* solve an *EQ* containing a list.

To alter the contents of *EQ*, use the command STEQ.

EXPR

EXPR contains the current algebraic expression (or the name of the variable containing the current expression) used by the SYMBOLIC application and its associated commands. The object in *EQ* must be an algebraic or a name.

IOPAR

IOPAR is a variable in the *HOME* directory that contains a list of the I/O parameters needed for a communications link with a computer. It is created the first time you transfer data or open the serial port (OPENIO), and is automatically updated whenever you change the I/O settings. All *IOPAR* parameters are integers.

Parameter (Command)	Description	Default Value
baud (BAUD)	The baud rate: 1200, 2400, 4800, or 9600.	9600
parity (PARITY)	The parity used: 0=none, 1=odd, 2=even, 3=mark, 4=space. The value can be positive or negative: a positive parity is used upon both transmit and receive; a negative parity is used only upon transmit.	0

Parameter (Command)	Description	Default Value
receive pacing	Controls whether receive pacing is used: a nonzero real value enables pacing, while zero disables it. Receive pacing sends an XOFF signal when the receive buffer is almost full, and sends an XON signal when it can take more data again. Pacing is not used for Kermit I/O, but is used for other serial I/O transfers.	0 (no pacing)
transmit pacing	Controls whether transmit pacing is used: a nonzero real value enables pacing, while zero disables it. Transmit pacing stops transmission upon receipt of XOFF, and resumes transmission upon receipt of XON. Pacing is not used for Kermit I/O, but is used for other serial I/O transfers.	0 (no pacing)
checksum (CKSM)	Error-detection scheme requested when initiating SEND: ■ 1=1-digit arithmetic checksum ■ 2=2-digit arithmetic checksum ■ 3=3-digit cyclic redundancy check.	3
translation code (TRANSIO)	Controls which characters are translated: ■ 0=none ■ 1=translate character 10 (line feed only) to/from characters 10 and 13 (line feed and carriage return) ■ 2=translate characters with numbers 128 through 159 (80–9F hex) ■ 3=translate characters with numbers 128 through 255.	1

Parameters without commands can be modified with a program by storing new values in the list contained in *IOPAR* (use the PUT command), or by editing *IOPAR* directly.

MHpar

MHpar stores the status of an interrupted Minehunt game. *MHpar* is created when you exit Minehunt by pressing (STO). If *MHpar* still exists when you restart Minehunt, the interrupted game resumes and *MHpar* is purged.

Mpar

Mpar is created when you use the Equation Library's Multiple-Equation Solver, and it stores the set of equations you're using.

When the Equation Library starts the Multiple-Equation Solver, it first stores a list of the equation set in *EQ*, and stores the equation set, a list of variables, and additional information in *Mpar*. *Mpar* is then used to set up the Solver menu for the current equation set.

Mpar is structured as library data dedicated to the Multiple Equation Solver application. This means that although you can view and edit *Mpar* directly, you can edit it only indirectly by executing commands that modify it.

You can also use the MINIT command ((←) (EQ LIB) NEXT MINIT) to create *MHpar* from a set of equations on the stack. See "Defining a Set of Equations" in chapter 25 of the *HP 48 User's Guide.*

n1, n2, . . .

The ISOL and QUAD commands return *general* solutions (as opposed to *principal* solutions) for operations. A general solution contains variables for arbitrary integers or arbitrary signs, or both.

The variable *n1* represents an arbitrary integer 0, ±1, ±2, etc. Additional arbitrary integers are represented by *n2*, *n3*, etc.

If flag −1 is set, then ISOL and QUAD return principal solutions, in which case the arbitrary integer is always zero.

Nmines

Nmines is a variable you create in the current directory to control the number of mines used in the Minehunt game. *Nmines* contains an integer in the range 1 to 64; if *Nmines* is negative, the mines are visible during the game.

PPAR

PPAR is a variable in the current directory. It contains a list of plotting parameters used by the DRAW command for all mathematical and statistical plots, by AUTO for autoscaling, and by the interactive (nonprogrammable) graphics operations.

Parameter (Command)	Description	Default Value
(x_{min}, y_{min}) (XRNG, YRNG)	A complex number specifying the lower left corner of *PICT* (the lower left corner of the display range).	(−6.50, −3.1)
(x_{max}, y_{max}) (XRNG, YRNG)	A complex number specifying the upper right corner of *PICT* (the upper right corner of the display range).	(6.5, 3.2)
indep (INDEP)	A name specifying the independent variable, or a list containing that name and two numbers that specify the minimum and maximum values for the independent variable (the plotting range).	*X*
res (RES)	Resolution. A real number specifying the interval between values of the independent variable. For plots of equations, this determines the plotting interval along the x-axis. A binary number specifies the *pixel* resolution (how many columns of pixels between points). An integer specifies the resolution in *user* units (how many user units between points). Resolution for statistical plots is different (see below).	0
axes (AXES)	A complex number specifying the user-unit coordinates of the plot origin, or a list containing the following: ■ the complex number specifying the origin ■ a real number, binary integer, or list containing two real numbers or binary integers specifying the tick-mark annotation (see ATICK) ■ two strings specifying labels for the horizontal and vertical axes	(0, 0)
ptype (BAR, etc.)	A command name specifying the plot type (BAR, CONIC, DIFFEQ, FUNCTION, GRIDMAP, HISTOGRAM, PARAMETRIC, PARSURFACE, PCONTOUR, POLAR, SCATTER, SLOPEFIELD, TRUTH, WIREFRAME, or YSLICE).	FUNCTION

Parameter (Command)	Description	Default Value
depend (DEPND)	A name specifying the dependent variable, or a list containing the name and two numbers that specify vertical plotting range. For DIFFEQ, the second element of the list may also be a real vector that represents the initial value.	*Y*

Parameters without commands can be modified with a program by storing new values in the list contained in *PPAR* (use the PUT command).

The RESET operation (◀ PLOT PPAR RESET) resets the *PPAR* parameters (except *ptype*) to their default values, and erases *PICT*.

Note that *res* behaves differently for the statistical plot types BAR, HISTOGRAM, and SCATTER than for other plot types. For BAR, *res* specifies bar width; for HISTOGRAM, *res* specifies bin width; *res* does not affect SCATTER.

PRTPAR

PRTPAR is a variable in the *HOME* directory that contains a list of printing parameters. It is created automatically the first time you use a printing command.

Parameter (Command)	Description	Default Value
delay time (DELAY)	A real number, in the range 0 to 6.9, specifying the number of seconds the HP 48 waits between sending lines. This should be at least as long as the time required to print the longest line. If the delay is too short for the printer, you will lose data. The delay setting also affects serial printing if transmit-pacing (in *IOPAR*) is not being used.	1.8
remap (OLDPRT stores the character-remapping string for the HP 82240A Infrared Printer)	A string defining the current remapping of the extended character set for printing. The string can contain as many characters as you want to remap, with the first character being the new character 128, the second being the new charcter 129, etc. (Any character number that exceeds the string length will not be remapped.) See the example below.	Empty string.
line length	A real number specifying the number of characters in a line for serial printing. This does *not* affect infrared printing.	80
line termination	A string specifying the line-termination method for serial printing. This does *not* affect infrared printing.	Control characters 13 (carriage return) and 10 (line feed).

Parameters without commands can be modified with a program by storing new values in the list contained in *PRTPAR* (use the PUT command).

A change in a parameter is effective immediately, *except* when printing the display using the simultaneous keystrokes ON I/O (because this does not use *PRTPAR*). This printing method is affected only by the delay parameter, a change in which will not affect ON I/O until after the next printing command has been executed. To use a new delay time with ON I/O immediately, use the DELAY command.

Example: If the remapping string were "ABCDEFGH" and the character to be printed had value 131, then the character actually printed would be "D", since 131−128=3 and "A" has the value zero. A character code of 136 or greater would not be remapped since 136−128=8, which exceeds the length of the string.

s1, s2, . . .

The ISOL and QUAD commands return *general* solutions (as opposed to *principal* solutions) for operations. A general solution contains variables for arbitrary integers or arbitrary signs or both.

The variable *s1* represents an arbitrary + or − sign. Additional arbitrary signs are represented by *s2*, *s3*, etc.

If flag −1 is set, then ISOL and QUAD return principal solutions, in which case the arbitrary sign is always +1.

VPAR

VPAR is a variable in the current directory. It contains a list of parameters used by the 3D plot types. The main data structure stored in VPAR describes the "view volume," the abstract three-dimensional region in which the function is plotted.

Parameter (Command)	Description	Default Value
(x_{left}, x_{right}) (XVOL)	Real numbers that specify the width of the view volume.	(−1, 1)
(y_{far}, y_{near}) (YVOL)	Real numbers that specify the depth of the view volume.	(−1, 1)
(z_{low}, z_{high}) (ZVOL)	Real numbers that specify the height of the view volume.	(−1, 1)
$(x_{eye}, y_{eye}, z_{eye})$ (EYEPT)	Real numbers that specify the point in space from which the plot is viewed.	(0, −3, 0)
(x_{step}, y_{step}) (NUMX,NUMY)	Real numbers that specify the increments between of x-coordinates and y-coordinates plotted. The increments are equal to the range for the axes divided by the number of steps. Used instead of (or in combination with) *res*.	(10, 8)
(xx_{left}, xx_{right}) (XXRNG)	Real numbers that specify the width of the input plane (domain). Used by GRIDMAP and PARSURFACE.	(−1, 1)
(yy_{far}, yy_{near}) (YYRNG)	Real numbers that specify the depth of the input plane (domain). Used by GRIDMAP and PARSURFACE.	(−1, 1)

Parameters without commands can be modified programmatically by storing new values in the list contained in *VPAR* (use the PUT command).

The RESET operation (← PLOT NXT 3D VPAR NXT RESET) resets the *VPAR* parameters to their default values.

ZPAR

ZPAR is a variable in the current directory. It contains a list of zooming parameters used by the DRAW command for all 2-D mathematical and statistical plots.

Parameter (Command)	Description	Default Value
h-factor	Real number that specifies the horizontal zoom factor.	4
v-factor	Real number that specifies the vertical zoom factor.	4
recenter flag	0 or 1 depending on whether the recenter at crosshairs option was selected in the set zoom factors input form.	0
{ *list* }	An empty list, or a copy of the last *PPAR*.	

Use the set zoom factors input form ( ) to modify *ZPAR*.

ΣDAT

ΣDAT is a variable in the current directory that contains either the current statistical matrix or the name of the variable containing this matrix. This matrix contains the data used by the Statistics applications.

Statistical Matrix for Variables 1 to m

var_1	var_2	...	var_m
x_{11}	x_{21}	...	x_{m1}
x_{12}	x_{22}	...	x_{m2}
⋮			
x_{1n}	x_{2n}	...	x_{mn}

You can designate a new current statistical matrix by entering new data, editing the current data, or selecting another matrix.

The command CLΣ clears the current statistical matrix.

ΣPAR

ΣPAR is a variable in the current directory that contains either the current statistical parameter list or the name of the variable containing this list.

Parameter (Command)	Description	Default Value
$column_{indep}$ (XCOL)	A real number specifying the independent-variable's column number.	1
$column_{dep}$ (YCOL)	A real number specifying the dependent-variable's column number.	2
intercept (LR)	A real number specifying the coefficient of intercept as determined by the current regression.	0
slope (LR)	A real number specifying the coefficient of slope as determined by the current regression.	0
model (LINFIT, etc.)	A command specifying the regression model (LINFIT, EXPFIT, PWRFIT, or LOGFIT).	LINFIT

E

New Commands

In the following tables, new commands (commands that were not available on a standard HP 48S series calculator) are arranged alphabetically and followed by brief descriptions. All of these commands are described in chapter 3.

New Commands Listed Alphabetically

Command	Brief Description
ADD	Adds list elements.
AMORT	Amortizes a loan or investment based upon the current amortization settings.
ANIMATE	Displays graphic objects in sequence.
ATICK	Sets the axes tick-mark annotation in the reserved variable *PPAR*.
CHOOSE	Creates a user-defined choose box.
CLTEACH	Removes the EXAMPLES subdirectory and its contents from the HOME directory.
COL+	Inserts an array (vector or matrix) into a matrix.
COL−	Deletes a column from of a matrix.
COL→	Transforms a series of column vectors and a column count into a matrix, or transforms a sequence of numbers and an element count into a vector.
→COL	Transforms a matrix into a series of column vectors, or transforms a vector into its elements.

New Commands Listed Alphabetically (continued)

Command	Brief Description
COND	Returns the 1-norm (column norm) condition number of a square matrix.
CONLIB	Opens the Constants Library catalog.
CONST	Returns the value of a constant.
CSWP	Swaps columns in a matrix.
CYLIN	Sets Cylindrical coordinate mode.
DARCY	Calculates the Darcy friction factor of certain fluid flows.
DIAG→	Takes an array and a specified dimension and returns a matrix whose main diagonal elements are the elements of the array.
→DIAG	Returns a vector that contains the major diagonal elements of a matrix.
DIFFEQ	Specifies differential equations as the plot type.
DOLIST	Applies commands, programs, or user-defined functions to lists.
DOSUBS	Applies a program or command to groups of elements in a list.
EGV	Computes the eigenvalues and right eigenvectors for a square matrix.
EGVL	Computes the eigenvalues of a square matrix.
ENDSUB	Provides a way to access the total number of sublists used while executing a program or command using DOSUBS.
EQNLIB	Starts the Equation Library application.
EYEPT	Specifies the coordinates of the eye point in a perspective plot.

New Commands Listed Alphabetically (continued)

Command	Brief Description
F0λ	Returns the fraction of total black-body emissive power.
FANNING	Calculates the Fanning friction factor of certain fluid flows.
FFT	Computes the one- or two-dimensional discrete Fourier transform of an array.
FREE1	Frees the previously merged RAM in port 1.
GRIDMAP	Specifies grid mapping as the plot type.
HEAD	Returns the first element of a list or string.
IFFT	Computes the one- or two-dimensional inverse discrete Fourier transform of a vector or matrix.
INFORM	Creates a user-defined dialog box (Input Form).
LIBEVAL	Evaluates unnamed library objects by their memory addresses.
LININ	Tests whether an algebraic is structurally linear for a given variable.
ΣLIST	Returns the sum of the elements in a list.
ΠLIST	Returns the product of the elements in a list.
ΔLIST	Returns the set of first differences.
LQ	Returns the LQ factorization of an $n \times m$ matrix.
LSQ	Returns the minimum norm least squares solution to any system of linear equations.
LU	Returns the LU decomposition of a square matrix.
MCALC	Designates a variable as a calculated value (not user-defined).
MERGE1	Merges the RAM from the card in port 1 with the rest of main user memory.

New Commands Listed Alphabetically (continued)

Command	Brief Description
MINEHUNT	Starts the MINEHUNT game.
MINIT	Creates the reserved variable *Mpar*.
MITM	Changes multiple equation menu titles and order.
MROOT	Solves for one or more variables.
MSGBOX	Creates a user-defined message box.
MSOLVR	Gets the Multiple-Equation Solver variable menu for the set of equations defined by *Mpar*.
MUSER	Designates a variable as user-defined.
NDIST	Returns the normal probability distribution.
NOVAL	Place holder for reset and initial values in user-defined dialog boxes.
NSUB	Provides a way to access the current sublist number during an iteration of a program or command applied using DOSUBS.
NUMX	Sets the number of x-steps for each y-step in 3D perspective plots.
NUMY	Sets the number of y-steps across the view volume in 3D perspective plots.
PARSURFACE	Specifies 3D parameterized surface grip mapping as the plot type.
PCOEF	Returns the coefficients of a monic polynomial.
PCOV	Calculates population covariance.
PCONTOUR	Specifies pseudo-contour as the plot type.
PEVAL	Evaluates an n-degree polynomial at x.
PINIT	Initializes the plug-in card ports.
PROOT	Returns all roots of an n-degree polynomial having real or complex coefficients.

New Commands Listed Alphabetically (continued)

Command	Brief Description
PSDEV	Calculates population standard deviation.
PVAR	Calculates population variance.
QR	Returns the QR factorization of an $n \times m$ matrix.
RANK	Returns the rank of a rectangular matrix.
RANM	Returns a matrix of random integers.
RCI	Multiplies a row of a matrix by a constant.
RCIJ	Multiplies a row of a matrix by a constant, and then adds the product to another row of the matrix.
RECT	Sets Rectangular coordinate mode.
REVLIST	Reverses the order of the elements in a list.
RKF	Computes the solution to an initial value problem for a differential equation, using the Runge-Kutta-Fehlberg method.
RKFERR	Returns the absolute error estimate for a given step when solving an initial value problem for a differential equation (using RKF method).
RKFSTEP	Computes the next solution step to an initial value problem for a differential equation.
ROW+	Inserts an array into a matrix.
ROW−	Deletes a row from a matrix.
RREF	Converts a rectangular matrix to reduced row echelon form.
RRK	Computes the solution to an initial value problem for a differential equation with known partial derivatives.
RRKSTEP	Computes the next solution step to an initial value problem for a differential equation, and displays the method used to arrive at that result.

New Commands Listed Alphabetically (continued)

Command	Brief Description
RSBERR	Returns an error estimate for a given step when solving an initial values problem for a differential equation (using the Rosenbrock method).
RSWP	Swaps rows in a matrix.
SCHUR	Returns the Schur decomposition of a square matrix.
SEQ	Returns a list of results generated by repeatedly executing an object on a specified range of elements.
SIDENS	Calculates the intrinsic density of silicon as a function of temperature.
SLOPEFIELD	Specifies slopefield as the plot type.
SNRM	Returns the spectral norm of an array.
SOLVEQN	Starts the solver for a specified set of equations.
SORT	Sorts the elements in a list in ascending order.
SPHERE	Sets Spherical coordinate mode.
SRAD	Returns the spectral radius of a square matrix.
STREAM	Applies an object to every element in a list.
SVD	Returns the singular value decomposition of an $n \times m$ matrix.
SVL	Returns the singular values of an $m \times n$ matrix.
TAIL	Returns all but the first element of a list or string.
TDELTA	Calculates a temperature change.
TEACH	Creates an EXAMPLES subdirectory in the HOME directory and loads HP 48 programming, graphing, and solver examples from ROM into it.
TINC	Calculates a temperature increment.
TRACE	Returns the trace of a square matrix.

New Commands Listed Alphabetically (continued)

Command	Brief Description
TVM	Start the TVM solver.
TVMBEG	Specifies that payments are made at the beginning of compounding periods.
TVMEND	Specifies that payments are made at the end of compounding periods.
TVMROOT	Solves for the specified TVM variable using values from the remaining TVM variables.
VERSION	Returns the software version and copyright message.
WIREFRAME	Specifies wireframe as the plot type.
XRECV	Receives an object via XModem.
XSEND	Sends an object via XModem.
XVOL	Sets the width of the view volume in the reserved variable *VPAR*.
XXRNG	Specifies the x range of an input plane (domain) for GRIDMAP and PARSURFACE plots.
YSLICE	Specifies y-slice cross sections as the plot type.
YVOL	Sets the depth of the view volume in the reserved variable *VPAR*.
YYRNG	Specifies the y range of an input plane (domain) for GRIDMAP and PARSURFACE plots.
ZFACTOR	Calculates the the gas compressibility correction factor for nonideal behavior of a hydrocarbon gas.
ZVOL	Sets the height of the view volume in the reserved variable *VPAR*.

F

Technical Reference

This appendix contains the following information:

- Object sizes.
- Mathematical simplification rules used by the HP 48.
- Symbolic differentiation patterns used by the HP 48.
- The EquationWriter's expansion rules.
- References used as sources for constants and equations in the HP 48 (other than those in the Equation Library).

Object Sizes

The following table lists object types and their size in bytes. (Note that characters in names, strings, and tags use 1 byte each.)

Object Size

Object Type	Size (bytes)
Algebraic	5 + size of included objects
Backup Object	12 + number of name characters + size of included object
Binary Integer	13
Command	2.5
Complex matrix	15 + 16 × number of elements
Complex number	18.5
Complex vector	12.5 + 16 × number of elements
Directory	6.5 + size of included variables
Graphics Object	10 + number of rows × CEIL(columns/8)
List	5 + size of included objects
Matrix	15 + 8 × number of elements
Program	12.5 + size of included objects
Quoted global or local name	8.5 + number of characters
Real number	10.5
String	5 + number of characters
Tagged Object	3.5 + number of tag characters + size of untagged object
Unit Object	7.5 +
real magnitude	2.5 or 10.5
each prefix	6
each unit name	5 + number of characters
each ×, ^, or /	2.5
each exponent	2.5 or 10.5
Unquoted global or local name	3.5 + number of characters
Vector	12.5 + 8 × number of elements
XLIB name	5.5

Automatic Simplification Rules

The following tables list the automatic simplification rules for the HP 48.

Addition and Subtraction

Object	Simplified
$x-x$	0
$0+x$	x
$(0,0)+x$	x
$0-x$	NEG(x)
$(0,0)-x$	NEG(x)
$x+0$	x

Object	Simplified
$x+(0,0)$	x
$x+-\mathrm{p}$	$x-\mathrm{p}$
$x-0$	x
$x-(0,0)$	x
$x--\mathrm{p}$	$x+\mathrm{p}$

Multiplication and Division

Object	Simplified
INV(i)	$-i$
$y\times$INV(x)	y/x
$y/$INV(x)	$y\times x$
$0\times x$	0
$(0,0)\times x$	$(0,0)$
$i\times i$	-1
$1\times x$	x
$(1,0)\times x$	x
$(-1)\times x$	NEG(x)
$(-1,0)\times x$	NEG(x)
$x\times 0$	0
$x\times(0,0)$	$(0,0)$
$x\times 1$	x

Object	Simplified
$x\times(1,0)$	x
$x\times(-1)$	NEG(x)
$x\times(-1,0)$	NEG(x)
$x/1$	x
$x/(1,0)$	x
$x/(-1)$	NEG(x)
$x/(-1,0)$	NEG(x)
$0/x$	0
$(0,0)/x$	$(0,0)$
$1/x$	INV(x)
$(1,0)x$	INV(x)
$(-1)/x$	$-$INV(x)
$(-1,0)/x$	$-$INV(x)

Powers

Object	Simplified	Object	Simplified
1^x	1	$x^{(1,0)}$	x
$(1,0)^x$	(1,0)	$x^{(-1)}$	INV(x)
SQ($\sqrt{(x)}$)	x	$x^{(-1,0)}$	INV(x)
SQ(y^x)	$y^{(2\times x)}$	$(\sqrt{x})^2$	x
SQ(i)	-1	$(\sqrt{x})^{(2,0)}$	x
x^0	1	i^2	-1
$x^{(0,0)}$	(1,0)	$i^{(2,0)}$	(−1,0)
$x^{\perp}$	x		

Parts

Object	Simplified	Object	Simplified
ABS(ABS(x))	ABS(x)	MIN(x,x)	x
ABS(NEG(x))	ABS(x)	MOD(0,x)	0
CONJ(CONJ(x))	x	MOD(x,x)	0
CONJ(IM(x))	IM(x)	MOD(x,0)	x
CONJ(RE(x))	RE(x)	x MOD y MOD y	x MOD y
CONJ(i)	$-i$	RE(CONJ(x))	RE(x)
IM(CONJ(x))	−IM(x)	RE(IM(x))	IM(x)
IM(IM(x))	0	RE(RE(x))	RE(x)
IM(RE(x))	0	RE(π)	π
IM(p)	0	RE(i)	0
IM(i)	1	SIGN(SIGN(x))	SIGN(x)
MAX(x,x)	x		

Symbolic Integration Patterns

This table lists the symbolic integration patterns used by the HP 48. These are the integrands that the HP 48 can integrate symbolically.

ϕ is a linear function of the variable of integration. The antiderivatives should be divided by the first-order coefficient in ϕ to reduce the expression to its simplest form. Also, patterns beginning with 1/ match INV: for example, $1/\phi$ is the same as INV(ϕ).

Symbolic Integration

Pattern	Antiderivative
ACOS(ϕ)	$\phi\times$ACOS(ϕ)−$\sqrt{(1-\phi^2)}$
ALOG(ϕ)	.434294481904×ALOG(ϕ)
ASIN(ϕ)	$\phi\times$ASIN(ϕ)+$\sqrt{(1-\phi^2)}$
ATAN(ϕ)	$\phi\times$ATAN(ϕ−LN(1+ϕ^2)/2
COS(ϕ)	SIN(ϕ)
1/(COS(ϕ)×SIN(ϕ))	LN(TAN(ϕ))
COSH(ϕ)	SINH(ϕ)
1/(COSH(ϕ)×SINH(ϕ))	LN(TANH(ϕ))
1/(COSH(ϕ)^2)	TANH(ϕ)
EXP(ϕ)	EXP(ϕ)
EXPM(ϕ)	EXP(ϕ)−ϕ
LN(ϕ)	$\phi\times$LN(ϕ)−ϕ
LOG(ϕ)	.434294481904×$\phi\times$LN(ϕ)−ϕ
SIGN(ϕ)	ABS(ϕ)
SIN(ϕ)	−COS(ϕ)
1/(SIN(ϕ)×COS(ϕ))	LN(TAN(ϕ))
1/(SIN(ϕ)×TAN(ϕ))	−INV(SIN(ϕ))
1/(SIN(ϕ)×TAN(ϕ))	−INV(SIN(ϕ))
1/(SIN(ϕ)^2)	−INV(TAN(ϕ))
SINH(ϕ)	COSH(ϕ)
1/(SINH(ϕ)×^{2}	−INV(SIN(ϕ))

Symbolic Integration (continued)

Pattern	Antiderivative
1/(SINH(ϕ)×COSH(ϕ))	LN(TANH(ϕ))
1/(SINH(ϕ)×TANH(ϕ))	−INV(SINH(ϕ))
SQ(ϕ)	$\phi^3/3$
TAN(ϕ)^2	TAN(ϕ)−ϕ
TAN(ϕ)	−LN(COS(ϕ))
TAN(ϕ)/COS(ϕ)	INV(COS(ϕ))
1/TAN(ϕ)	LN(SIN(ϕ))
1/TAN(ϕ)×SIN(ϕ))	−INV(SIN(ϕ))
TANH(ϕ)	LN(COSH(ϕ))
TANH(ϕ)/COSH(ϕ)	INV(COSH(ϕ))
1/TANH(ϕ)	LN(SINH(ϕ))
1/TANH(ϕ)×SINH(ϕ))	−INV(SINH(ϕ))
$\sqrt{\phi}$	2×$\phi^{1.5}/3$
$1/\sqrt{\phi}$	2×$\sqrt{\phi}$
1/(2×$\sqrt{(\phi)}$)	2×$\sqrt{(\phi)}$×.5
ϕ^z (z symbolic)	IFTE(z==−1,LN(ϕ),$\phi^{(z+1)}/(z+1)$)
ϕ^z (z real, ≠0,−1)	$\phi^{(z+1)}/(z+1)$
ϕ^0	ϕ
ϕ^{-1}	LN(ϕ)
$1/\phi$	LN(ϕ)
$1/(1-\phi^2)$	ATANH(ϕ)
$1/(1+\phi^2)$	ATAN(ϕ)
$1/(\phi^2+1)$	ATAN(ϕ)
1/($\sqrt{(\phi-1)}$×$\sqrt{(\phi+1)}$)	ACOSH(ϕ)
$1/\sqrt{(1-\phi^2)}$	ASIN(ϕ)
$1/\sqrt{(1+\phi^2)}$	ASINH(ϕ)
$1/\sqrt{(\phi^2+1)}$	ASINH(ϕ)

Trigonometric Expansions

The following tables list expansions for trigonometric functions in Radians mode when using the →DEF, TRG*, and →TRG operations. These operations appear in the EquationWriter RULES menu.

→DEF Expansions

Function	Expansion
SIN(x)	$\frac{EXP(x \times i) - EXP(-(x \times i))}{2 \times i}$
COS(x)	$\frac{EXP(x \times i) + EXP(-(x \times i))}{2}$
TAN(x)	$\frac{EXP(x \times i \times 2) - 1}{(EXP(x \times i \times 2) + 1) \times i}$
SINH(x)	$-(SIN(x \times i) \times i)$
COSH(x)	$COS(x \times i)$
TANH(x)	$TAN(x \times i) \times -i$
ASIN(x)	$-i \times LN(\sqrt{1-x^2} + i \times x)$
ACOS(x)	$\frac{\pi}{2} + i \times LN(\sqrt{1-x^2} + i \times x)$
ATAN(x)	$-i \times LN(\frac{(1+i \times x)}{\sqrt{1+x^2}})$
ASINH(x)	$-LN(\sqrt{1+x^2} - x)$
ACOSH(x)	$\sqrt{-(\frac{\pi}{2} + i \times LN(\sqrt{1-x^2} + i \times x))^2}$
ATANH(x)	$-LN(\frac{(1-x)}{\sqrt{1-x^2}})$

TRG* Expansions

Function	Expansion
SIN$(x + y)$	$SIN(x) \times COS(y) + COS(x) \times SIN(x)$
COS$(x + y)$	$COS(x) \times COS(y) + SIN(x) \times SIN(y)$
TAN$(x + y)$	$\frac{TAN(x) + TAN(y)}{1 - TAN(x) \times TAN(y)}$
SINH$(x + y)$	$SINH(x) \times COSH(y) + COSH(x) \times SINH(x)$
COSH$(x + y)$	$COSH(x) \times COSH(y) + SINH(x) \times SINH(y)$
TANH$(x + y)$	$\frac{TANH(x) + TANH(y)}{1 + TANH(x) \times TANH(y)}$

→TRG Expansion

Function	Expansion
EXP(x)	$COS(\frac{x}{i}) + SIN(\frac{x}{i}) \times i$

Source References

The following references were used as sources for many of the constants and equations used in the HP 48. (See "References" in chapter 4, "Equation Reference," for the references used as sources for the Equation Library.)

1. E.A. Mechtly. *The International System of Units, Physical Constants and Conversion Factors*, Second Revision. National Aeronautics and Space Administration, Washington DC, 1973.

2. *The American Heritage Dictionary*. Houghton Mifflin Company, Boston, MA, 1979.

3. *American National Standard Metric Practice ANSI/IEEE Std 268-1982*. The Institute of Electrical and Electronics Engineers, Inc., New York, 1982.

4. *ASTM Standard Practice for Use of the International System of Units (SI) E380-89a*. American Society for Testing and Materials, Philadelphia, 1989.

5. *Handbook of Chemistry and Physics*, 64th Edition, 1983-1984. CRC Press, Inc, Boca Raton, FL, 1983.

6. *International Standard publication No. ISO 31/I-1978 (E)*.

7. *The International System of Units (SI)*, Fourth Edition. The National Bureau of Standards Special Publication 330, Washington D.C., 1981.

8. *National Aerospace Standard*. Aerospace Industries Association of America, Inc., Washington D.C., 1977.

9. *Physics Letters B*, vol 204, 14 April 1988 (ISSN 0370-2693).

G

Parallel Processing with Lists

Parallel processing is the idea that, generally, if a command can be applied to one or more individual arguments, then it can also be extended to be applied to one or more *sets* of arguments.

Some examples:

- `5 INV` returns `.2`, so `{ 4 5 8 } INV` returns `{ .25 .2 .125 }`.
- `4 5 *` returns `20`, so `{ 4 5 6 } { 5 6 7 } *` returns `{ 20 30 42 }`, and `{ 4 5 6 } 5 *` returns `{ 20 25 30 }`.

General Rules for Parallel Processing

As a rule-of-thumb, a given command can use parallel list processing if all the following are true:

- The command checks for valid argument types. Commands that apply to all object types, such as DUP, SWAP, ROT, and so forth, do not use parallel list processing.
- The command takes exactly one, two, three, four, or five arguments, none of which may itself be a list. Commands that use an indefinite number of arguments (such as →LIST) do not use parallel list processing.
- The command isn't a programming branch command (IF, FOR, CASE, NEXT, and so forth).

The remainder of this appendix describes how the many and various commands available on the HP 48 are grouped with respect to parallel processing.

Group 1: Commands that cannot parallel process

A command must take arguments before it can parallel process, since a zero-argument command (such as RAND, VARS, or REC) has no arguments with which to form a group.

Group 2: Commands that must use DOLIST to parallel process

This group of commands cannot use parallel processing directly, but can be "coerced" into it using the DOLIST command (see "Using DOLIST for Parallel Processing" later in this appendix). This group consists of several subgroups:

- **Stack manipulation commands.** A stack manipulation command cannot parallel process because the stack is manipulated as a whole and list objects are treated the same as any other object. Stack commands (such as DROPN) that take level 1 arguments will not accept level 1 list arguments.
- **Commands that operate on lists as wholes.** Certain commands accept lists as arguments but treat them no differently than any other data object. They perform their function on the object as a whole without respect to its elements. For example, →STR converts the entire list object to a string rather than converting each individual element, and the == command tests the level 1 object against the level 2 object regardless of the objects' types.
- **List manipulation commands.** List manipulation commands will not parallel process since they operate on list arguments as lists rather than as sets of parallel data. However, a list manipulation command can be forced to parallel process lists of lists by using the the DOLIST command. For example, `{ { 1 2 3 } { 4 5 6 } } « ΣLIST » DOLIST` returns `{ 6 120 }`.
- **Other commands that have list arguments.** Because a list can hold any number of objects of any type, it is commonly used to hold a variable number of parameters of various types. Some commands accept such lists, and because of this are insensitive to parallel processing, except by using DOLIST.
- **Index-oriented commands.** Many array commands either establish the size of an array in rows and columns or manipulate individual elements by their row and column indices. These commands expect these row and column indices to be real number pairs collected in lists. For example, `{ 3 4 } RANM` will generate a random integer matrix having 3 rows and 4 columns. Since these commands can normally use lists as arguments, they cannot perform parallel processing, except by using DOLIST.
- **Program control commands.** Program control structures and commands do not perform parallel processing and cannot be forced to do so. However, programs containing these structures can be made to parallel process by using DOLIST. For example, `{ i 2 3 4 5 6 } i « IF DUP 3 ≤ THEN DROP END » DOLIST` returns `{ 3 4 5 6 }`.

Group 3: Commands that sometimes work with parallel processing

Graphics commands that can take pixel coordinates as arguments expect those coordinates to be presented as two-element lists of binary integers. Since these commands can normally use lists as arguments, they cannot parallel process, except by using DOLIST.

For the two-argument graphics commands (BOX, LINE, TLINE), if either argument is not a list (a complex number, for example), then the commands will parallel process, taking the list argument to be multiple complex number coordinates. For example, `(0,0) { (1,1) (3,2) } LINE` will draw two lines—between (0,0) and (1,1) and between (0,0) and (3,2).

Group 4: ADD and +

On HP 48S and HP 48SX calculators, the + command has been used to append lists or to append elements to lists. Thus `{ i 2 3 } 4 +` returns `{ i 2 3 4 }`. With the advent of parallel processing in the HP 48G series, the ADD command was created to perform parallel addition instead of +.

This has several ramifications:

- To add two lists in parallel, you must do one of the following:
  - Use ADD from the MTH LIST menu.
  - Create a custom menu containing the ADD command.
  - Assign the ADD command to a user-defined key.
- User programs must be written using ADD instead of + if the program is to be able to perform direct parallel processing, or written with + and applied to their arguments by using DOLIST. For example, programs such as `« →x '×+2' »` will produce list concatenation when x is a list rather than parallel addition, unless rewritten as `« →x '×ADD 2' »`
- Algebraic expressions capable of calculating with variables containing lists (including those intended to become user-defined functions) cannot be created in RPN syntax since using

ADD to add two symbolic arguments concatenates the arguments with + rather than with ADD. For example, `'X' DUP 2 ^ SWAP 4 * ADD 'F(X)' SWAP =` produces `'F(X)=X^2+4*X'` rather than `'F(X)=X^2 ADD 4*X'`

Group 5: Commands that set modes/states

Commands that store values in system-specific locations so as to control certain modes and machine states can generally be used to parallel process data. The problem is that each successive parameter in the list cancels the setting established by the previous parameter. For example, `{ 1 2 3 4 5 } FIX` is effectively the same as `5 FIX`.

Group 6: One-argument, one-result commands

These commands are the easiest to use with parallel processing. Simply provide the command with a list of arguments instead of the expected single argument. Some examples:

`{ 1 -2 3 -4 } ABS` returns `{ 1 2 3 4 }`

`DEG { 0 30 60 90 } SIN` returns `{ 0 .5 .866025403784 1 }`

`{ 1 A 'SIN(Z)' } INV` returns `{ 1 'INV(A)' 'INV(SIN(Z))' }`

Group 7: Two-argument, one-result commands

Two-argument commands can operate in parallel in any of three different ways:

- *{ list } { list }*
- *{ list } object*
- *object { list }*

In the first form, parallel elements are combined by the command: `{ 1 2 3 } { 4 5 6 } %` returns `{ .04 .1 .18 }`.

In the second form, the level 1 object is combined with each element in the level 2 list in succession:
`{ 1 2 3 } 30 %CH` returns `{ 2900 1400 900 }`.

In the third form, the level 2 object is combined with each element of the level 1 list in succession:
`50 { 1 2 3 } %T` returns `{ 2 4 6 }`.

Group 8: Multiple-argument, one-result commands

Commands that take multiple (3, 4, or 5) arguments can perform parallel processing only if all arguments are lists. For example, `{ 'SIN(X)' 'COS(X)' 'TAN(X)' } { X X X } { 0 0 0 } ROOT` returns `{ 0 90 0 }`. Notice that lists must be used even though the level 1 and level 2 lists each contain multiples of the same element.

Group 9: Multiple-result commands

Any command that allows parallel processing, but produces multiple results from its input data, will return its results as a single list. For example, `{ 1 2 3 } { 4 5 6 } R→C C→R` produces `{ 1 4 2 5 3 6 }` rather than the more expected `{ 1 2 3 } { 4 5 6 }`.

The following *UNMIX* program will unmix the data given the number of expected result lists:

```
« OVER SIZE → l n s
 « 1 n
  FOR j j s
   FOR l l 1 GET n
   STEP s n / →LIST
  NEXT
 »
»
```

Taking `{ 1 4 2 5 3 6 }` from above as the result of C→R (a command which should return two results), `2 UNMIX` gives `{ 1 2 3 } { 4 5 6 }`.

Group 10: Quirky commands

A few commands behave uniquely with respect to parallel processing:

- **DELALARM.** This command can take a list of arguments. Note, however, that deletions from early in the alarm list will change the alarm indices of the later alarm entries. Thus, if there are only three alarms, `{ 1 3 } DELALARM` will cause an error, whereas `{ 3 1 } DELALARM` will not.
- **DOERR.** This command forces an error state that causes all running programs and commands to halt. Thus, even though providing the command with a list argument will cause the command to perform parallel processing, the first error state

will cause the command to abort and none of the rest of the list arguments will be used.

- **FREE, MERGE.** Only port 1 can be freed or merged on the HP 48GX. Thus, even though a list argument is acceptable, an error will occur for any list except { 1 }.
- **RESTORE.** This command performs a system warmstart after installing the backup object into memory. All functions are terminated at that time. Thus, only the first backup object in a list will be restored.
- **_ (Attach Unit).** This command will create unit objects in parallel only if level 1 contains a list. Thus 1 { ft in m } _ produces { 1_ft 1_in 1_m } while { 1 2 3 } 'm' _ produces an error.
- **STO+.** STO+ performs parallel list addition only if both arguments are lists. If one argument is a list and the other is not, STO+ appends the non-list argument to each element in the list.
- **STO−, STO*, STO/.** These commands perform parallel processing if both arguments are lists, but fail otherwise.

Using DOLIST for Parallel Processing

Almost any command or user program can be made to work in parallel over a list or lists of data by using the DOLIST command. Use DOLIST as follows.

- Level 1 must contain a command, a program object, or the name of a variable that contains a command or program object.
- Level 2 must contain an argument count unless the level 1 object is a command that accepts parallel processing, a program containing only one command that accepts parallel processing, or a user-defined function. In these special cases, Level 2 contains the first of the list arguments.
- If level 2 was the argument count, then level 3 is the first of the argument lists. Otherwise, levels 2 through n are the argument lists.

As an example, the following program takes three objects from the stack, tags them with the names a, b, and c, and displays them one after the other in line 1 of the display.

```
« → a b c
  « { a b c } DUP « EVAL » DOLIST
    SWAP « →TAG » DOLIST
    CLLCD 1 « i DISP i WAIT » DOLIST
  »
»
```

Index

D

E

F

G

H

I

J

K

L

M

N

T

U

V

W

X

Y