PET USER'S GROUP NEWSLETTER

VOLUME Ø NUMBER 3

T-A-B-L-E O-F C-O-N-T-E-N-T-S

N-E-W-S-L-E-T-T-E-R S-U-B-S-C-R-I-P-T-I-O-N

This is the first issue of a PET NEWSLETTER subscription.

A subscription to the PET NEWSLETTER is defined as a six and only six issues commencing with Vol.0, No.3 and terminating with issue 8, the last issue of volume 0, regardless of when you place your subscription. Issues 0, 1 and 2 are back issues and are not included in the subsription.

PET USERS' GROUP NEWSLETTER SUBSCRIPTION AND BACK ISSUE ORDER FORM

Please send me:

( )	Six-issue subscription to PET Newsletter	$4.50
( )	Back Issue, Volume 0, Number 0	.75
( )	Back Issue, Volume 0, Number 1	.75
( )	Back Issue, Volume 0, Number 2	.75

Your Name _____

Address _____

City/State _____ ZIP _____

Send order to: PET Newsletter / Computer Project
Lawrence Hall of Science
University of California
Berkeley, CA 94720

MAKE ALL CHECKS PAYABLE TO: REGENTS OF THE UNIV.OF CALIF.

S-O-F-T-W-A-R-E R-E-V-I-E-W

In early May, Lawrence Hall of Science received three cassettes from the Peninsula School loaded with programs and each accompanied by a user's manual containing instructions and complete listings. Tape #1 contains PILOT, an easy to learn conversational programming language and five programs written in PILOT. Peninsula School PILOT closely conforms to PILOT 73, a Bay Area standard. They have added two new PILOT commands: H for Head which is the same as T:[clear] and F for Foot which types "Press RETURN to go on", waits for the user to type any key and executes an H command. The user in entering a PILOT program is actually entering BASIC statements without BASIC commands. The PILOT interpreter PEEK's into memory to decode the PILOT statements and when the user SAVEs his PILOT program, the PILOT interpreter is saved right along with his PILOT code.

PILOT itself does not need line numbers; it uses alphameric labels for branching. The manual does not explain the use of labels, and in fact, Peninsula PILOT differs from PILOT 73 in requiring that every label be on a separate line.

All the PILOT programs included on TAPE #1 except HAMURABI are good examples of the simplicity and transparency of the language. HAMURABI was originally written in DEC FOCAL and most of us have copies in BASIC. HAMURABI in itself is a great simulation requiring minimal interaction with the user, but when translated into PILOT, results in unreadable code since the game relies on more computation then PILOT is designed for.

Tape #2 contains WSFN, Which Stands For Nothing robot/graphic language designed by Lichen Wang; LEMON, a simulation of a lemonade stand; and a couple of "freebees": KALEIDOSCOPE and RENUMBER, which are in the PUG/SPHINX libraries. The user's manual for tape #2 is very complete. It even contains Wang's original article and some good sample programs written in WSFN. Like PILOT, young children can quickly pick up the WSFN language, but don't try to feed them the entire command set, let them discover parts of the language as they need to. Despite WSFN's simplicity, it clearly has the power to create very complex patterns with minimal statements, some of which have been presented in Scientific American. The graphic movements of the robot are made using the "quarter square" plotting found in the Pet Plotting article on page 3.

LEMON lets you "run a lemonade stand" where you have to decide on how many glasses to make, number of signs to post and what selling price to ask for each glass. Your win/loss potential is expressed in the PROFIT margin that appears in the table for each day. Random "incidents" throughout the game add interest. The original game was from the Minnesota Educational Computer Consortium where it was a multi-player simulation. In fact, it still is a multi-player simulation on the Video Brain color computer available at Macy's.

The last tape, #3, we received contained QUEST and DRAW. DRAW is the full-blown super-sketch program seen in recent issues of People's Computers.

What can I say about QUEST? If you haven't played QUEST, you haven't played a real computer game. QUEST is a derivative of ADVENTURE and DUNGEONS & DRAGONS. Mini-"epic" games are discussed in the March-April issue (Vol.6,No.5) of People's Computers. I've spent hours with this game, searching through caves, being attacked by the Giant, sliding down giant stalactites,.... Once you start playing, you will not be able to stop!

In general, the manuals that accompany these cassettes are good models for w[illegible]-be authors for they are complete and concise.

[illegible]e cassettes and manuals can be obtained from: Computer Projec Peninsula School, Peninsula Way, Menlo Park, CA 94025 for $19.95 for PILOT et.al., $14.95 for WSFN, LEMON, et al., and $9.95 for QUEST and DRAW.

F-L-E-A C-O-L-L-A-R C-R-I-M-E

A disturbing event was reported at a recent PET meeting. A program bearing a copyright was being copied and "shared". The function of the club is to pool our knowledge and share our interest in computing; it is not to provide a base for violating copyright laws.

It is difficult in an ambience of sharing to keep in mind that some programs are not to be copied. But, as anyone who has written a significant program knows, writing good software takes a lot of time. If someone wants to receive financial acknowledgement, indeed earn a living, from his programs and places a copyright upon it, that must be honored. As a club we have a responsibility to foster this attitude. Perhaps the underlying notion of value is unclear. It is easy to accept that something concrete and material like hardware has value; it is more difficult to accept that software has value too.

From both a legal and ethical point of view the issue is clear -- copyrighted programs are not to be copied. From a practical point of view as well, it is in our own self interest to discourage abuse. It is important [illegible] maintain an atmosphere which will encourage novel and imaginative software development. Copyright protection provides that environment.

P-E-T P-L-O-T-T-I-N-G

Last year, Commodore produced a demo program that graphed a sine-curve using "quarter squares". That means for every standard character position, we could plot up to four points, two vertical and two horizontal, increasing the resolution of the PET screen to 50 vertical and 80 horizontal. The crucial problem is to avoid destroying the existing points in the character position when adding a new point. This is accomplished by examining the pattern at the character position and computing which of the 16 possible patterns must replace it. The following subroutine written by Arthur Luehrmann is such an algorithm using the quarter square symbols found on the , ; < > ? ! and " keys and their video reverses. The actual routine starts at line 1000 and uses variable names starting with letter "O". Therefore, you should avoid using any variables starting with "O". The first time thru, the routine dimensions and assigns the O() array and OO=9999. From then on, the routine will skip this initialization (since OO now equals 9999) and just computes the quarter square points starting at line 2000. All points are POKEd onto the screen by line 2100. Lines 100 to 160 are included as a demo driver, assigning X and Y before calling the subroutine at line 140. The routine expects an integer X ranging from 0 to 79 and integer Y from 0 to 49, with the origin (X=0 and Y=0) at the lower left of the screen.

```
  10 REM LAWRENCE HALL OF SCIENCE
  20 REM UNIVERSITY OF CALIFORNIA
 100 PRINT "[clear]"
 120 FOR X=0 TO 79
 130 Y=INT((SIN(X/5))*25)+25              :REM  Compute Y given X
 140 GOSUB 1000                           :REM  Call plot subroutine
 150 NEXT X
 160 END

1000 IF OO=9999 THEN 2000                 :REM  If flag set, OO=9999, skip to 2000
1010 DIM O(19)                            :REM  Dimension
1020 FOR O=0 TO 19
1030 READ O(O)                            :REM  Assign
1040 NEXT O
1050 DATA 96,123,108,98,126,97,127,252
1060 DATA 124,255,225,254,226,236,251
1070 DATA 224,1,2,4,8
1080 OO=9999                              :REM  Set flag, OO=9999

2000 O1=INT(X/2):O2=INT(Y/2)              :REM  Compute character position
2010 O3=X-2*O1:O4=Y-2*O2:O5=2*O4+O3
2020 O6=O(16+O5):O7=32768+40*(24-O2)+O1
2030 O8=PEEK(O7)                          :REM  is another point already there?
2040 IF O8<=96 THEN 2100
2050 FOR O=1 TO 16
2060 IF O8<>O(O) THEN 2090
2070 O6=O6 OR O:GOTO 2100                 :REM  Found point. "OR" it with new point
2090 NEXT O
2100 POKE O7,O(O6)                        :REM  POKE it to the screen
2110 RETURN
```

PERFORMANCE OF ON-GOTO

by Chuck Johnson, SPHINX

The PET ON-GOTO statement has a few subtle undocumented traits that can possibly be used to the programmer's advantage. The accompanying program and sample run test ON-GOTO for a variety of conditions. (Note in the PRINT statement of line 20 that "[←]" denotes a single "cursor left" control.)

In the sample run, #1, #2, and #3 perform as expected: we input integer values of 1, 2, and 3 for I and the ON-GOTO branches accordingly.

In #4 and #5, however, we input positive integers which are outside the ON-GOTO branch list (i.e., 40, 50, and 60). First, note that these out-of-range values do not cause an error. Instead they "fall through" to the next executable statement. Second, that "next executable statment" is on the same line number as the ON-GOTO, separated by a colon. This contrasts with the more commonly used IF-THEN statement. When IF-THEN "falls through" (i.e., fails) the next executable statement must be on the next numbered line.

Looking at #6 shows that π (shift - 4 , just above RETURN) is considered to be a symbol, rather than a numerical value. Further, #6 and #7 show that ON-GOTO automatically converts the argument value to integer and makes the conversion by truncation (rather than rounding-off).

In conclusion, ON-GOTO will accept and adjust for any non-negative values, but a negative argument causes an unrecoverable error, as shown in #8.

```
LIST
10 N=0
20 N=N+1 : PRINT "#";N;"[←]:  I"; : INPUT I
30 ON I GOTO 40, 50, 60 : PRINT "FALL THROUGH" : GOTO 20
40 PRINT "@40 I=1" : GOTO 20
50 PRINT "@50 I=2" : GOTO 20
60 PRINT "@60 I=3" : GOTO 20
READY.

RUN
#1:  I? 1
@40 I=1
#2:  I? 2
@50 I=2
#3:  I? 3
@60 I=3
#4:  I? 0
FALL THROUGH
#5  I? 4
FALL THROUGH
#6  I? π
?REDO FROM START
? 3.14159
@60 I=3
#7  I? 2.71828
@50 I=2
#8  I? -1
?ILLEGAL QUANTITY ERROR IN 30
READY.
```

commodore

COMMODORE INTERNATIONAL LIMITED
901 CALIFORNIA AVENUE
PALO ALTO, CALIFORNIA 94304
TELEPHONE: (415) 326 4000 TELEX: 345 569
CABLE ADDRESS COMBUSMAC PLA

April 14, 1978

PET Newsletter
Lawrence Hall of Science
Computer Project
University of California
Berkely, CA 94720

Dear PET Users,

As you may all well know, the current PET software is undergoing a revision. Hopefully, the majority of bugs will be corrected. This is also a perfect opportunity to make known your wants and needs in the system software. Some are already planned for incorporation such as the cursor fix and the RAM vector for I/O.

If you have any ideas, write a letter to me at Commodore. Please do not call, as I am usually overloaded. Also, please excuse me if I can't give a personal reply to your letter. We will keep you posted.

Yours truly,

John Feagans

John Feagans

JF:jh

P.E.T. EDGE CONNECTORS

Fits the IEEE-488 & User Port attachments. These are 24 contact TRW connectors of the type specified by Commodore. Two polarizing keys included. Solder eyelets for wiring. $3.50 each

Contact: Chuck Johnson 415/278-6595
17104 Via Alamitas
San Lorenzo, CA 94104

Note: I can get wirewrap and/or 2nd ssette connectors, if there is interest.

RESEQUENCE--JOE TRIMBLE,5/9/78
**TYPE 'LOAD', THEN PRESS PLAY BEFORE 'RETURN'. AFTER 'READY' PRESS 'HOME' AND 9 'RETURN'S.
**TYPE 'GOTO 63988'. YOU MIGHT WANT TO DELETE 63988-63999 BEFORE SAVING THE RESULTING PROGRAM WHICH IS SEQ'CED BY 10'S.

This program will change not only line no.'s but all references to line no.'s. CAUTION: changing a digit to >n-digit line no's. Can you make this program shorter?

```
63988 DIML(999):L=1025:DEFFNR(X)=PEEK(X)+256*PEEK(X+1):DEFFNM(X)=INT(10*X/256)
63989 N=FNR(L):X=FNR(L+2):IF X<63988 THEN A=A+1:L(A)=X:L=N:GOTO 63989
63990 L=1025:FOR B=1 TO A:N=FNR(L):POKE(L+3),FNM(B):POKE(L+2),10*B-256*FNM(B)
63991 F=0:FOR C=L+4TON-1:P=PEEK(C):IF P=137OR P=141OR P=167THEN F=1:GOTO 63999
63992 IF F=0 GOTO 63999
63993 IF P>47 AND P<59 THEN D=10*D+P-48:G=G+1:GOTO 63999
63994 IF D=0 GOTO 63999
63995 FOR E=1 TO A:IF D=L(E)GOTO 63997
63996 NEXT E:D=0:G=0:GOTO 63999
63997 D=0:E$=STR$(E*10)+"   ":H=LEN(E$)-4:C=C-G:IF H>G THEN C=C-1:G=H
63998 FOR I=1 TO G:POKE C,ASC(MID$(E$,I+1,1)):C=C+1:NEXT I:G=0
63999 NEXT C:L=N:NEXT B:END
```

T-R-Y T-H-I-S

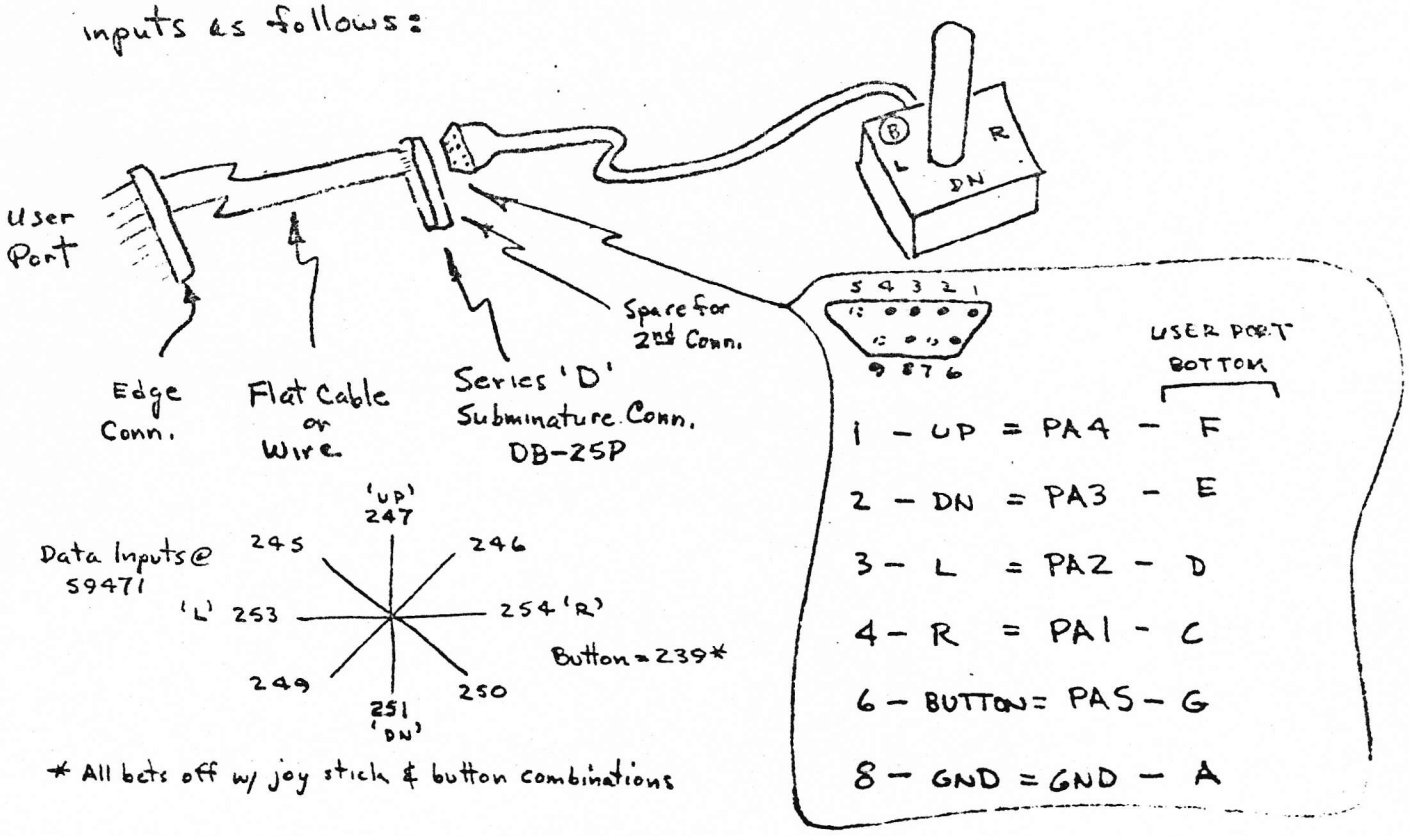
1. Turn on your PET
2. Type L then shift O
3. Press the RETURN key

What does your PET do now?

Try L then shift I and press RETURN. Try S then shift A and press RETURN. At the April meeting of SPHINX, a young PET programmer located this shortcut. You can save some time by typing only two characters for a command. You will have to type three characters for RETURN and RESTORE. Example: Inside a BASIC program, typing R then E then shift S is the minimum number of characters necessary to distinguish RESTORE from RETURN and READ. What happens if you type R then shift E inside a program?

ATARI® JOY STICK INTERFACE

The joy stick furnished with the ATARI TV game provides simple contact closure for each of the four positions. Moving the stick at 45° produces two contact closures. Interfacing to the User's Port requires only connecting to the inputs as follows:



'BREAKOUT' PATCH - Delete or change

```
200 J = PEEK(59471): IF J=255 THEN PM=0: GOTO 220
205 IF J=253 THEN PM=-1
210 IF J=254 THEN PM=+1
220 PP = PP + PM  (No Change)
```

You Can Develop the Patch to have the Button Start a New Game

TEST PROGRAM - RUN 1000

```
1000 ? "♡" ← Screen Clear
1010 ? PEEK(59471)
1015 ? "S" ← Home
1020 GOTO 1010
```

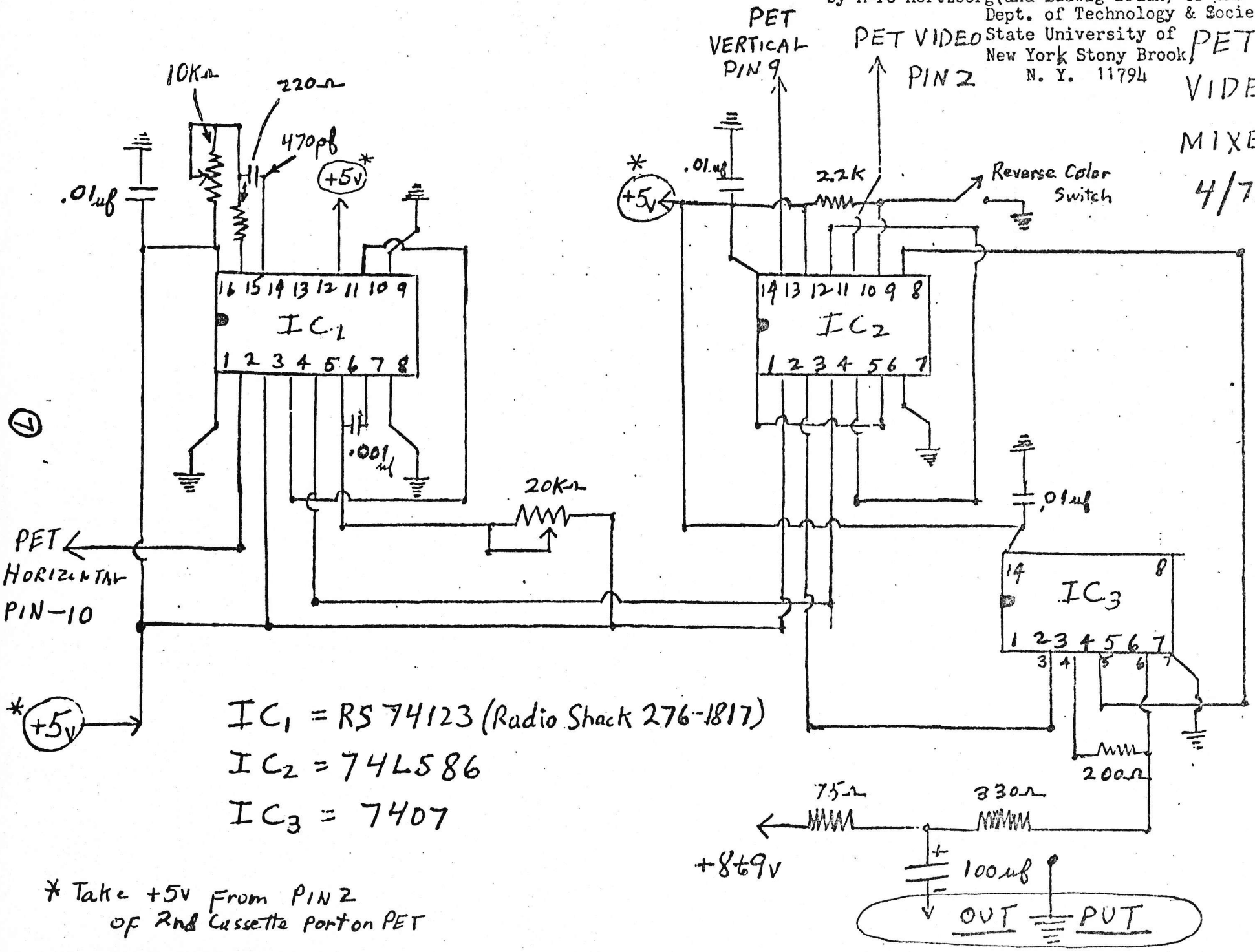
George Milum
(415) 284-1856

4/24/75

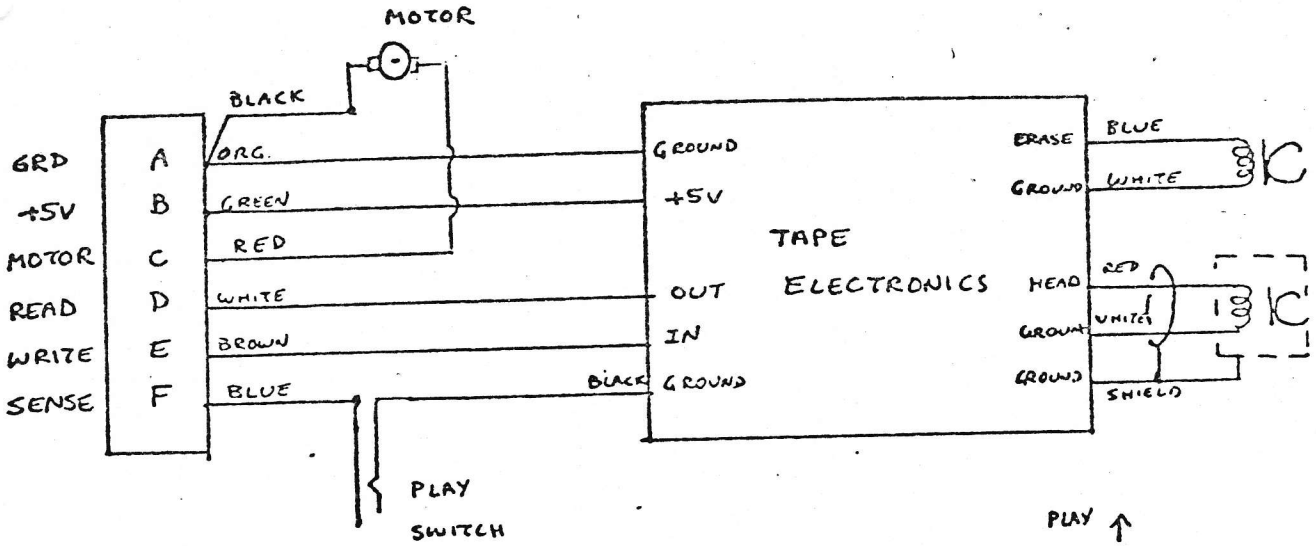
IC Circuitry C Modified by Randall Julin
of Daly City, CAL. Original circuit designed
by Marc Hertzberg (and Ludwig Braun) of the
Dept. of Technology & Society
State University of
New York Stony Brook
N. Y. 11794

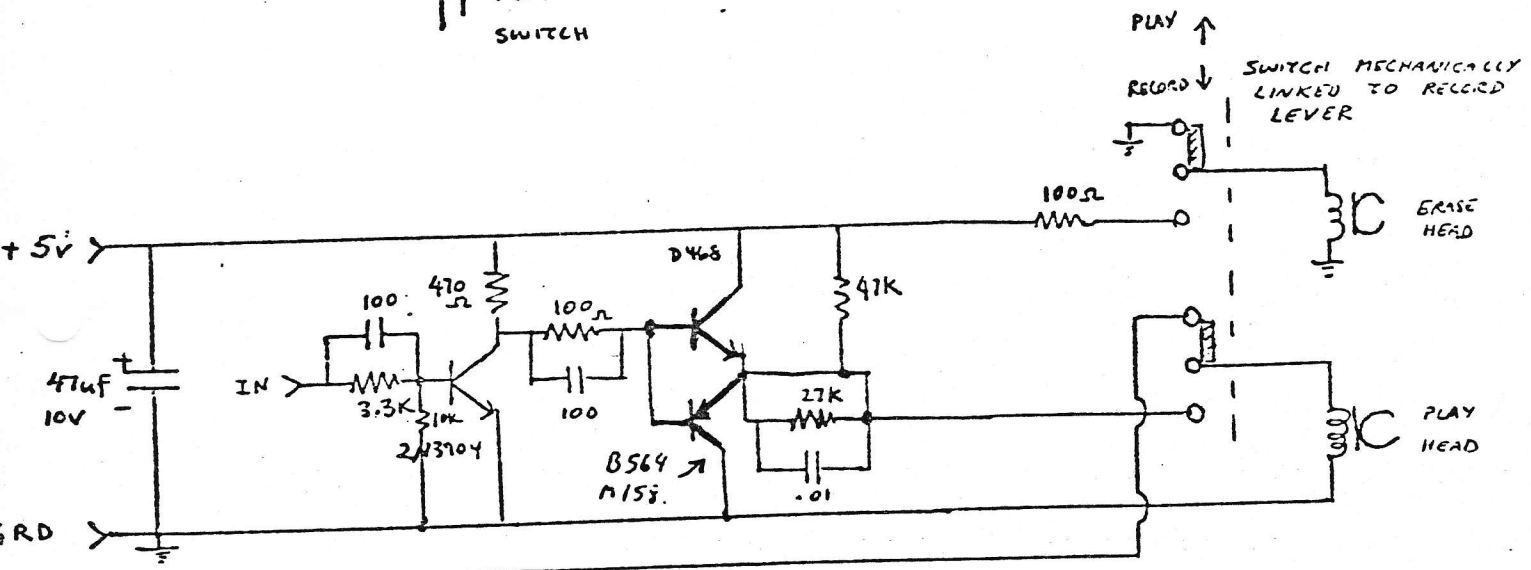
PET VIDEO MIXER

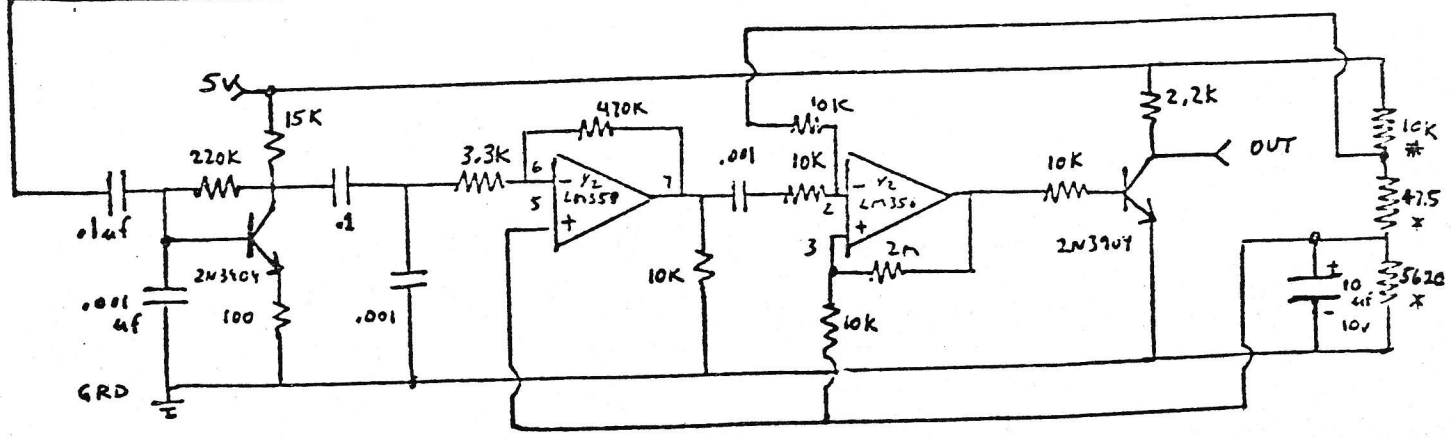
4/78



PET TAPE INTERFACE







NOTES * PRECISION RESISTORS

LM358 V+ = 5V PIN 8

V- = 0V PIN 4

ALL RESISTORS 1/4 W 5% TOL. UNLESS OTHERWISE SPECIFIED

PARALLEL-TO-SERIAL COMMUNICATION LINK

FOR THE COMMODORE PET

P. K. Govind

National Center for Atmospheric Research
P.O. Box 3000
Boulder, Colorado 80307

Introduction

This article describes a simple parallel-to-serial and serial-to-parallel interface which allows the Commodore PET (through the 8-bit user port) to converse with external devices which employ an RS232C serial input/output.

Parallel-to-Serial Communication Link

Fig. 1 is a functional block diagram of the serial communication link which can be implemented by configuring the 8-bit user port to operate in either one of two modes: parallel output mode or parallel input mode.

Parallel Output Mode

A parallel output procedure is used to send data bytes from the PET to the input port of the UART (Universal/Asynchronous Receiver/Transmitter) at TTL levels. The data bytes are then transmitted serially to a level converter in order to drive devices that support an RS232C interface (e.g., remote input ports of other computers). The TRANSMITTER BUFFER EMPTY signal notifies the PET that parallel-to-serial conversion requested by the DATA STROBE signal has been completed.

Parallel Input Mode

A parallel input procedure can be used to allow the PET to receive data bytes from a device that sends serial data. The RS232C serial input level is first converted to TTL level. The serial input

line of the UART accepts the bit stream and the receiver section of the UART converts that to parallel data. With the data direction switch in the RECEIVER ENABLE position, the RECEIVER DATA AVAILABLE signal is used to tell the PET that it has valid data available on its input port.

Circuit Boards

The serial communication interface was built using the following circuit boards available from Electronic Systems (P.O. Box 9641, San Jose, CA 95157):

1. UART and baud generator (Part 101A: $35)
2. RS232/TTL interface (Part 232A: $7)

NOTE: The DC power supply cost is not included. Typical costs could range from $45 to $75).

The UART and baud generator board has two functions: (1) to convert parallel data to a serial bit stream with start, parity, and stop characters; (2) to convert serial bit stream to parallel data. The baud generator allows the user to choose the following data rates: 110, 150, 300, 600 1200, or 2400 baud. A seven-section DIP switch allows the user to select the polarity of the input and output strobe separately, to select 5 to 8 data bits, 1 or 2 stop bits, and odd or even parity. Power required is +5V and -12V, if the General Instruments UART AY-5-1013 is used. However, there is no need for the -12V if the AY-5-1014 UART is used. All connections go to a 44-pin gold plated edge connector.

The RS232/TTL interface has two separate sections: (1) the RS232 driver section amplifies TTL levels to RS232C voltage levels; (2) the RS232 receiver section converts RS232C level to TTL level. The driver section is implemented with a general purpose operational amplifier circuit (741). The receiver section employs a single transistor circuit. The interface board has a 10-pin edge connector.

The overall intercabling diagram is shown in Fig. 2. Simple

modifications made to the UART and baud generator board are shown in Fig. 3.

Software

Listing 1 contains the program written in BASIC for the parallel output mode. This program can be used to generate a hard copy listing of the image displayed on the PET screen using a line printer with an RS232C interface. Although this program works at all settings (110 - 2400 baud) the effective data transfer rate is about 6 characters per second.

Listing 2 is a simple program which reads incoming serial data that has been converted to a parallel format by the communications interface. This program works for data transfer at 110 baud. An input driver written in assembly language should enable data transfer at much higher rates.

Acknowledgement: I would like to thank Jacques Brun for his assistance with this project.

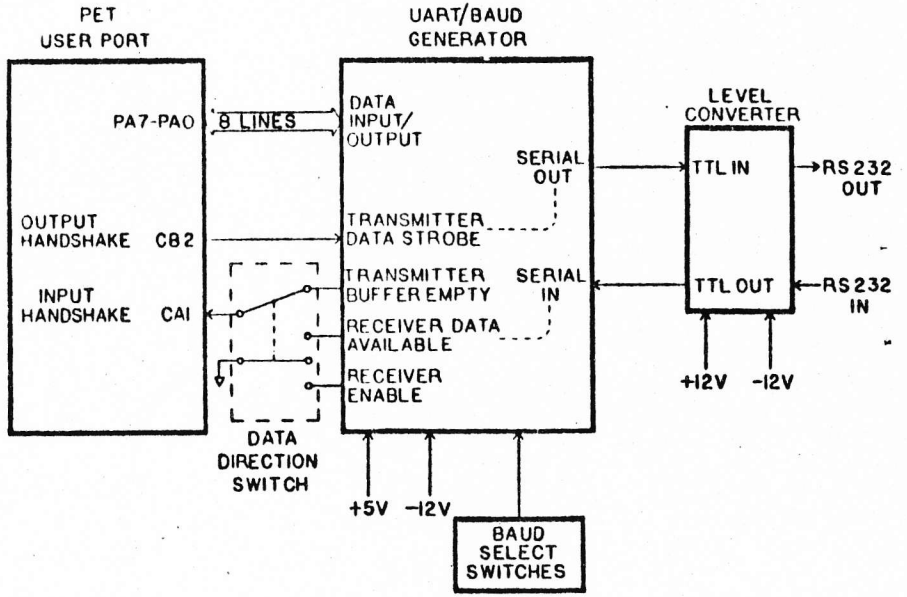


Fig. 1 Functional block diagram of the serial communication link between external devices and the 8-bit user port on the Commodore PET.

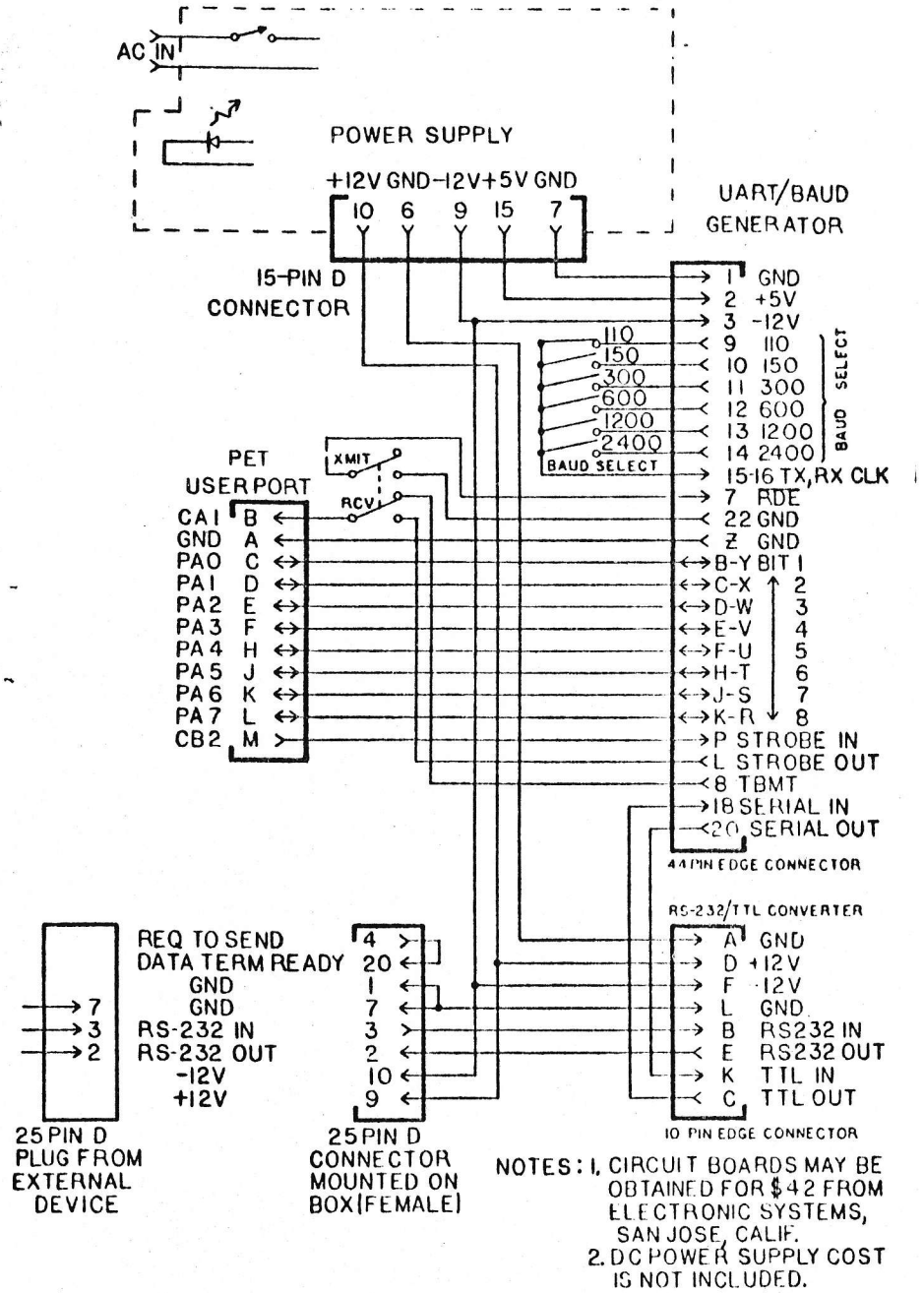


Fig. 2 Intercabling diagram for the parallel/serial communication link.

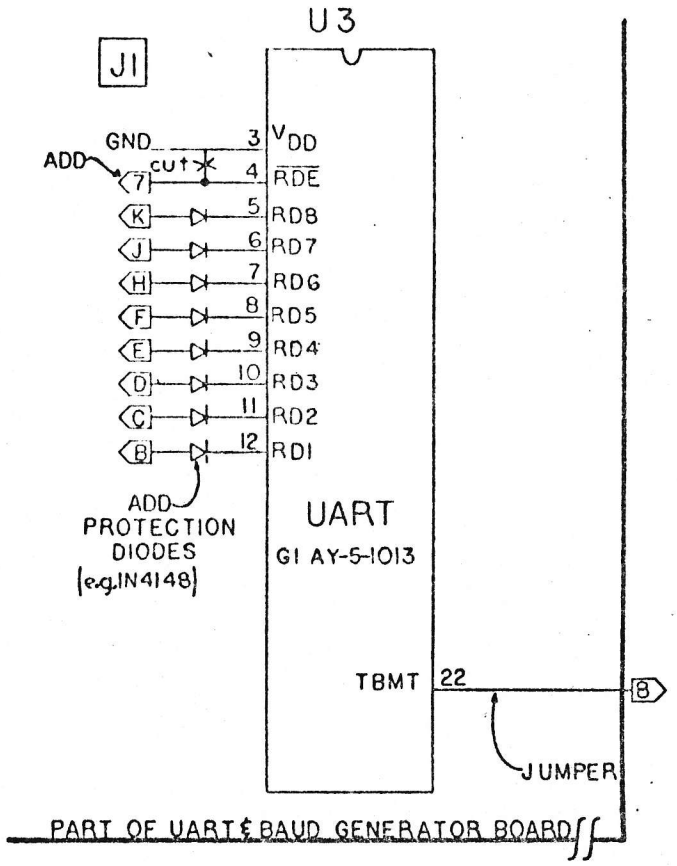


Fig. 3 Simple modifications to the electronic systems UART/baud generator board.

```
READY.

 1 REM FILENAME "PRINTSCREEN"
 2 REM THIS IS A GENERAL PROGRAM FOR
 3 REM FOR PARALLEL OUTPUT FROM THE PET.
 4 REM IT CAN BE USED WITH A PRINTER
 5 REM THAT HAS A DIRECT PARALLEL
 6 REM PORT SUPPORTING STROBE/ACKNLG
 7 REM OR WITH ONE HAVING A STANDARD
 8 REM SERIAL INTERFACE CONNECTED TO
 9 REM A PARALLEL/SERIAL CONVERTER.
 10 REM OUTPUT DATA TO EXTERNAL DEVICE
 15 REM  HANDSHAKE WITH LINE PRINTER
 16 REM CB2 FOR DATA STROBE;TO DEVICE
 18 REM CA1 FOR ACKNOWLEDGE;FROM DEVICE
 19 REM
 20 POKE 59459,255:REM DIRECTION OUT
 25 GOSUB 100:REM HANDSHAKE NOT READY
 34 FOR I=1 TO 25 :REM SCAN ROWS
 35 FOR J=1 TO 40 :REM SCAN COLUMNS
 36 V=PEEK(32767+J-1+40*(I-1))
 37 IF V>64 THEN V=V+32 :REM LOWER CASE
 38 IF V<=26 THEN V=V+64:REM UPPER CASE
 39 IF V=128 THEN V=V-96:REM SPACE
 40 IF J=1 THEN 180 :REM PRINT SPACE
 50 POKE 59457,V AND 127:REM SEND VALUE
 51 GOSUB 150:REM READY TO OUTPUT
 52 GOSUB 100:REM NOT READY
 56 ACK=PEEK(59469)AND2:REM INT FLG REG
 58 IF ACK <> 2 THEN 56:REM ACKNOWLEDGE
 70 NEXT J
 72 POKE 59457,13:REM CR
 73 GOSUB 150:REM READY
 74 GOSUB 100:REM NOT READY
 76 POKE  59457,10:REM LF
 78 GOSUB 150:REM READY
 80 NEXT I
 82 GOSUB 100
 84 POKE 59457,128 :REM STOP PRINT
 85 PRINTCHR$(147) :REM CLEAR SCREEN
 86 END
 98 REM       SUBROUTINES
 100 REM SET CB2 TO LOGIC 1:NOT READY
 110 POKE(59468),PEEK(59468) OR 224
 120 RETURN
 150 REM SET CB2 TO LOGIC 0 :REM READY
 160 POKE (59468),PEEK(59468)AND310R192
 170 RETURN
 180 V=32 AND 127 :REM SPACE
 182 GOSUB 150:REM READY
 184 GOSUB 100:REM NOT READY
 186 GOTO 50
 200 PRINT" Upper and Lower Case "
 240 PRINT"ABCDEFGHIJKLMNOPQRSTUVWXYZ"
 250 PRINT"abcdefghijklmnopqrstuvwxyz"
 300 PRINT" These listings were made on
 310 PRINT" TI Model 810 printer"
READY.
```

Listing 1 Parallel output driver in BASIC

```
READY.

 400 REM PARALLEL INPUT MODE
 405 REM READ A LINE OF ASCII
 410 REM INCOMING SERIAL CONVERTED TO
 420 REM PARALLEL FORMAT BY THE
 430 REM SERIAL TO PARALLEL INTERFACE.
 440 REM
 450 POKE 59459,0:REM DIRECTION IN
 460 D=PEEK(59457) AND 127 :REM GET DATA
 470 ACK=PEEK(59469) AND 2:REM INT FLG
 480 IF ACK <> 2 THEN 470
 490 PRINT CHR$(D);
 500 GOTO 460
READY.
```

2. Parallel input driver in BASIC.

A SHORT DESCRIPTION OF THE IEEE-488 BUSS FOR THE PET

This description covers the pin-out and signal designations for the IEEE-488 as plemented on the PET. A brief description of the PET basic commands for the IEEE-488 uss is also included.

INTERCONNECTION

The PC card edge on the left-rear of the PET labeled J1 has the IEEE-488 signals. For reasons of economy, a standard IEEE-488 connector is not included.

A standard 12-position, 24-contact edge connector with .156" spacing is ttached to the PET PC card. Some typical connectors and part numbes are:

EDGE CONNECTOR BRAND	PART #
SYLVANIA	6AGØ1-12-1A1-Ø1
AMP	53Ø657-3
AMP	53Ø658-3
AMP	53Ø654-3
CINCH	251-12-9Ø-16Ø

In a pinch, a larger edge connector (such as 15 or 22 positions) can be cut with a hacksaw to provide a temporary substitute.

The IEEE-488 (or HP-GB/IB) connector is available from:

IEEE CONNECTOR BRAND	PART #
CINCH	571Ø24Ø SOLDER PLUG
CINCH	572Ø24Ø SOLDER RECEPTACLE
AMP	5523Ø1-1 INSULATION DISP PLUG
AMP	5523Ø51-1 INSULATION DISP RECP

The pin designations and numbers are identical for both connectors. A short cable (i.e., 15 conductor ribbon, etc.) may be used to join the connectors.

PET PC-CONNECTION

TOP	DIO1	DIO2	DIO3	DIO4	EOI	DAV	NRFD	NDAC	IFC	SRQ	ATN	CHASSIS GND	
PIN	1	2	3	4	5	6	7	8	9	10	11	12	Polarization slots between 2-3 and 9-10
PIN	A	B	C	D	E	F	H	J	K	L	M	N	
BOTTOM	DIO5	DIO6	DIO7	DIO8	(REN)GND	GND							

IEEE-488 CONNECTION

IEEE DESIGNATION	PIN		IEEE DESIGNATION
DI01	1	13	DI05
DI02	2	14	DI06
DI03	3	15	DI07
DI04	4	16	DI08
EOI	5	17	REN
DAV	6	18	GND6
NRFD	7	19	GND7
NDA	8	20	GND8
IFL	9	21	GND9
SRQ	10	22	GNDI0
ATN	11	23	GNDII
SHIELD	12	24	LOGIC GND

SCHEMATIC OF

CONNECTOR POLARIZATION .

SUGGESTION: When wiring the edge-connector to the IEEE connector, include a 16 pin dip socket to jumper the control lines. This permits easy modification of the connection to IEEE-488 Interface. (These are described later.)

II. SOME PHYSICAL LIMITATIONS:

1. Maximum length: 20 meters

II. SOME PHYSICAL LIMITATIONS: (cont'd)

2. Maximum inter-device spacing: 5 meters
3. Maximum number of devices: 15
4. Maximum date rate: 250 KHZ (I MIIZ with tristate drivers)

III. GENERAL CONCEPTS

The IEEE-488 BUSS is comprised of three functional groups of lines:

DI01, DI02, DI03, DI04, DI05, DI07, DI08 — DATA BUSS

The data buss transfers data at a rate controlled by the slowest device on the buss. The form is byte-serial/bit-parallel (i.e., a byte at a time).

Also transferred on the data buss are peripheral addresses or control information.

NRFD, DAV, NDL — TRANSFER BUSS

This set of lines controls the transfer of data on the data buss. This buss ensures that date is valid and that all transfers are complete before new data is sent.

ATN, SRQ, IFL, REN, EOI — MANAGEMENT BUSS

The management buss controls the state of the buss, commands for the devices, etc

The buss can support three classes of devices:

1. TALKERS. At any given time, only one device may transmit data to the buss. Devices capable of this are talkers.
2. LISTENERS. As many devices as required may receive data from the buss.
3. CONTROLLERS. At any moment, only one device may control the buss. Control can be passed to other devices capable of controlling the buss.

BUSS SIGNALS

A. THE DATA BUSS

Lines DIO1 - DIO8 are the data buss. These are active-low bidirectional lines. is means a line is normally high. Any device can ground the line, making a signal present.

Data is transferred in bytes, one bit per line, with the MSB in DIO8. The forms data are:

1. Data from instruments
2. Address - primary or secondary
3. Control words

B. THE TRANSFER BUSS

The transfer of data over the data buss is controlled by these three lines. The ndshake sequence ensures complete transmission and reception by the slowest device on the ss.

B. THE TRANSFER BUSS (cont'd)

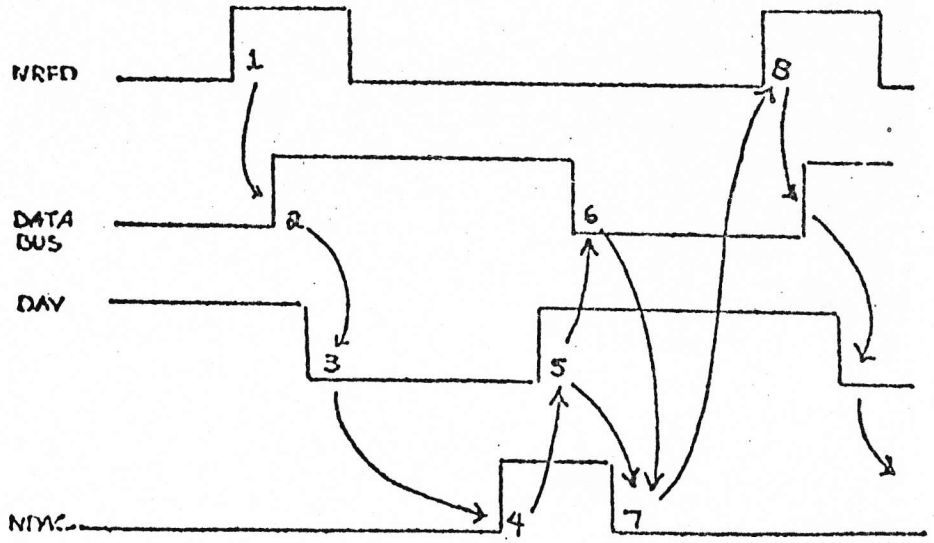
LINE

NRFD NOT READY FOR DATA. When this line is low, one or more listeners are not ready for the next byte of data. When all devices are ready, NRFD goes high. This informs the talker to put the next byte on the data buss.

DAV DATA VALID. When this line goes low, the listeners may read the data byte on the data buss. The talker cannot put DAV low if NRFD is low (All listeners must be ready first).

NDAC DATA NOT ACCEPTED. Each listener holds this line low until it has finished reading the data byte. When NDAC goes high, the talker can remove the data buss and go to the next byte.

A simplified diagram of the handshake sequence looks like this:



EVENT

1. When NRFD goes high, the talker is permitted to put data on the data buss.
2. The data is put on the buss and after a settling interval
3. DAV is set low to indicate data is valid. The devices accept data.
4. When all devices have accepted the data, NDAC goes high, permitting
5. The talker to move DAV and
6. Take the data off the data buss.
7. The listeners note the removal of DAV and resets NDAC in preparation for the
8. Next data transfer cycle

NOTE: When PET is a listener, it expects DAV within 64 milliseconds of NRFD going low (1-3 within 60 milliseconds).
When PET is a talker, it expects NDAC within 64 milliseconds of DAV 3-4.
Failure to observe these limitations may result in loss of data.

DATA PROTOCOLS

1. Any series of bit patterns is valid on the buss.
2. ASCII Data transfer:
   a. Numeric data is transmittable in either floating point or scientific format, with most significant digit first. Valid numeric characters are: 0 - 9, E, e, ±, -, .
   b. Strings are terminated with return or activation of the EOI line or both.

C. THE MANAGEMENT BUSS

Five signal lines control the activity of the buss and define the meaning of the data being transferred (data, address or control).

LINE

ATN ATTENTION. The controller sets this line to low when it is assigning devices as listeners and talkers. When ATN is low, only peripheral addresses and control messages are on the data buss. When ATN is high, only assigned devices can transfer data.

SRQ SERVICE REQUEST. Any device can set SRQ low to alert the controller that a device requires service. When the controller sets SRQ, it sets ATN low and does a "service poll" to find out which device wants service. NOTE: This bit is accessible in the PET. However, the PET 488 software does not include this function, and it is up to the user to do so.

IFC INTERFACE CLEAR. The controller sets this line to initialize the buss. NOTE: PET only activates this line when it is reset or powered up. The signal is low for about 100 milliseconds. If the user wants this function, it is suggested he place a switch on this line.

REN REMOTE ENABLE. Some devices hang the option of either operating from their front panels or the IEEE buss. When REN is low, control is via the buss. NOTE: The PET has this line set permanently low (the pin is grounded).

Put a switch in the line of REN control is desired.

EOI END OR IDENTIFY. When a talker is finished with data transfer, it sets EOI low. (This is optional). The controller always sets EOI low when it is finished. (EOI is set low during last byte transferred).

V. PET COMMANDS/BASIC STATEMENTS PERTINENT TO IEEE-488 BUSS

It is assumed the user knows how to read and write data to the tape casette files. See the casette tutorial bulletin for coverage of this area.

The IEEE-488 buss appears as a file to BASIC. The following BASIC items are pertinent:

OPEN CLOSE	Open/Close files (assign devices)
PRINT# INPUT# GET#	Transfer data
CMD	Direct PET's output elsewheres
ST	.I/O status variable

The following descriptions are only about the aspects which pertain to the buss.

OPEN (Logical Address), (Physical Device), (Secondary Address), "Filename"

The Logical Address is 1-255 and is referenced by the CLOSE, PRINT #, INPUT#, AND GET # statements.

The Physical Device is the Primary Device Address, and the range is 4-15.

The Secondary Address is optional. If omitted, none is sent. The range is 0-31. Bits 6 and 7 are set when sent to the buss.

If the address was 2, 00000010
is sent as 01100010

The Secondary Address is sent only on execution of the OPEN and CLOSE statements.

A specific form of the Secondary Address is sent if a Filename is specified for OPEN and CLOSE. Bit 8 is set in both cases, and bit 5 set on OPEN. As bit 5 is used to specify a control command,

SA 0-15 are files
16-31 are commands

CLOSE This will send the Secondary Address (if any) to the device specified by the open command.

PRINT# This will send ASCII characters to the IEEE-488 buss. If it is desired to set the most significant bit, use variations of:

PRINT# 2, CHR$ (X) range: 0 - 255

INPUT# Receives characters according to BASIC INPUT rules.

GET# Gets a character or a digit.

NOTE: PRINT, INPUT, and GET all refer to the Logical Address specified in the OPEN Statement.

CMD LOGICAL ADDRESS. All BASIC output is not sent to the device specified by a prior OPEN statement or command. This has two useful properties:

1. BASIC programs can be listed to a file or device.
2. CMD leaves the IEEE buss active, permitting more than one listener on the IEEE buss.

NOTE: Each time a PRINT# statement is executed, the following sequence happens:

1. The device specified in the corresponding OPEN statement is designated a listener.
2. The data is sent.
3. All devices are set to "not listen" status (UNL).

A similar sequence is used for INPUT#, with designation of a talker, and an untalk (UNT) command.

If a CMD is executed first, the specified device will also be able to listen when the PRINT# is executed. Note that CMD must be executed again if more than one PRINT# statement is used for multiple devices or PRINT #'s.

ST STATUS WORD. The following bits in the BASIC variable, ST, pertain to the IEEE-488 buss:

BIT	AND MASK	
0	1	Time out on data transfer
1	2	.... read error
....		
6	64	EOI

BIT	AND MASK	
7	128	Device not present

Use the form: IF (ST) AND MASK then --- (1, 2, 64, or 128) to detect these conditions. The test should be done immediately after the I/O operation of interest.

TIME OUT. BIT 4 MASK: 1 The IEEE device has not responded within 65 milliseconds (time out interval).

READ ERROR. BIT 1 MASK: 2 The IEEE device has not provided DAV within the time out - INPUT# or GET#.

EOI. This is set when an IEEE device finishes transmission of data (see the manual for the instrument as some devices won't do this). A convenience feature.

DEVICE NOT PRESENT. When I/O is initiated, the device did not respond to its physical address. This generates an error message and returns you to BASIC command level.

VI. IEEE-488 REGISTER ADDRESSES

If you are bold, here are the IEEE-488 hardware addresses for the PET. Attempting to control the buss via peek and poke will probably fail as the timeouts for the 488 devices may be exceeded.

NAME	HEX ADDRESS	DECIMAL ADDR	BITS	IEEE LINES	
IEEI	$1820	59424	0-7	DIO 1-8	(INPUT)
IEE	$E822	59426	0-7	DIO 1-8	(OUTPUT)
IEEIS	$E821	59425	3	$\overline{\text{NDAC}}$	(OUTPUT)
IEEOS	$E823	59427	3	$\overline{\text{DAV}}$	(INPUT)
			4	SRQ	***
PIAL	$E810	59408	6	$\overline{\text{EOI}}$	(INPUT)
PIA	$E840	59456	0	$\overline{\text{NDAC}}$	(INPUT)
			1	$\overline{\text{NRFD}}$	(OUTPUT)
			2	$\overline{\text{ATN}}$	(OUTPUT)
			6	$\overline{\text{NRFD}}$	(INPUT)
			7	$\overline{\text{AV}}$	(OUTPUT)

*** CBI input of VIA 6522 (see MOS Tech 6522 specs)

AT LAST !

The PET Manual

© by:

Gregory Yob

ANNOUNCING ---

A complete manual for the Commodore PET Personal computer. This book describes the PET in detail including a thorough manual for PET's BASIC language.

Also included are instructions for using the User Port, Cassette Files, and the IEEE-488 Port.

The details of BASIC are shown by many examples including several programs for your use which teach programming techniques as well.

It is organized in an easy to use reference format for continuing usefulness with your PET.

Take a look at the Table of Contents printed on the other side of this sheet.

Mind's Eye is pleased to accept your check, money order or Mastercharge/Visa card, and warns you that The PET Manual will not be available until July 1, 1978 (hopefully sooner).

Number of PET Manuals: _____
@ $ 16.00 each $ _____

Calif. Residents add
6 or 6½% Sales tax _____

Shipping & Handling $ 2.00

TOTAL: ═══

Your Name: _____
Address: _____
_____
State: _____ Zip: _____
MC/VISA Card # _____
Expiration date _____
Signature: _____

Mind's Eye SOFTWARE

GREGORY YOB
(415)326-4039

PO. Box 354
Palo Alto, CA.
94301

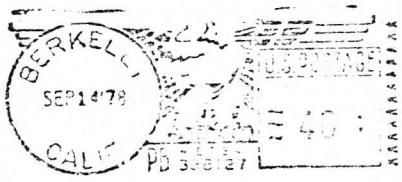
What's Here

Mind's Eye
SOFTWARE

PO. Box 354
Palo Alto, CA.
94301

(415)326-

LAWRENCE HALL OF SCIENCE
UNIVERSITY OF CALIFORNIA
BERKELEY, CALIFORNIA 94720



Barbara Vail*
4319 Cavalier St. N.E.
Cedar Rapids, IA 52402