```
PPPP                  t           PPPP    AAA   RRRR   III   OOO
P   P                 t           P   P  A   A  R   R   I   O   O
P   P                ttt          P   P  A   A  R   R   I   O   O
PPPP   r rr    ooo    t     ooo   PPPP   AAAAA  RRRR    I   O   O
P      rr  r  o   o   t    o   o  P      A   A  R R     I   O   O
P      r      o   o   t t  o   o  P      A   A  R  R    I   O   O
P      r       ooo     t    ooo   P      A   A  R   R  III   OOO
```

by

PROTOTECH, INC.

A universal 10-bit 5-volt I/O interface for the HP 41C

ProtoPARIO OWNERS MANUAL

TABLE OF CONTENTS

INTRODUCTION

The ProtoPARIO allows the HP 41C user to interface to and from almost any 5 volt device, providing 10 input lines, 10 output lines, and 2 output handshake lines. It attaches to the ProtoEPROM/ProtoSYSTEM combination, and looks like an 8K ROM to the calculator. The lower 4K is occupied by the PARIO-1A or some other EPROM set, and the upper 4K is interpreted as Input/Output signals by the ProtoPARIO. Although the PARIO-1A EPROM set is not mandatory, it greatly simplifies programming and automates use of the ProtoPARIO.

PHYSICAL CONNECTIONS

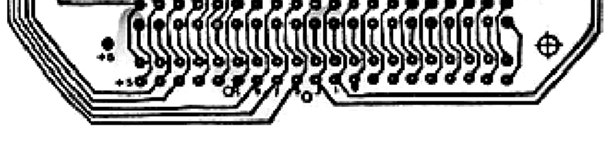
The three IC sockets on the ProtoPARIO attach directly to the ProtoEPROM board. Flip the EPROM socket levers forward and line up the pins on the ProtoPARIO sockets into the ProtoEPROM sockets, then reach between the boards with a pencil or small knife and pull the levers back to lock the ProtoPARIO in place. In addition to this, the free wire on the ProtoPARIO must be connected to V+ in the ProtoSYSTEM. V+ is available as the leftmost hole in the 25 pin connector. The 4K controlling EPROM is plugged into the rightmost 2 sockets in the ProtoPARIO, oriented the same as for the ProtoEPROM. Set the ProtoEPROM select switches 1-2-3 to ON-OFF-ON, and set switches 4-5-6-7 to an even page.

Two complete sets of data lines are available at the bottom of the ProtoPARIO. Each set is organised as 2 rows of holes centered .1" apart for standard connectors. The leftmost 12 holes of the upper row are (left to right) I0, I1, I3, I5, I7, I9, Output Accepted, O8, O6, O4, O2, and O0. The lower row (left to right) is V+, I2, I4, I6, I8, Output Ready, O9, O7, O5, O3, O1, and GND. All inputs (I9-I0 and OA) are pulled to GND with 100K resistors if unused. With two complete sets of I/O lines, it is possible for the user to hook up two devices or to customize the connector by using the unused pads to the right.

ELECTRICAL CHARACTERISTICS

Both inputs and outputs are buffered to minimize possible damage to the ProtoSYSTEM or calculator from external signals (overvoltage, etc.). All inputs should be in the range 3.5-5.5 volts for ON and 0-1.5 volts for OFF, and will require at most 1 uamp drive current per data bit. Propagation delay time (and set/reset time for OA and OR) is at most 180 nsec. OR can source .5 mamp and the outputs O9-O0 can source 1.75 mamp each, both at 4.5-5.5 volts. Outputs are latched in two CMOS flipflop arrays: 74C174, 74C175. The OR signal is latched in a 4013, and can only be reset by asserting OA. The OR signal is provided for handshaking with devices that are faster than the HP 41C. It does not prevent subsequent outputs from the calculator from being accepted. Inputs are buffered through two 4503 (70C97) CMOS chips. All buffer chips are socketed for easy replacement. All specifications given above are for V+=5 volts and ambient temperature 25C.

The GND line should always be connected to the external device ground. The V+ output should only be used through passive components such as switches back to I9-I0 since it is the regulated power from the calculator which does not provide much current capability. No external signal should be connected to V+.



PROGRAMMING WITHOUT THE PARIO-1A EPROM SET

The PARIO-1A EPROM set provides various I/O functions for simplified use of the ProtoPARIO, but is not required. The ProtoPARIO is programmed by using the CXISA (FETCH - hex 330) microcode instruction. For the following discussion it is assumed that the ProtoEPROM board is addressed at page E which places the ProtoPARIO in page F.

In this arrangement, a fetch to addresses F000-F3FF or FC00-FFFF will return 000. A fetch to any address F800-FBFC which has the final digit 0, 4, B, or C will return I9-I0 in the exponent field of the C register. A fetch to any address F801-FBFF which has the final digit 1-3, 5-7, 9-B, or D-F will return I7-I0 in digits 1-0 of C and 0 in digit 2.

Outputs are generated by a fetch to addresses F400-F7FF. Add F400 to the 10-bit binary number to be output. For example, to output 000, do a fetch at F400. A fetch at F654 will output 254 hex. Whenever an output is received by the ProtoPARIO, a flipflop in the 4013 chip (available as Output Ready) is set. If OR was already set, the new output data overwrites what was previously output. OR can only be reset to 0 by asserting Output Accepted. This provides handshaking capabilities with external devices that are faster than the HP 41C - see the interface for the TRS-80 Color Computer in the example at the end of this manual.

PARIO-1A EPROM SET

This EPROM set provides the user with a variety of input and output functions to control the ProtoPARIO:

A-XB	converts the last 12 or less binary characters (0-1) in ALPHA into the exponent of X
A-XD	converts the last 4 or less decimal characters (0-9) in ALPHA into the exponent of X
A-XH	converts the last 3 or less hex characters (0-F) in ALPHA into the exponent of X
A-XO	converts the last 4 or less octal characters (0-7) in ALPHA into the exponent of X
ESCAPE	converts the next to last character in ALPHA to be >= hexcode 20
F-X	converts flags 11-0 into the exponent of X
FETCH	executes a CXISA instruction at address in digits 3-0 of X
GOKEY	uses ProtoPARIO inputs to specify keycode from external keyboard
LISTU2	lists upper 2 bits of any ROM page in EPROM format
PACK5	packs 5 10-bit blocks of data into X
PACK7	packs 7 8-bit blocks of data into X
READ	immediate read of I9-I0 into exponent of X
READ1	loops with wait inputting data into consecutive registers (1 input/reg)
READ5	loops with wait inputting data into consecutive registers (5 inputs/reg)
READ7	loops with wait inputting data into consecutive registers (7 inputs/reg)
READA	asynchronous looped read waits for non zero changed input (1 input/reg)
ROM	displays PARIO message (try it)
RVIEW	views first 3 digits of X, Y, Z, T
STK<>Σ	provides alternate stacks by exchanging with sum regs (no normalize)
U2	microcode subroutine used by LISTU2
UNPACK5	unpacks X into 5 registers each containing a 10-bit block of data
UNPACK7	unpacks X into 7 registers each containing an 8-bit block of data
WAITNZ	loop until I9-I0 is not zero then return input in exponent of X
WAITX	loop until I9-I0 matches exponent of X
WRIT	immediate write of O9-O0 from exponent of X

WRIT1	loop with wait writing 09-00 from consecutive regs (1 output/reg)
WRIT5	loop with wait writing 09-00 from consecutive regs (5 outputs/reg)
WRIT7	loop with wait writing 09-00 from consecutive regs (7 outputs/reg)
WRITA	asynchronous write upon input equal to 0
X-AB	converts exponent of X to up to 12 binary digits in ALPHA
X-AD	converts exponent of X to up to 4 decimal digits in ALPHA
X-AH	converts exponent of X to up to 3 hexadecimal digits in ALPHA
X-AO	converts exponent of X to up to 4 octal digits in ALPHA
X-F	converts exponent of X to user flags 11-0
X<>REG	exchanges X with absolute user register specified by Y
XE-M	converts hex exponent of X to decimal mantissa of X
XM-E	converts decimal mantissa of X to hex exponent of X

CREATING DATA IN X FOR OUTPUT

The functions A-XB (binary), A-XD (decimal), A-XH (hexadecimal), A-XO (octal), F-X, and XM-E can be used to create data in the exponent of X. To use A-XB, A-XD, A-XH, or A-XO, set ALPHA to contain the number in the appropriate base to be put in the exponent of X:

```
"1010010" A-XB will set exponent of X to 052 hex
"1023"    A-XD will set exponent of X to 3FF hex
"7C"      A-XH will set exponent of X to 07C hex
"47"      A-XO will set exponent of X to 027 hex
```

F-X converts flags 11-0 as a binary number into the exponent of X: if flags 11, 10, 9, 8, 6, 4, 3, 2 are clear and flags 7, 5, 1, 0 are set then F-X will set exponent of X to 0A3 hex.
XM-E converts decimal number in mantissa of X to hex number in exponent of X: if X contains 64.0000 then XM-E will set exponent of X to 040 hex.

DECODING DATA IN X AFTER INPUT

The functions X-AB (binary), X-AD (decimal), X-AH (hexadecimal), X-AO (octal), X-F, and XE-M can be used to decode the hex data in the exponent of X. To use X-AB, X-AD, X-AH or X-AO, execute the function for the appropriate base and the exponent of X will be returned in ALPHA in that base. If the exponent of X contains hex 0FD:
X-AB will return '11111101' in ALPHA
X-AD will return '253' in ALPHA
X-AH will return 'FD' in ALPHA
X-AO will return '375' in ALPHA
X-F will set flags 7-2 and 0 and clear flags 11-8 and 1
XE-M will return 253.0000 in X

INPUTTING DATA

Five functions are provided for inputting data from the ProtoPARIO: READ, READ1, READ5, READ7, and READA.

READ performs a single read without any waiting and returns the input in the exponent of X. Digits 12-3 are returned as 0 and digit 13 is 1. This causes X to look like ALPHA DATA so that it will not be normalized.

READ1 performs a set of reads, storing inputs in consecutive registers in the same format as READ: 10 00 00 00 00 0I II. X contains the destination registers: eee.bbb where eee is the last register to be written and bbb is the first. Y contains the wait loop constant (0-999) which is counted down before each read occurs. Experimentation will provide the actual time delay between reads. The instruction can be terminated before completion by pressing R/S. If in a program, execution will continue with the next instruction.

READ5 is identical to READ1 except that 5 consecutive reads of 10 bits are stored per register instead of 1. The data word is initialized to 0 then at each read the register is shifted 10 bits to the left and the data is transferred into the bottom 10 bits. After 5 reads or if the X-loop terminates, digit 13 is set to 1 so that the data will not be normalized. See instructions for READ1 above for X and Y register usage and loop termination.

READ7 is identical to READ1 except that 7 consecutive reads of 8 bits are stored per register instead of 1. The data word is initialized to 0 then at each read the register is shifted 8 bits to the left and the data is transferred into the bottom 8 bits. After 7 reads or if the X loop terminates, the data is written to a user register. Note that all 56 bits are used so that a RCL, VIEW, or X<> instruction will normalize the register, changing the data. To get around this, use UNPACK7 or X<>REG so that normalization is avoided. See instructions for READ1 above for X and Y register usage and loop termination.

READA is identical to READ1 except that the Y register is not used as a timing loop. Data is read continuously, but is only stored at 10 bits per register when the input changes from the previously stored input AND is non zero.

OUTPUTTING DATA

Five functions are provided for outputting data to the ProtoPARIO: WRIT, WRIT1, WRIT5, WRIT7, and WRITA.

WRIT provides a single write from the exponent of X without any waiting loop then returns. The exponent of X should contain 0-0-09-08 07-06-05-04 03-02-01-00.

WRIT1 performs a set of writes from the exponent of consecutive registers at 1 data output per register in the same format as for WRIT. See instructions for READ1 above for X and Y register usage and loop termination.

WRIT5 performs a set of writes from consecutive registers at 5 data outputs per register in the same format as for READ5. See instructions for READ1 above for X and Y register usage and loop termination.

WRIT7 performs a set of writes from consecutive registers at 7 data outputs per register in the same format as for READ7. See instructions for READ1 above for X and Y register usage and loop termination.

WRITA is identical to WRIT1 except that the Y register is not used as a timing loop. WRITA outputs one data word then continuously reads data until 000 appears at the input. This can be used to synchronize the calculator with an external by jumpering Output Ready to an input and asserting Output Accepted after each output from the calculator has been received by the external device.

PAUSING

Two functions are provided to introduce wait loops into I/O control for the ProtoPARIO: WAITNZ and WAITX.

WAITNZ continuously reads data from the ProtoPARIO until the input is non zero. The input is returned in X in the format 10 00 00 00 00 0I II. WAITNZ can be aborted by pressing R/S which will return X as 0 and continue with the next program line (if any).

WAITX continuously reads data from the ProtoPARIO until the input matches the contents of the exponent of X in the same format as for WAITNZ. WAITX can be aborted by pressing R/S.

REFORMATTING DATA

Four functions are provided to convert data between the 3 storage formats of 1, 5 or 7 data words per register: PACK5, PACK7, UNPACK5, and UNPACK7. The register formats are:
10 00 00 00 00 0D DD (1-10 bit datum per register, in digits)
dd DD dd DD dd DD dd (7-8 bit data per register, in digits, dd and DD are consecutive data)
0001 000D DDDD DDDD dddd dddd ddDD DDDD DDDD dddd dddd ddDD DDDD DDDD (5-10 bit data per register, in bits, dd and DD are consecutive data).

Conversion is done between the X register and the first 5 (for PACK5 or UNPACK5) or first 7 (for PACK7 or UNPACK7) statistics registers. Any block of consecutive registers can be selected by using the ΣREG function - see the HP 41C Owners Manual.

PACK5 compresses the data in the first 5 statistics registers which are in 1-10 bit data word per register format into the X register in 5-10 bit data words per register format.

UNPACK5 reverses PACK5 by separating the X register into the 5 statistics registers.

PACK7 compresses the data in the first 7 statistics registers which are in 1-10 bit data word per register format into the X register in 7-8 bit data words per register format. The upper two bits in each data word are ignored.

UNPACK7 reverses PACK7 be separating the X register into the 7 statistics registers. The upper two bits are set to 0.

USING AN EXTERNAL KEYBOARD

The GOKEY function is designed to accept any 6-bit non zero input and map it onto the calculator keyboard as a key press. Note that the input is accepted as a keycode, therefore to enter ALPHA characters, you must first input the code that maps onto the ALPHA key. The table below shows the key press mapping for all 6-bit input combinations.

	LEAST SIGNIFICANT DIGIT															
MSD	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0		Σ+	1/X	SQR	LOG	LN	X-Y	RDN	SIN	COS	TAN	XEQ	STO	RCL	ENT	CHS
1	EEX	-	7	8	9	+	4	5	6	*	1	USR	PGM	ALP	ENT	BSP
2	0	ON	R/S	SHF	SST	/	/	/	COS	TAN	*	+	.	-	.	/
3	0	1	2	3	4	5	6	7	8	9	/	.	COS	2	TAN	3

X-Y is X<>Y, SQR is SQRT, ALP is ALPHA, ENT is ENTER, BSP is back-arrow, SHF is SHIFT. Note that in ALPHA mode, hex inputs of 01-1A will generate alpha characters A-Z. Row 2 and 3 characters are mapped as closely as possible to ALPHA mode inputs versus ASCII; some may need to be SHIFTed. Upon execution, GOKEY will loop until any non zero input in the bottom 6 bits is received, then jump to the system routine to handle that keypress. GOKEY can be aborted by pressing R/S which will continue with the next program line (if any).

MISCELLANEOUS UTILITIES AND ROUTINES

ESCAPE will examine the character that is second from the right in ALPHA. If this character has a hex code < 20, it will be replaced with a space (hex code 20). This may be used in conjunction with DISASM on the NFCROM EPROM set to remove hex codes that would be interpreted as control codes or escape sequences by an external printer.

FETCH executes a CXISA (FETCH) microcode instruction at the address given in digits 3-0 of X. The result from the fetch is returned in X, and the fetched address is incremented and stored in L.

LISTU2 prints the encoded contents for the "U" EPROM in an EPROM set. This is useful when programming the contents of the Prototech, Inc. ProtoCODER (a user-programmable ROM emulator) onto EPROMs. To use, get the hex starting address (a multiple of 4) into digits 3-0 of X then execute LISTU2. A printer is required. U2 is a microcode subroutine used by LISTU2.

ROM displays a PARIO message.

RVIEW displays the first 3 digits of X, Y, Z, and T, separated by dots. The registers are not changed.

STK<>Σ provides the user with multiple stack capability. By using ΣREG (see the HP 41C Owners Manual) any block of consecutive registers can be selected. When executed, registers T, Z, Y, X, and L are exchanged with the first 5 summation registers. No normalization occurs.

U2 is a subroutine used by LISTU2 (see LISTU2).

X<>REG exchanges the X register with the absolute RAM register specified by the exponent of Y. No normalization occurs.

WARRANTY, SERVICE, ASSISTANCE

LIMITED WARRANTY: The ProtoPARIO and PARIO-1A EPROM set as well as all ProtoSYSTEM devices manufactured by Prototech, Inc. are warranted against defects in material and workmanship for a period of ninety (90) days from the date shipped from Prototech, Inc. Within this warranty period, Prototech, Inc. will repair or at its option replace a defective part at no charge to the owner, provided that Prototech, Inc. is contacted within the warranty period for shipping instructions. There will be a charge for repairs after the warranty period has expired. Prototech, Inc. assumes no responsibility for damage, either direct or consequential, from the use of its products. Prototech, Inc. will have no obligation to modify or update products after sale. This warranty does not apply to products damaged by accident or misuse, or to products that have been modified by anyone other than Prototech, Inc. Since the ProtoPARIO requires some user interfacing to signals from the outside world not controlled by Prototech, Inc., any IC chips on the ProtoPARIO which are found to be damaged will be replaced by the owner. Prototech, Inc. guarantees that the chips will be functioning properly at time of shipment. This warranty is made in lieu of all other warranties, either express or implied.

If your ProtoPARIO or any other Prototech, Inc. product requires service, contact Prototech, Inc. for instructions.

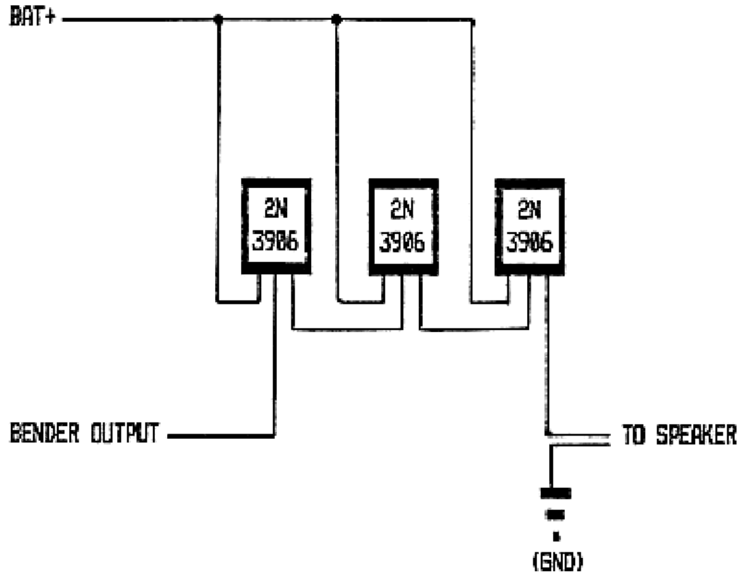
If you need technical or applications assistance relating to the use of any Prototech, Inc. product, please contact Prototech, Inc. at (303)-447-9883 (no collect calls), or write to:

PROTOTECH, INC.
P O BOX 12104
BOULDER, CO 80303 USA

INTERNAL BENDER AMPLIFIER

This simple curcuit can be built within the calculator to provide a large volume increase from the bender outout into an external speaker. The only parts you need are a miniature speaker (about 1 1/2 in.), 3 transistors (2N3906), a small plug and jack, and some wire and solder; total cost about $3. Note that this modification is not supported by HP and may void your warranty. Prototech, Inc. assumes no responsibility for the use of this amplifier. It is provided for the users reference only.

Remove the battery pack and all modules then remove the four screws from under the rubber pads on the back of the calculator and lift the back of the calculator off. Locate the bender (1-inch flag metal disk stuck onto the CPU) and unstick it. There are two wires connected to the bender. The inner one on the smaller section of the bender is the bender output signal. Solder a wire on top of the wire that is already there. Locate the plastic-copper battery contacts (where the battery pack plugs in) and scrape a small hole in the plastic at some location on both the BAT+ and GND contacts that will not get in the way of the battery pack. Solder a wire to each contact. Locate a place to put the output jack. I used a Radio Shack plug and jack combination that fits tightly in the battery charger hole. You should now have 3 wires added on to your calculator: bender output, BAT+, and GND. Solder the 3 transistors together as shown below and attach to the 3 wires and the jack. Also wire the two speaker contacts to the plug. The transistors will fit easily in the calculator along the side of the I/O ports. After verifying that you have wired everything correctly, reassemble and try it out. Note that this DOES draw significantly more power than the bender alone. Transistors below are shown with their flat face forwards.

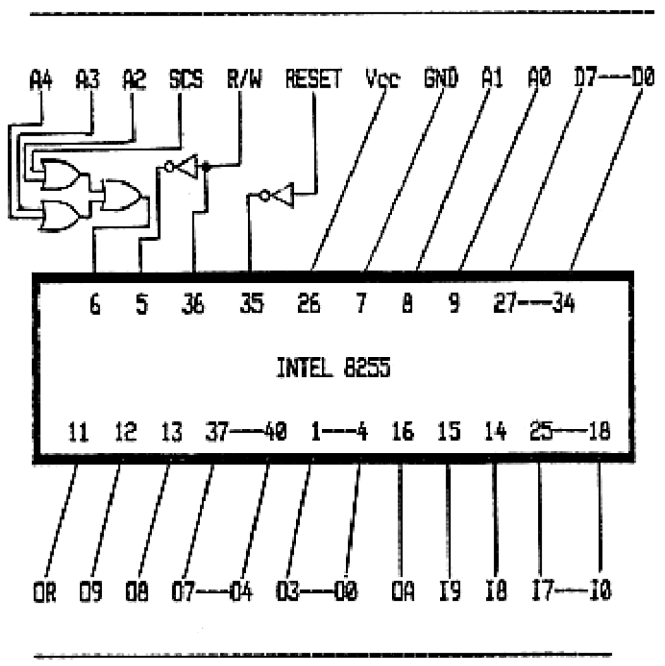


ProtoPARIO INTERFACE TO TRS-80 COLOR COMPUTER

The following circuit diagram illustrates a possible interface between the HP 41C and the Radio Shack Color Computer. The bus connections to the 74LS04, 74LS32 and Intel 8255 are all available at the plug-in ROM port (sound similar to the calculator??) on the side of the computer. Use the PARIO-1A EPROM for programming on the calculator side. To initialize the interface, POKE &HFF43,152. This will set up the A port and C7-C4 as outputs from the calculator, and the B port and C3-C0 as inputs to the calculator on the 8255. Input O7-O0 from the calculator with PEEK (&HFF40). Output I7-I0 to the calculator with POKE &HFF41,I where I is 0-255 decimal. O9, O8, and OR are available as C5, C4, and C6 with PEEK (&HFF42). I9, I8, and OA (C1, C0, C2) can be programmed by POKEing to &HFF42.

If you build this interface, test it carefully before attaching to the HP 41C. This circuit is presented as an example only: Prototech, Inc. assumes no responsibility for the accuracy or use of this information.
Parts required are: 74LS32 QUAD OR, 74LS04 HEX INVERTER, INTEL 8255 PIA, and 3-.1 uf bypass capacitors - one per chip.

SIGNALS TO TRS-80 COLOR COMPUTER ROM PORT BUS


A4 A3 A2 SCS R/W RESET Vcc GND A1 A0 D7---D0
6 5 36 35 26 7 8 9 27---34
INTEL 8255
11 12 13 37---40 1---4 16 15 14 25---18
OR O9 O8 O7---O4 O3---O0 OA I9 I8 I7---I0

SIGNALS TO ProtoPARIO BOARD ON HP 41C

Use CODE & DISASM

F4ØØ - clears all outputs

add $O_9 O_8 | O_7 O_6 O_5 O_4 | O_3 O_2 O_1 O_Ø$

(3 hex digits) to F4ØØ

to turn on particular

output. e.g. F5Ø2 will

turn off all but O_8 and O_1

FARIO BOARD