HEWLETT-PACKARD

HP-65

Quick Reference Guide

This booklet is primarily intended for reference use after you read the *HP-65 Owner's Handbook*. The Procedures section is a brief digest for the user who needs a quick review of the more important operating procedures. The Key Dictionary 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., [+], [π]) are at the front. A general Index is provided at the end.

Procedures

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

Compute	By Pressing
8 − 2 = 6	8 [ENTER↑] 2 [−]
8 ÷ 2 = 4	8 [ENTER↑] 2 [÷]

Compute: $\frac{(4\times5)}{(2+3)}-6=-2$ using the keys shown below.

Contents of Stack Registers

T						20	20				
Z					20	2	2	20		4	
Y		4	4	20	2	2	3	5	4	6	-2
X	4	4	5	20	2	2	3	5	4	6	-2
Key	4	↑	5	×	2	↑	3	+	÷	6	−

Note: ENTER↑ *is here abbreviated as* ↑ .

3. Data Entry

The number building keys are: 0 thru 9, ·, CHS, and EEX. Every other key is a number terminating key. ENTER↑ and CLX are special cases.

Entering Negative Numbers. ■ Press CHS (change sign) after keying in the positive value. **Example:** to key in −12, press 12 CHS.

Entering Big and Small Numbers *(Scientific Format).* Method: ① Key in mantissa. ② Press CHS if negative number. ③ Press EEX. ④ Key in exponent. ⑤ Press CHS if exponent is negative. ■ **Example:** To key in 5×10^{-8}, press 5 EEX 8 CHS.

Correcting Mistakes. Press CLX and reenter the number.

4. Automatic Stack Lift

If the number is terminated, the stack lifts upon the entry of a new number. ENTER↑ and CLX are exceptions to the rule.

5. Display

Controlling Display. To display x, rounded to n fixed decimal places, press DSP . n (where n = 0 thru 9). To display x in scientific notation, rounded to n places, press DSP n . ■ Display can also be set by a program.

Blinking Display. ■ Blinking display occurs when an illegal operation is attempted **Example:** 5 ÷ 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: LN, LOG, $\sqrt{x}$, SIN, COS, D.MS←, →D.MS, →OCT, $1/x$, y^x, ÷.

Multiple Decimal Points. The display also indicates low battery power (all decimal points light up).

6. Prefix Operations

Performing "Gold" Functions (Upshift). Rule: Precede gold functions by f to do the function or f^{-1} to do the inverse. ■ **Example:** Calculate log (100) = 2 by pressing 100 f LOG. ■ **Example:** Calculate antilog (2) = 100 (the inverse) by pressing 2 f^{-1} LOG.

Performing "Blue" Functions (downshift). ■ Precede blue functions by g. ■ **Example:** Calculate 5! = 120 by pressing 5 g n!.

Correcting or Cancelling a Prefix. ■ To correct a wrong prefix, merely press the correct prefix. ■ To cancel a prefix press f PREFIX. f PREFIX also cancels these additional keys: DSP, GTO, STO, RCL, LBL.

7. Angular Mode

Operations involving angles (namely, →D.MS, SIN, COS, TAN, R→P) assume the angles to be in the units (degrees, radians, or grads) of the prevailing angular mode as set by DEG (or power ON), RAD, or GRD. ■ 360 degrees = 2π radians = 400 grads.

Converting from One Angular Mode to Another:

g { DEG / GRD / RAD } f →D.MS g { DEG / GRD / RAD } f^{-1} →D.MS

8. Storage Register Operations

Storing a Number in Addressable Register R_n. ① Key in number to be stored. ② Press STO n (where n is a digit 1 thru 9).

Recalling a Number from R_n. Press RCL n (where n is a digit 1 thru 9).

Note: R_9 is used to store intermediate results for the *trigonometric functions and their inverses:* SIN, COS, TAN, and rectangular to polar conversion R→P The *relational tests:* x≠y, x≤y, x=y, x>y store Last X in R_9.

Doing Storage Register Arithmetic.

Subtraction. ($r_n - x \rightarrow R_n$): STO − n

Addition. ($r_n + x \rightarrow R_n$): STO + n

Multiplication. ($r_n \times x \rightarrow R_n$): STO × n

Division. ($r_n \div x \rightarrow R_n$): STO ÷ n

where n is a digit key 1 thru 9. The value in X is unchanged in these operations.

9. Clearing Registers

■ Press CLX to clear X-register. ■ Press f STK to clear entire stack (X, Y, Z, and T) ■ Press f REG to clear all storage registers (R_1 thru R_9) ■ Power ON clears all registers.

10. Programming

Program Memory. 100 usable locations and a top of memory marker. All keys can be stored in memory in W/PRGM mode except SST, f PRGM, and g DEL.

Keycodes. Keys are stored in memory as codes. The codes for keys 0 thru 9 and the functions

on these keys are 00 - 09. Top of memory shows code 00 00. For other keys, the code denotes the row and key position in the row. **Example:** Code for RTN (row 2, 4th key) is 24.

To Write a Program: identify the beginning by LBL followed by the top row key that is to call it. End the program with RTN. **Example:** LBL E (ENTER↑ ENTER↑ × ×) RTN. The keys in parentheses calculate x^3.

To Key in a Program: 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 Program (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.

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. 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. To delete a step or correct a mistake, press g DEL in W/PRGM mode.

Direct Branching. A direct branch in a program consists of the GTO key and a digit key 0 thru 9 or a program control key A thru E. Each such direct branch should be paired with a corresponding label. When the calculator executes a direct branch, the pointer searches downward in memory for the label from the GTO. Program execution continues at the corresponding label.

Subroutine Branching. Any program controlled by one of the program control keys (A thru E) can be called in another program by the program control key. The subroutine is executed by a secondary pointer. Upon executing the RTN at the end of the subroutine, program execution transfers back to the main program and the main program pointer is reactivated. Program execution then continues sequentially in the main program.

Conditional Testing. Nine different instructions modify program execution depending on conditions with the program. They are: DSZ, f TF1, f TF2, f^{-1} TF1, f^{-1} TF2 and x≠y, x≤y, x=y, x>y. If the condition is met, program

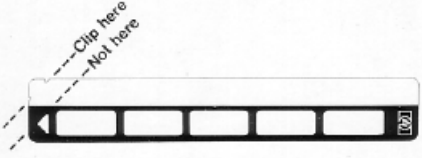
execution continues sequentially. If the condition is not met, the program pointer skips two steps before continuing.

Both overflow and underflow of a register will stop a program.

11. Using Magnetic Cards

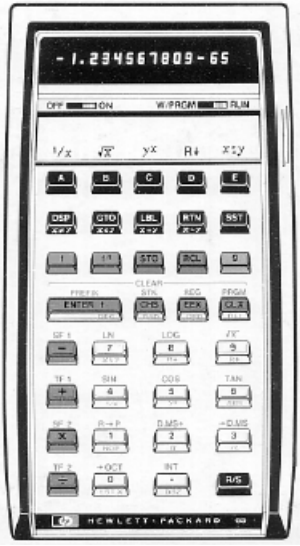
Reading a Pre-recorded Card. ■ Set W/PRGM-RUN switch to RUN. ■ Insert the card in the right lower slot. ■ If the card does not read properly, display will flash and program memory will be cleared (but with no effect on registers). Press RTN and reinsert the card.

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.



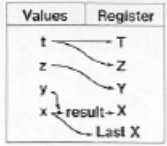
Key Dictionary

This section contains reference entries for all keys with their associated symbols. In addition, the following entries are included. Insert, Last X, Merged Codes, OFF-ON Switch, Stack Lift, W/PRGM-RUN Switch.



12.

−, **+**, **×**, **÷** **Arithmetic Operations.** Calculate respectively: (y — x), (y + x), y × x), (y ÷ x). ■ Save x in Last X. ■ Drop stack as follows:



For storage register arithmetic, refer to paragraph 59.

13.

0 **thru** **9** **Digits.** Digits are number building keys and are used: ① to enter data (3), ② to specify registers, ③ to label program steps, ④ to specify displayed decimal places.

14.

. **Decimal Point.** The **.** key is a number building key. It is also used to specify fixed decimal display. (Refer to paragraph 25.)

15.

π **Recalls** π (3.141592654) to the X-register.

16.

A, **B**, **C**, **D**, **E** **Program Control Keys.** These keys control your programs if they begin

with corresponding labels. They can be used from the keyboard to call a program or in a program to call a subroutine. ■ Used as suffix for LBL and GTO. ■ **Note:** Power ON automatically inserts 5 programs in memory. Press f PRGM in W/PRGM mode to clear them.

17.

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

B
C Refer to paragraph 16.

18.

CHS **Change Sign.** CHS is a number building key. It changes the sign of the number or exponent in the X-register. Press CHS after keying in a number to change its sign. To change the sign of an exponent, press CHS before or after keying in the exponent. ■ Once a number is terminated the CHS key cannot change the exponent sign. In such a case CHS changes the sign of the number.

19.

CLX **Clear X-Register.** CLX replaces the number in the displayed X-register with zero and prepares the X-register for a new number. The new number then writes over the zero in X.

20.

COS **Cosine** (f prefix) / **Arc Cosine** (f^{-1} prefix) (principal value, $0° \leq$ result $\leq 180°$ or equivalent in radians or grads). ■ Saves x in Last X. ■ Stores intermediate result in R_0. ■ x less than -1 or greater than $+1$ gives error (blinking zero) for arc cosine.

D. See paragraph 16.

21.

DEG **Set Degree Angular Mode.** ■ Affects angle operations (→D.MS , SIN, COS, TAN, R→P).

22.

DEL **Delete Program Step.** ■ With W/PRGM-RUN switch in W/PRGM position, g DEL ① deletes the program step denoted by the program pointer, ② moves all the following steps up one step, and ③ inserts a g NOP code in the vacated bottom position of memory. ■ Inoperative during run mode, (g DEL acts as CLX) ■ g DEL can be used to back up the pointer (after which you reinsert the deleted codes). ■ g DEL loses the bottom memory step. ■ If the program pointer is at the bottom (i.e., 2 dashes in the display), g DEL deletes two locations.

23.

[→D.MS] **Converts to** ([f] prefix) or **from** ([f⁻¹] prefix) **Degrees, Minutes, Seconds.** ■ Converts the contents of the X-register (in degrees, radians, or grads) to the form DDDDD.MMSS (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.

24.

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

25.

[DSP] **Display.** ■ [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] thru [9]. ■ Does not alter internal value of x. ■ Power ON sets [DSP] [•] [2] (0.00). ■ If x is too small for

a specified display, zero is displayed (with minus sign if the number is negative). If x is too large for the specified format, DSP 9 format is used.

26.

DSZ **Decrement and Skip on Zero.** DSZ is a conditional operation. It subtracts 1 from a value in register R_s, then tests for a non-zero value. If the condition is met ($r_s \neq 0$) execution continues sequentially. If the condition is not met ($r_s = 0$), the program skips two steps before continuing execution. DSZ does not work if r_s falls outside the range $-10^{10} \leq r_s \leq 10^{10}$ and (in general) is not designed to work for non-integer values less than one.

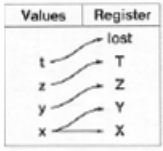
E See paragraph 16.

27.

EEX **Enter Exponent.** EEX is a number building key. It 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, EEX sets up a mantissa of 1.

28.

ENTER↑ Prepares the X-register for a new number by terminating the old number and copying it into the Y-register. A new number then writes over the number in the X-register without lifting the stack.



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

29.

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 preceded by f^{-1}. ■ No inverses are defined for clear functions (4th row keys). For these keys f^{-1} or f gives the clear function. ■ To cancel an unwanted f or f^{-1}, press PREFIX.

30.

g **Downshift Prefix.** When g is followed by a key having a blue symbol below it, the operation denoted by the symbol is performed. ■ To cancel an unwanted g, press f PREFIX.

31.

GRD **Sets Grad *Angular Mode.*** Affects angle operations (→D.MS , SIN , COS , TAN , R→P).

32.

GTO **Go to.** When followed by a digit (0 thru 9) or a letter (**A** thru **E**), **GTO** advances the program pointer downward to the first occurrence of the corresponding program label. If executed from the keyboard, the program does not execute. If executed in a program, execution continues at the label.

33.

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: ① For PRGM , DEL , **SST** ② For the second key of a merged code. ③ When the pointer is at the bottom.

34.

INT **Truncates x to Integer** (f prefix) **or fraction** (f' prefix). ■ Saves x in Last X. ■ Retains the sign of the number.

35.

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, press 9 [LST x]. ■ The following save x in Last X before performing their functions: + − × ÷ [ABS] [COS] [→D.MS] [D.MS+] [INT] [LN] [LOG] [→OCT] [R→P] [SIN] [TAN] [n!] [√x] [1/x] [y^x] ■ Note that [CLX] does not affect the Last X register.

36.

LBL Label. [LBL] identifies its suffix (a digit [0] thru [9], or top row key, [A] thru [E]) as a label in a stored program. Most programs should begin with [LBL] ([A] thru [E]) and end with [RTN]. Such a program is executed from the keyboard by pressing one of the program control keys [A] thru [E]. A direct branch to the labelled part of the program can then be made by executing [GTO] followed by the same suffix.

37.

[LN] **Natural Log(x)** ([f] prefix) **or e^x** ([f] [1/x] prefix). ■ Saves x in Last X. ■ [f] [LN] gives error (blinking zero) if x is zero or negative.

38.

LOG **Common Log(x)** (f prefix) **or 10^x** (f^{-1} prefix). ■ Saves x in Last X. ■ f LOG gives error (blinking zero) if x is zero or negative.

39.

Merged Codes. Program codes for the following are merged with their respective prefix codes: LSTX , NOP, x≷y , R↓ , R↑ , x≠y , x≤y , x=y , x>y , 1 thru 8 when prefixed by STO or RCL . ■ Example: g LSTX in program mode is merged and displayed as 35 00; STO 5 as 33 05, etc. For explanation of codes, see paragraph 10.

40.

n! **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.

41.

NOP **No Operation.** When the program pointer executes a g NOP , no operation occurs. ■ Useful as a filler in tests. ■ f PRGM in W/PRGM mode clears the entire memory to g NOP (merged code 35 01).

42.

[→OCT] **Convert Integer x to/from Octal.** ■ Saves x in Last X. ■ Non-integer or x larger in magnitude than 1073741823_{10} gives error (blinking zero).

43.

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 programs at the top of memory that are callable from the top row keys (as marked in white above these keys).

44.

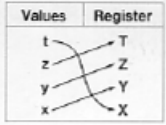
[PREFIX] **Clear Prefix.** When preceded by [f] or [f⁻¹], [PREFIX] cancels an unwanted prefix key. ■ If a wrong prefix key is pressed when another prefix key is wanted, the error can be corrected by simply pressing the correct prefix and proceeding from there.

45.

[PRGM] **Clear Program Memory.** In program mode, [f] [PRGM] (or [f⁻¹] [PRGM]) clears the entire program memory to "no operation" codes (35 01), setting the pointer at the top of memory.

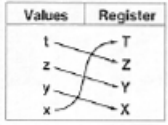
46.

R↓ **Roll Stack Up.**



47.

R↑ **Roll Stack Down.**



48.

RAD **Set Radian Angular Mode.** ■ Affects angle operations (→D.MS, SIN, COS, TAN, R→P).

49.

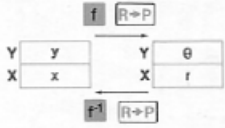
RCL **Recall.** Recalls storage register R_n (n is a digit 1 thru 9) to the X-register.

50.

REG **Clear Registers.** When preceded by f or f^{-1}, REG clears storage registers R_1 thru R_9 to zero.

51.

R→P **Rectangular to Polar.** ■ Transforms the respective contents of the X- and Y-registers from rectangular form (x, y) to polar form (r, θ) (f prefix) or the inverse (f^{-1} prefix). ■ Saves x in Last X. ■ Stores intermediate results in R_0.



52.

R/S **Run/Stop.** As a program step R/S interrupts program execution at an intermediate point, allowing you to key in data, make additional calculations, etc. From the keyboard, R/S will start a program or halt a running program. R/S can also be used as a program control key. If you plan to initiate execution of a program with R/S, a R/S must be used as a program step to halt the program. ■ If a 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. Digits occurring as program steps immediately following a

R/S should be separated from the R/S by an ENTER↑.

53.

RTN **Return.** ■ If executed from the keyboard, RTN merely resets the program pointer to the top of memory. RTN should be used to end all programs beginning with labels. If a program is executed from the keyboard using program control keys A thru E, execution stops at the RTN. If the program is used as a subroutine, execution of RTN deactivates the secondary pointer. Program execution automatically transfers back to the step following the subroutine call in the main program.

54.

SF1, SF2 **Set flag 1, Set flag 2.** ■ f SF1 sets flag 1 ON while f⁻¹ SF1 sets it OFF. ■ f SF2 performs similarly, but using flag 2. To test the flags, refer to paragraph 61.

55.

SIN **Sine** (f prefix) /**Arc Sine** (f⁻¹ prefix). ■ Arc sine calculates the principal value ($-90^o \leq$ result $\leq +90^o$ or equivalent in radians or grads). ■ Saves x in Last X. ■ Stores intermediate results in R_0. ■ x less than -1 or greater than $+1$ gives error (blinking zero) for arc sine.

56.

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 subroutine, the entire subroutine executes (as one step) before returning control to the keyboard. **SST** does not terminate numbers.

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 respectively higher registers Y, Z and T (t is lost) for future use. Refer to paragraph 4. If a number is terminated it will be lifted automatically in the stack upon the entry of a new number.

58.

STK **Clear Stack.** (f or f^{-1} prefix).

59.

STO **Store.** **STO** n copies the contents of register X into storage register R_n (n = a digit 1 thru 9. To perform an arithmetic

function (+, −, ×, ÷) of x and a storage register R_n, insert the corresponding arithmetic key between STO and n.

STO + n	adds	$(r_n + x) \rightarrow R_n$
STO − n	subtracts	$(r_n - x) \rightarrow R_n$
STO × n	multiplies	$(r_n \times x) \rightarrow R_n$
STO ÷ n	divides	$(r_n \div x) \rightarrow R_n$

■ Stack registers and Last X are unchanged. ■ Storage arithmetic codes are unmerged.

60.

TAN **Tangent** (f prefix)/**Arc Tangent** (f⁻¹ prefix). ■ Arc tangent calculates the principal value ($-90° \leq$ result $\leq +90°$ or equivalent in grads or radians). ■ Saves x in Last X.

61.

TF1, TF2. **Test Flag 1, Test Flag 2.** These are conditional tests. f TF1 tests flag 1 to see if it is ON. f⁻¹ TF1 tests flag 1 to see if it is OFF. Similarly, f TF2 tests flag 2 to see if it is ON. And f⁻¹ TF2 tests flag 2 to see if it is OFF. In each case, if the condition is met, program execution continues sequentially. If the condition is not met, the program pointer skips two steps before continuing. To set the flags, refer to paragraph 54.

62.

W/PRGM-RUN Switch. ■ Set to W/PRGM position in order to: ■ create and edit a stored program or ■ write program memory on a magnetic card. ■ Set to RUN position in order to: ■ read a magnetic card into program memory ■ do calculations ■ execute stored programs.

63.

[1/x] **Reciprocal of x.** ■ Saves x in Last X. ■ Gives error (blinking zero) if reciprocal of 0 is attempted.

64.

[√x] **Square Root** ([f] prefix) / **Square** ([f²] prefix). ■ Saves x in Last X. ■ For [f] [√x], negative x gives error (blinking zero).

65.

[x≷y] **Exchange x and y.**

Values	Register
y → x	Y → X

66.

[x≠y], [x≤y], [x=y], [x>y] **Numerical Comparisons.** These are conditional tests which compare values in the X- and Y-registers. If the test condition is met, program execution

continues sequentially. If the condition is not met, the program pointer skips two steps before continuing.

67.

y^x **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 arithmetic operations). ■ Negative or zero value of y gives error (blinking zero).

Index

The numbers in the following index refer to paragraphs, not pages. To reference keys, please use the preceding Key Dictionary.

D

E

M

N

O

P

R

S

Sales and service from 172 offices in 65 countries.
19310 Pruneridge Avenue, Cupertino, CA 95014

For Additional Sales and Service Information Contact Your Local Hewlett-Packard Sales Office or Call 408/996-0100 (Ask for Calculator Customer Service).

00065-90203 Rev. 8/74 Printed in U.S.A.