This Document describes the seting up and the operation of the PSB-01 dual serial port.

1) Introduction

The PSB-01 is a circuit card that plugs into the PC-8012 expansion interface. This card contains two RS-232 serial channels. The two channels are implemented with uPD 8251 USARTS. The two channels have independently controllable Baud rates. The capability for interrupting the processor when receive characters are available is also provided.

When the I/O addresses and interrupt pins are properly connected the board will function with NBASIC. Statements such as PRINT% and INPUT% can then be used.

2) General Specifications

Two complete RS-232 channels

USARTS are uPD-8251

Baud rates are independently selectable from the following; 300,600,1200,2400,4800,9600

I/O addresses are settable anywhere within the range of 80H to OFFH

Each channel can be set up to provide an interrupt on Rx RDY Interrupts are settable within INT0 to INT7

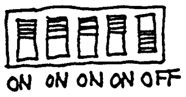
The board is functional within a temperature range of 40 to 90 degress Farenheit. The relative humitidy should be in the range of 20% to 80%

The board size is approximetly 220mm by 95mm

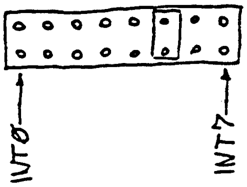
To maintain compatability with NBASIC the interrupts must also be set according to the following table.

Interrupt Table Addresses		Content	Channel
16K System	32K System		
C000 1	8000 1	Used by N-BASIC	IEEE-488
2 3	2 3	"	IEEE-488
4 5	4 5	Low-order byte High-order byte	Real-Time Clock
6 7	6 7		
8 9	8 9	Used by N-BASIC	RS-232C Ch.1
A B	A B	"	RS-232C Ch.2
C D	C D	Low-order byte High-order byte	INT9
E F	E F	Low-order byte High-order byte	INT8
10 11	10 11	Low-order byte High-order byte	INT7
12 13	12 13	Low-order byte High-order byte	INT6
14 15	14 15	Low-order byte High-order byte	INT5
16 17	16 17	Low-order byte High-order byte	INT4
18 19	18 19	Low-order byte High-order byte	INT3
1A 1B	1A 1B	Low-order byte High-order byte	INT2
1C 1D	1C 1D	Low-order byte High-order byte	INT1
1E C01F	1E S01F	Low-order byte High-order byte	INT0

If the I/O addresses are set at the addresses C0H to C3H then the board is usable with NBASIC. The switch settings corresponding to this setting are shown below.



Rx RDY interrupt capability is provided by Jumper sockets S4 and S5. S4 controlls channel 1 while S5 controlls channel 2. The jumper socket is illustrated below.



4) Usage with NBASIC

When all of the following conditions are met the board can be used with NBASIC.

I/O address   C0H-C3H

SW0(A2)	ON	SW3(A5)	ON
SW1(A3)	ON	SW4(A6)	OFF
SW2(A4)	ON		

Interrupt address

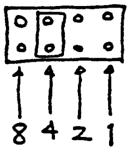
S5	INT3
S4	INT2

The PC-8012 must also be jumpered as follows;

CN2	1-2
CN1	1-2
CN5	1-2

The board is now ready for use with NBASIC.

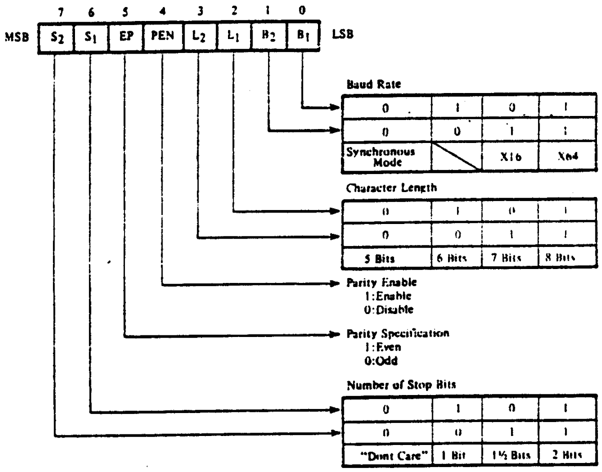
The baud rates are independently controllable by jumpers at sockets S3 (for channel 1) and S2 (for channel 2). The socket is illustrated below.



The baud rates which result from setting the jumpers are illustrated below.

	Divisor mode	
Jumper Pin	64	16
8	300	1200
4	600	2400
2	1200	4800
1	2400	9600

For Asynchronous mode:



Baud Rate

0	1	0	1
0	0	1	1
Synchronous Mode		X16	X64

Character Length

0	1	0	1
0	0	1	1
5 Bits	6 Bits	7 Bits	8 Bits

Parity Enable
1:Enable
0:Disable

Parity Specification
1:Even
0:Odd

Number of Stop Bits

0	1	0	1
0	0	1	1
"Dont Care"	1 Bit	1½ Bits	2 Bits

The first thing that must be done is to enable the interrupt controller. This is done with the following statement:

```
OUT  &HE4,&HFF
```

The channels must now be programed for use by the use of the following statement:

```
INIT%n,m,c
```

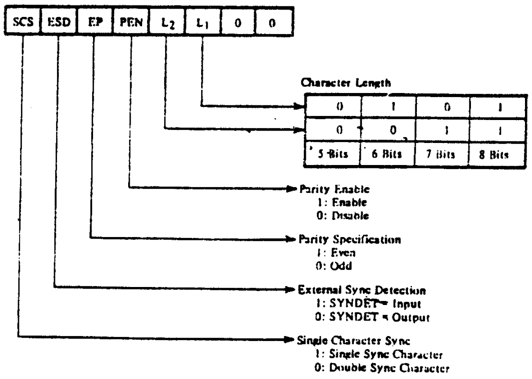
where n, m, and c, represent the following:

n: Channel number (either 1 or 2)

m: Mode word

c: Command word

Mode words are avaiiable from the following tables. For Synchronous mode:



0	1	0	1
0	0	1	1
5 Bits	6 Bits	7 Bits	8 Bits

You can change the command specification anytime after the INIT statement by using the OUT statement. The format of the OUT statement follows;

```
OUT<a>,<c>
                 where a: is the port address
                       c: is the data to be sent
```

The port address for channel 1 is &HC1

The port address for channel 2 is &HC3

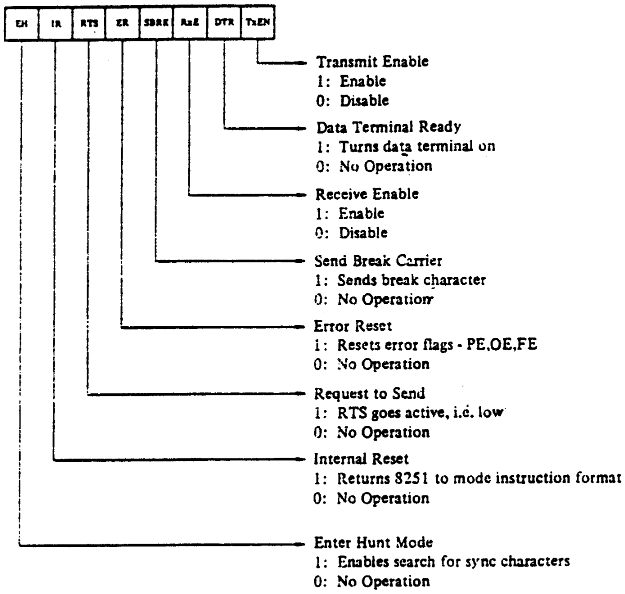
The status of the 8251 USART can be read by using the INP function.

```
A=INP(a)
                  where a: is the port address
```

EXAMPLES

```
OUT  &HC1,&HC5
A=INP(&HC1)
A$=HEX$(INP(&HC1))
```

Command words are available from the following table.



Example:

To set the USART for synchronous mode, no parity, 7 bits per character, baud rate divisor of 16, and 2 stop bits the following statement can be used

```
INIT%1,&HCA,&H15
```

Data can be sent to the RS-232 channel by use of the following statement;

```
PRINT%n,d1,d2,.......
```

where n is the channel number (1 or 2)
d1,d2,... is the data to be sent.

EXAMPLE

```
PRINT%1,A,B,C
PRINT%1,"FOX",A$
```

For inputing data the following statements can be used:

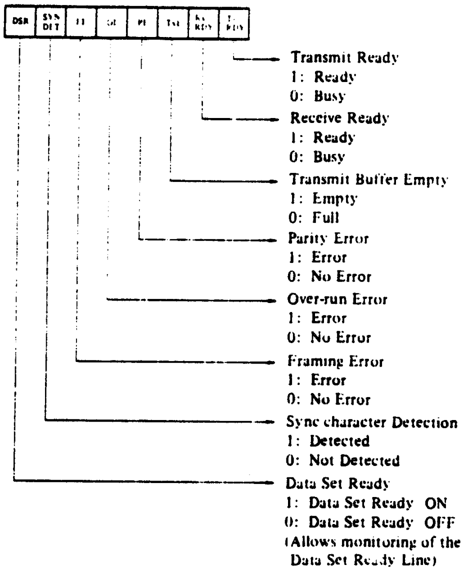
```
INPUT%n,d1,d2........
```

where n is the channel number (1 or 2)
d1,d2,... is the data to be received

EXAMPLES

```
INPUT%1,A,B,C
INPUT%1,A$,B$,C$
```

The actual format of the status word is shown below.



Transmit Ready
1: Ready
0: Busy

Receive Ready
1: Ready
0: Busy

Transmit Buffer Empty
1: Empty
0: Full

Parity Error
1: Error
0: No Error

Over-run Error
1: Error
0: No Error

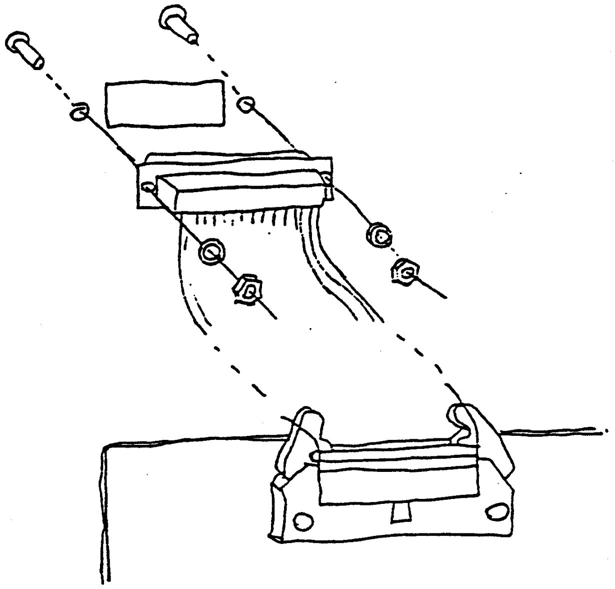
Framing Error
1: Error
0: No Error

Sync character Detection
1: Detected
0: Not Detected

Data Set Ready
1: Data Set Ready ON
0: Data Set Ready OFF
(Allows monitoring of the
Data Set Ready Line)

4) Installation

The PSB-01 and cable should be installed according to the following pictorial. For more installation information refer to the PC-8012 users manual.



A method for inputing character strings is also available. The form of this command is:

```
     INPUT$(m,%n)
```

where m is the number of characters to be input and n is the channel number.

EXAMPLE

```
     A$=INPUT$(5,%1)
```

The following statement can be used to determine how many characters are waiting in the input buffer for the specified port.

EXAMPLE

```
          A=PORT(1)
```

NOTES

If you attempt to use a USART without initializing it first the error message "PORT NOT INITIALIZED" will be issued. The serial input buffer contains space for 127 characters. If you do not read the characters in the time it takes for 127 characters to be received, the 128'th character will cause the following error message to be printed; "COMMUNICATION BUFFER OVERFLOW".

IMPORTANT NOTES:

Pins 2 and 3, and pins 4 and 5 are reversed from normal RS-232 male conventions. Take care when hooking up external equipment.

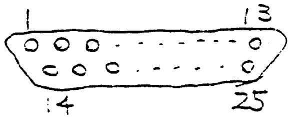
The pin Clear to Send is not pulled up internally, and hence you must do this yourself. If only pins 2,3, and 7 are to be used pin 4 must be connected to pin 5 otherwise the device will be unable to transmit data.

IMPORTANT NOTE:
IF THE INTERRUPTS ARE SET FOR NBASIC, (JUMPERS S4 AND S5), THE BOARD WILL NOT WORK CORRECTLY WHEN BEING USED WITH THE TERM II OR CP/M COMMUNICATIONS PROGRAMS THAT DO NOT REQUIRE INTERRUPTS. FOR THESE PROGRAMS, REMOVE THE JUMPERS ON S4 AND S5. SET THEM ASIDE AND SAVE THEM FOR LATER USE.

5) Connection pins

The following table lists the pin connections for the RS-232 25 pin connector.

pin #	Signal
1	Ground
2	Transmit Data
3	Receive Data
4	Request to Send
5	Clear to Send
6	Data Set Ready
7	Ground
8-19	Not connected
20	Data Terminal Ready
21-25	Not connected



XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX

The file HELP . COM is a public-domain piece of software that will give you help on different subjects. Such as - (whatever help files
re on the disk along with it---MBASIC , CBASIC , MASM , HELP , etc.

Help is used in this way--:

HELP MBASIC	-for help with Mbasic
HELP	-for help with the HELP file
HELP MASM	-for help with Microsofts Macro Assembler

and so on, Help uses the files with the .HLP extension.

The file COPY-15 . COM is also public-domain software, & it's a nice litle program that allows a multiple copy with the wildcard symbo
or a simple copy from one disk to another without the stiff syntax of PIP.

Some samples are:

CPY HELP.COM B: Would copy from the currently logged disk
to drive B: , the file HELP.COM

CPY B:*.* A:Would prompt you for all the dir files on B:
and ask if you wanted them transferred to A:

------ CSE

DOCUMENTATION ON THE MODEM PROGRAM AS OF 01/12/80 V2

----

by Ward Christensen

MODEM.ASM is a multi-function communications program for use with the CP/M operating system. Its primary function is the transfer of files between CP/M systems. The transfer is accomplished via a block mode, with headers and checksums to ensure data validity. Automatic retry, up to 10 times, is attempted for every sector transmitted. The user is given an option to quit, or retry after 10 consecutive errors occur.

Also supported is a terminal mode which allows the computer to function as a terminal to a time sharing system, CBBS, etc, and a similar program, but which echos all received characters, such that two people running MODEM can communicate keyboard-to-keyboard, one running in terminal mode, and one running in echo mode. This is frequently done to "test the line" or to see how high a baud rate can be supported before beginning file transfers.

Planned future enhancements include the ability to place data received (while in terminal mode) into memory, and write it out to a disk file.

As "delivered" the program supports a PMMI S-100 modem at primary address 0C0H, and will run unmodified if this is what you have. Keith Peterson added equates for the Hayes, and a serial modem.

The program is named MODEM2.ASM (or COM) on the disk, to indicate that it is an extension of the MODEM.ASM which was distributed on the CP/M user's group disk 25, and the MODEM which is/was distributed by PMMI itself. It is expected that the user will rename it back to MODEM once any earlier versions of this program are erased.

Using the program P.COM

The program is started by typing in the letter P and then hitting return, make sure your printer is turned on before you run this program. It contains two pages of menu selections for setting-up the NEC PC8023A Printer. The first page is a selection of character sizes with or without enhancements and the second page is to set up the line spacing for 1/6 or 1/8 of an inch.

When the program is done you will be back in CP/M. This program may also be run from the main menu of the wordprocessor called 'SELECT', distributed by NEC, using the 'R' option, used in this way, you will return to SELECT main menu when done.

note 'O' and 'A' are n-o-t defaulted (i.e. send to originate), etc. When using MODEM to send files under a remote console program, use the Q option, but O-M-I-T the O or A - this will leave the modem (if PMMI) in the same mode as before. Otherwise, the ability of the modem to hang up on loss of carrier will be lost, since the MODEM program grabs the line and doesn't leave it able to hang up on loss of carrier. This is because MODEM was written as an ATTENDED program, and only by proper use will work adequately under a remote console program (such as BYE).

Ex: MODEM sq.600 b:foo.asm

Note the baud rate defaults to 300, so if you want to transfer at another rate, don't forget the .600 or whatever.

R: Show characters as received
S: Show characters as sent
V: View the file (suppresses non-error
       status messages)

T: Go to terminal mode after file transfer
E: Go to echo mode after file transfer
D: Disconnect after program execution

EXAMPLES:

send 'test.fil' in originate mode, 'quietly',
disconnect after transfer

COMMAND FORMAT:

```
MODEM option
or MODEM option.baudrate
or MODEM option fn.ft
or MODEM option.baudrate fn.ft
```

"option" consists of a single character PRIMARY OPTION, and 0, 1, or more characters of SECONDARY OPTIONS.

fn.ft is the filename to be received or sent

PRIMARY OPTIONS:

S: To send a file
R: To receive a file
T: Terminal - i.e. the system being communicated with must echo
E: Terminal mode but with echo - this would be used when 2 people using the MODEM program are talking keyboard to keyboard - one uses MODEM t, the other, MODEM E.
D: Disconnect the phone (if your MODEM supports this)
H: Help (prints usage documentation)
X: Prints usage examples

SUB-OPTIONS:

Q: A'Q' may be appended to either 'S' or 'R' to transfer 'quietly' i.e. w/ no console I/O.

This is for several purposes: 1) if you have a slow terminal (such as a tty), you must use the q option; 2) if you are using this program on a "remote" CP/M system, in which the "remote console" is the same as the line for sending a file, then you must use MODEM sq or rq to suppress the console msgs.

S,R,T,C, or (or Q) may be followed by the following if your MODEM is capable of supporting these: (the program currently supports the PMMI). If anyone adds the IDS or HAYES, please send me a list of equates and changes. (address in MODEM.asm file)

O: go to originate mode
A: go to answer mode
D: disconnect (otherwise, keeps the line)

PLINK65.DOC

PLINK is a CP/M transient command which allows the user to communicate with a remote computer that is not running a special program. It allows the user to selectively copy received text into memory and to later write the text onto a CP/M disk file. It also allows text to be transmitted from a disk file to the remote computer a line at a time, suppressing the line feeds and waiting for a trigger character such as a line feed from the remote computer before sending the next line of text. Optionally the trigger character may be a BELL code, an X-ON, or text may be sent without pause including line feeds after the carriage return. Note that none of these options work for sending to a DEC which uses an X-OFF, X-ON protocol when it gets more input< than it can handle.

This program currently supports the following modems or computers via conditional assembly:

1. PMMI modem
2. Any serial I/O board (TUART included)
3. TRS-80 model 1
4. TRS-80 model 2
5. Heath H8 with 8251 UART at port 330q
6. D.C. Hayes 80-103A or Micromodem 100
7. MITS 2SI/O board, ports 10h&12h=console,12h&13h=modem
8. Intel SBC or National BLC multi-bus boards using 8251 USART

Originally written by L.E. Hughes (EDCAM) in July, 1977. Many modifications have been made since this time, as shown in the following summary.

Fixes/updates (in reverse order to minimize reading time):

June 26, 1981. Added message when exiting if last buffer was not saved. Ted Shapin.

June 14, 1981, by Keith Petersen, W8SDZ. Changed port equate to ´equ´ instead of ´set´. ASM doesn´t like ´set´ when later conditionals are based on a label defined that way.

```
MODEM SOQD test.fil
```

receive 'test.fil' in answer mode at 450 baud, and don't disconnect after.

```
MODEM RA.450 test.fil
```

Suppose you have sent 1 or more files already, did not 'D' (disconnect) and want to send or receive another, just:

```
MODEM S test.fil
```

(note this defaults to 300, so s.450 or whatever if that is the rate you were going at)

After transferring the last file, use the D sub-option (MODEM sod.450 name) or re-type the MODEM command with the D primary option

```
MODEM D
```

to disconnect the phone.

When sending to another computer, PLINK waits for a trigger response after sending each line before it will send the next. The following "trigger" equate is set to "lf" (linefeed) by default. An optional trigger char may be passed via fcb1

ie: PLINK B will set trigger to "bell"

The following options are allowed:

1. B = bell 07h
2. X = xon 11h
3. U = upload no trigger check at all and send line feeds

any other ascii character may be passed through fcb1.

June 7, 1981, by Tom Jorgenson (CP-MIG). Changed CP/M origin from being via SETs to referenced to BASE, added TRUE/FALSE rather than numeric values (for readability), changed Q function to W (write) because some systems (notably Micronet) use S/Q to suspend/resume output, changed page 0 references in TRS routines to use BASE equate properly, changed PORT equates to default to TRUE, reinserted Heath equates, and cleaned up code in several places.

June 7, 1981, by Keith Petersen, W8SDZ. Fixed problem with equates which prevented assembly by 'ASM' when TUART option was selected.

June 6, 1981, by Keith Petersen, W8SDZ. Added version number, cleaned up file.

May 12, 1981, by T. Shapin. Added code for 8251 USART on Intel SBC or National BLC multibus board with modified CP/M origin. Added prompt to signon. Added toggle to Y to save or ignore incoming text. Added C abort on file name response.

(for earlier update info, see PLINK65.AQM. Use USQ.COM to unsqueeze)

PLINK currently supports two way transfer of text files between the CP/M disk and the remote computer. The following control codes may be initiated from the console keyboard:

Control-E Exit PLINK to CP/M "warm-boot".

Control-T Transmit ASCII file to remote system, asks for drive (A,B,etc.) and filename.typ.

Control-C Aborts transmition of file to remote system.

Control-Y Switches between saving and ignoring incoming ASCII data in RAM buffer, for later transfer to disk.

Control-Q Writes RAM buffer to disk, and asks for drive and filename.typ.

Del (delete) Backspace when in command mode (e.g. T or Q).

Control-U Aborts current line when in command mode.

(Note: all other control codes are passed to modem output, and may be interpreted by the remote system as various control functions.)

Examples:

UNSPOOL TEST.PRN
will send the file TEST.PRN from the current
default drive to the current LST: device.

UNSPOOL A:TEST.PRN LST
is exactly equivalent to the above, assuming
drive A is the default drive. Note that the device name has no
trailing colon.

UNSPOOL B:ZINGER.HEX PUN
will send the file ZINGER.HEX from the B drive
to the current PUN: device regardless of which drive is
currently the default. Note that the device is "PUN"
not "PUN:".

UNSPOOL . OFF
UNSPOOL *.* OFF
UNSPOOL OFF OFF

all cause an operator prompt: "Do you want to
cancel UNSPOOL?" A single "Y" or "N" (Yes or No) is accepted
from the console as a response. Any other character is assumed
to mean "No".

UNSPOOL OFF
causes the file "OFF.   " to be sent to the
list device.

OPERATION:

Upon loading, the program checks to see that the the BIOS vector table pointed to by the word at location 0001H is valid, i.e. is a table of JMP instructions containing at least 16 entries. If an error is detected, the program will display an error message on the console and attempt to warm-boot CP/M.

If found, the BIOS vector table is copied into the program segment which will remain active during unspooling so that subsequent application programs running concurrently with UNSPOOL will still have access to the BIOS.

This table is modified to trap attempts to warm boot the system or perform direct console input.

The address of the old BIOS vector table, the BDOS

```
UNSPOOL.ASM   81-11-21
for CP/M 2.0 and up.


AUTHOR:                 Gary P. Novosielski
                        Rutherford, NJ
                        (201) 935-4087


INTRODUCTION:

UNSPOOL (Ver 3.0) is a program to send a standard
CP/M file such as a .PRN or .ASM file to the system's list
or punch device, while still allowing other system operations
to take place.  The file is transferred during periods when
console is waiting for input.

SYNTAX:

UNSPOOL (d:) filename.ext (dev)
or
UNSPOOL dummy OFF


square brackets denote optional parameters

Where d:  is an optional drive spec
                        such as A: or B:.  If not
                        entered, the current default
                        drive is assumed.

filename.ext is the name of the CP.M file
                        to be printed/punched

dev  is the symbolic name of the
                        output device to be used.
                        Valid devices are LST and
                        PUN.  Note that the colon
                        (:) usually present is these
                        names is NOT entered.  If not
                        specified, the LST: device is
                        assumed.

OFF  If OFF is specified
                        instead of a device name, the
                        operator will be offered the
                        option of cancelling UNSPOOL
                        if it is already running.

dummy  Because the word OFF is the
                        second parameter, a filespec
                        is still needed.  A dummy name
                        of "." will suffice as a place holder.
```

If a jump to BOOT is attempted, this is also intercepted by the UNSPOOL supervisor segment. The message "Unspooling in progress" is displayed on the console, and no actual boot takes place. Control is returned to the protected copy of the CCP instead. Before returning, a disk reset is performed and the default DMA address is reset to 0080H to simulate a true warm-boot as closely as possible.

If warmboot is attempted using BDOS function 0, the program will prompt the operator with the option to cancel. If the response is "Y", warmboot is performed using BDOS function 0 as requested. Otherwise the request is handled as with normal warmboot, above.

When the input file is completely transferred, or a 1AH end-of-file character is detected, the supervisor becomes inactive, and passes on all previously intercepted requests immediately, without checking console status. When the next warm-boot request is detected, the supervisor removes itself from memory by executing a true warm-boot, and informs the operator with the message "UNSPOOL completed.".

NOTES:

While UNSPOOL makes every effort to restore the values of the DMA address, USER number, IOBYTE, and default disk drive before returning control to the program, a hardware reset may leave these values in an undetermined state if unspooling was actually taking place at the time.

When function 10, Read Console Buffer is used, UNSPOOL will transfer characters only until the first key is pressed. At that time, no characters will be transferred until the input line is completed by pressing a carriage return.

UNSPOOL requires that the List Status function in the BIOS was properly implemented at system installation time. UNSPOOL will not send characters to the LST: device unless it recieves a ready condition from the List Status routine. If the PUN: device is used, no status check is provided for by CP/M, so a not-ready condition on the punch may cause the system to hang up if PUN was specified on the command line. See the CP/M Alteration Guide for a discussion of the BIOS List Status routine.

entry address, and the CCP return address (from the top of the stack on entry) are saved in memory.

The "dev" parameter from the command line is checked. If not valid, an error message is typed and control returns to the CCP.

If the "dev" parameter is the literal OFF, the program executes a BDOS funcion 0 (System Reset) and terminates.

The file named in the command line is opened for input. If not present, the command is echoed to the console followed by a "?" and control returns to CCP.

If the drive is not explicit, the current default drive number is recorded internally in case the default drive is changed while UNSPOOL is active.

The current user number and IOBYTE values are checked and stored internally so that if the user number is changed, UNSPOOL will still be able to read the input file. If the device is changed (using STAT LST:=TTY: for example) unspool will continue to use the physical device in effect at the time the program was initiated. Any application programs will, however, use the new values of the user number and the new physical device assignments.

If no errors were detected, the active program segment which monitors all calls to BDOS is relocated into high memory just below the CCP. This reduces the available user program area by 3 K: 4 pages for the UNSPOOL supervisor segment, and 8 pages for the CCP which is commonly overwritten by user program buffers. The BOOT and BDOS jumps in low memory are modified to protect the CCP and UNSPOOL supervisor segment.

Control is then returned to the console. Normal CP/M operation will then be possible. Characters will be sent from the input file to the output device whenever the console is idle.

Whenever an application program or the CCP requests console input using BDOS functions 1 or 10, or a direct call into the jump table at C(BOOT + 1) + 6, the supervisor segment intercepts these requests and checks to see if the console is idle. If it is, characters are transferred from the input file to the output device until the console becomes ready, i.e. a key is actually pressed. At that time the BDOS function or BIOS call is executed normally, and control returns to the application program.

****

Plinktty.com and Modemtty.com are configured to work with the TTY: port in the PC-8001A, using the 8062A RS-232 link distributed by NEC. They need no reconfiguration to work with this device under CP/M. Please note that this is not the RS-232 dual channel board that NEC will be distributing shortly, but the built-in RS-232 port with the 8062A interface attached. As it stands now, it is initializing that port for an (8) bit word, no parity, 1 stop bit and with the jumpers as they come from the factory, 300 baud.

The programs labelled, Modem206.com and Plink65.com are set-up for the dual-channel RS-232C board using channel 1 and same word length and stop-bits, etc.

****

Although the console is polled frequently during the unspooling of the file, some of the diskette operations may take a second or two to complete, for example when a new extent is opened. Since the console is not polled during this period, high speed typing may cause one or more characters to be lost. This effect will vary depending upon the program being run, the types of input reqests (character or line) it uses, and the relative locations on disk of the spool file and any files in use by the program. As a result, heads-down typing is not reccommended while UNSPOOL is running. Some experience with UNSPOOL will teach the user when caution is required.

INSTALLATION:

The source file is written for assembly with the MAC assembler. The .HEX file produced is LOADed to a .COM file and executed just as any normal program. Relocation is done at execution time as described above.

If the assembly option EXPAND is set to TRUE, tab characters in the input file will be expanded to spaces with assumed tab stops at every eighth print position. This option should be set to FALSE if the printer driver or the printer itself can properly handle the tab character. If the option PHYSBS is set to TRUE, a backspace character will cause the tab expansion algorithm to recognize backspace characters and decrement the column count when a backspace is encountered in the input file. This option should be set to FALSE if backspace characters are ignored by the printer. All other control codes except carriage return are assumed to be non-printing, and are ignored by the algorithm.

If tab expansion is included, the version number in the signon message will be followed by "/T".

Gary Novosielski

RS232C BOARD

INSTALLATION & INFORMATION

CAUTION: Before you install the RS232c board, **TURN OFF ALL THE UNITS OF YOUR COMPUTER SYSTEM.** The RS232c board is static sensitive. Be very careful in handling the board. A static charge could damage the board.

INTRODUCTION

The NEC RS232c board has two complete RS232 ports. This gives you the capability to handle full communications with up to two serial devices (which could be a modem, a serial printer, or a terminal). It can be used with N-BASIC, Racet Computes NECDOS, or CP/M (with the TERM II software communications package). There are also some public-domain communications programs which are available for downloading from the CP/M Bulletin Board systems.

The NEC RS232c board has the communications capability of up to 9600 Baud. It could link-up your computer to other computer systems over telephone lines. It allows you to do program or data-file transfers. It allows you to connect to a serial printer (with the appropriate driving software). With the NEC RS232c, you have the capability of linking up to large communications systems like "The Source", "Micronet" for instant news or stock reports, or through "mail facilities" with other micro-computer users around the world.

DESCRIPTION

Near the center of the board, there are two large black Integrated Circuit (IC) chips. These two chips are the USARTS (Universal Synchronous-Asynchronous Receiver-Transmitter). The USARTS are the "brains" of the RS232c board.

On the top of the board are two black plastic assemblies. These assemblies are where you plug-in the two gray plastic cables. The gray cables will connect the board to the outside rear of the I/O box.

On the left side of the board is a small black block with a set of switches (1-5). These switches are used for address settings. This set of switches is labelled **S1**. More about **S1** later.

To the lower right side of the of **S1** are the **S4** and **S5** jumpers with the little white jumper blocks. These little jumper blocks are used for setting the interrupt addresses for channels 0 and 1. For now, these are used primarily by the N-BASIC commands (INIT%, PRINT%, and INPUT%). Most of the other available software are not running off of the interrupts at this time.

Approximately four inches to the right of **S4** and **S5** is another set of jumpers labelled **S2** and **S3**. These are the jumpers that set the board's communication Baud rate.

INTERRUPT SETTINGS

The jumpers on the board labelled **S4** and **S5** tell the system where the interrupts will be set. They are only being used by the NBASIC commands and must the set as follows:

1. The set of jumpers labelled **S4** sets the interrupts for Channel 0. The white jumper block should be gently pushed onto the 3rd set of the pins. This is the third set of pins from the white letters on the board that says **S4**.

2. The set of jumpers labelled **S5** sets the interrupts for Channel 1. The white jumper-block should be gently pushed onto the 4th set of pins. This is the fourth set of pins from the white letters on the board that says **S5**.

This takes care of the setting for the interrupts. For more information on the setting of these, for a software package that might be released in the future, refer to the sheets of information that come with the RS232c board.

BAUD RATE SETTINGS

The baud rate settings tell the USARTS the rate to send and receive data. **S2** sets the baud rate for Channel 0, and **S3** sets the baud rate for Channel 1. On each set of these jumpers, there are four posibble settings. The settings are dependent on the software commands that are being sent to the USARTS. For instance, if you want to operate at 300 baud (which is common for most modems), you should set the jumpers onto the set of pins that are the farthest away from the white ID letters **S2** and **S3**.

This would set it at 300 baud for the commands under N-BASIC or Racets NECDOS. This setting could also be used with the TERM II software package. If you want to use other baud rates for specialized applications, a good understanding of Z80 assembly programming is needed. For further information, consult the application notes that came with the RS232c board.

SETTING THE ADDRESS SWITCH

The jumpers are now set for use. Now, we have to tell the system where the board is located. This is done by·the small set of switches on the left side of the board marked 1 thru 5 (S1).
The switches set the port address for use with NBASIC and TERM II.
These should be set to 1,2,3 and 4 **UP** or to the **ON** position.
And 5 should be set **DOWN** or to the **OFF** position. This completes the board settings.

INSERTING THE BOARD

First, attach the gray plastic cables that come with the board to the black plastic sockets marked **CH0** and **CH1**. Please note that these have a key, and may only be inserted one way into the plug socket. The triangle on the plug socket must line up with the triangle on the black plastic sockets.

There are four phillips-head screws holding the cover on the PC-8012A unit on each side. Remove these and put them in a safe place. Once the screws are removed, lift the cover carefully and set it aside. Inside the I/O box, you will see 6 empty slots. The RS232c board may be placed into any of the slots, but for the sake of convention, insert the board gently into the slot marked **CN7**, which is the last slot towards the back of the I/O box. Do not force the board into the card-edge connector, but apply firm pressure.

The board should now be inserted into the last slot on the expansion slot row. All that is left to do is to line-up the other end of the gray ribbon cables to the small rectangular slots on the rear of the I/O box. Use the screws that came with the board to anchor the cables firmly to the inside back-face of the I/O box. At this point, go back over the instructions on the jumper settings and switch settings to make sure that you have not inadvertently changed them during the insertion of the RS232c board. If all is well, the cover may now be replaced and screwed securely back on.

This ends the installation aspect.

INTERFACE

Replace all the plugs and make sure that all the components are connected properly. You will need to plug-in the cable that connects the modem to the plugs at the back of the I/O box. To do this, refer to the instructions that accompany the modem. If you sense that something is not working right, turn off all your units immediately and check to see what the problem is.

REMEMBER: The channels are numbered 0 and 1. The TERM II package refers to channel 1 and 2. So, when using this package, keep in mind that your system's channel 0 is channel 1 for TERM, while your system's channel 1 is channel 2 for TERM.

NEC HOME ELECTRONICS USA, PERSONAL COMPUTER DIVISION

This copy belongs to C.R. Myers

This Document describes the seting up and the operation of the PSB-01 dual serial port.

1) Introduction

The PSB-01 is a circuit card that plugs into the PC-8012 expansion interface. This card contains two RS-232 serial channels. The two channels are implemented with uPD 8251 USARTS. The two channels have independently controllable Baud rates. The capability for interrupting the processor when receive characters are available is also provided.

When the I/O addresses and interrupt pins are properly connected the board will function with NBASIC. Statements such as PRINT% and INPUT% can then be used.

2) General Specifications

Two complete RS-232 channels

USARTS are uPD-8251

Baud rates are independently selectable from the following;
300,600,1200,2400,4800,9600

I/O addresses are settable anywhere within the range of
80H to 0FFH

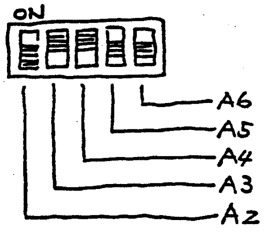
Each channel can be set up to provide an interrupt on Rx RDY
Interrupts are settable within INT0 to INT7

The board is functional within a temperature range of 40 to 90 degress Farenheit. The relative humitidy should be in the range of 20% to 80%

The board size is approximetly 220mm by 95mm

3) Setting the board up for use

The I/O address at which the board resides is adjustable by means of a 5 position dip switch. The switch is located at the upper left hand corner of the board. The address bits which correspond to the switch are listed below.



ON represents a "0"

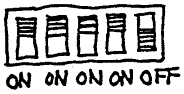
OFF represents a "1"

When the switch has been programmed the USARTS will appear at the following:

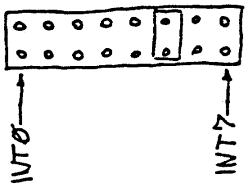
1	A6	A5	A4	A3	A2	X	X

0 0	Channel 1	DATA
0 1	Channel 1	STATUS
1 0	Channel 2	DATA
1 1	Channel 2	STATUS

If the I/O addresses are set at the addresses C0H to C3H then the board is usable with NBASIC. The switch settings corresponding to this setting are shown below.



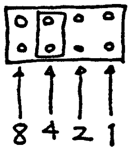
Rx RDY interrupt capability is provided by Jumper sockets S4 and S5. S4 controlls channel 1 while S5 controlls channel 2. The jumper socket is illustrated below.



To maintain compatability with NBASIC the interrupts must also be set according to the following table.

Interrupt Table Addresses		Content	Channel
16K System	32K System		
C000 1	8000 1	Used by N-BASIC	IEEE-488
2 3	2 3	"	IEEE-488
4 5	4 5	Low-order byte High-order byte	Real-Time Clock
6 7	6 7		
8 9	8 9	Used by N-BASIC	RS-232C Ch.1
A B	A B	"	RS-232C Ch.2
C D	C D	Low-order byte High-order byte	INT9
E F	E F	Low-order byte High-order byte	INT8
10 11	10 11	Low-order byte High-order byte	INT7
12 13	12 13	Low-order byte High-order byte	INT6
14 15	14 15	Low-order byte High-order byte	INT5
16 17	16 17	Low-order byte High-order byte	INT4
18 19	18 19	Low-order byte High-order byte	INT3
1A 1B	1A 1B	Low-order byte High-order byte	INT2
1C 1D	1C 1D	Low-order byte High-order byte	INT1
1E C01F	1E S01F	Low-order byte High-order byte	INT0

The baud rates are independently controllable by jumpers at sockets S3 (for channel 1) and S2 (for channel 2). The socket is illustrated below.



The baud rates which result from setting the jumpers are illustrated below.

	Divisor mode	
Jumper Pin	64	16
8	300	1200
4	600	2400
2	1200	4800
1	2400	9600

4) Usage with NBASIC

When all of the following conditions are met the board can be used with NBASIC.

I/O address C0H-C3H

	SW0(A2) ON	SW3(A5) ON
	SW1(A3) ON	SW4(A6) OFF
	SW2(A4) ON	
Interrupt address		
	S5 INT3	
	S4 INT2	

The PC-8012 must also be jumpered as follows;

CN2	1-2
CN1	1-2
CN5	1-2

The board is now ready for use with NBASIC.

The first thing that must be done is to enable the interrupt controller. This is done with the following statement:

```
OUT  &HE4,&HFF
```

The channels must now be programed for use by the use of the following statement:

```
INIT%n,m,c
```

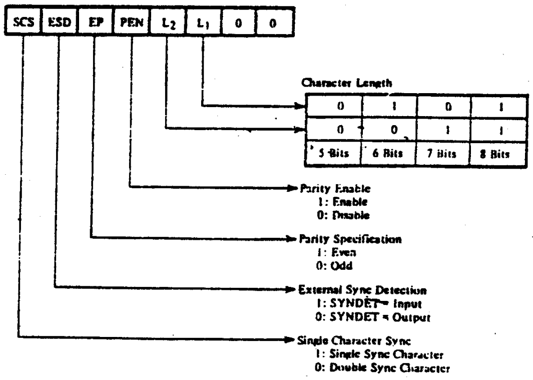
where n, m, and c, represent the following:

n: Channel number (either 1 or 2)

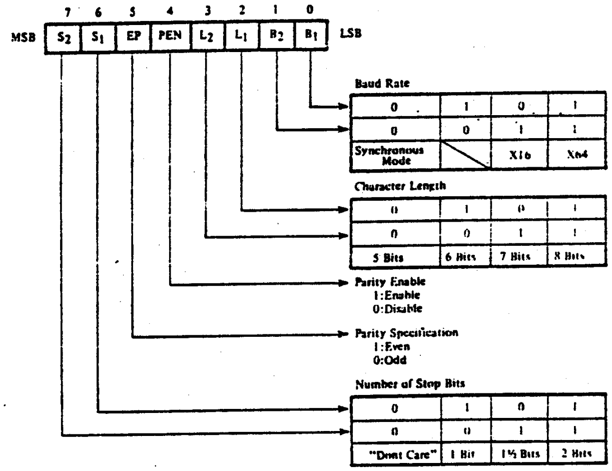
m: Mode word

c: Command word

Mode words are available from the following tables. For Synchronous mode:



For Asynchronous mode:



Baud Rate

0	1	0	1
0	0	1	1
Synchronous Mode		X16	X64

Character Length

0	1	0	1
0	0	1	1
5 Bits	6 Bits	7 Bits	8 Bits

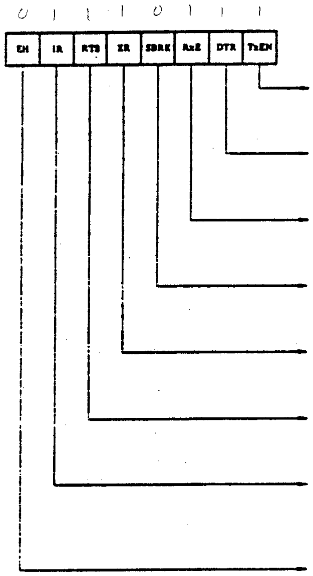
Parity Enable
1:Enable
0:Disable

Parity Specification
1:Even
0:Odd

Number of Stop Bits

0	1	0	1
0	0	1	1
"Dont Care"	1 Bit	1½ Bits	2 Bits

Command words are available from the following table.



Transmit Enable
1: Enable
0: Disable

Data Terminal Ready
1: Turns data terminal on
0: No Operation

Receive Enable
1: Enable
0: Disable

Send Break Carrier
1: Sends break character
0: No Operation

Error Reset
1: Resets error flags - PE,OE,FE
0: No Operation

Request to Send
1: RTS goes active, i.e. low
0: No Operation

Internal Reset
1: Returns 8251 to mode instruction format
0: No Operation

Enter Hunt Mode
1: Enables search for sync characters
0: No Operation

Example:

To set the USART for synchronous mode, no parity, 7 bits per character, baud rate divisor of 16, and 2 stop bits the following statement can be used

```
INIT%1,&HCA,&H15
```

You can change the command specification anytime after the INIT statement by using the OUT statement. The format of the OUT statement follows;

```
OUT<a>,<c>
                where a: is the port address
                      c: is the data to be sent
```

The port address for channel 1 is &HC1

The port address for channel 2 is &HC3

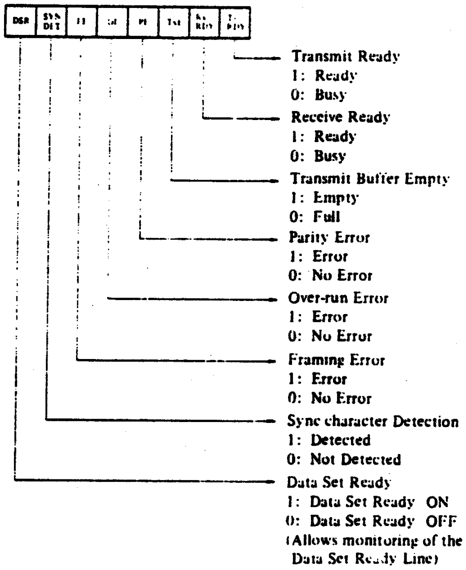
The status of the 8251 USART can be read by using the INP function.

```
A=INP(a)
                where a: is the port address
```

EXAMPLES

```
OUT  &HC1,&HC5
A=INP(&HC1)
A$=HEX$(INP(&HC1))
```

The actual format of the status word is shown below.



Data can be sent to the RS-232 channel by use of the following statement;

```
PRINT%n,d1,d2,.......
```

where n is the channel number (1 or 2)
d1,d2,... is the data to be sent.

EXAMPLE

```
PRINT%1,A,B,C
PRINT%1,"FOX",A$
```

For inputing data the following statements can be used:

```
INPUT%n,d1,d2........
```

where n is the channel number (1 or 2)
d1,d2,... is the data to be received

EXAMPLES

```
INPUT%1,A,B,C
INPUT%1,A$,B$,C$
```

A method for inputing character strings is also available. The form of this command is:

```
     INPUT$(m,%n)
```

where m is the number of characters to be input and n is the channel number.

EXAMPLE

```
     A$=INPUT$(5,%1)
```

The following statement can be used to determine how many characters are waiting in the input buffer for the specified port.

EXAMPLE

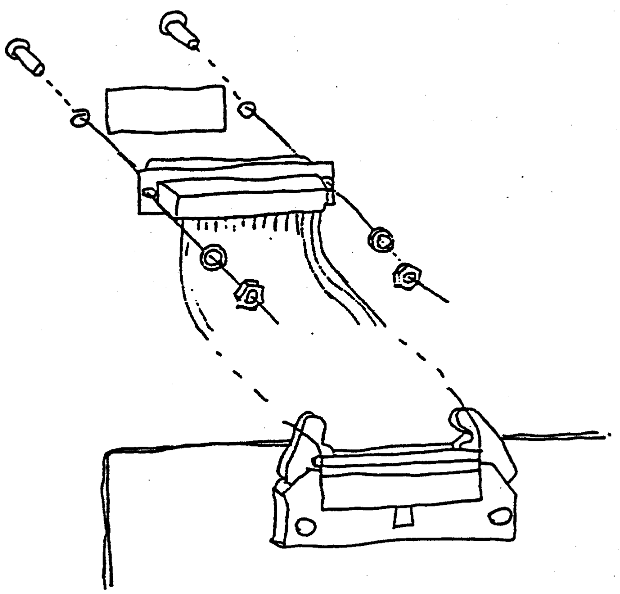
```
          A=PORT(1)
```

NOTES

If you attempt to use a USART without initializing it first the error message "PORT NOT INITIALIZED" will be issued. The serial input buffer contains space for 127 characters. If you do not read the characters in the time it takes for 127 characters to be received, the 128'th character will cause the following error message to be printed; "COMMUNICATION BUFFER OVERFLOW".

4) Installation

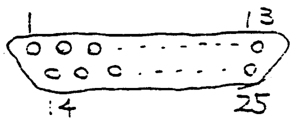
The PSB-01 and cable should be installed according to the following pictorial. For more installation information refer to the PC-8012 users manual.



5) Connection pins

The following table lists the pin connections for the RS-232 25 pin connector.

pin #	Signal
1	Ground
2	Transmit Data
3	Receive Data
4	Request to Send
5	Clear to Send
6	Data Set Ready
7	Ground
8-19	Not connected
20	Data Terminal Ready
21-25	Not connected



IMPORTANT NOTES:

Pins 2 and 3, and pins 4 and 5 are reversed from normal RS-232 male conventions. Take care when hooking up external equipment.

The pin Clear to Send is not pulled up internally, and hence you must do this yourself. If only pins 2,3, and 7 are to be used pin 4 must be connected to pin 5 otherwise the device will be unable to transmit data.

IMPORTANT NOTE:
IF THE INTERRUPTS ARE SET FOR NBASIC, (JUMPERS S4 AND S5), THE BOARD WILL NOT WORK CORRECTLY WHEN BEING USED WITH THE TERM II OR CP/M COMMUNICATIONS PROGRAMS THAT DO NOT REQUIRE INTERRUPTS. FOR THESE PROGRAMS, REMOVE THE JUMPERS ON S4 AND S5. SET THEM ASIDE AND SAVE THEM FOR LATER USE.