This booklet is primarily intended for reference use after you read the HP-65 Owner’s Handbook. The Procedures section (1-110) is a brief digest for the user who needs a quick review of the more important operating procedures. The Key Dictionary (11-67) provides easy access to the details of the individual key operations and switches. All symbols on the HP-65 keyboard are presented in alphabetic order in the dictionary; non alphabetic symbols (e.g., + , [ ] ) are at the front (11-714). A general Index is provided at the end.

**Procedures (1-110)**

1. W/PRGM-RUN Switch

Set to: RUN position to calculate, to run a program, to read a pre-programmed card.

Set to: W/PRGM position to clear program memory, to key in a program, to edit a program, to write a program on a magnetic card.

2. Doing Arithmetic in the Stack

<table>
<thead>
<tr>
<th>Compute</th>
<th>By Pressing</th>
</tr>
</thead>
<tbody>
<tr>
<td>8 − 2 = 6</td>
<td>8 ENTER↑ 2 −</td>
</tr>
<tr>
<td>8 ÷ 2 = 4</td>
<td>8 ENTER↑ 2 ÷</td>
</tr>
</tbody>
</table>
Compute: \(\frac{(4 \times 5)}{(2 + 3)} = -6 = -2\) using the keys shown below.

### Contents of Stack Registers

<table>
<thead>
<tr>
<th>T</th>
<th>4</th>
<th>4</th>
<th>4</th>
<th>20</th>
<th>20</th>
</tr>
</thead>
<tbody>
<tr>
<td>Z</td>
<td>20</td>
<td>20</td>
<td>20</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>Y</td>
<td>4</td>
<td>4</td>
<td>5</td>
<td>20</td>
<td>2</td>
</tr>
<tr>
<td>X</td>
<td>4</td>
<td>4</td>
<td>5</td>
<td>20</td>
<td>3</td>
</tr>
</tbody>
</table>

Key: \[4 \uparrow 5 \times 2 \uparrow 3 \div 6 -\]

**NOTE:** \(\text{ENTER}^\uparrow\) is here abbreviated as \(\uparrow\)

### 3. Data Entry

**Entering Negative Numbers.** Press \(\text{CHS}\) (change sign) after keying in the positive value.  
**Example:** to key in \(-12\), press \(12 \text{ CHS}\).

**Entering Big and Small Numbers (Scientific Format).**  
**Method:**  
1. Key in mantissa  
2. Press \(\text{CHS}\) if negative number  
3. Press \(\text{EEX}\)  
4. Key in exponent  
5. Press \(\text{CHS}\) if exponent is negative.  

**Example:** To key in \(5 \times 10^{-8}\), press \(5 \text{ EEX} 8 \text{ CHS}\).

**Correcting Mistakes.** Press \(\text{CLX}\) and reenter the number.
4. Display

Controlling Display. To display \( x \), rounded to \( n \) fixed decimal places, press \( \text{DSP} \ \bullet \ n \) (where \( n = 0, 1, \ldots, 9 \)). To display \( x \) in scientific notation, rounded to \( n \) places, press \( \text{DSP} \ [n] \). Display can also be set by a program. Blinking Display. Blinking display occurs when an illegal operation is attempted (Example: \( 5 \div 0 \)). Depressing any key stops the blinking without doing the key function. Blinking occurs when a program card is misread (or blank). See the Key Dictionary for the limitations on the following: \( \text{LN} \), \( \text{LOG} \), \( \sqrt{x} \), \( \text{SIN} \), \( \text{COS} \), \( \text{D.MS}+ \), \( +\text{D.MS} \), \( +\text{OCT} \), \( 1/x \), \( y^x \), \( \div \). Multiple Decimal Points. The display also indicates low battery power (all decimal points light up).

5. Prefix Operations

Performing “Gold” Functions (Upshift). Rule: Prefix gold function keys by \( f \) to do the function or \( f^1 \) to do the inverse. Example: Calculate \( \log(100) = 2 \) by pressing \( 100 \ f \ \text{LOG} \). Example: Calculate antilog (2) = 100 (the inverse) by pressing \( f^1 \ \text{LOG} \). Performing “Blue” Functions (downshift). Prefix blue function keys by \( g \). Example: Calculate \( 5! = 120 \) by
Correcting or Cancelling a Prefix.  To correct a wrong prefix, merely press the correct prefix.  To cancel a prefix, press \[f\] \(\text{PREFIX}\).  Other prefix keys are \[DSP\], \[GTO\], \[STO\], \[RCL\], \[LBL\].

### 6. Angular Mode

Operations involving angles (namely, \[\rightarrow \text{D.MS}\], \[\text{SIN}\], \[\text{COS}\], \[\text{TAN}\], \(\rightarrow \text{P}\)) assume the angles to be in the units (degrees, radians, or grads) of the prevailing angular mode as set by \[\text{DEG}\] (or power on), \[\text{RAD}\], or \[\text{GRD}\].  360 degrees = \(2\pi\) radians = 400 grads.  Converting from One Angular Mode to Another:

\[
\begin{align*}
\text{g} & \quad \begin{cases} 
\text{DEG} \\
\text{GRD} \\
\text{RAD}
\end{cases} \\
\begin{cases} 
\text{f} \quad \rightarrow \text{D.MS} \\
\text{g}
\end{cases} & \quad \begin{cases} 
\text{DEG} \\
\text{GRD} \\
\text{RAD}
\end{cases} & \quad \begin{cases} 
\text{f}^{-1} \quad \rightarrow \text{D.MS}
\end{cases}
\end{align*}
\]

### 7. Storage Register Operations

Storing a Number in Addressable Register \(R_n\).  ① Key in number to be stored. ② Press \[\text{STO} \quad n\] (where \(n\) is a digit 1, \ldots, 9).  Recalling a Number from \(R_n\).  Press \[\text{RCL} \quad n\] (where \(n\) is a digit 1, \ldots, 9).
Doing Storage Register Arithmetic.

**Subtraction.** \((r_n - x \to R_n)\): \(\text{STO} - \boxed{n}\).

**Addition.** \((r_n + x \to R_n)\): \(\text{STO} + \boxed{n}\).

**Multiplication.** \((r_n \times x \to R_n)\): \(\text{STO} \times \boxed{n}\).

**Division.** \((r_n \div x \to R_n)\): \(\text{STO} \div \boxed{n}\).

where \(\boxed{n}\) is a digit 1, . . . , 9. \(x\) is unchanged in these operations.

8. Clearing Registers

- Press \(\text{CLX}\) to clear \(X\) register.
- Press \(\text{f} \ STK\) to clear entire stack (\(X, Y, Z,\) and \(T\)).
- Press \(\text{f} \ REG\) to clear all storage registers \((R_1, ..., R_9)\).
- Power on clears all registers.

9. Programming

**Program Memory.** Used to contain user’s stored program. Capacity: 100 locations
\- **Top** is above 1st location.
\- **Bottom** is last location.

**The Pointer,** an internal part of the calculator, determines which memory location is executed or displayed. In W/PRGM Mode, keystrokes are stored in memory as codes: The address for top of memory is 00 00. The codes for keys 0–9, are 00 – 09. For other keys, the code denotes row and column. Example: Code for \(R/S\) (row 8, column 4) is 84.

**User Defined Functions.** The top row keys are used to call a user defined function. When the calculator is turned on, default functions are de-
fined as shown in the window above the top row. Alternate functions may be keyed in or read from a magnetic program card. Pressing a top row key finds the function and executes it. **To Write a Function:** identify the beginning by **LBL** followed by the top row key that is to call it. End the function with **RTN**. **Example:**

```
E ( ENTER↑ ENTER↑ x x ) RTN
```

The keys in parentheses calculate \( x^3 \). **Keying in a function:**

1. Set W/PRGM-RUN switch to W/PRGM.
2. Press **f** [PRGM] to clear program memory.
3. Press the keys in the order shown ( **LBL** E . . . **RTN** in the sample case). If you make a mistake, press **g** [DEL] (to delete the error) and press the correct key. **To execute the function** (from keyboard):

1. Set W/PRGM-RUN switch to RUN.
2. Press the appropriate top row key. For the sample case, to compute \( 2^3 = 8 \), press 2 E ; to compute \( 3^3 = 27 \), press 3 E ; etc. A user defined function can also be executed from a program by merely including the corresponding top row key in the program. **Revising (Editing) a Program.**

- To move the pointer to the top of memory, press **RTN** in RUN mode.
- To move the pointer to a label, press (in RUN mode) **GTO** \( n \) where \( n \) is the same digit or top row key as in **LBL** \( n \) (the label).
- To step
through your program, use SST in W/PRGM mode (See ¶56). You will see the successive program codes in the display. To insert a step below the currently displayed step, just key in the new operation in W/PRGM mode. (See ¶32). To delete a step or correct a mistake, press g DEL in W/PRGM mode (See ¶21). Both overflow and underflow of a register will stop a program.

10. Using Magnetic Cards

Reading a Pre-recorded Card. Set W/PRGM-RUN switch to RUN. Insert the card in the right lower slot. If the card does not read properly, display will flash and program memory will be cleared (but, with no effect on registers). Press R/S and reinsert the card.

Recording (Writing) on a Program Card. Set W/PRGM-RUN switch to W/PRGM. Insert unprotected (unclipped) card in right lower slot. Protect the card by clipping the notched corner.
Key Dictionary (§11–§67)

This section contains reference entries for all keys with their associated symbols. In addition, the following entries are included. **Insert**, §32. **Last X**, §34. **Merged Codes**, §39. **OFF-ON Switch**, §43. **Stack Lift**, §57. **W/PRGM-RUN Switch**, §62.
111 
\[ - , + , \times , \div \] **Arithmetic Operations.** Calculate respectively: \( (y - x) \), \( (y + x) \), \( (y \times x) \), \( (y \div x) \). ▪ Enable lift ▪ Save \( x \) in \( \text{Last X} \). ▪ Drop stack as follows:

<table>
<thead>
<tr>
<th>Contents</th>
<th>Register</th>
</tr>
</thead>
<tbody>
<tr>
<td>( t )</td>
<td>( T )</td>
</tr>
<tr>
<td>( z )</td>
<td>( Z )</td>
</tr>
<tr>
<td>( y )</td>
<td>( Y )</td>
</tr>
<tr>
<td>( x )</td>
<td>( \text{result-X} )</td>
</tr>
<tr>
<td></td>
<td>( \text{Last X} )</td>
</tr>
</tbody>
</table>

For **storage register arithmetic**, see ¶59.

112

0 . . . 9 **Digits.** Digits are used: ① to enter data (¶3), ② to specify registers (¶59, ¶49), ③ to label program steps (¶35, ¶31), ④ to specify displayed decimal places (¶24). ▪ When used to enter numbers, digits enable the stack lift (¶57).

113

□ **Decimal Point.** The □ key is also used to specify fixed decimal display (¶24). ▪ Enables stack lift.

114

π **Inserts** \( \pi (3.141592654) \) into the stack \( X \) register. ▪ Lifts stack (¶57) if enabled. ▪ Enables stack lift.

115

A, B, C, D, E **Top Row Keys.** When used
without a prefix, a top row key finds and executes a user defined function. Used as suffix for `LBL` (¶35) and `GTO` (¶31). Precaution: Power on automatically inserts 5 functions `A` - `E` in memory. Press `f PRGM` in `W/PRGM` mode to clear them. Top row keys leave stack lift unaffected.

16

**ABS** Absolute Value. If x is negative, **ABS** makes it positive. Saves x in Last X. Enables stack lift.

17

**CHS** Change Sign. Changes the sign of the number in the stack X register unless the `EEX` key has been pushed; in this case the exponent sign is reversed, instead. First non zero digit of a number entry gives x a positive sign. Once a number is terminated (by a key other than a digit, `CHS` or `EEX`), the `CHS` key cannot reverse the exponent sign. In such a case `CHS` reverses the sign of the number. No effect on stack lift.

18

**CLX** Clear X Register. Disables stack lift.
Cosine ([prefix]) / Arc Cosine ([prefix]) (principal value, \(0^\circ \leq \text{result} \leq 180^\circ\) or equivalent in radians or grads).  ■ Saves \(x\) in Last X.  ■ Destroys register \(R_y\).  ■ \(x\) less than \(-1\) or greater than \(+1\) gives error (blinking zero) for arc cosine.  ■ Enables stack lift.

Set Degree Angular Mode.  ■ Affects angle operations (\(\rightarrow\text{D.MS}\), \(\text{SIN}\), \(\text{COS}\), \(\text{TAN}\), \(\text{R} \rightarrow \text{P}\)).  ■ No effect on stack lift.

Delete Program Step.  ■ With W/PRGM-RUN switch in W/PRGM position, \(\text{g DEL}\) ① deletes the program step denoted by the program pointer (¶9), ② moves all the following steps up one step, and ③ inserts a \(\text{g NOP}\) code in the vacated bottom position of memory.  ■ Inoperative during run mode, (\(\text{g DEL}\) acts as \(\text{CLX}\))  ■ \(\text{g DEL}\) can be used to back up the pointer (after which you reinsert the deleted codes).  ■ If memory is full (i.e., a minus sign shows in display), \(\text{g DEL}\) loses bottom memory step.  ■ If the program pointer is at the bottom (i.e., 2 minus signs show in display), \(\text{g DEL}\) deletes 2 locations.  ■ No effect on stack lift.
**+D.MS** Converts to (f prefix) or from (r⁻¹ prefix) Degrees, Minutes, Seconds. Converts the contents of the X register (in degrees, radians, or grads) to the form DDDMMSS (degrees, minutes, seconds) or the inverse, depending on the prefix. Saves x in Last X. Gives error (blinking zero) if the magnitude of x (degrees or equivalent in radians or grads) exceeds 99999.99999. Enables stack lift.

**D.MS+** Adds (f prefix) / Subtracts (r⁻¹ prefix) Degrees, Minutes, Seconds. Used for adding/subtracting (y—x) degrees-minutes-seconds or hours-minutes-seconds in the stack X and Y registers. Operands are of the form DDDMMSS in the stack X and Y registers. Saves x in Last X. Drops stack (as in ¶11). Gives error (blinking 0.00), if the magnitude of an operand or the result exceeds 99999.59599 (degrees, minutes, seconds). Enables stack lift.

**DSP** Display (prefix). DSP ⊛ n displays x rounded to n fixed decimal places. DSP n sets scientific display of x rounded to n decimal places where n = 0, 1, . . . , 9.
- Does not alter internal value of x.

Power on sets \texttt{DSP} \texttt{2} (0.00). 
- If x is too small for a specified display, (signed) zero is displayed.
- If x is too large for the specified format, \texttt{DSP} \texttt{9} format is used.
- No effect on stack lift.

\textbf{Decrement and Skip on Zero.} Subtracts 1 from an integer in register R_s, then skips two program memory locations if R_s contains zero. 

- The decrement operation is suppressed outside the limits: \(1 < |r_s| < 10^{10}\).

- Useful for looping. \textbf{Rule:} To execute a labelled program segment n times, preset n in R_s and use \texttt{DSZ} to determine whether or not to repeat the segment.

\textbf{Example:} The following loop executes 12 times, then stops: 12 \texttt{STO 8 LBL 1} \ldots \texttt{DSZ GTO 1 R/S}.

- No effect on stack lift.

\textbf{See \texttt{115}.}

\textbf{Enter Exponent.} Terminates the mantissa portion of a number and institutes the entry of a power of 10 multiplier (exponent) into the X register. If no mantissa was previously entered, \texttt{EEX} sets up a mantissa of 1.

- Enables stack lift.
127

**ENTER**  Copy x to Y.

<table>
<thead>
<tr>
<th>Contents</th>
<th>Register</th>
</tr>
</thead>
<tbody>
<tr>
<td>t</td>
<td>T</td>
</tr>
<tr>
<td>z</td>
<td>Z</td>
</tr>
<tr>
<td>y</td>
<td>Y</td>
</tr>
<tr>
<td>x</td>
<td>X</td>
</tr>
</tbody>
</table>

- Copies the contents of the X register into the Y register, pushing y into Z and z into T (t is lost).
- Disables stack lift.

128

**f**  **f⁻¹** **Upshift** (prefix). The gold symbol above a key denotes the function of the key if preceded by **f**. **The inverse or complement is done if the key is prefixed by **f⁻¹**. **No inverses are defined for clear functions (4th row keys). For these keys **f⁻¹** gives the clear function. **To cancel an unwanted **f** or **f⁻¹**, press **PREFIX**. **No effect on stack lift.**

129

**g**  **Downshift** (prefix). When **g** is followed by a key having a blue symbol below it, the function denoted by the symbol is done.
- To cancel an unwanted **g**, press **f** **PREFIX**. **No effect on stack lift.**
Sets Grad Angular Mode. Affects angle operations (→D.MS, SIN, COS, TAN, R+P). ■ No effect on stack lift.

Go to (prefix). When followed by a digit ([0, ..., 9]) or a letter ([A, ..., E]), GTO advances the program pointer downward to the first occurrence of the corresponding program label: LBL followed by the same digit or letter. ■ No effect on stack lift.

Insert ■ Pressing a key in W/PRGM mode stores the instruction code in program memory between the displayed code and the following instruction code and moves the pointer to display the code just inserted. The bottom location drops off. ■ Insert is not performed: (a) For PRGM, DEL, SST. (b) For the second key of a merged code (¶39). (c) When the pointer is at the bottom. ■ No effect on stack lift.

Truncates x to Integer (.prefix) or fraction (prefix). ■ Saves x in Last X. ■ Retains the sign of the number. ■ Enables stack lift.
Last-X Register. Contains the value of x before the latest \( f(x) \) or \( f(x, y) \) was computed. Used to: compute functions that make multiple use of the same operand. enable recovery from keystroke errors in certain instances. To Recall Last x, See \( \text{LST X} \) (¶38). The following save x in Last X before performing their functions:

\[
\begin{align*}
+ & \quad - & \quad \times & \quad \div & \quad \text{ABS} & \quad \text{COS} & \quad \rightarrow \text{D.MS} \\\n\text{D.MS+} & \quad \text{INT} & \quad \text{LN} & \quad \text{LOG} & \quad \rightarrow \text{OCT} & \quad \text{R→P} \\
\text{SIN} & \quad \text{TAN} & \quad ! & \quad \sqrt{x} & \quad \frac{1}{x} & \quad y^x 
\end{align*}
\]

Note that \( \text{CLX} \) does not affect the Last-X register.

Label (prefix). \( \text{LBL} \) identifies its suffix (a digit \( \text{0, ... , 9} \), or top row key, \( \text{A, ... , E} \)) as a label in a stored program. A branch to the part of the program thus labelled can then be done by executing \( \text{GTO} \) followed by the same suffix. For user defined functions, the label suffix must be a top row key (\( \text{A, ... , E} \)). No effect on stack lift.

Natural Log (x) (\( \text{f} \) prefix) or e^x (\( \text{f}^{-1} \) prefix).

\[
\begin{align*}
\text{Saves x in Last X.} & \quad \text{f} & \quad \text{LN} & \quad \text{gives error (blinking zero) if x is zero or negative.} \\
\text{Enables stack lift.} & \quad \text{f} & \quad \text{LN} & \quad \text{gives error (blinking zero) if x is zero or negative.} \\
\text{Enables stack lift.}
\end{align*}
\]
137
Common Log(x) ( \textbf{f} prefix) or 10^x ( \textbf{f}^\text{\textsuperscript{-1}} prefix). □ Saves x in Last X. □ \textbf{f} \textbf{LOG} gives error (blinking zero) if x is zero or negative. □ Enables stack lift.

138
Recall Last x to the X Register. □ Performs automatic stack lift unless the lift is disabled. See ¶34. □ Enables stack lift.

139
Merged Codes. Program codes for the following are merged with their respective prefix codes: \textbf{LST X}, \textbf{NOP}, \textbf{x:y}, \textbf{R↓}, \textbf{R↑}, \textbf{x≠y}, \textbf{x≤y}, \textbf{x=y}, \textbf{x>y}, 1 \ldots 8 when prefixed by \textbf{STO} or \textbf{RCL}. □ Example: \textbf{g} \textbf{LST X} in program mode is merged and displayed as 35 00; \textbf{STO} 5 as 33 05, etc. For explanation of codes, see ¶9.

140
Integer Factorial. □ Computes the factorial of a nonnegative integer n in the X register. \[ n! = 1 \cdot 2 \cdot 3 \cdot \ldots \cdot (n-1) \cdot n \]
0! = 1 □ Saves x in Last X; □ Negative or non-integer x gives error (blinking zero) □ If x exceeds 69, overflow occurs. □ Enables stack lift.
**No Operation.** Useful as a filler in tests. 
- in W/PRGM mode clears the entire memory to (merged code 35 01). No effect on stack lift.

**Convert Integer x to/from Octal.** Saves x in Last X. 
- Non-integer or x larger in magnitude than 1073741823 gives error (blinking zero). Enables stack lift.

**OFF-ON Switch.** “Power on”: clears all registers and flags. 
- sets the display rounding to 2 fixed places (0.00). 
- leaves program pointer at top of memory. 
- inserts 5 functions at the top of memory that are callable from the top row keys to allow single stroke execution of the 5 functions shown in the window above the top row.

**Clear Prefix.** When prefixed by or cancels the effect of an impending prefix, so that a nonprefix operation can be done. 
- If a wrong prefix key is depressed before keying in the suffix of a prefixed operation, the error can be cor-
rected by simply pressing the correct prefix and proceeding from there. ■ No effect on stack lift.

**145**

**PRGM** Clear Program Memory. In program mode, \( f \text{ PRGM} \) (or \( f^\uparrow \text{ PRGM} \)) clears the entire program memory to “no operation” codes (35 01), leaving the pointer at the top of memory. ■ In run mode, \( f \text{ PRGM} \) is equivalent to \( \text{CLX} \). ■ No effect on stack lift.

**146**

**R↓** Roll Stack Up. Enables stack lift.

**147**

**R↑** Roll Stack Down. Enables stack lift.

**148**

**RAD** Set Radian Angular Mode. ■ Affects angle operations (\( \rightarrow \text{D.MS} \), \( \text{SIN} \), \( \text{COS} \), \( \text{TAN} \), \( \text{R-P} \)) ■ No effect on stack lift.

**149**

**RCL** Recall (prefix). Recalls storage register \( R_n \)
(n is a digit 1, \ldots, 9) to the X register. \[=\]
Lifts stack unless stack lift is disabled. \[=\]
Enables stack lift.

150
\textbf{REG} \textbf{Clear Registers.} When prefixed by \( f \) or \( f^{-1} \), \textbf{REG} clears storage registers \( R_1 \ldots R_9 \) to zero.

151
\textbf{R\textasciitilde P} \textbf{Rectangular to Polar.} \[=\] Transforms the respective contents of the X and Y stack registers from rectangular form \((x, y)\) to polar form \((r, \theta)\) (\( f \) prefix) or the inverse (\( f^{-1} \) prefix). \[=\] Saves \( x \) in Last X. \[=\] Destroys previous contents of register \( R_0 \). \[=\] Enables stack lift.

![Diagram of rectangular to polar conversion]

152
\textbf{R/S} \textbf{Run/Stop.} \[=\] If \textbf{R/S} is pressed and a stored program is not executing, the stored program starts executing at the step denoted by the program pointer. \[=\] If exe-
cuted in a stored program, \textbf{R/S} stops the program, allowing keystrokes from the operator. The program pointer is positioned at the \textbf{R/S}. \textbf{R/S} in a program is immediately preceded by a numerical entry from the program, the automatic lift is disabled upon return to the keyboard. This allows a program to display prompting information that will not be lifted in the stack if you enter a number from the keyboard. Except for this case, \textbf{R/S} does not affect the stack lift.

\section*{Return}

\textbf{Return}. \textbullet{} If executed from the keyboard, \textbf{RTN} merely resets the program pointer to the top of memory. \textbullet{} In a stored program, \textbf{RTN} is the logical end of a user defined function. \textbullet{} If a function is executed from the keyboard, \textbf{RTN} stops the program. \textbullet{} If a function is executed in a program, execution of \textbf{RTN} resumes the calling program. \textbullet{} A function executed from the keyboard or a nonfunction program can execute another function. The latter, however, cannot properly execute yet another function. \textbullet{} No effect on stack lift.

\section*{Set flag 1, Set flag 2}

\textbf{SF1}, \textbf{SF2} \textbf{Set flag 1, Set flag 2}. \textbullet{} \textbf{f} \textbf{SF1} sets flag 1 on while \textbf{f'} \textbf{SF1} sets it off. \textbullet{} \textbf{f} \textbf{SF2} performs similarly, but using flag
2. To test the flags, see f61. ■ No effect on stack lift.

**SIN**

Sine (f prefix) / arc sine (f₁ prefix). ■

Arc sine calculates the principal value (−90° ≤ result ≤ +90° or equivalent in radians or grads). ■ Saves x in Last X. ■ Destroys register R₀. ■ x less than −1 or greater than +1 gives error (blinking zero) for arc sine. ■ Enables stack lift.

**SST**

Single Step. ■ In W/PRGM mode, SST advances the program pointer to the next memory location, displaying the step code. Repeated use of the key enables you to review a program and to position the pointer for editing. ■ In RUN mode, SST executes the program step denoted by the program pointer. In the case of single stepping a call to a user defined function, the entire function executes (as one step) before returning control to the keyboard. ■ No direct effect on stack lift.

**57**

Stack Lift. The stack lift is an automatic response of the HP-65 to allow you to put a new value in the X register and to simultaneously lift the previous values x, y, and z into the respectively higher registers Y, Z and T (t is lost) for future use—
this obviates the use of $\text{ENTER}^+$ in many instances. If the lift is enabled (i.e., not disabled), the lift takes place (as shown in the left diagram) at the beginning of $\text{EE}, \text{LST X}, \text{RCL} \ n$, and on the first key-stroke of a new number. If the lift is disabled, the previous $x$ is lost as shown in the right diagram.

<table>
<thead>
<tr>
<th>Contents</th>
<th>Register</th>
</tr>
</thead>
<tbody>
<tr>
<td>lost</td>
<td></td>
</tr>
<tr>
<td>$t$</td>
<td>$T$</td>
</tr>
<tr>
<td>$z$</td>
<td>$Z$</td>
</tr>
<tr>
<td>$y$</td>
<td>$Y$</td>
</tr>
<tr>
<td>$x$</td>
<td>$X$</td>
</tr>
</tbody>
</table>

Lift Disable Keys: $\text{R/S}$ in a program if the program has just put a new number in the $X$ register from program memory; $\text{CLX}$, $\text{ENTER}^+$ at any time.

Lift Enable Keys: All number entry keys $[0], \ldots, [9], \cdot, \text{EEX}, \text{TT}$, but not $\text{CHS}$. ■

All calculating keys $-, +, \times, \div, \text{ABS}, \cos, \pm\text{D.MS}, \text{D.MS+}, \text{INT}, \text{LN}, \text{LOG}, \text{LST X}, \text{n!}, \pm\text{OCT}, \text{R+P}, \text{SIN}, \text{TAN}, \sqrt[\text{x}], \sqrt[\text{y}], y^x$. ■ Stack manipulating keys $\text{R+}, \text{R-}, x; y$, but not $\text{ENTER}^+$ ■

Storage registers keys: $\text{STO}, \text{RCL}$.■
**No Effect Keys:** All other keys have no effect on the lift status. They include: all programming keys, angular mode keys, display control keys, clear keys (except \(\text{CLX}\)), and \(\text{CHS}\).

```
| 58 |
| **STK** |
| **Clear Stack.** ( \(\text{f}\) or \(\text{f}^{-1}\) prefix). |

```

```
| 59 |
| **STO** |
| **Store** (prefix). \(\text{STO}\) \(n\) copies the contents of register \(X\) into storage register \(R_n\) \((n = \text{a digit } 1, \ldots, 9)\). To perform an arithmetic function \((+, -, \times, \div)\) of \(x\) and a storage register \(R_n\), insert the corresponding arithmetic key between \(\text{STO}\) and \(n\).

\[
\begin{align*}
\text{STO} & + \quad \text{adds} \quad (r_n + x) \rightarrow R_n \\
\text{STO} & - \quad \text{subtracts} \quad (r_n - x) \rightarrow R_n \\
\text{STO} & \times \quad \text{multiplies} \quad (r_n \times x) \rightarrow R_n \\
\text{STO} & \div \quad \text{divides} \quad (r_n \div x) \rightarrow R_n
\end{align*}
\]

- Stack registers and Last X are unchanged.  
- Storage arithmetic codes are unmerged.  
- Enables stack lift.

```

```
| 60 |
| **TAN** |
| **Tangent** ( \(\text{f}\) prefix / **Arc Tangent** ( \(\text{f}^{-1}\) prefix).  
- **Arc tangent** \(x\) calculates the principal value \((-90^\circ \leq \text{result} \leq +90^\circ\) or equivalent in grads or radians).  
- Saves \(x\) in Last X.  
- Destroys register \(R_n\).  
- Enables stack lift.

```

**Key Dictionary**
161  

Test Flag 1, Test Flag 2.  \[ f \] TF1 tests flag 1, skipping 2 memory locations if flag 1 is off, while \( f^{-1} \) TF1 skips if flag 1 is on.  \( f \) TF2 performs similarly, but using flag 2.  To set the flags, see \( \|$54. \)  No effect on stack lift.

162  

W/PRGM-RUN Switch.  \( W/PRGM \) position sets program mode, used to:  create and edit a stored program or  write program memory on a magnetic card.  RUN position sets run mode, used to:  read a magnetic card into program memory  do calculations  execute stored programs.

163  

Reciprocal of x.  \( \sqrt{x} \) Saves x in Last X.  Gives error (blinking zero) if reciprocal of 0 is attempted.  Enables stack lift.

164  

Square Root ( \( f \) prefix) / Square ( \( f^{-1} \) prefix).  \( \sqrt{x} \) Saves x in Last X.  For \( f \) \( \sqrt{x} \), negative x gives error (blinking zero).  Enables stack lift.
Exchange x and y.

<table>
<thead>
<tr>
<th>Contents</th>
<th>Register</th>
</tr>
</thead>
<tbody>
<tr>
<td>y</td>
<td>Y</td>
</tr>
<tr>
<td>x</td>
<td>X</td>
</tr>
</tbody>
</table>

Enables stack lift.

Relational Tests of x and y. Each test compares the value in the X and Y registers and skips 2 memory locations if the test condition is not met. ■ Destroys register R₉. ■ No effect on lift.

Exponential. Raises the contents of the stack Y register to the power specified in the X register. ■ Saves x in Last X. ■ Drops stack (as in ¶11). ■ Negative or zero value of y gives error (blinking zero). ■ Enables stack lift.
Index

NOTE: To reference keyboard symbols, please see the preceding Key Dictionary.

10^x (common antilogarithm), ¶37

A

Absolute value, ¶16
Addition
- stack, ¶11, ¶12
- degrees, minutes, seconds, ¶23
- storage register, ¶59
Angular mode, ¶6
Antilogarithm
- common (10^x), ¶37
- natural (e^x), ¶36
Arc cosine, ¶19, ¶6
Arc sine, ¶55, ¶6
Arc tangent, ¶60, ¶6
Arithmetic operations
- stack, ¶11
- storage registers, ¶59, ¶7

B

Branch, ¶31

C

Cards, magnetic, ¶10
Change sign, ¶17
Clearing
- flags, ¶54
- program step, ¶21, ¶9
- program memory, ¶45
- registers, ¶50
- stack, ¶58
- whole calculator, ¶43
X register, ¶18

Codes
  defined, ¶9
  merged, ¶39

Comparisons, ¶66, ¶25

Cosine, ¶19, ¶6

D

Decimal part of number, ¶33

Decimal point
  display, ¶24
  number, ¶13

Decimal to octal, ¶42

Degrees, minutes, seconds
  addition, ¶23
  conversion, ¶22, ¶6

Degree angular mode, ¶20, ¶6

Delete program step, ¶21, ¶9

Digits, ¶12

Disable stack lift, ¶57

Display
  scientific and fixed, ¶24
  blinking, ¶4
  (low battery indication), ¶4
  W/PRGM mode codes, ¶9

E

e^x (natural antilogarithm), ¶36

Editing a program, ¶9

Enable stack lift, ¶57

Equals (test), ¶66

Errors, ¶4

Exchange x and y, ¶65

Exponent entry, ¶26

Exponential function, ¶67
Factorial function, ¶40

Flags
  setting, ¶54
  testing, ¶61

Flashing zero, ¶4

Fraction part of number, ¶33

Grad angular mode, ¶30, ¶6

Greater than (test), ¶66

Insert program step, ¶32

Integer function, ¶33

Inverse operations, ¶28
  $10^x$ (common antilogarithm), ¶37
  arc cosine, ¶19
  arc sine, ¶55
  arc tangent, ¶60
  degrees, minutes, seconds to
    decimal angle, ¶22
  $e^x$ (natural logarithm), ¶36
  fraction of number, ¶33
  octal to decimal, ¶42
  polar to rectangular, ¶51
  set flags off, ¶54
  square, ¶64
  subtract degrees, minutes, seconds, ¶23
  test flags for off, ¶61

Label, ¶35, ¶9

Last x, ¶34

Letters, ¶15

Lift (stack), ¶57
Logarithm
  common (base 10), ¶37
  natural (base e), ¶36
Loop control, ¶25, ¶54, ¶61, ¶66

M
  Magnetic cards, ¶10
  Merged codes, ¶39
  Multiplication
    stack, ¶11, ¶2
    storage registers, ¶59

N
  Negative numbers, ¶17, ¶3
  Nonprogrammable operations, ¶56, ¶45, ¶21
  No operation, ¶41
  Not equal (test), ¶66
  Number entry keys, ¶12, ¶13, ¶17, ¶26

O
  Octal to decimal, ¶42
  OFF-ON switch, ¶43

P
  Polar to rectangular, ¶51, ¶6
  Prefix keys, ¶5
  Program mode, ¶62
  Program pointer, ¶9
  Programming, ¶9

R
  Radian angular mode, ¶48, ¶6
  Recall last x, ¶49
  Recall storage register $R_n$, ¶49
Reciprocal, \[63\]
Rectangular to polar, \[51, 6\]
Return, \[53, 9\]
Roll stack down, \[47\]
Roll stack up, \[46\]
Rounding display, \[24, 4\]
Run mode, \[62\]
Run/stop, \[52\]

S

Scientific display, \[24, 4\]
Scientific notation (data entry), \[3, 26, 17\]
Set flag, \[54\]
Sign of numbers and exponents, \[17, 3\]
Sine, \[55, 6\]
Single stepping, \[56, 9\]
Skip, two step, \[25, 61, 66\]
Square root, \[64\]
Square, \[64\]
Stack
  arithmetic, \[2, 11\]
  clear, \[18, 58\]
  lift, \[57\]
  manipulation, \[27, 46, 47, 65\]
Starting a program, \[52, 15, 53, 31, 9\]
Stopping a program, \[52\]
Storing in register \(R_n\), \[59, 7\]
Storage register arithmetic, \[59, 7\]
Subtraction
  degrees, minutes, seconds, \[23\]
  stack, \[11, 2\]
  storage registers, \[59\]
Switches
  OFF-ON, \[43\]
  W/P RGM-RUN, \[62\]
Tangent, ¶60, ¶6

Testing
  flags, ¶61
  relation of x to y, ¶66
  index rₜ, ¶25

Trigonometric functions
  cosine/arc cosine, ¶19, ¶6
  sine/arc sine, ¶55, ¶6
  tangent/arc tangent, ¶60, ¶6
  rectangular to/from polar, ¶51, ¶6

Truncating to fraction/to integer, ¶33

U

User defined functions, ¶9, ¶15, ¶53

W

W/PRGM-RUN switch, ¶62