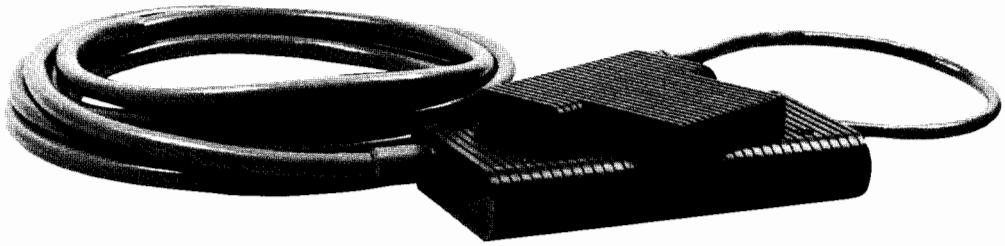
Hewlett-Packard 9815A/S Calculator
98133A BCD Interface
Operating and Service

Operating and Service

HP 98133A BCD Interface



Hewlett-Packard Desktop Computer Division
3404 East Harmony Road, Fort Collins, Colorado 80525
(For World-wide Sales and Service Offices see back of manual.)

HP Computer Museum
www.hpmuseum.net

For research and education purposes only.

TABLE OF CONTENTS

Section 1: Installation

Section 2: Operation

Section 3: Service

Appendix

Figures

Tables

1
Installation

Introduction

The HP 98133A BCD Interface allows you to connect an HP 9815A Calculator to a wide variety of peripheral devices. The interface provides the calculator with 40 bit-parallel input lines and 8 bit-parallel, character serial output lines. The interface transfers data in a "half-duplex" fashion; that is, it can input and output data, but not both at the same time. The interface provides buffer-storage for both input and output data and all lines are compatible with standard TTL levels.

This manual describes the installation, operation and servicing of the BCD Interface.

Technical Specifications

Temperature Range

5° C to 45° C Ambient.

Power

Provided by the calculator.

Cable Length

The interface has overall length of approximately 4.6 meters (15 feet) and is equipped with an unterminated cable. Cable wire colors, "pin outs", and material list are in Section 3.

Data Input Lines

Data is input by the interface via 40 input lines. These lines are divided into ten 4-bit digits. Each digit has four lines through which binary data is input.

Data Output Lines

Eight output lines with open-collector SN 7406 TTL inverters are available. Data is output in negative-true logic and remains valid on the output lines after an output operation until it is modified by another output.

Control Lines

Device Control (CTL1 and CTL2) – Initiates an I/O operation when driven to logical 1 and terminates data transfer when it goes to logical 0 (Handshake Mode 1 and 2 only). The CTL1 line is used for input operations and the CTL2 line is used for output operations. The logic level used by the CTL lines is set by the HANDSHAKE Instruction (see Section 2).

Device Ready (FLG1 and FLG2) – The peripheral device indicates "output data accepted" or "input data valid" by forcing the line to logical 1 (Handshake Mode 1 only). The FLG1 line is used for input operations and the FLG2 line is used for output operations. The logic level used by the FLG lines is set by the HANDSHAKE Instruction (see Section 2).

I/O Status (I/O) – Indicates that the calculator has started an input operation (high) or an output operation (low).

Signal Level

Data is input in the logic level specified by bit 4 of the data code. Data is output in negative-true logic only. The logic levels used by the CTL and FLG lines can be either positive-true or negative-true TTL logic levels (see the "HANDSHAKE Instruction" in Section 2). Positive-true logic means that a logical 1≥2.4 V = high and a logical 0≤0.7 V = low. Negative-true logic means that a logical 1≤0.7 V = low and a logical 0≥2.4 V = high.

Plug-In Procedure

Calculator Option 002

Before an interface can be installed the calculator must be equipped with option 002, two channel I/O. This option provides the calculator with the interface connectors and internal I/O compatibility.

The BCD Interface can be plugged into either I/O channel on the calculator back-panel. Before plugging in the interface, however, be sure that the calculator is switched off – if not, the calculator will not recognize I/O operations which are executed or programmed. After the interface is plugged in, switch the calculator back on.

If "SELECT CODE ERR" is printed when the calculator is switched on, both interfaces in the calculator are set to the same select code. If this occurs, switch the calculator off immediately and change the select code on one of the interfaces.

Memory Usage

When the BCD interface is plugged in, it uses 16 steps of program memory. This loss of 16 steps is indicated by the remaining number of steps shown in the display when the Program mode is set.

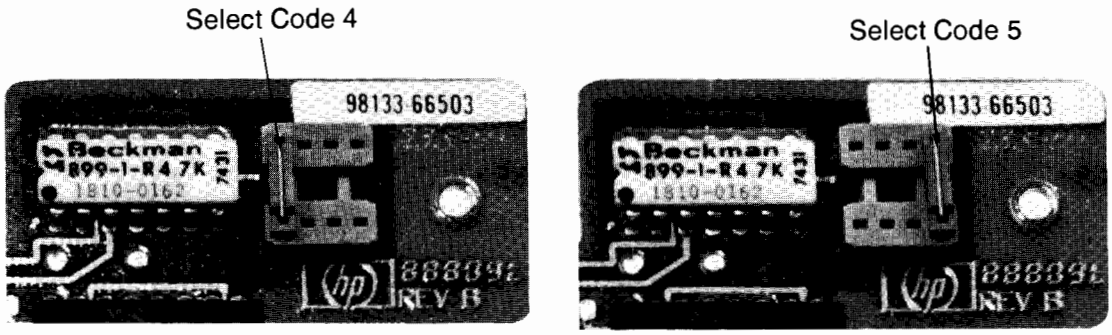
Select Code

Each interface connected to the calculator has a unique address, or select code, so that you can specify which interface should respond to each instruction. This select code is a one-digit number and must be included in each interface instruction.

Although the BCD interface is preset to select code 4 at the factory, you can change the setting to 5 by following this procedure.

1. Switch the calculator and the peripheral device off.
2. Disconnect the interface from the calculator. Remove the four screws on the top of the interface connector and remove the plastic cover.
3. Locate the select code plug (see the next figure), and move it to its alternate position.

4. Replace the plactic cover and secure it with the four screws. Then place a new select code label on the side of the connector to indicate the new code.
5. Reconnect the interface to the calculator; then turn the calculator and the peripheral on. Verify interface operation by executing an I/O instruction (or running a program) which specifies the new select code.

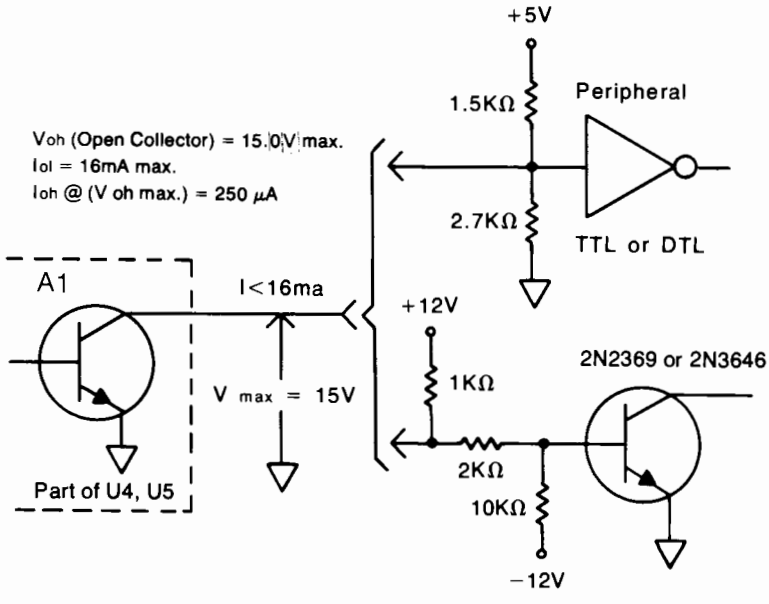


Interface Select Code

Typical Peripheral Circuits

Receiving Circuits

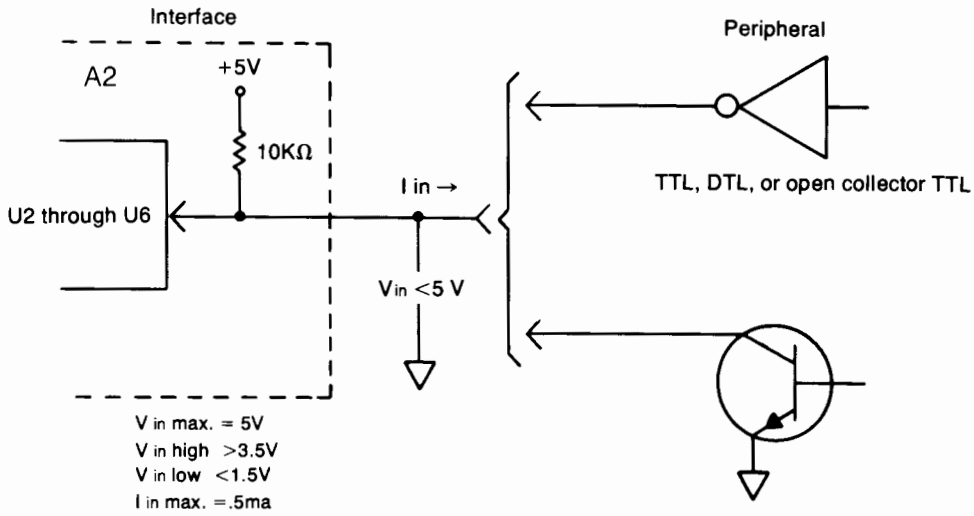
Each interface output line has a device with an open-collector output. The current-sinking capability of each device is 16 mA and the breakdown voltage is 15 V. Since each output device has an open collector, the peripheral receiving circuit must have a positive pull-up voltage (not to exceed 15 V) and must be restricted to sourcing (back to the transmitter) less than 16 mA. Typical receiving circuits are shown here.



Typical Receiving Circuits

Transmitting Circuits

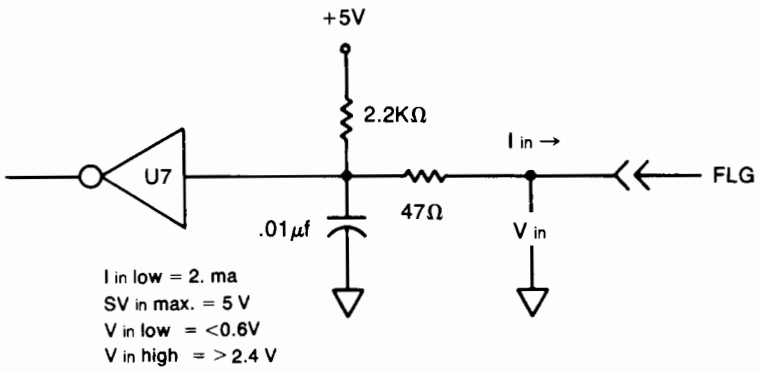
Each data-input line on the interface goes directly to a data shift register. The input voltage must not exceed 5 V.



Typical Transmitting Circuits

If a peripheral device without isolated or buffered output lines is connected to the interface and is switched on while the calculator is switched off, the peripheral circuitry may become overloaded due to the low impedance load on the interface input lines when the calculator is switched off. This situation can be eliminated by either keeping the calculator switched on or disconnecting the peripheral device from the interface when the calculator is off.

The FLAG signals are received by a TTL Schmitt trigger gate (A2U7) and the low-pass filter network shown next. Either of the peripheral transmitting circuits just shown can be used as a flag transmitter.



Interface Flag Circuit

Input Data Stability

Each data character placed on the interface input lines must be settled 2 μs before the "data ready" signal (FLG line) is transmitted. Then the data must be held stable until the handshake is completed.

2
Operation

Introduction

This section explains the calculator instructions used to control the interface. The preceding section should be read before continuing, to insure that the interface is properly connected.

> **NOTE**
> This section assumes that the reader is familiar with the operation of the calculator, as described in the HP 9815A Operating and Programming Manual.

The interface provides the calculator with sixteen key-sequence instructions. These instructions can be executed from either the keyboard or a program. The instructions can be used to specify the type of I/O control used for data transfer, the format of the incoming data samples and to check each sample for specified conditions. Data can be input and stored in single or multiple samples and recalled for use as either complete samples or individual digits. Data can be output in either binary or ASCII code.

A brief summary of the instructions begins on page 6 and a detailed description of each instruction comprises the remainder of the section.

Definition of Terms

The following terms are used in this section in describing the function of the interface instructions.

Digit - One BCD coded character.

Digit A through Digit J - Designates the 10 input digits; Digit A is the most significant.

Byte - One 8-bit binary number or two BCD coded digits in either binary or decimal representations.

Sample - A number (in fixed-point or exponential form) consisting of a specified number of characters or BCD digits.

Block - The total data latched during one input operation and consists of 5 bytes (10 BCD digits).

Illegal Characters - The BCD characters 10, 11, 12 and 13 are interpreted as illegal characters by the calculator. See the Appendix for an explanation of BCD code.

Data Code - The data code uses the specified digit's four bits to specify the following for each incoming data sample:

- **Bit 1** - Sign of the exponent (logical 1 = +, logical 0 = −).
- **Bit 2** - Sign of the mantissa (logical 1 = +, logical 0 = −).
- **Bit 4** - Logic sense (voltage ≥2.4V = positive true, voltage ≤.7V = negative true).
- **Bit 8** - Overload indicator (logical 1 = overload, logical 0 = not overload).

Summary of the Interface Instructions

I/O Control

HANDSHAKE (HNDSK)

CALL · ALPHA · (select code) K

Specifies the I/O control mode (handshake) and sets the logic level of the CTL1, CTL2, FLG1 and FLG2 lines. See page 8.

Data Control

FORMAT (FRMT)

CALL · ALPHA · (select code) L

Specifies the form of the incoming data for either "single sample" (one input device) or "split sample" operations (data from two input devices). See page 11.

MASK

CALL · ALPHA · (select code) M

Enables an auto check for either overload or illegal characters or both. See page 13.

Input Instructions

START

CALL · ALPHA · (select code) F

Sets the interface to automatically input and hold a data sample when it becomes available from a peripheral device. See page 14.

STATUS (STAT)

CALL · ALPHA · (select code) G

Checks the peripheral device or the handshake status. See page 14.

READ

CALL · ALPHA · (select code) A

Inputs one data block into X and Y according to the currently set FORMAT instruction. See page 14.

Automatic Data Input

START, STATUS and READ instructions can be used to program an automatic data-input capability. See page 14.

BURST

CALL · ALPHA · (select code) C

Inputs and stores data blocks directly into data registers at a maximum rate of 2000 data blocks per second. See page 15.

PACK

CALL · ALPHA · (select code) H

Inputs and stores data with either two 8-digit samples or four 4-digit samples per register at a rate of 1000 blocks per second. See page 17.

INPUT

CALL · ALPHA · (select code) E

Inputs data in 8-bit binary bytes and stores 8 bytes per data register. See page 20.

SAMPLE (SAMPL)

CALL · ALPHA · (select code) B

Recalls a data block that was stored by a BURST intruction. The sample or samples are entered in decimal form in X and Y. See page 15.

DECODE (DCODE)

CALL · ALPHA · (select code) D

Converts a specified number of data blocks previously stored by BURST instructions into their decimal form (single sample only). See page 17.

UNPACK (UNPAK)

CALL · ALPHA · (select code) I

Recalls data, previously stored in a register by a PACK instruction, into the stack. See page 19.

BYTE

CALL · ALPHA · (select code) N

Recalls or stores individual binary bytes of data in a register. See page 21.

Output Instructions

OUTPUT (OUTPT)

CALL · ALPHA · (select code) J

Outputs 8-bit-binary bytes of data from data registers. See page 21.

WRITE BYTE (WBYTE)

CALL · ALPHA · (select code) O

Outputs the binary equivalent of the decimal value in X to a peripheral device. See page 22.

WRITE ALPHA (WRT∝)

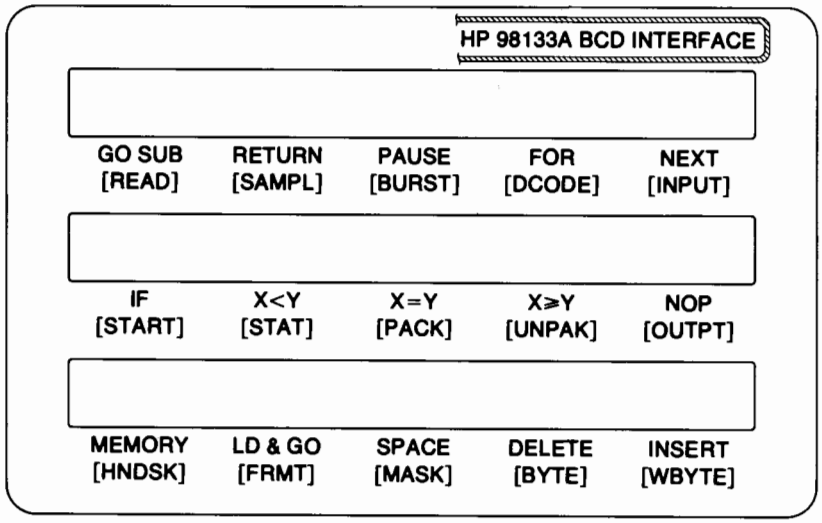
CALL · ALPHA · (select code) CALL · ALPHA ·

Sets the calculator to an Alpha mode which redefines each key to output a specified ASCII-coded character. See page 22.

All output instructions send data in negative-true logic.

Key Overlay

A key overlay (HP Part No. 7120-5354) showing the BCD Instructions is provided with the interface. The overlay (shown below) should be placed over the keys A through 0.



Key Overlay

I/O Control Instructions

HANDSHAKE Instruction

CALL ALPHA- (select code) K

Stack

Handshake Mode ±(0, 1 or 2) → Y
±(any value) → X

The I/O control mode and the logic levels used on the CTL and FLG lines are specified by the HANDSHAKE instruction (HNDSK).

The handshake-mode parameter specifies the handshake mode as follows:

- 0 specifies no handshake.
- 1 specifies a complete handshake.
- 2 specifies a pulsed handshake.

Timing diagrams and an explanation of each handshake mode begin below.

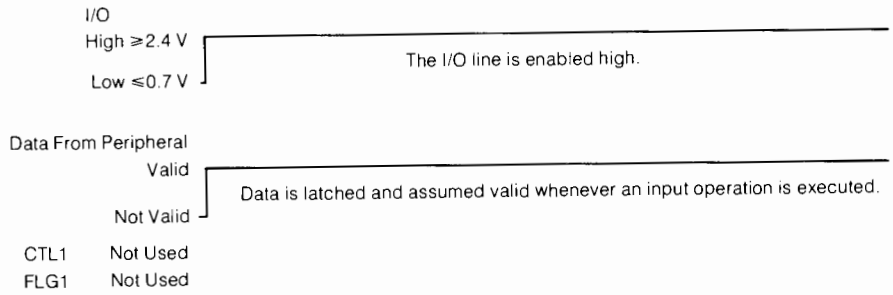
The logic level used by the CTL1 and CTL2 lines is set by the sign of the Handshake-mode parameter in Y. Negative-true logic is set with a negative parameter and positive-true logic is set with a positive parameter.

The logic level used by the FLG1 and FLG2 lines is set by the sign of the value in X.

The calculator, when first switched on, automatically sets a mode 1 (complete) handshake with positive-true CTL and FLG logic levels.

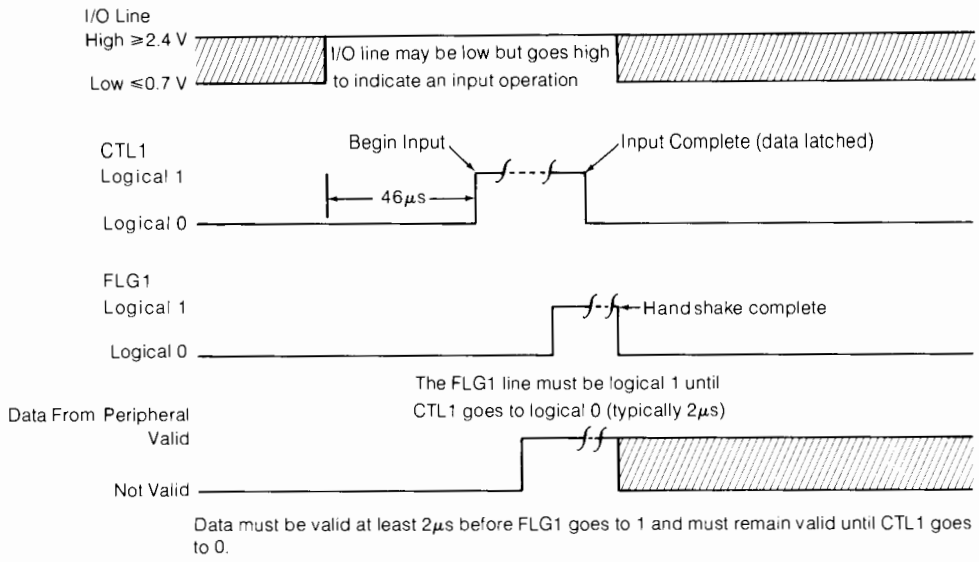
Input Timing Diagrams

Handshake Mode 0 (No Handshake)



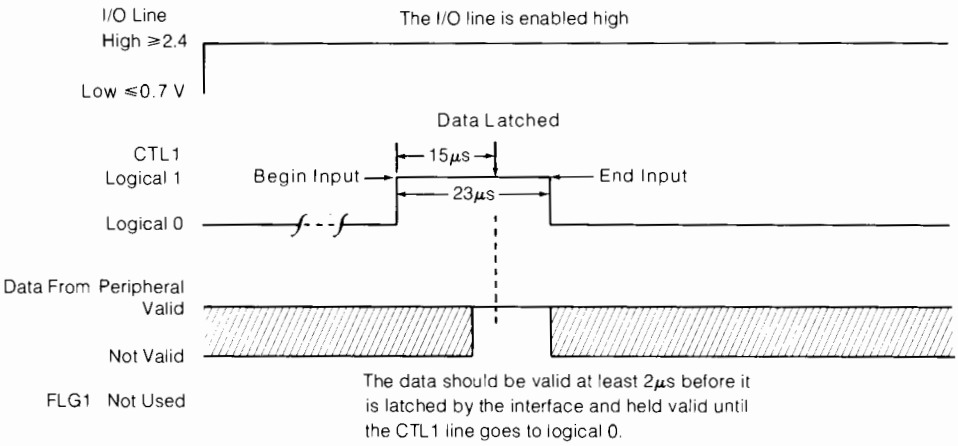
Handshake mode 0 does not use the CTL1 and FLG1 lines. The calculator sets the I/O line high (≥2.4 V) when the mode is set. When an input instruction is executed, whatever data is on the input lines is latched.

Handshake Mode 1 (Complete Handshake)



Handshake mode 1 is a complete peripheral-interface handshake. The calculator initiates the handshake by driving the I/O line high. After 46μs, the calculator drives the CTL1 line to a logical 1. The peripheral device should have the data sample valid at least 2μs before driving the FLG1 line to a logical 1. The interface then latches the data and drives the CTL1 line to a logical 0. The FLG2 pulse must be held at logical 1 until the CTL1 line goes to logical 0 (typically 2μs).

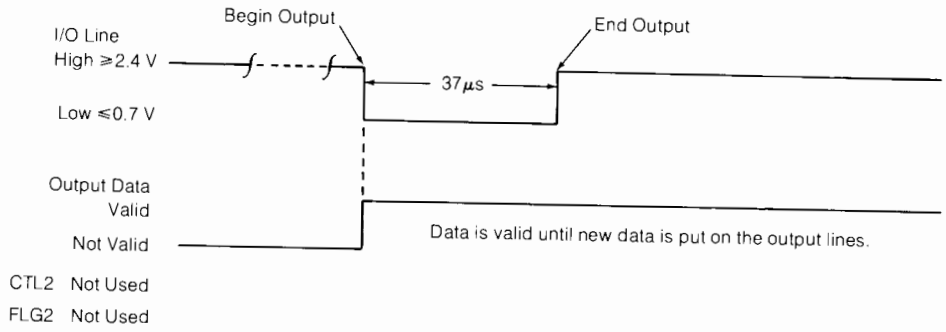
Handshake Mode 2 (Pulsed Handshake)



Handshake mode 2 is a pulsed interface handshake. The calculator enables the I/O line high (≥2.4 V) when the mode is set. When an input instruction is executed, the calculator drives the CTL1 line to logical 1 for a period of 23μs. The interface latches a data sample 15μs after the leading edge of theCTL1 pulse. The peripheral should have valid data on the input lines at least 2μs before the latch occurs, and the data should be valid until the calculator drives CTL1 line to logical 0. The FLG1 line is not used.

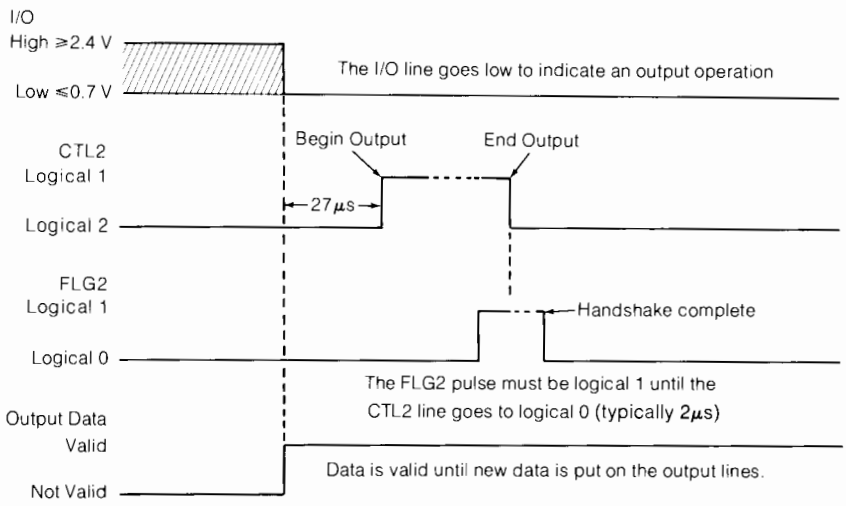
Output Timing Diagrams

Handshake Mode 0 (No Handshake)



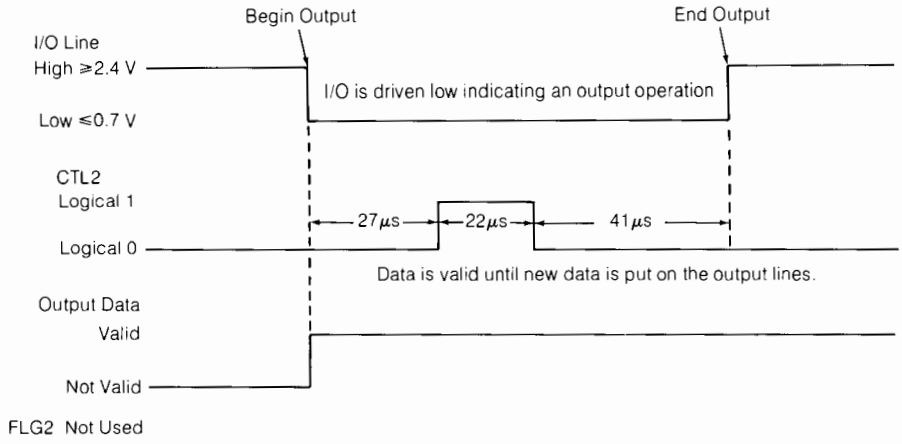
Handshake mode 0 does not use the CTL2 and FLG2 lines for output operations. The calculator drives the I/O line to a logical 0 and puts valid data on the output lines. After a period of 37μs the calculator drives the I/O line to logical 1, which ends the calculator output operation. The data remains valid on the output lines until new data replaces it.

Handshake Mode 1 (Complete Handshake)



Handshake mode 1 is a complete interface-peripheral handshake. The calculator drives the I/O line low to initiate an output operation and puts data on the output lines. After 27μs, the calculator drives the CTL2 line to logical 1. The CTL2 line remains at logical 1 until the peripheral device drives the FLG2 line to a logical 1. The calculator then drives the CTL2 to logical 0, ending the calculator output operation. The FLG2 line can be driven to logical 0 any time after the CTL2 line goes to logical 0 (typical minimum FLG2 pulse is 2μs). The data on the output lines remains valid until new data replaces it.

Handshake Mode 2 (Pulsed Handshake)



Handshake mode 2 is a pulsed interface handshake. The calculator inititates the output operation by driving the I/O line low and puts data on the data output lines. 27μs after the leading edge of the I/O pulse, the calculator drives the CTL2 line to logical 1 for a period of 22μs then drives it to logical 0. 41μs after the trailing edge of the CTL2 pulse, the I/O line is driven high to end the output operation. The data remains valid on the output lines until new data replaces it. The FLG2 line is not used.

Data Control Instructions

FORMAT Instruction

[CALL ALPHA] (select code) [L]

Data Mode (0 or 1) → Z
±(Number of Digits for Sample 1) → Y
±(Number of Digits for Sample 2) → X

The FORMAT instruction (FRMT) specifies the form of the data for single-sample operations (one input peripheral) or split-sample operations (two input peripherals).

The Z parameter specifies the Data Mode Type. 0 specifies that one data code (digit A) will apply to the entire data block. 1 specifes that two data codes (digits A and B) will apply; digit A containing the data code for Sample 1 and digit B containing the data code for Sample 2.

The Y parameter specifies the number of digits in the mantissa of Sample 1.

The X parameter specifies the number of digits in the mantissa of Sample 2.

The sign of the X and Y parameters determines if an extra digit should be input and used as an exponent:

- + specifies a fixed-point (no exponent) form.
- − specifies an exponential form.

The calculator, when first switched on, automatically sets a FORMAT of Data Mode 0 with 9 digits in fixed-point form for sample 1 and 0 digits for sample 2.

The following table shows the maximum range for the X and Y parameters.

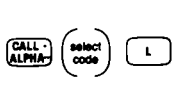
Format Parameter Range

Data Mode 0 One data code (digit A) is used for both samples; 9 data digits maximum; n equals the number of digits specified for sample 1.		
Case 1 - Both samples are in a fixed-point form.	Sample 1 Sample 2	$(n) \rightarrow Y$ $(9-n) \rightarrow X$
Case 2 - The first sample is an exponential form and the second sample is a fixed-point form.	Sample 1 Sample 2	$-(n) \rightarrow Y$ $(8-n) \rightarrow X$
Case 3 - The first sample is a fixed-point form and the second sample is an exponential form.	Sample 1 Sample 2	$(n) \rightarrow Y$ $-(8-n) \rightarrow X$
Case 4 - Both samples are in an exponential form.	Sample 1 Sample 2	$-(n) \rightarrow Y$ $-(7-n) \rightarrow X$

Data Mode 1 Two data codes (digit A is used for sample 1 and digit B is used for sample 2); 8 data digits maximum; n equals the number of digits specified for sample 1.		
Case 1 - Both samples are in a fixed-point form.	Sample 1 Sample 2	$n \rightarrow Y$ $(8-n) \rightarrow X$
Case 2 - The first sample is an exponential form and the second sample is a fixed-point form.	Sample 1 Sample 2	$-n \rightarrow Y$ $(7-n) \rightarrow X$
Case 3 - The first sample is a fixed-point form and the second sample is an exponential form.	Sample 1 Sample 2	$(n) \rightarrow Y$ $-(7-n) \rightarrow X$
Case 4 - Both samples are in an exponential form.	Sample 1 Sample 2	$-(n) \rightarrow Y$ $-(6-n) \rightarrow X$

To use either format mode to input a single sample, enter the number of digits for that sample into Y and a zero into X. This will cause the calculator to ignore the data on any remaining input lines after the first sample is input.

Example 1: The FORMAT instruction specifies the following parameters:



$0 \rightarrow Z$
$-4 \rightarrow Y$
$3 \rightarrow X$

The digits (A through J) will be interpreted as follows:

Sample 1 - A (data code) B C D E Exponent F
Sample 2 - A (data code) G H I

Since the total digits for both samples is less than 9, any excess digits (in this case only the digit J) are ignored.

Example 2: The FORMAT instruction specifies the following parameters:

[CALL ALPHA] [select code] [L] $1 \rightarrow Z$, $3 \rightarrow Y$, $-4 \rightarrow X$

The digits (A through J) will be interpreted as follows:

Sample 1 - A (data code) C D E
Sample 2 - B (data code) F G H I Exponent J

MASK Instruction

[CALL ALPHA] [select code] [M] Mask Code $\rightarrow$ X

The MASK instruction enables an automatic check of incoming data for an overload condition and illegal characters.

The instruction also specifies the action to be taken (set flag 4 and abort reading) if the specified condition is found. The table below shows the conditions checked and the resulting actions that each mask code specifies.

When the calculator is first switched on, the automatic check is disabled.

Table of Mask Code Parameters

Mask Code	Check	Action
5	Overload condition	Set flag 4
6	Illegal characters	Set flag 4
7	Overload condition or illegal characters	Set flag 4
8	Overload condition	Abort reading
10	Illegal characters	Abort reading
11	Overload condition or illegal characters	Abort reading
13	Overload condition	Set flag 4 and abort reading
14	Illegal characters	Set flag 4 and abort reading
15	Overload condition or illegal characters	Set flag 4 and abort reading

Input Instructions

START Instruction

CALL · ALPHA- (select code) F

The START instruction sets the CTL1 line at logical 1 which initiates a data input operation. Once this instruction is executed the calculator can return either to a program or keyboard operation. Then as data becomes available, the interface automatically inputs the data (according to the current handshake mode) and holds it until it is input into the calculator with a READ. Executing any input or output instruction other than READ clears this data.

STATUS Instruction

CALL · ALPHA- (select code) G

The STATUS (STAT) instruction is used to check either the peripheral device status or the handshake status. The instruction returns one of the following interface status codes to the X register.

Status Codes

Code	Condition
1	Handshake complete (CTL1 = logical 0)
−1	Handshake not complete (CTL1 = logical 1)
0	Peripheral FLG1 line = logical 0
−0	Peripheral FLG1 line = logical 1

If a START instruction is in effect, STAT checks the handshake status and returns either a 1 or −1 code to X. If a START instruction has not been executed, STAT checks the peripheral status and returns either a 0 or −0 code to X.

READ Instruction

CALL · ALPHA- (select code) A

The READ instruction inputs data from the output device into the stack. The samples are taken according to the currently set FORMAT instruction and are displayed according to the calculator number format (FIX, SCI or SCI 3). If two samples are specified by the FORMAT instruction, the first is entered into Y and the second into X.

A READ instruction must be used to input data being held by the interface as the result of a START instruction.

Automatic Data Input

The START, STAT and READ instructions can be used to program an automatic data-input cycle for use with slow-speed peripheral devices or in systems using slow sample cycles.

This program example uses START, STAT and READ to input and print data from a slow peripheral device while the calculator computes and displays values of A^2.

The START instruction sets the interface to input and hold a data sample when it becomes available. The STAT instruction monitors the interface status. If the interface status is −1 (handshake not complete) the program branches to step 0013 where a value of A^2 is calculated and displayed. When the interface status is +1 (handshake complete) the data is input with a READ and then printed. The program resets the interface with a START instruction and continues displaying values of A^2 until the next data sample becomes available.

```
0000 CLRA→J
0001 START   4
0003 STAT    4
0005 IF -
0006 GOTO    0013
0008 READ    4

0010 PRINT
0011 GOTO    0011
0011 GOTO    0001
0013 RCL     A
0014 ENTER↑
0015 *
0016 PAUSE
0017 1
0018 STO+    A
0019 GOTO    0003
0021 END
```


BURST Instruction

[CALL ALPHA] [select code] [C]

Number of Data Blocks → Y
Starting Data Register Number → X

The BURST instruction inputs data blocks, each consisting of 10 digits (including status digits), and stores them in data registers. The number of data blocks to be input is entered in Y and the starting data-register number is entered into X. Enough data registers to hold the specified number of data blocks must be assigned prior to executing a BURST instruction. If there are not enough data registers assigned, the calculator prints the error message "ILLEGAL ARGUMENT". If the starting address parameter specifies an unassigned data-register number, the calculator prints the error message "ILLEGAL ADDRESS".

The data blocks are stored in negative-true logic in the data registers, regardless of the logic sense specified by the data codes. The data blocks can only be recalled using SAMPLE and DECODE instructions. When the data is recalled, it is decoded in the logic sense specified by the data codes. The SAMPLE and DECODE instructions are explained next.

SAMPLE Instruction

[CALL ALPHA] [select code] [B]

Data Register Number → X

Data blocks that have been stored in registers by BURST instructions can be recalled to the stack with SAMPLE instructions (SAMPL). A SAMPLE instruction recalls the data from the register number specified in X and decodes it according to the current FORMAT instruction. The sample (or split-samples) are entered into the stack in the order shown below:

Single-Sample	Split-Sample
Sample → X	Sample 1 → Y
	Sample 2 → X

The following program uses FRMT, BURST and SAMPL instructions to input and recall these two data blocks:

	+Exponent +Mantissa +True Logic Not Overload	+Exponent −Mantissa +True Logic Not Overload	3	6	7	1	0	1	3	2
Block 1	0111	0101	0011	0110	0111	0001	0000	0001	0011	0010
	A	B	C	D	E	F	G	H	I	J

	+Exponent +Mantissa +True Logic Not Overload	−Exponent +Mantissa +True Logic Not Overload	4	3	7	1	5	2	3	2
Block 2	0110	0111	0100	0011	0111	0001	0101	0010	0011	0010
	A	B	C	D	E	F	G	H	I	J

The FORMAT instruction sets a Data Mode 1 with a split-sample of digits and an exponent (−3 parameter) for each sample (steps 0000-0005). The BURST instruction inputs the two data blocks and stores them in registers 1 and 2 (steps 0007-0010). A SAMPLE instruction recalls the data from reg 1 and it is printed (steps 0012-0020). Another SAMPLE instruction recalls the data from reg 2 and it is printed (steps 0021-0028.).

```
0000 1
0001 ENTER↑
0002 3
0003 +≒-
0004 ENTER↑
0005 FRMT    4
0007 2
0008 ENTER↑
0009 1
0010 BURST   4
0012 1
0013 SAMPL   4
0015 X≒Y
0016 PRINT
0017 ROLL↓
0018 PRINT
0019 SPACE
0020 SPACE
0021 2
0022 SAMPL   4
0024 X≒Y
0025 PRINT
0026 ROLL↓
0027 PRINT
0028 END
```

Here is the resulting printout.

```
 3670.00
-1300.00


 4370.00
    5.23
```

DECODE Instruction

CALL · ALPHA– (select code) D

Number of Data Blocks → Y
Starting Data Register Number → X

The DECODE instruction (DCODE) decodes, or translates, data blocks which have been input and stored by BURST instructions. This instruction decodes the number of data blocks specified by the parameter in Y, beginning with the data-register number in X. After each data block is decoded into a valid sample, it is returned to the same register in which it was previously stored.

Data is assumed to be single-sample by DECODE instructions. The instruction decodes the sample according to the sample 1 parameters specified by the currently set FORMAT instruction. If a split sample is specified, the second sample is ignored. If a MASK instruction has been executed, the specified checks and actions are performed on each data sample. After a DECODE instruction has been executed, the decoded data can be recalled directly to the stack with RECALL.

PACK Instruction

CALL · ALPHA– (select code) H

Pack Type (0 or 1) → Z
Number of Data Blocks → Y
Starting Data Register Number → X

The PACK instruction allows you to input data blocks and store either two or four samples per data register.

The Z parameter specifies the pack type (0 or 1) to be used to store the data samples. Pack type 0 stores two 8-digit data samples (each includes data code, mantissa and exponent digits) per data register. Pack type 1 stores four 4-digit data samples (each consists of data code, mantissa, and exponent digits) per data register. The data is stored in positive-true logic of the logic level specified by the data code.

The Y parameter specifies the number of data blocks to be input and stored. If more data blocks are specified than can be stored in assigned registers, the calculator will print out the error message "ILLEGAL ARGUMENT". You should assign enough data registers to store the specified number of data blocks before executing a PACK instruction.

The X parameter specifies the starting register number in which data will be stored. If the register number specified is unassigned, the error message "ILLEGAL ADDRESS" is printed.

Typical assignment of data blocks for each pack type are shown in the diagrams on the next page.

Pack Type 0 –

Two Blocks (Single Sample) Per Register or **One Block (Split Sample) Per Register**

Block 1	Data Code A D F H	C E G I
Block 2	Data Code A D F H	C E G I

Data Code A D	C E
Data Code B G	F H

Pack Type 1 –

Four Blocks (Single Sample) Per Register or **Two Blocks (Split Sample) Per Register**

Block 1	Data Code A D	C E
Block 2	Data Code A D	C E
Block 3	Data Code A D	C E
Block 4	Data Code A D	C E

Block 1	Data Code A D	C E
	Data Code B G	F H
Block 2	Data Code A D	C E
	Data Code B G	F H

The data samples are input in the form specified by the current FORMAT instruction, **except that the data mode is assumed to be 1**. If more digits per sample are specified than can be accepted by the specified pack type, the excess digits are ignored. The FORMAT parameters should be set to match the number of digits acceptable by the pack type specified. This will prevent the loss of digits or exponents from the sample being input.

The following table gives the maximum number of digits per sample for FORMAT instruction parameters to be used with PACK instructions.

Format Parameters for Use With Pack Instructions

Pack Type	Samples Per Block	Blocks Stored Per Register	Range of FORMAT Instruction Parameters	
0	1 (Single-Sample)	2	(1 through 7) → Y Fixed 0 → X	
			−1(through 6) → Y Exp. 0 → X	
0	2 (Split-Sample)	1	(n) → Y Fixed (6−n) → X Fixed	−(n) → Y Exp. (5−n) → Fixed
			(n) → Fixed −(5−n) → X Exp.	−(n) → Y Exp. −(4−n) → X Exp.
1	1 (Single-Sample)	4	(≤3) → Y Fixed 0 → X	
			−(≤2) → Y Exp. 0 → X	
1	2 (Split-Sample)	2	(≤3) → Y Fixed (≤3) → X Fixed	−(≤2) → Y Exp. 3 → X Fixed
			(≤3) → Y Fixed −(≤2) → X Exp.	−(≤2) → Y Exp. −(≤2) → X Exp.

UNPACK Instruction

CALL · ALPHA (select code) 1

Pack Type (0 or 1) → Y
Register No. → X

The UNPACK instruction (UNPAK) must be used to recall data from a register after it has been input and stored by a PACK instruction. The data is recalled according to the pack type specified and the current FORMAT instruction. The pack type parameter specified in Y should be the same type that was specified by the PACK instruction that stored the data. The FORMAT instruction should also be the same as the one set when the data was input.

If a MASK instruction has been executed, any checks and resulting actions are performed on the data before it is entered into the stack. Data is entered into the stack in the following order by UNPACK instructions:

Pack Type 0
Sample 1 → Y
Sample 2 → X

Pack Type 1
Sample 1 → T
Sample 2 → Z
Sample 3 → Y
Sample 4 → X

The following program uses FORMAT, PACK and UNPACK instructions to input and recall these data blocks:

+Exponent +Mantissa +True Logic Not Overload	−Exponent +Mantissa +True Logic Not Overload	2	3	1	1	2	1	Not	Not
0011	0110	0010	0011	0001	0001	0010	0001	Used	Used
A	B	C	D	E	F	G	H	I	J

+Exponent −Mantissa +True Logic Not Overload	+Exponent −Mantissa +True Logic Not Overload	1	4	1	0	3	1	Not	Not
0101	0101	0001	0100	0001	0000	0101	0001	Used	Used
A	B	C	D	E	F	G	H	I	J

The FORMAT instruction specifies that both data samples will be two digits and in exponential form (steps 0000-0005). The two data blocks are input and stored by the PACK instruction specifying Pack type 1 and register 0 (steps 0007-0012). The UNPAK instruction recalls the data samples to the stack and the values are then printed.

```
0000 1
0001 ENTER↑
0002 2
0003 +⁄-
0004 ENTER↑
0005 FRMT    4
0007 1
0008 ENTER↑
0009 2
0010 ENTER↑
0011 0
0012 PACK    4
0014 1
0015 ENTER↑
0016 0
0017 UNPAK   4
0019 PRTSTK
0020 END
```

Here is the resulting printout.

```
 230.00
   1.20
-140.00
 -30.00
```

INPUT Instruction

CALL · ALPHA · (select code) E

Number of Data Bytes → Y
Starting Register Number → X

The INPUT instruction inputs 8-bit binary bytes of data and stores 8 bytes per data register. The data is input on the digit A and B input lines with A-8 being the most significant bit and B-1 the least significant bit. The currently set handshake is used to input each byte.

The Y parameter specifies the number of data bytes to be input. The X parameter specifies the starting data register number in which the bytes will be stored.

Before executing an INPUT instruction, be sure that enough data registers have been allocated to hold the number of bytes (Y parameter) specified. If not, the calculator will print the error message "ILLEGAL ARGUMENT". If the starting register number (X parameter) specifies a data register that has not been allocated, the calculator will print the error message "ILLEGAL ADDRESS". In either case, the instruction will not be executed.

Data is stored in negative-true logic by INPUT instructions and can only be recalled with BYTE instructions or output to another device with OUTPUT instructions. These instructions are described next in this section.

BYTE Instruction

CALL ALPHA · select code N

Decimal Equivalent of the Byte to be Stored → Y
±(Register . Byte) → X

The BYTE instruction is used to either recall (+parameter) a specific 8-bit binary byte from a data register or to store (−parameter) an 8-bit binary byte into a specific location in a data register.

To recall a specific byte from a register, the parameter is of the form:

+(Register Number) . (Byte Position 0 through 7)

For example to recall the third byte in register 1, the parameter entered in X would be 1.2. The decimal-equivalent value of the byte is returned to X when the instruction is executed.

To store an 8-bit binary byte in a register, the decimal equivalent of the byte is entered into Y. The register-number and byte-position parameter is entered in X in the form:

−(Register Number . Byte Position 0 through 7)

For example, to store the binary byte 11010110 in register 2 at byte position 5, the parameters would be entered as follows:

214 → Y
−2.4 → X

When the instruction is executed, the binary equivalent of 214_{10} is stored in register 2 at byte position 5. If there is a byte already in that position, the specified byte is stored in its place. Any other bytes in register 2 remain unchanged.

Output Instructions

OUTPUT Instruction

CALL ALPHA · select code J

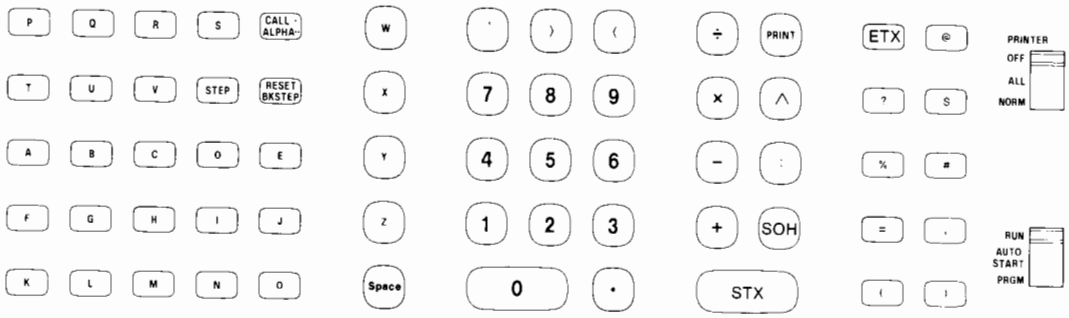
Number of Data Bytes → Y
Starting Register Number → X

The OUTPUT instruction (OUTPT) outputs 8-bit binary data that has been stored in data registers. The instruction outputs the number of bytes specified in Y, starting with the register specified in X. The bytes are output in negative-true logic.

WRITE ALPHA Instruction

CALL · ALPHA- (select code) CALL · ALPHA-

The WRT ∝ instruction sets an Alpha mode, similar to the internal printer's Alpha mode. Now, pressing each key outputs (immediately) its corresponding ASCII character. To terminate the Alpha mode press CALL · ALPHA- again. The keyboard below shows each key's corresponding ASCII character.



Alpha Keyboard

When the Alpha mode is set, the PRINT key outputs the value in the X register in ASCII code. The value is output, in the currently-set number format (FIX, SCI or SCI 3) with a leading space or minus sign followed by the number. All data output from the Alpha mode is in negative-true logic.

WRITE BYTE Instruction

CALL · ALPHA- (select code) 0 Decimal Equivalent Value (0 to 255) → X

The WRITE BYTE instruction (WBYTE) outputs the 8-bit binary equivalent of the integer value in X. The number can range from 0 through 255. The error message "ILLEGAL ARGUMENT" indicates that a number is outside this range. A table of decimal and binary-equivalent numbers is in the Appendix.

Values are output with WBYTE instructions in negative-true logic and the data remains valid on the output lines until another instruction is executed.

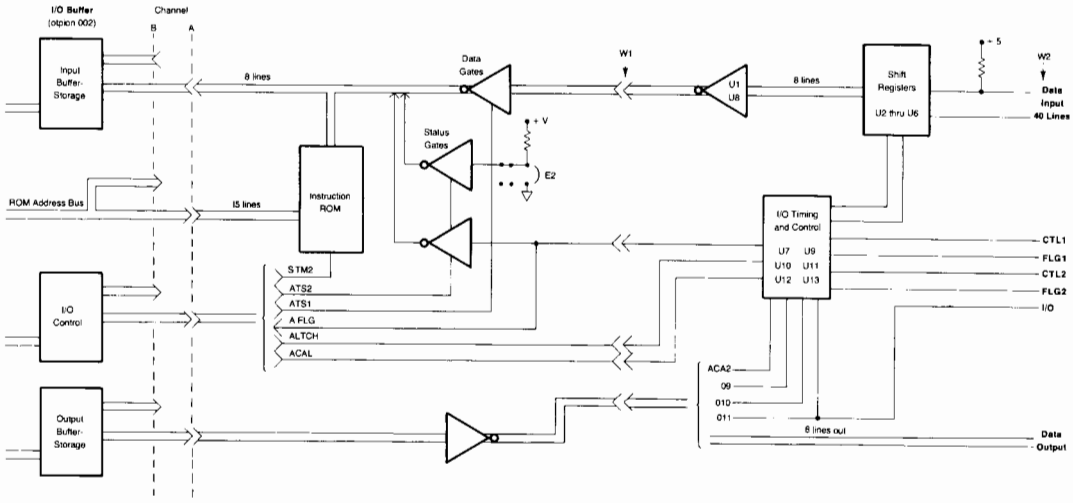
3
Service

This section contains a brief description of interface operation and instructions to help you repair the interface. A complete circuit diagram and a list of replaceable parts are at the back of this section.

If you have difficulty repairing the interface or if you would rather have HP repair it, contact the nearest sales and service office for assistance; office locations are listed in the Appendix.

Theory of Operation

A block diagram of the interface is shown below. The A1 circuit board contains an instruction ROM, data buffers and interface status circuits. The A2 circuit board contains the input shift registers, the interface I/O control and the timing circuits.



Interface Block Diagram

As shown in the block diagram, input data, interface status, and ROM information are transmitted to the calculator on the same Data In Lines. Control signals ATS1, ATS2, and STM2 determine which information is currently being transmitted.

The interface select code is determined by the position of programming plug E2. The interface responds to either select code 4 when the plug is in place, or select code 5 when the plug is removed (in its alternate position).

Data Output Sequence

This sequence occurs for each character output.

1. As an output instruction is executed from the calculator, the status gates are enabled.
2. The calculator puts data on the output lines and drives the I/O line low indicating an output operation.
3. The output operation is terminated according to the handshake mode specified (see "HANDSHAKE Instruction" in Section 2).

    - Mode 0: 37μs after the I/O line goes low, it is driven high again, terminating the output operation.
    - Mode 1: 27μs after the I/O line goes low, the ACAL line drives CTL2 to a logical 1 (the logic sense set by A2U10 and A2U12). The CTL2 line stays at 1 until the peripheral drives the FLG2 line to logical 1 indicating "done". The CTL2 line is then driven to 0 terminating the operation. The FLG2 line can go to 0 any time after CTL2 does (typical minimum FLG2 pulse is 2μs).
    - Mode 2: 27μs after the I/O line goes low, the ACAL line drives CTL2 to a 1 (as set by A2U10 and A2U12) and it is held at 1 for 22μs before it is driven to 0. 41μs after CTL2 goes to 0, the I/O line is driven high which terminates the operation.

4. In all handshake modes, the data remains valid on the output lines until new data replaces it.

Data Input Sequence

1. The status gates are enabled when an input instruction is executed.
2. The interface timing circuits (A2U9 and A2U10) are cleared and the shift registers (A2U2 through A2U6) are set to a parallel load condition.
3. The I/O line is driven high to indicate an input operation.
4. The input operation is completed according to the handshake mode specified (see "HANDSHAKE Instruction" in Section 2).

    - Mode 0: Data is assumed valid and then latched by pulsing the ACA2 line.
    - Mode 1: 46μs after the I/O line is drive high, the ACAL line drives CTL1 to a logical 1 (as set by A2U10 and A2U12) signalling "calculator ready" to the peripheral. The peripheral then drives the FLG1 line to logical 1 when valid data is available. The CTL1 line is driven to 0 (ACAL is cleared) when the data is latched by pulsing ACA2. The peripheral can drive the FLG1 line to 0 any time after CTL1 goes to 0 (typical minimum FLG1 pulse is 2μs).
    - Mode 2: The ACAL line drives CTL1 to a logical 1 (as set by A2U10 and A2U12). 15μs after the leading edge of the CTL1 pulse, the data is latched by pulsing ACA2. The data should be valid a minimum of 2μs before the latch occurs. 8μs after the data is latched, the CTL1 line is driven to 0 (ACAL is cleared). Total CTL1 pulse time is 23 μs.

5. Each byte is input into the calculator and the next one shifted into position by 8 pulses from A2U9 and A2U10 (controlled by ACA2).
6. The data is transferred and stored in the calculator as specified by the input operation being executed.

Additional Theory

Refer to the fold-out circuit diagram while reading the following sections.

ROM Enable

A1U8 is a BCD-to-decimal decoder which, together with its associated circuitry, enables the ROM as each I/O instruction is executed. The signals on the ROM address bus are strobed continuously and have no steady state.

ROM Power-up

A1Q1 controls the −12 V supply for A1U7. A1Q1 is switched off except when the ROM is enabled by an I/O instruction. A1C1 enables A1Q1 to switch-on more rapidly.

Troubleshooting & Repair

Read the remainder of this section before attempting to service the interface. When you suspect that the interface is defective, first disconnect it from the calculator and verify calculator operation by running all applicable tests using the Utility and Test Cartridge. See Appendix 2 of the calculator operating manual for instructions.

Broken Trace Repair

If an internal trace on the circuit board is open or has high resistance, it should be bridged using insulated wire, on the back of the board.

> **CAUTION**
> TO HELP PREVENT DAMAGE TO THE CIRCUIT BOARD, USE A LOW-TEMPERATURE SOLDERING IRON WHEN REPLACING PARTS.

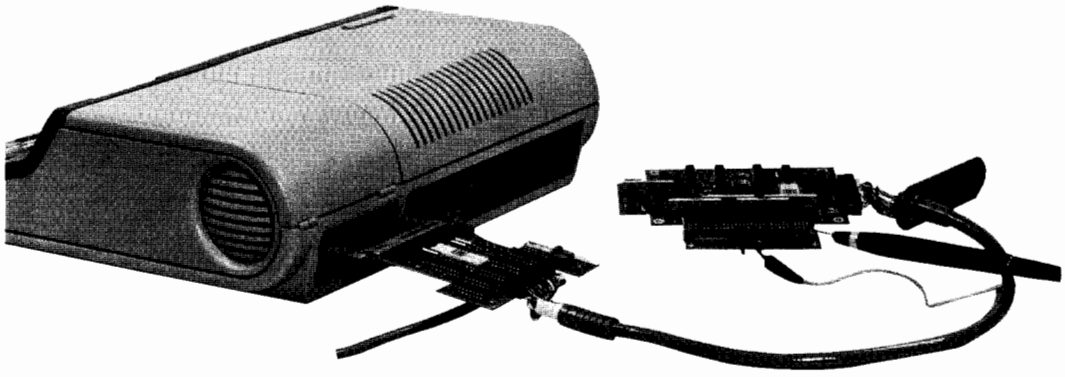
Equipment Required

- An HP 9815A Calculator, with option 002 (two channel I/O).
- Utility and Test Cartridge, HP P/N 09815-10004.
- An HP 427A voltmeter (or equivalent).
- A board extender card P/N 09815-66582.
- Test Connector HP P/N 1251-3334 (supplied with the Interface).
- An HP 10525A Logic Probe (or equivalent) with an appropriate power supply.

Test Setup

Switch the calculator off and disconnect all interfaces. Remove the BCD interface circuit boards from their cases. Connect the A1 circuit board to the calculator using the board extender (see the next photo). Connect the A2 board to A1 with the XA1 connecting cable. Be sure that both boards are connected with the component sides up, as shown in the photo.

Replace the XA2 connector on the A2 board with the Test Connector. The Test Connector (Rev. B) should be connected with the word "TOP" facing up. The photo below shows a Rev. A test connector which should be connected with the printing facing down.

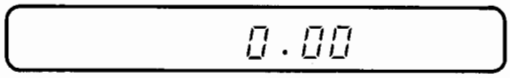


Troubleshooting the Interface

Before switching the calculator on, first verify that the select code plug is in position for select code 4 (see page 3). Then check for broken wires on the connecting cable between the circuit boards; refer to the schematic on page 31.

Power-Up Check

1. Switch the calculator on.

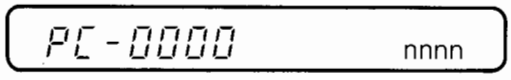


2. If the display shown above does not appear, switch the calculator off, disconnect the interface boards and switch the calculator back on.
3. If the correct display appears when the interface is not connected, but does not appear when the interface is connected (try each I/O channel), disconnect the circuit boards and check for a shorted or low-resistance line on each calculator-to-interface pin. Use the GND pin on the Test Connector as a ground reference. All pins should measure greater than 1 MΩ to ground except these (± 10% tolerance allowed):

   A1 = 1KΩ
   A19, 20 = 200Ω
   A21, 22, 23 = 0Ω

   B1 = 26KΩ
   B25 = 700Ω

   If an incorrect measurement is found, check the related components and circuit-board traces. Then reconnect the interface to the calculator and repeat step 1.
4. Set 



The number displayed on the right should be either 0376 (basic memory) or 1912 (option 001). If not, perform the ROM check procedure.

ROM Check

To check the ROM (A1U7), perform the ROM test included in the Utility and Test Cartridge supplied with your calculator. The test procedure is described in Appendix 2 of the Calculator Operating Manual under "ROM Test".

If the test fails, perform the following checks:

1. Using the logic probe to monitor A1U7 pin 15, press the sequence [CLEAR] [CALL ALPHA] [4] [M]. A negative pulse should be seen each time the sequence is executed. If not, replace A1U3 or A1U8.
2. Monitor A1U7 pin 13 and press the sequence [CLEAR] [CALL ALPHA] [4] [M]. A positive pulse should be seen each time the sequence is executed. If not, check A1Q1 and its related components.
3. If pulses are seen at A1U7 pins 13 and 15 in the checks above and the ROM test fails, replace A1U7.

Interface Operation Test

Insert the Utility and Test Cartridge into the calculator tape drive. To load the Interface Test Program from file −25:

Press: [CLEAR] 25 [+/-] [LOAD]

When the tape drive halts:

Press: [END] [RUN STOP]

If the interface passes the test, the calculator will print "Test Passed".

If the calculator prints "Test Failed" or does not print anything within 10 seconds, switch the calculator off, then on again. Check the connectors on the boards to be certain that they are properly connected and repeat the test procedure. If the test still fails, contact the nearest HP Sales and Service Office for assistance. See the back of the manual for the location of an office near you. If you wish to repair the interface yourself, follow the troubleshooting procedure given next in this section.

Output Data Check

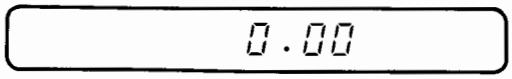
This procedure verifies A1U4, A1U5, Part of A1U2, and the CTL2 and FLG2 circuits.

1. Connect the Test Connector to the A2 board as shown on page 26.
2. Set 

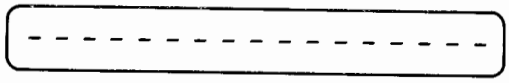
Press

[1] [ENTER ↑] [CALL ALPHA] [4] [K]

[CLEAR] [CALL ALPHA] [4] [0]



If the display remains "busy" as shown below, press [RUN STOP] and go to step 5.



3. Now check each of the Data Out lines (OØ-O7) on the A2 board. Each line should be high (≥2.4 V). If not, check its corresponding inverter, A1U4 or A1U5. Then repeat this step.

4. Press 255 [CALL ALPHA] [4] [0] and check each output line (OØ-O7). Each line should be low (≤0.7).

5. Using the logic probe, check for a CTL pulse at each of the pins listed below, press this sequence for each check:

[CLEAR] [CALL ALPHA] [4] [0] (press [RUN STOP], if needed, to cancel the busy display).

If a pulse is not seen, replace the indicated component(s).

Monitor:	Check:
XA1 pin A10 (ACAL)	(Calculator requires service)
XA2 pin A11 (CTL2)	A1R1, A2U11, A2U12, A2U13
XA2 pin A10 (FLG2)	Test Connector (see step 1)
A1U2 pin 7	A1U2

Data Input Check

This procedure verifies the following components:

A2U2 through A2U6, A2U9, A2U10, and the CTL1 and FLG1 circuits.

1. Connect the Test Connector to the A2 board as shown on page 26.

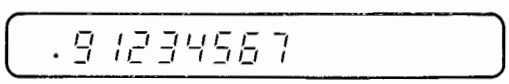
2. Set RUN/AUTO START/PRGM Press

[■] [+/-] 8 1 [ENTER ↑] [CALL ALPHA] [4] [K]

[CLEAR] 8 [+/-] [ENTER ↑] 0 [CALL ALPHA] [4] [L]

105 [CALL ALPHA] [4] [0] [CALL ALPHA] [4] [A]

3. The value returned to the display should be:



If any other value is returned, check the shift registers A2U2 through A2U6. If the display remains busy instead of returning a value, press [RUN STOP] and go to step 4.

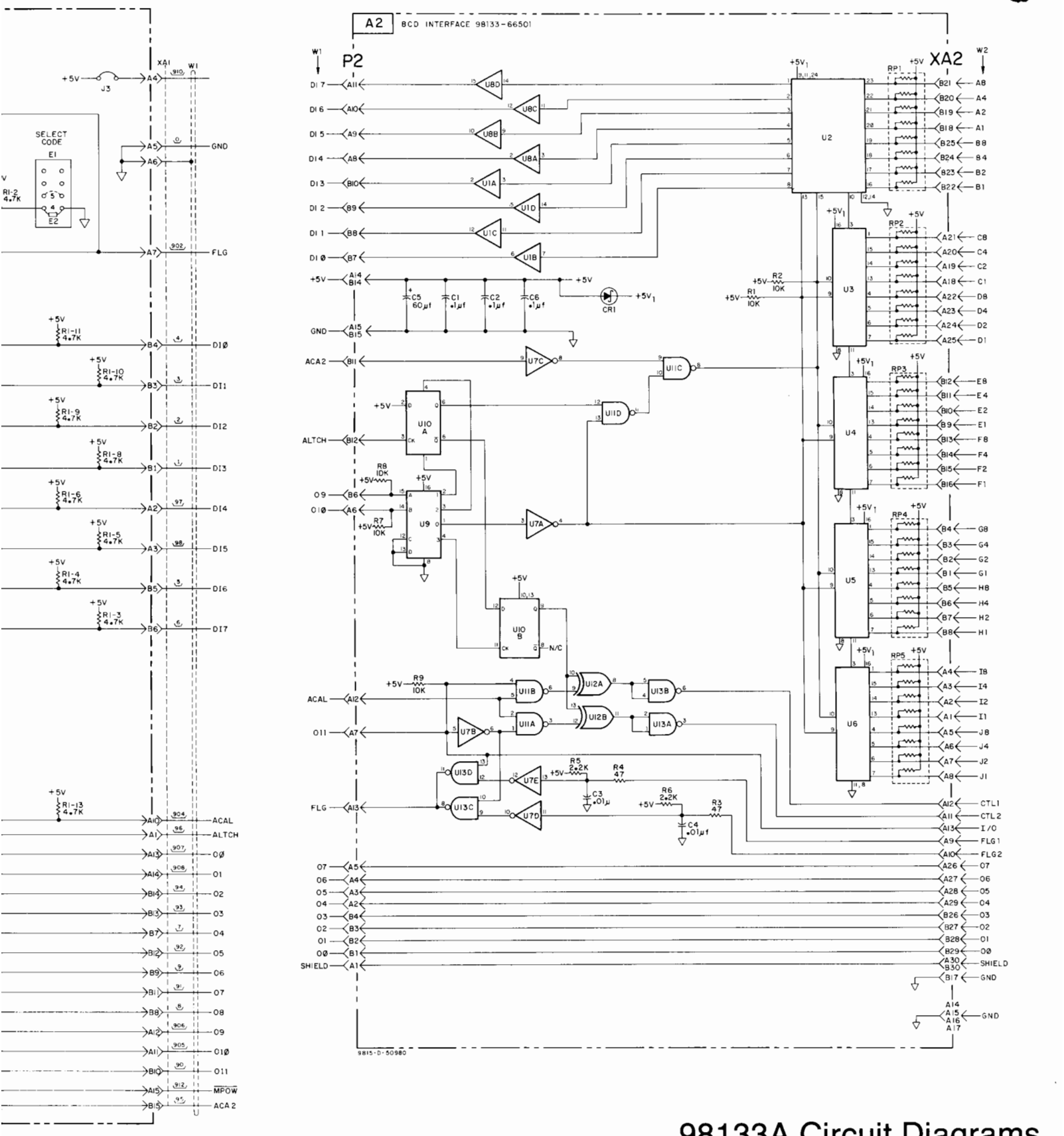
4. Using the logic probe, check for a CTL pulse at each of the pins listed below; press this sequence for each check: [CALL ALPHA] [4] [A] (press [RUN STOP], if needed, to cancel the busy display). If a pulse is not seen, replace the indicated components.

Monitor:	Check:
XA1 pin A10 (ACAL)	(Calculator requires service)
XA1 pin A2 (CTL1)	A1R1, A2U11, A2U12, A2U13
XA2 pin A9 (FLG1)	Test Connector (see step 1)
A1U2 pin 7	A1U2

Replaceable Parts

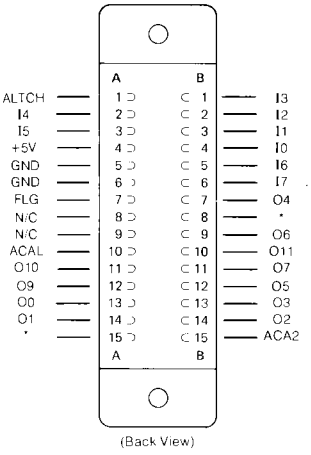
REFERENCE DESIGNATOR	-hp- PART NO.	TQ	DESCRIPTION
A1 ROM Board	98133-69503	1	ROM PC Assembly
A1C1	0160-4387	1	Cap: 47μf, 200V
A1C2	0180-0228	1	Cap: 22μf, 15 V
A1C3, C4	0180-1704	2	Cap: 47μf, 6V
A1E1	1200-0471	1	Socket: 8-pin DIP
A1E2	1258-0124	1	Plug, select code
A1Q1	1853-0058	1	Transistor: Si., PNP
A1R1	1810-0162	1	R-Network: 14 pin DIP
A1R2, R4	0683-1025	2	Res: 1k, ¼W, 5%
A1R3	0683-6825	1	Res: 6.8k, ¼W, 5%
A1U1,U2	1820-1255	2	IC: Hex Inverter, DM8098
A1U3	1820-1144	1	IC: Quad, 2-input NOR Gate, SN74LS02N
A1U4, U5	1820-0471	2	IC: Hex Inverter/Driver, SN7406N
A1U7	1818-2639	1	IC: ROM
A1U8	1820-1418	1	IC: Decoder, SN74LS42J
W1	98133-61601	1	Cable Ass'y w/connectors
XA1	1251-4327	2	Connector, 30 pin
	5040-7781	1	A1 Cover, Top
	4040-7782	1	A1 Cover, Bottom
A2 I/O Board	98133-66501	1	I/O PC Assembly
A2C1, C3, C7, C8	0160-3847	4	Cap: .01μf, 25V
A2C2, C9	0160-3878	2	Cap: .001μf, 50V
A2C4	0180-0106	1	Cap: 60μf, 6V
A2C5, C6	0160-3879	2	Cap: .01μf, 100V
A2CR1	9101-0535	1	Diode, Hot Carrier
A2R1,R2,R8,R9, R11,R13	0683-1035	6	Res: 10KΩ, ¼W, 5%
A2R3, R4	0683-4705	2	Res: 47Ω, ¼W,5%
A2R5,R10,R12	0683-1515	3	Res: 150Ω, ¼W, 5%
A2R6, R7	0683-2225	2	Res: 2.2KΩ, ¼W, 5%
A2RP1-RP5	1810-0055	5	R-Network
A2U1, U8	1820-1146	2	IC: CD4050AE
A2U2	1820-0977	1	IC: CD4034AD
A2U3, U4, U5, U6	1820-0970	4	IC: CD4014AE
A2U7	1820-1416	1	IC: SN74LS14N
A2U9	1820-1418	1	IC: SN74LS42N
A2U10	1820-1112	1	IC: SN746LS4N
A2U11	1820-1197	1	IC: SN74LS00N
A2U12	1820-1211	1	IC: SN74LS86N
A2U13	1820-0514	1	IC: SN7426N
W2	98133-61602	1	Cable Ass'y w/connectors
XA2	1251-3339	1	Connector, 60 pin
	5040-8035	1	A2 Cover, Top
	5040-8034	1	A2 Cover, Bottom
	5040-8025	2	A2 Cover, Ends
	1251-3339	1	Test Connector, 60 pin
	98133-90000	1	Operating and Service Manual
	98133-90010	1	Reference Card
	7120-5054	1	Key Overlay

A2 I/O Circuit Board Schematic



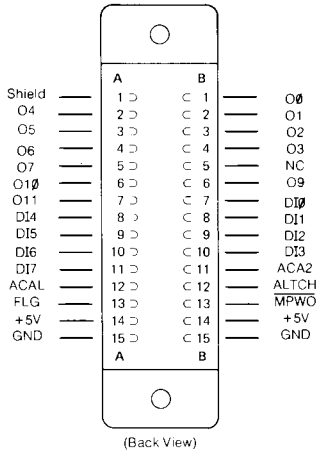
98133A Circuit Diagrams

XA1 Connector to A1 Board



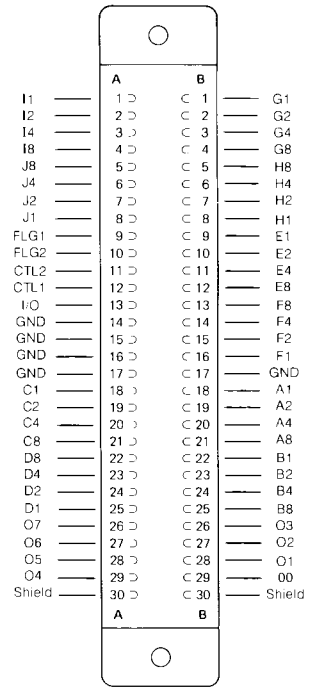
(Back View)

P2 Connector to A2 Board



(Back View)

XA2 Connector TO INTERFACE CARD



X A2 Wire Connections

Signal	Wire Color	Signal	Wire Color
A8	Wht/Yel/Blu	H8	Wht/Brn/Blu
A4	Wht/Yel/Grn	H4	Wht/Brn/Vio
A2	Wht/Orn/Gry	H2	Wht/Brn/Gry
A1	Wht/Orn/Vio	H1	Wht/Red/Orn
B8	Wht/Grn/Vio	I8	Grn
B4	Wht/Grn/Blu	I4	Yel
B2	Wht/Yel/Gry	I2	Orn
B1	Wht/Yel/Vio	I1	Red
C8	Wht/Gry	J8	Blu
C4	Wht/Vio	J4	Vio
C2	Wht/Blu	J2	Gry
C1	Wht/Grn	J1	Wht
D8	Wht/Blk/Brn	07	Wht/Blk/Grn
D4	Wht/Blk/Red	06	Wht/Blk/Blu
D2	Wht/Blk/Orn	05	Wht/Blk/Vio
D1	Wht/Blk/Yel	04	Wht/Bik/Gry
E8	Wht/Red/Vio	03	Wht/Grn/Gry
E4	Wht/Red/Blu	02	Wht/Blu/Vio
E2	Wht/Red/Grn	01	Wht/Blu/Gry
E1	Wht/Red/Yel	00	Wht/Vio/Gry
F8	Wht/Red/Gry	FLG1	Wht/Blk
F4	Wht/Orn/Yel	FLG2	Wht/Brn
F2	Wht/Orn/Grn	CTL2	Wht/Red
F1	Wht/Orn/Blu	CTL1	Wht/Orn
G8	Wht/Brn/Grn	I/O	Wht/Yel
G4	Wht/Brn/Yel	GND	Blk
G2	Wht/Brn/Orn	GND	Brn
G1	Wht/Brn/Red	Shield	Shield

tic Notes:

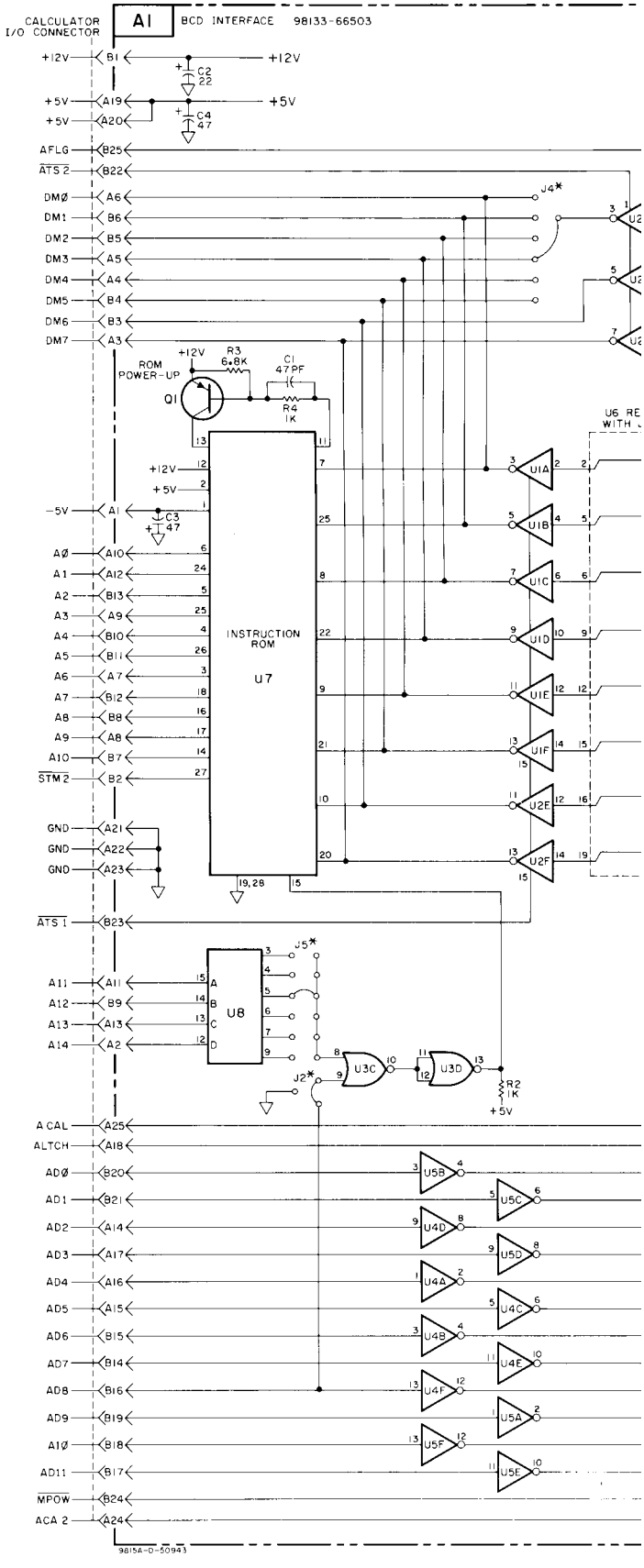
ss indicated otherwise, resistor values are shown in ohms and acitor values are shown in microfarads.

page 2 before changing the position of E2, the select code plug.

color code is the same as resistor color code: The first number ates the base color, second number indicates the wider strip, and color indicates the narrower strip (e.g., 924 = white, red, yellow).

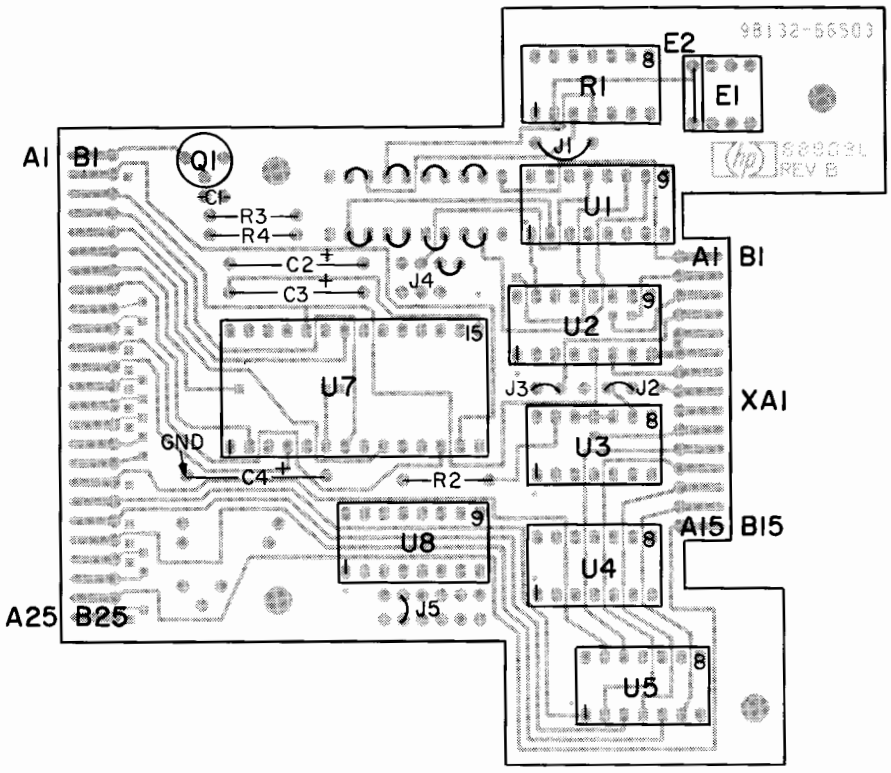
5 = Green
6 = Blue
7 = VBiolet
8 = Grey
9 = White

A1 ROM Board Schemat

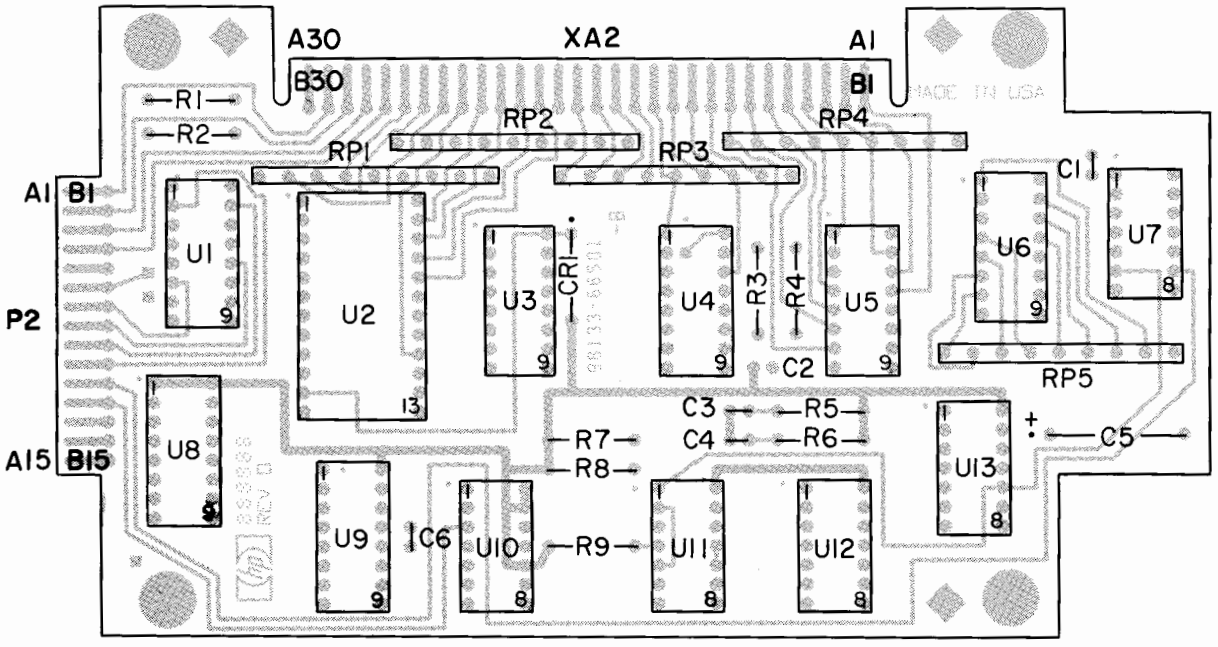


Component Locator

A1 ROM Board



A2 I/O Board



Schema

1. Unle capa
2. See
3. Wire indic third

0 = Black
1 = Brown
2 = Red
3 = Orang
4 = Yellow

Appendix

Binary Coding and Conversions

Binary is a base 2 number system using only 1's and 0's. By giving the 1's and 0's positional value, any decimal number can be represented. For example, this diagram shows how decimal 41 = binary 101001:

Decimal		Binary					
10^1	10^0	2^5	2^4	2^3	2^2	2^1	2^0
↓	↓	↓	↓	↓	↓	↓	↓
10	1	32	16	8	4	2	1
4	1	1	0	1	0	0	1

Binary-Coded Decimal

The primary code for representing numeric values with this interface is Binary Coded Decimal (BCD). BCD is a four-bit binary code; each four bits represents a decimal digit from 0 through 9.

For example, to convert the BCD code 010000110110, to decimal:

BCD representation	0100	0011	0110
	↓	↓	↓
Decimal Value	4	3	6

The decimal equivalent number is found by breaking up the 12-bit binary number into groups of 4 bits (starting from the right) and converting each group into a decimal number. Valid BCD characters include 0 through 9 and the decimal point (14 or 15).

Binary-Decimal Conversions

To convert from binary to decimal, the positional values for the 1's are added up. From the above example this would be:

$$2^5 + 2^3 + 2^0 = 32 + 8 + 1 = 41$$

To convert from decimal to binary, the decimal number is repeatedly divided by 2. The remainder is the binary equivalent. For example:

		Remainder (read up)
$2\overline{)41}$	→	1
$2\overline{)20}$	→	0
$2\overline{)10}$	→	0
$2\overline{)5}$	→	1
$2\overline{)2}$	→	0
$2\overline{)1}$	→	1

Octal-Binary Conversions

Octal is a base 8 number system. Octal numbers are often used since conversion from binary to octal and vice-versa is easy by using electronic circuits.

To convert from binary to octal, the octal number is broken up into groups of three bits (starting from the right). Each group of 3 bits represent an octal number.

For example, to convert binary 10110100011001 to octal:

Binary Number	10	110	100	011	001
	↓	↓	↓	↓	↓
Octal Number	2	6	4	3	1

Notice that only values from 0 through 7 are used in octal.

To convert from octal to binary, the process is reversed:

Octal Number	1	4	0	7	2	6
	↓	↓	↓	↓	↓	↓
Binary Number	001	100	000	111	010	110

ASCII

Binary is often used as a code to represent not only numbers, but also alphanumeric characters such as "A" or "," or "?" or "x" or "2". One of the most common binary codes used is ASCII[1]. ASCII is an eight-bit code, containing seven data bits and one parity bit. The BCD Interface uses 7 bit ASCII without a parity bit. For example:

Character	ASCII Binary Code	ASCII Decimal Code
A	01000001	65
B	01000010	66
?	00111111	63

A complete list of ASCII characters and their equivalent binary and decimal representations is on pages 36-37.

[1]American Standard Code for Information Interchange.

NOTES

ASCII Character Codes

ASCII Char.	EQUIVALENT FORMS			Alpha Mode
	Binary	Octal	Dec	
NULL	00000000	000	0	
SOH	00000001	001	1	
STX	00000010	002	2	ENTER
ETX	00000011	003	3	
EOT	00000100	004	4	
ENQ	00000101	005	5	
ACK	00000110	006	6	
BELL	00000111	007	7	
BS	00001000	010	8	
HTAB	00001001	011	9	
LF	00001010	012	10	
VTAB	00001011	013	11	
FF	00001100	014	12	
CR	00001101	015	13	
SO	00001110	016	14	—
SI	00001111	017	15	—
DLE	00010000	020	16	
DC1	00010001	021	17	
DC2	00010010	022	18	
DC3	00010011	023	19	
DC4	00010100	024	20	
NAK	00010101	025	21	
SYNC	00010110	026	22	
ETB	00010111	027	23	
CAN	00011000	030	24	—
EM	00011001	031	25	—
SUB	00011010	032	26	—
ESC	00011011	033	27	—
FS	00011100	034	28	—
GS	00011101	035	29	—
RS	00011110	036	30	—
US	00011111	037	31	—

ASCII Char.	EQUIVALENT FORMS			Alpha Mode
	Binary	Octal	Dec	
space	00100000	040	32	RUN STOP
!	00100001	041	33	
"	00100010	042	34	
#	00100011	043	35	LOG
$	00100100	044	36	LN
%	00100101	045	37	COS
&	00100110	046	38	
'	00100111	047	39	
(	00101000	050	40	P→R
)	00101001	051	41	ACC +
*	00101010	052	42	×
+	00101011	053	43	+
,	00101100	054	44	Σ+
–	00101101	055	45	–
.	00101110	056	46	.
/	00101111	057	47	÷
0	00110000	060	48	0
1	00110001	061	49	1
2	00110010	062	50	2
3	00110011	063	51	3
4	00110100	064	52	4
5	00110101	065	53	5
6	00110110	066	54	6
7	00110111	067	55	7
8	00111000	070	56	8
9	00111001	071	57	9
:	00111010	072	58	R↓
;	00111011	073	59	
<	00111100	074	60	CL X
=	00111101	075	61	TAN
>	00111110	076	62	E EX
?	00111111	077	63	SIN

ASCII Char.	EQUIVALENT FORMS Binary	Octal	Dec	Alpha Mode
@	01000000	100	64	f⁻¹
A	01000001	101	65	A
B	01000010	102	66	B
C	01000011	103	67	C
D	01000100	104	68	D
E	01000101	105	69	E
F	01000110	106	70	F
G	01000111	107	71	G
H	01001000	110	72	H
I	01001001	111	73	I
J	01001010	112	74	J
K	01001011	113	75	K
L	01001100	114	76	L
M	01001101	115	77	M
N	01001110	116	78	N
O	01001111	117	79	O
P	01010000	120	80	LOAD
Q	01010001	121	81	REWIND
R	01010010	122	82	RECORD
S	01010011	123	83	LIST
T	01010100	124	84	GO TO
U	01010101	125	85	LABEL
V	01010110	126	86	SFG CFG
W	01010111	127	87	CLEAR
X	01011000	130	88	STORE
Y	01011001	131	89	
Z	01011010	132	90	END
[	01011011	133	91	
\	01011100	134	92	
]	01011101	135	93	
^	01011110	136	94	R/S
_	01011111	137	95	

ASCII Char.	EQUIVALENT FORMS Binary	Octal	Dec	Alpha Mode
`	01100000	140	96	
a	01100001	141	97	
b	01100010	142	98	
c	01100011	143	99	
d	01100100	144	100	
e	01100101	145	101	
f	01100110	146	102	
g	01100111	147	103	
h	01101000	150	104	
i	01101001	151	105	
j	01101010	152	106	
k	01101011	153	107	
l	01101100	154	108	
m	01101101	155	109	
n	01101110	156	110	
o	01101111	157	111	
p	01110000	160	112	
q	01110001	161	113	
r	01110010	162	114	
s	01110011	163	115	
t	01110100	164	116	
u	01110101	165	117	
v	01110110	166	118	
w	01110111	167	119	
x	01111000	170	120	
y	01111001	171	121	
z	01111010	172	122	
{	01111011	173	123	
|	01111100	174	124	
}	01111101	175	125	
~	01111110	176	126	
DEL	01111111	177	127	

p

r

s

t

u

w

x

ERROR MESSAGES

* OVERFLOW	Number or result exceeds calculating range.
* SQRT OF NEG #	
* DIVISION BY ZERO	
* LOG OF # <=0	
* NO I/O DEVICE	Peripheral device or interface not connected.
ILLEGAL ADDRESS	Improper step address or storage register specified.
ILLEGAL ARGUMENT	The number of data blocks exceeds the number of assigned registers or the range of an instruction has been exceeded.
MEMORY OVERFLOW	Program instruction, storage register assignment, or program loaded from tape exceeds available memory.
GOSUB OVERFLOW	More than seven subroutines (including special functions) nested at a time.
KEY NOT DEFINED	Special function just called is not defined.
IMPROPER SYNTAX	Incorrect use of FIX, SCI, SCI 3, CALL·ALPHA·, SFG·CFG··, D FOR, E NEXT, or F IF
* CHECKSUM ERROR	Program or data loaded into calculator not identical to that in file; this usually indicates a dirty tape head or a worn tape.
* VERIFY FAILED	Program or data in file not identical to that in calculator.
WRONG FILE TYPE	Attempting to load an empty, extra, or binary file; recording on an extra file.
END OF TAPE	End of tape reached during MARK operation. Also indicates a broken or defective tape; if the tape does not appear to be broken, (advance it using the drive wheel), replace the cartridge, press REWIND, and continue.
PROTECTED TAPE	The cartridge RECORD slide is positioned to prevent MARK and RECORD operations.
SECURED MEMORY	Attempting to list, edit, or record a secured program.
MISSING FOR STMT	
LABEL NOT FOUND	
FILE NOT FOUND	
CARTRIDGE OUT	
MISSING GOSUB	

*These messages are suppressable; see "Flags" in Section 3 of the calculator operating manual.

HP 98133A BCD Interface

HEWLETT hp PACKARD

PART NO. 98133-90000
MICROFICHE NO. 98133-99000

PRINTED IN U.S.A.
May 4, 1979