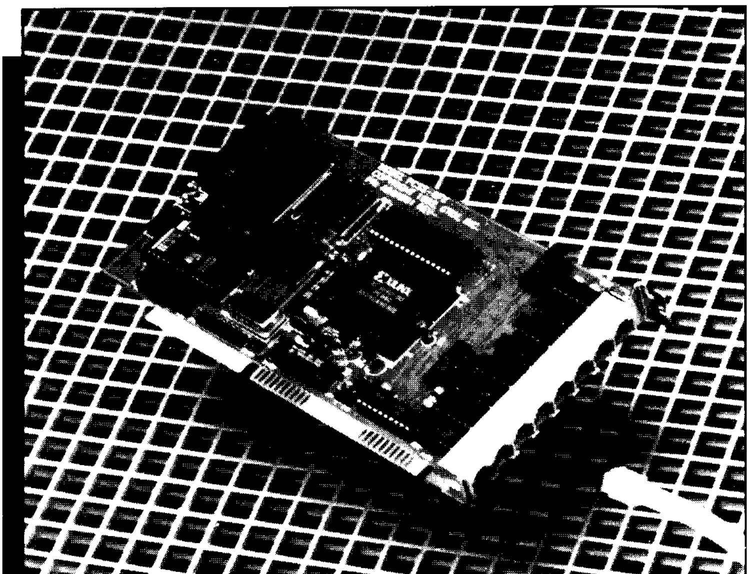
PCSS–8FA

and

PCSS–8FX

INTELLIGENT SERIAL COPROCESSORS



User Manual

GTEK®, INC.

$10.00

Operating Manual for the GTEK Model PCSS–8FA and Model PCSS–8FX

First Printing, July 13, 1993

Table of Contents

1—General Overview

The GTEK PCSS–8FA/FX Super Serial Cards are very powerful serial enhancement products for the IBM Personal Computer (PC, XT, AT, and PS2 Models 30 and 25) family and all compatibles. The PCSS–8FA/FX products provide 8 independent RS–232 asynchronous serial communication channels per board. Baud rates of up to 115.2k baud can be selected. Automatic handshaking of either DTR/CTS or XON/XOFF type is available in both directons. Both transmit and received data is buffered on the board and all of this is done with no overhead on the host computer's processor.

Each board occupies only 4 I/O addresses. Using all of page one and page two of the computers I/O map, it is possible to support 128 cards or 1024 channels of communication. The default address range (no jumpers) is 2E0–2E3h. Jumpering JB1 as the chart in Chapter 3, Step 1 selects the desired base address.

GTEK's operating system for its intelligent serial cards implements DYNAMEMORY™. Dynamemory allocates & deallocates memory to queues that need it or are finished with it, in real time. This gives you a virtual memory space that can be as much as 16 times the size of actual memory. A standard PCSS–8FA/FX with 32K total of Dynamemory can approximate the performance of an operating system with fixed length queues totaling 512K bytes.

The PCSS–8FA/FX GTEK Intelligent Super Serial Cards are supplied with a utility program called RES8F to make it compatible with programs that use BIOS software interrupt 14H. This BIOS Interface utility is a "Terminate and Stay Resident" program that enhances the BIOS INT14 function call. RES8F limits the number of cards that can be installed in one computer to sixteen (or 128 channels), if you can call that a limitation. The PCSS–8FA/FX can take the place of a PCSS–8 or 8X physically, but must use the new BIOS Interface program RES8F in place of RES14. RES14 may also run in conjunction with RES8F. So any of our serial cards can be mixed and matched in the same computer.

SSI is a program supplied on the utilities disk to remap COM1:—COM4: to any of the channels on the intelligent card. Compatability is so complete you can even use the DOS "Mode" command to set up an intelligent channel that has been remapped to one of the COMx: ports.

GTEK Intelligent Super Serial Cards can also be operated in a command driven mode that bypasses INT14. In this mode you can write programs to do many different types of serial I/O.

Since you are not operating with the Uart directly, the I/O bus speed is of no immediate concern. The GTEK Intelligent Super Serial Cards can operate with I/O bus speeds of over 60MHz. They can be used in a number of different applications, some of which do not even require it to be in a computer! By supplying power to the board and installing the proper co–processor firmware, the board can be operated stand alone in a data multiplexing mode or as a spooler with multiple inputs or outputs. One such application could be as a print spooler, taking input from a number of computers and outputting to a single device such as a laser printer.

The board can also be operated as a data multiplexer or spooler within a computer, with the application program on the computer controlling the data flow. By using optional I/O addresses, application programs such as bulletin boards with as many as 1024 channels (depending on the number of available slots) can be written. Used with data multiplexing on external GTEK Intelligent Super Serial boards, any number of channels could be available. You could have literally hundreds of addressable RS–232 channels for many applications.

Dynamemory is a trademark of GTEK, Inc.

—NOTES—

2—Compatibility

GTEK's Intelligent Super Serial Cards are completely compatible with all existing IBM PC/XT/AT models when the application uses the BIOS 14H software interrupt and the GTEK RES8F program has been installed. It is also compatible with Compaq and other IBM compatibles.

The solution for enhancing new and many existing applications is to use GTEK's RES8F program. SSI.com can be used to remap the computer's COM1:–COM4: ports to any of the intelligent board's channels. The selected intelligent channel can then be used as a standard COM port. Programs that are not well behaved (ie. write directly to the computer's uarts instead of using INT14) will require some modifications to work with the GTEK Intelligent Super Serial Cards.

GTEK's Intelligent Super Serial Cards may also be addressed directly by writing and reading registers normally located at 2E0–2E3h. You can write your own application program to use them directly, rather than through RES8F if you have a special need. Multiple boards can be used by selecting another I/O space (normally 2E4–2E7h), or you can request I/O space at any normally unused I/O address space in your computer.

Unlike the GTEK model PCSS–8, the GTEK Intelligent Super Serial Cards have 8 built–in RJ–12 type jacks. This allows the connection of 8 telephone type modular plugs directly to the card in just one slot space, rather than through an external bracket with DB connectors as on the PCSS–8. Available signals are CTS, DTR, TXD, RXD, CD and Signal/Frame GND.

—NOTES—

3—Installation

The PCSS–8FA/FX card can be inserted into any available expansion slot in a PC, XT or AT type computer. The PCSS–8FA/FX takes a half size slot.

CAUTION

Be sure that the power is off and the power cord is removed from the computer before installing or removing any equipment.

STEP 1—PCSS–8FA/FX I/O and IRQ Configuration

Address Selections

Jumper block JB1 selects one of seven base addresses for the PCSS–8FA. In the following chart "on" means a jumper installed between the two pins under the number indicated. "Off" means no jumper installed. All three jumpers installed is not allowed and will not work.

JB1 (JP5 on an PCSS-8FX)

BASE ADDRESS	ONE	TWO	THREE
2E0h	off	off	off
2E4h	on	off	off
210h	off	on	off
214h	on	on	off
218h	off	off	on
21Ch	on	off	on
220h	off	on	on
illegal	on	on	onbase address

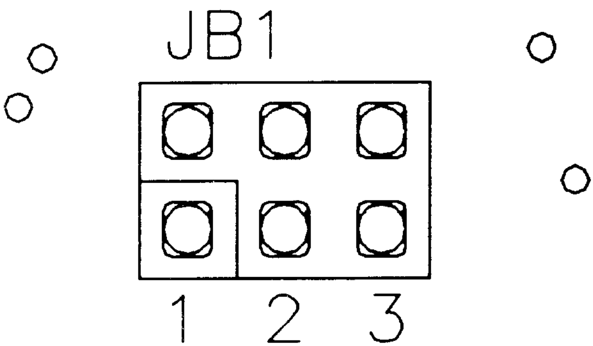


Figure 3.1—Base Address Selections

Simply place a shorting jumper as indicated. NOTE: You need only 2 jumpers to cover all possible selections, and you need NO jumpers to use the default selections.

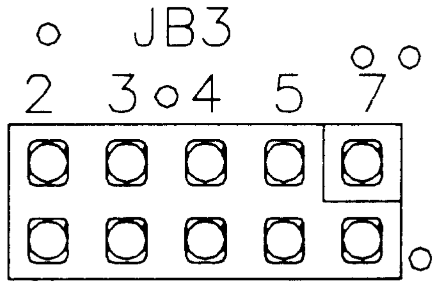


Figure 3.2 Irq Selections for PCSS–8FA

IRQ Selections

It is not necessary to use any interrupts with the PCSS-8F or 8FX, however, if special interrupting functions are desired, irq selections are made with a SINGLE shorting jumper on jumper block JB3 on a PCSS-8F, or JP6 on a PCSS-8FX. Interrupt numbers 2, 3, 4, 5, and 7 (10-15 on 8FX also) are available. NOTE: It is necessary to give Command 1Eh followed with data 0Dh and then data 02h

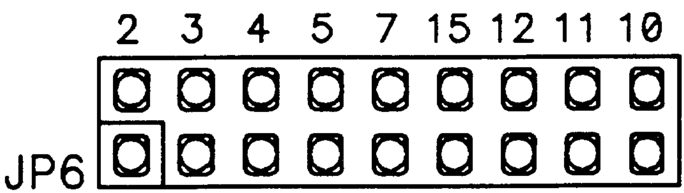


Figure 3.3 Irq Selections for PCSS–8FX

prior to using hardware interrupts. After this is done, an event detected by command 30h will cause a hardware interrupt. However, it is not necessary to use hardware interrupts to use the event detection provided for by commands 30h, 2Fh and 3Bh.

STEP 2—Select Placement

Select an open expansion slot. Locate the metal bracket that covers the cut–out in the back panel for the slot you've selected. Remove the bracket–retaining screw using a small screw or nut driver. Remove the bracket. Save the screw and store the bracket.

STEP 3—Insert Board

Keeping the top edge of the board level, lower the card until its edge connector is resting on the expansion slot receptacle. The 8 jack telephone connector will fit into the rear panel slot. Using evenly distributed pressure, press the card straight down until it seats in the expansion slot. Install the bracket–retaining screw (that was removed at the beginning of STEP 2) to secure the bracket to the rear of the computer chassis. You should be able to plug the proper connector into any of the open jacks. Channel 0 is at the top and channel 7 is at the bottom.

STEP 4—Reinstall Cover

Replace the system unit cover by carefully sliding the cover over the chassis from the front until it stops securely against the rear panel. Reinstall the screws you removed earlier to secure the system cover.

STEP 5—Re–Install Connectors

Replace the power cord to the system unit and be sure that the keyboard and the monitor connectors are plugged in.

STEP 6—Initialize

The operating system for the PCSS–8FA/FX initializes all eight channels with the following configuration: 9600 Baud, no parity, 8 data bits, 1 stop bit, DTR low (negated), and no automatic handshaking enabled. If you wish to change any of the settings you can do so by using either RES8F as outlined in Appendix B or by commanding the board directly as outlined in Appendix A. Either method results in the same wide flexibility of configurations for any or all of the channels available to you through the PCSS–8FA/FX.

RES8F.COM is a program to enhance the BIOS of your computer so that it can communicate with the PCSS–8FA/FX in a DOS compatible manner. The syntax to install RES8F is:

C>**RES8F aaa bbb ... ppp<enter>**

The operands (aaa bbb etc. to ppp) are simply the base addresses of all the boards that you wish to install. RES8F limits you to 16 boards in a computer. The channel numbers are assigned in the order that the boards appear on the command line. The first board gets channels 0–7, and the second board gets channels 8–16 etc. Please see Appendix B for a more detailed look at the operation of RES8F.

SSI.COM is a program to allow you to attach any of the normal COM ports (ie. COM1:–COM3:) to any of the intelligent channels defined previously with RES8F . The proper syntax is:

C>**SSI m nnn<enter>**

The **m** is 1, 2, 3 or 4 representing COM1:, COM2:, COM3: or COM4:. The **nnn** represents the intelligent channel that you wish to attach. After this program is executed, any I/O that is directed to the attached COM port is channeled through the appropriate intelligent channel.

—Notes—

4—Operating Instructions

Initialization:

The programs necessary to set up the PCSS–8FA/FX are outlined in the previous chapter. They must be run in the order specified. Set up for the PCSS-8FA/FX can be done from a batch file. You will probably want to include this in your autoexec.bat file so that the board comes up ready to run whenever you turn your computer on.

Methods of Operation:

The PCSS–8FA/FX will operate in the background queuing any received or transmitted data. Using either the command driven interface or through RES8F the board can be polled for data. The co–processor in either case can return a global poll of all eight channels. The byte of information returned will have a bit set for each channel that has received data. Bit 0 set indicates channel 0 has received data, bit 1 set indicates channel 1 has received data up to bit 7 set indicates channel 7 has received data. See Appendix A command 25h or Appendix B function call 19h for more information about global polling.

Because the PCSS–8FA/FX stores any received data in its on–board buffer, the host computer can poll the board for received data at your leisure. This reduces the load on the computer and allows you to execute task swaps between serial communications and some other task. This method of operation could be used with programs written in high level languages like “C”, “PASCAL”, “BASIC”, etc. For example, in Microsoft QuickBasic:

```
ON TIMER(1) GOSUB POLL    'subroutine poll checks
'the PCSS-8F for received data and processes it
'as necessary
POLL:
ax=&H1900: CALL RES(ax,dx)
al=ax-(ax\256)*256              'get AL bit pattern
IF al=0 THEN RETURN             '=0 if no channel with data
FOR I=0 TO 7
          ch=2^I
          alold=al
          IF (ch AND al)>0 THEN         'we found one
                    'put characters into indexed variable ch$(I)
          END IF
          al=alold
NEXT I
RETURN
```

Remember the time between polls of the board must not be so large that the received data could overflow the buffer on the board. This maximum time is about 1 second for the 32K PCSS–8FA/FX running at 9600 baud.

—NOTES—

5—PCSS–8FA/FX Specifications

Board dimensions:
8FA—4.2 x 6.5" or 106.7 x 165.7mm (½ size)
8FX—4.2 x 7.4" or 106.7 x 188mm

Weight: 8FA—5.4oz, 154g
8FX—6.7oz, 190g

Power Requirements:
+5 Volts: 8FA/FX is .7 amp typical
on all models:
+12 volt @ 200ma typical
—12 volt @ 20ma typical

Operating Environment:
45–95 deg F., 7–35 deg C.
5% to 95% non–condensing relative humidity.

RJ–12 Telco Connector Pin–out (6 pins). Pin 1 is the top pin of each jack, or the pin closest to the metal mounting bracket at the top of the card:

PIN	abbrv.	Name	Signal Direction
1	CTS	Clear To Send	(input)
2	TXD	Transmitted Data	(output)
3	CD	Carrier Detect	(input)
4	SFG	Signal and Frame Ground	
5	RXD	Received Data	(input)
6	DTR	Data Term. Ready	(output)

In the tables below, hook only the pins necessary to the operation of your device. Signal names might be changed to accomodate the type of device you are attaching to. This is controlled by your interface software and/or the setup that was used for RES8F .

Typical hookup with a male DB–25 connector (to a DCE device such as a modem)

Telco	DB-25P
1–CTS	5–CTS
2–TXD	2–TXD
3–CD	8–CD
4–SFG	7–SG
5–RXD	3–RXD
6–DTR	20–DTR
4–SFG	1–Frame Ground

Pin 1 is the top pin of each jack.

Typical hookup with a female DB–25 connector (to a DTE device such as a terminal):

```
1–CTS           4–RTS
2–TXD           3–RXD
3-CD           20–DTR
4–SFG           7–SG
5–RXD           2–TXD
6–DTR           5–CTS
4–SFG           1–Frame ground
```

Hookup for the external power connections, when it is used as a stand alone are as follows:

```
1–Buffered Reset
2–Ground
3–Vcc (+5v)
4–Ground
5—-12V
6–+12V
```

—Notes—

—Notes—

6—Software License Agreement

"This software is a proprietary product of GTEK, INC. It is protected by copyright and trade secret laws. It is licensed (not sold) for use on a single microcomputer system, and is licensed only on the condition that you agree to this License Agreement." GTEK, INC. provides this program and licenses its use worldwide. You assume responsibility for the use of this software to achieve your intended results, and for the installation, use and results obtained from the software.

6.1—License

The Licensee may:

a. use the program on a single machine;

b. copy the program into any machine readable or printed form for backup or modification purposes in support of your use of the program on the single machine;

c. modify the program and/or merge it into another program for your use on the single machine (Any portion of this program merged into another program will continue to be subject to the terms and conditions of this Agreement.): and,

d. transfer the program and license to another party if the other party agrees to accept the terms and conditions of this Agreement. If you transfer the program, you must at the same time either transfer all copies whether in printed or machine readable form to the same party or destroy any copies not transferred; this includes all modifications and portions of the program contained or merged into other programs.

You must reproduce and include the copyright notice on any copy, modification or portion merged into another program.

You May Not Use, Copy, Modify, Or Transfer The Program, Or Any Copy, Modification Or Merged Portion, In Whole Or In Part, Except As Expressly Provided For In This License. If You Transfer Possession Of Any Copy, Modification Or Merged Portion Of The Program To Another Party, Your License Is Automatically Terminated.

6.2—Term

The license is effective until terminated. You may terminate it at any other time by destroying the program together with all copies, modifications and merged portions in any form. It will also terminate upon conditions set forth elsewhere in this Agreement or if you fail to comply with any term or condition of this Agreement. You agree upon such termination to destroy the program with all copies, modifications and merged portions in any form.

7—Limited Warranty

7.1—Hardware

GTEK, Inc., warrants to the original purchaser of this GTEK, Inc., product that it is to be in good working order for a period of one year from the date of purchase from GTEK, Inc., or an authorized GTEK, Inc., dealer. Should this product, in GTEK, Inc.'s opinion, malfunction during the warranty period, GTEK will, at its option, repair or replace it at no charge, provided that the product has not been subjected to misuse, abuse, or non–GTEK authorized alterations, modifications, and / or repairs.

Products requiring Limited Warranty service during the warranty period should be delivered to GTEK with proof of purchase. If the delivery is by mail, you agree to insure the product or assume the risk of loss or damage in transit. You also agree to prepay the shipping charges to GTEK.

All Express And Implied Warranties For This Product Including, But Not Limited To, The Warranties Of Merchantability And Fitness For A Particular Purpose, Are Limited In Duration To The Above 1 Year Period. Some states do not allow limitations on how long an implied warranty lasts, so the above limitations may not apply to you.

Under No Circumstances Will GTEK, Inc. Be Liable In Any Way To The User For Damages, Including Any Lost Profits, Lost Savings, Or Other Incidental Or Consequential Damages Arising Out Of The Use Of, Or Inability To Use, Such Product. Some states do not allow the exclusion or limitation of incidental or consequential damages for consumer products, so the above limitations or exclusion may not apply to you.

This Warranty Gives You Specific Legal Rights, And You May Also Have Other Rights Which May Vary From State To State.

The limited warranty applies to hardware products only.

7.2—Software

This Product Is Not A Consumer Product Within The Meaning Of The Uniform Commercial Code And Applicable State Law. The Program Is Provided "AS IS" Without Warranty Of Any Kind, Either Expressed Or Implied, Including, But Not Limited To The Implied Warranties Of Merchantability And Fitness For A Particular Purpose. The Entire Risk As To The Quality And Performance Of The Program Is With You. Should The Program Prove Defective, You (Not GTEK, Inc.) Assume The Entire Cost Of All Necessary Servicing, Repair Or Correction. Some States Do Not Allow The Exclusion Of Implied Warranties, So The Above Exclusion May Not Apply To You. This Warranty Gives You Specific Legal Rights And You May Also Have Other Rights Which Vary From State To State.

GTEK, Inc. does not warrant that the functions contained in the program will meet your requirements or that the operation of the program will be uninterrupted or error free. However, GTEK, Inc. warrants the diskette(s) on which the program is furnished, to be free from defects in materials and workmanship under normal use for a period of ninety (90) days from date of delivery to you as evidenced by a copy of your receipt.

Licensee herein acknowledges that the software licensed hereunder is of the class which inherently cannot be tested against all contingencies by Licensor. Licensee acknowledges Licensee's obligation to test all programs produced by the licensed software to determine suitability and correctness prior to use.

7.3—Limitations Of Remedies

GTEK, Inc.'s entire liability and your exclusive remedy shall be:

1. the replacement of any diskette(s) not meeting GTEK's "Limited Warranty" and which is returned to GTEK, Inc. with a copy of your receipt, or
2. if GTEK, Inc. or the dealer is unable to deliver a replacement diskette(s) which is free of defects in materials or workmanship, you may terminate this Agreement by returning the program and your money will be refunded.

In No Event Will GTEK, Inc. Be Liable To You For Any Damages, Including Any Lost Profits, Lost Savings Or Other Incidental Or Consequential Damages Arising Out Of The Use Or Inability To Use Such Program Even If GTEK, Inc. Has Been Advised Of The Possibility Of Such Damages, Or For Any Claim By Any Other Party.

Some States Do Not Allow The Limitation Or Exclusion Of Liability For Incidental Or Consequential Damages So The Above Limitation Or Exclusion May Not Apply To You.

7.4—General

You may not substitute, assign or transfer the license or the program except as expressly provided in this Agreement. Any attempt otherwise to sublicense, assign or transfer any of the rights, duties or obligations hereunder is void.

This Agreement will be governed by the laws of the State of Mississippi.

Should you have any questions concerning this Agreement, you may contact GTEK, Inc. by writing to:

GTEK, Inc. Sales and Service
P. O. Box 2310
Bay St. Louis, MS 39521-2310

8—Service

For **warranty or non warranty service**, contact GTEK, INC. at (601) 467-8048 to obtain an RMA (Return of Material Authorization number). We will need the serial number and date of purchase. Send the board, **freight prepaid** to:

GTEK, INC.
RMA #####
399 Highway 90
Bay St. Louis, MS. 39520

Be sure to include the RMA number on and in the package so we will know what to do with it. Out of warranty service charges are determined on an hourly labor plus materials basis. GTEK will pay the return freight via UPS Surface for "in warranty" service within the continental United States.

—Notes—

Appendix—A

Table of Contents

PCSS–8FA and PCSS–8FX Operating System

Before describing the rich set of communications commands available to the programmer, a few notes are in order. Take a moment to read them.

1—In the following command descriptions, "CSC" means Currently Selected Channel.

2—Some of the commands have operands. Command operands are written to the data register. If a command returns information, then that information is read from the data register.

3—Communicating with the PCSS–8FA/FX is accomplished through two of the board's four contiguous I/O addresses in the computer's I/O map. The default base address is 2E0H. See Chapter 3 for other available base addresses.

3.a—The register at the boards' base address is known as the DATA REGISTER.

3.b—The register at the base address+1 is known as the COMMAND/STATUS Register. When written to, the base+1 address is the command register. On reads, the base address + 1 is a status register as follows:

Bit 7 Command Register Ready. When set (this is normally the case), the board is ready to accept a command. Note: ON PCSS–8FA only, status bits 0 through 6 must be qualified by bit 7. That is, they are only valid when bit 7 is high. On the PCSS–8FX, however, bit 7 need only be high prior to issuing a command. Data (and command operands if desired) may always be qualified by the status bits TxRdy, TxEmp, RxRdy. That is to say, on the PCSS-8FX, that while a command is in progress, not only will bit 7 be low, but TxRdy=0, RxRdy=0, Txemp=0 so that you would not transfer any data even if you were just monitoring these 8250 compatible status bits.

Bit 6 TxEmpty. Set when the transmit buffer is empty.

Bit 5 TXready. Set when the transmit buffer is ready, clear when the transmit buffer is full.

Bit 4 Break detected.

Bit 3 Framing error on the received byte.

Bit 2 Parity error on the received byte.

Bit 1 Overrun error on the received byte.

Bit 0 RXready. Set when there is received data. The data is already available on the data port at the board base address. When you read the data port, the status port is updated, and new data is applied to the data port if more is available.

3.d—On the PCSS–8FA/FX boards, the base address + 2 (2E2H) is reserved. Do not use it.

3.e—The base address + 3 is the RESET REGISTER. A write to this register of any data will cause a continuous hardware reset of the PCSS–8FA/FX. The reset condition is cleared by a write of any data to the DATA REGISTER. A hardware reset of the computer also resets the board, but a warm boot (ctl-alt-del) does not.

3.f—The following subroutines can be used to communicate with the PCSS–8FA/FX. You can assemble them into a linkable .OBJ file and use them with any high level language. They only use the AL register in your computer. They are used in the examples in the command descriptions.

```
;to issue command, call this routine with the command number in
AL.
xcom:   push    dx      ;execute command subroutine
        push    ax
        call    waitr   ;make sure card is ready, wait if nec.
```

```
        pop     ax          ;waitr sets dx=com base address+1
        out     dx,al       ;send command to card
        pop     dx
        ret
;to transfer data to the board, call this routine with the data in AL.
xdat:   push    dx          ;execute command to send
        push    ax          ;data to card
        call    waitr       ;wait until card is ready
        pop     ax
        dec     dx          ;point to data port
        out     dx,al       ;send data to card
        pop     dx
        ret
;to transfer data from board, call this routine. Data is returned in AL.
rdat:   push    dx
        call    waitr       ;make sure card is ready, wait if nec.
        dec     dx
        in      al,dx       ;get data
        pop     dx
        ret
;waitr returns status if board is ready or waits until it is ready.
waitr:  push    cx                      ;wait until board is ready
        mov     dx,base_addr+1          ;point to cmd register
        mov     cx,0h                   ;maximum wait count
waitr2: in      al,dx                   ;get status
        test    al,80h                  ;check ready bit (bit 7)
        jz      waitr1                  ;uh oh, board not ready
        pop     cx
        ret                             ;return when board ready
waitr1: loop    waitr2                  ;try again if cx not timed out
        ;If we get here, the board failed to become ready in a
        ;reasonable period of time. This means the board has
        ;failed, so output an error message (see following)
        mov     dx,offset ermsg
        mov     ah,9        ;display message function
        int     21h         ;display the message
        mov     ax,4C01h;exit with errorlevel
        int     21H         ;abort program all way back to DOS.

ermsg:  db      cr,lf,7,'Card Fault - Card not ready.',cr,lf,'$'
```

Following is an example:

```
;assume ah=channel, al=data
xmit:   xchg al,ah            ;get channel , save data in ah
        call    xcom          ;select channel
xmit0:  call    waitr         ;get status
        test    al,20h        ;check txrdy bit
        jz      xmit0         ;ok to xmit
;here if ok to xmit data
        xchg    al,ah         ;get data back to AL
        call    xdat          ;send the data
        ret
```

OK, enough said. Now for the command descriptions.

Commands 00h Thru 07h
Channel Select.

These commands select the channel (CSC) upon which the following commands operate (except global commands). usage:

```
        mov     al,chan_no        ;desired channel
        call    xcom              ;select the channel
```

Command 08h
Force PCSS-8I compatiblilty mode.
Only available on the PCSS-8F

Command 09h
(MCSS–9IM Only)
Select Channel 9

Reserved for the MCSS–9IM.

Command 0Ah
(MCSS–9IM Only)
Change Baud Reserved for the MCSS–9IM.

Reserved for the MCSS–9IM.

Commands 0Bh and 0Ch
Reserved

Command 0Dh
Get The Currently Selected Channel.

```
mov     al,0dh
call    xcom
call    rdat        ;returns csc in AL
```

Block Commands 0Eh

Command 0Eh, Data 00h

Turns off special mode to transmit the the same data to all eight channels.

Command 0Eh, Data 01h

Turns on special mode to transmit the same data to all eight channels.

Command 0Eh, Data 02h

Toggles between the standard and alternate baud rate sets.

Issuing command 0Eh followed by data 02h toggles the baud rate set between the standard set and the extended set. The standard set is the default set selected when the card is initialized. This command affects all channels on the PCSS-8FA/FX. The baud rate selected by commands 18h and 1Ah are modified as shown following:

Standard Set	Extended Set
75	7200
110	880
150	14.4k
300	**28.8k**
600	57.6k
1200	**115.2k**
2000	2000
2400	57.6k
4800	4800
1800	**14.4k**
9600	9600
19.2k	19.2k
38.4k	38.4k
57.6k	57.6k

Command 0Eh, Data 03h

Enable DMA transfers from the PC to the PCSS–8FX. DMA NOTE: DMA transfers are only available on the PCSS-8FX versions 2.27 and above. DMA transfers must be enabled by selecting a DACK line with jumper JP2 and a DRQ line with jumper JP1.

Command 0Eh, Data 04h

Disable DMA transfers from the PC to the PCSS–8FX. See DMA note above.

Command 0Eh, Data 05h

Enable DMA transfers from the PCSS–8FX to the PC. See DMA note above.

Command 0Eh, Data 06h

Disable DMA transfers from the PCSS–8FX to the PC. See DMA note above.

Command 0Fh
Enable PCSS–8FA Mode
(PCSS–8F Only)

This command will make a PCSS–8F behave exactly like the PCSS–8FA described in this manual.

Command 10h
Start Break.

The transmit data line on CSC goes to +12 volts. The break is continuous until the stop break commmand is issued.

```
usage:  mov     al,10h   ;start break
        call    xcom     ;do it.
```

Command 11h
End Break.

The transmit data line on the CSC is restored to –12 volts.

usage:

```
        mov     al,11h   ;stop break
        call    xcom     ;do it.
```

Command 12h
Assert DTR (+12v)

Sets the DTR line on the currently selected channel to allow external device to send. The default is DTR inactive (–12v). If you are using auto DTR handshaking, do not use this command.

usage:

```
mov     al,12h   ;set dtr high
call    xcom     ;do it.
```

Command 13h
Negate DTR (–12V—Inactive)

Resets DTR to tell external device to stop sending. This happens in real time and is not inserted in the transmit queue. The default is DTR inactive. If you are using auto DTR handsahaking, then you should not use this command. Updated channel status for the CSC is placed on the data port.

usage:

```
mov     al,13h   ;set dtr low
call    xcom     ;do it.
```

Command 14h
Get Global CD Conditions.

This command will return a bit pattern on the data port showing the condition of all 8 CD lines. Bit 7 is channel 7, bit 6 is channel 6, bit 0 is channel 0, etc. If the bit is set, this means that an external device is asserting that line to +12 volts. Bits reset (0) mean that the CD line is –12 volts. This information is "real time" showing the condition of the CD lines, not from the queue. See also command 16h. After reading the the global cd bit pattern on the port, updated channel status for the CSC is placed on the data port.

usage:

```
mov     al, 14h    ;command
call    xcom       ;give command
call    rdat       ;get info
```

Command 15h
Get Global CTS Conditions.

This command will return a bit pattern on the data port showing the condition of all 8 CTS lines. Bit 7 is channel 7, bit 6 is channel 6, bit 0 is channel 0, etc. If the bit is set, this means that an external device is asserting that line to +12 volts. Bits reset (0) mean that the CTS line is –12 volts. This information is "real time" showing the condition of the CTS lines, not from the queue. See also command 16h. After reading the the global CTS bit pattern on the port, updated channel status for the CSC is placed on the data port.

usage:

```
mov     al, 15h   ;command
call    xcom      ;give command.
call    rdat      ;get info
```

Command 16h
Get Modem Status.

This command will get the handshaking status of the CSC. A byte of data is placed on the data port in the following format.

Modem Status

bit 7	CD status. If set then CD currently being asserted (+12)
bit 6	reserved
bit 5	Remote TX status. If set, card has inhibited the remote transmitter using automatic handshaking, i.e., Xoff sent or DTR negated.
bit 4	CTS status. If set, CTS is currently asserted (+12)
bit 3	Delta CD. Set if CD changed since the last time read.
bit 2	reserved
bit 1	Received Xoff. If set, the card has received an Xoff, cleared when the card receives an Xon (OK to transmit). Valid only if Xon/Xoff TX flow control is on.
bit 0	Delta CTS. CTS changed since last time read.

Bit five set indicates that the card has attempted to inhibit the equipment that is transmitting to the card by the currently selected automatic handshaking method. Bit one set indicates that the card has received an Xoff if that method of receive handshaking has been enabled. After reading the the MODEM STATUS bit pattern on the port, updated channel status for the CSC is placed on the data port.

usage:

```
mov     al, 16h   ;command
call    xcom      ;give command.
call    rdat      ;get info
```

Command 17h
Return Version Number.

This command will return the version of the operating system running in the card. Write the command number (17h) to the command port. Successive reads of the data port will return the version number. The first byte of data returned will be the number of bytes (n) in the version number. The next n bytes returned will be the version number. The last byte returned is the checksum of all the previous bytes.

```
;the following routine returns the version in ax. Returns ax=0 if
;version no. is bad
getvers: push   cx
         push   bx
         mov    al, 17h
         call   xcom      ;give command
         call   rdat      ;get data byte count (currently 2)
         mov    bl, al    ;generate check sum
         call   rdat      ;get data
         mov    cl, al
         add    bl, al    ;generate check sum
         call   rdat      ;get data
         mov    ch, al
         add    bl, al    ;generate check sum
         call   rdat      ;get check sum
         add    bl, al    ;finish check sum
         cmp    bl, 0
         jz     ident1    ;check sum is ok
         mov    cx, 0     ;invalid information in cx
ident1:  mov    ax, cx    ;return version in ax
         pop    bx
         pop    cx
         ret
```

The current version reads like this: ax=126Fh, which means Operating System Version 1.26F.

Command 18h
Initialize Port (IBM Style)

Baud Rate, parity, stop bit and word size initialization for the CSC. This command also has the effect of flushing the currently selected channel queue. The data byte written to the data port following the command is in the following format:

```
7       6       5       4       3       2               1       0
—baud rate—             -parity-        stop            word length
000- 110                x0-none         0-1 stop        10-7 bits
001-19200*              01-odd          1-2 stop        11-8 bits
010-300                 11-even
011-600
100-1200       (*)150 bps in IBM bios. (Who needs that?)
101-2400       you can use the dos MODE command to
110-4800       set 19.2k by using 150 in the command
111-9600       MODE COM1:150 will yield 19.2K baud
```

See Command 0Eh Data 02h to select between the standard and extended baud rate set. Extended set follows:

```
7       6       5       4       3       2               1       0
—-baud rate---          -parity-        stop            word length
000-     880            x0-none         0-1 stop        10-7 bits
001- 19,200             01-odd          1-2 stop        11-8 bits
010- 28,800             11-even
011- 57,600
100-115,200
101- 57,600
110-  4,800
111-  9,600
```

As an example, to set 1200 baud, even parity, 7 data bits with 1 stop bit, the data byte would be 10011010B or 9AH. Here is an example:

```
mov     al, 18h    ;the command
call    xcom
mov     al, 9Ah    ;the operand
```

```
        call    xdat
```

DTR is not affected by this command.

Command 19h
Handshake Init of the CSC.

You can use either Xon/Xoff handshaking or DTR/CTS handshaking. When this command is performed, it also has the effect of flushing the currently selected channel queue.

Auto Xon/Xoff handshaking controls the transmitting and receiving of characters to and from the PCSS–8FA/FX with Xon and Xoff characters.

If Auto Xon/Xoff Transmit Handshaking is enabled, an Xoff character received from an external device will automatically cause the PCSS–8FA/FX to stop sending any characters from its buffer until an Xon is received.

If Auto Xon/Xoff Receive Handshaking is enabled then when the "high water mark" is reached, an Xoff is automatically sent. This condition remains until characters are taken out of the receive queue for that channel and the "low water mark" is reached. A Xon is then sent.

Auto DTR/CTS handshaking controls the transmitting and receiving of characters to and from the PCSS–8FA/FX using the DTR and CTS hardware lines.

If Auto CTS Tx Handshaking is enabled and the CTS line goes low (–12), then no more characters will be transmitted from the PCSS–8FA/FX until it goes high again (+12).

If Auto DTR Rx Handshaking is enabled, then DTR is asserted automatically. Then when the "high water mark" is reached in the PCSS–8FA/FX receive buffer, DTR is set low (–12) until the receive buffer "low water mark" is reached by taking characters out of the receive queue. DTR is then set high (+12). If you select this method of handshaking, DO NOT use commands 12h or 13h since DTR is being controlled automatically.

The byte written to data register after the write to the command register is as follows (by bit):

7—reserved

6—reserved

5—Special Multi–Drop mode for GTEK's RS–485 adapter, on operating system versions 1.18d and later.

4—Enable Auto CTS Tx Handshake. Stops PCSS–8FA/FX from transmitting when CTS goes low (–12).

3—Enable Auto Xon/Xoff Tx Handshake. Stops PCSS–8FA/FX from transmitting when Xoff is received.

2—Enable Auto DTR Rx Handshake. Stops external device from transmitting when "high water mark" in the receive buffer is reached by setting DTR low (–12).

1—Enable Auto Xon/Xoff Rx Handshake. Stops external device from transmitting by sending Xoff when the "high water mark" in the receive buffer is reached.

0—reserved

usage:

```
mov     al, 19h    ;the command
call    xcom
mov     al, 14h    ;cts/dtr both directions
call    xdat       ;give data—DTR is now asserted.
```

NOTE: You **CANNOT** use both auto DTR and auto Xon/Xoff Rx handshaking at the same time. You **SHOULD NOT** use auto CTS and auto Xon/Xoff Tx handshaking at the same time.

Command 1Ah
Extended Baud Rates.

The parameter byte written to the data register following the command allows a greater degree of flexibility in setting baud rates than does command 18h. It is composed of two baud rate specification nibbles from the following table. Bits 7-4 control the baud rate of the receiver while bits 3-0 control that of the transmitter. See Command 0Eh to select the Normal or Extended Baud Rate Set.

Nibble data	Baud Rate Standard	Baud Rate Extended
0	75	7,200
1	110	880
3	150	14,400
4	300	28,800
5	600	57,600
6	1,200	115,200
7	2,000	2,000
8	2,400	57,600
9	4,800	4,800
A	1,800	14,400
B	9,600	9,600
C	19,200	19,200
2	38,400	38,400
D	57,600	57,600
E	reserved	reserved
F	reserved	reserved

For example, to transmit at 300 baud and receive at 1200 baud on the same channel, write 64h as the data to base address + 0 after the write of 1Ah to base address +1.

usage:

```
mov     al,1Ah    ;command
call    xcom
mov     al,64h
call    xdat      ;give the parameter
```

Command 1Bh
Set Handshaking Low Water Mark

This command and command 1Ch are used to adjust the levels for the automatic flow control on the receive queue. To set the receive low water mark on the csc, issue command 1Bh followed by the LSB and then the MSB of the new low water mark. There is a minimum value of one. If some form of handshaking is enabled and if high water mark is reached (upon which DTR goes low or XOFF is transmitted), then the re-enabling of the handshake signal (raising DTR or sending XON) will occur when enough characters are removed from the receive queue to reach the low water mark again.

IF the sum of all the transmit and receive limits exceeds the available system memory then there is the possibility that when a queue attempts to allocate a memory block, there will be none available. If this happens, the handshake flow control will take place (DTR low or XOFF sent) and the RESERVE block will be allocated to the channel. Each receive channel keeps a reserved block in case this condition occurs. If this happens, then the flow control will be re-enabled as soon as system memory becomes available. That is the DTR high or XON will occur as soon as the memory shortage clears rather than wait until the low water mark is reached.

```
;Set the receive low water mark
        mov     al,1Bh   ;command to set the low water mark
        call    xcom     ;issue the command
        mov     al,2h    ;low byte of the low water mark
        call    xdat     ;send the low byte
        mov     al,0     ;high byte of the low water mark
        call    xdat     ;send the high byte
;now the low water mark on the csc is set to two
```

Command 1Ch
Set The Receive High Water Mark

This command and command 1Bh are used to adjust the levels for the automatic flow control on the receive queue. To set the receive high water mark on the csc, issue command 1Ch followed by the LSB and then the MSB of the new high water mark. The high water mark is the place where DTR is lowered or an XOFF is sent if automatic handshaking is enabled with command 19h.

```
;Set the receive high water mark
        mov     al,1Ch   ;command to set the high water mark
        call    xcom     ;issue the command
        mov     al,10h   ;low byte of the high water mark
        call    xdat     ;send the low byte
        mov     al,0     ;high byte of the high water mark
        call    xdat     ;send the high byte
;now the high water mark on the csc is set to ten
```

Command 1Dh
Set Non–standard Baud Rate

Command 1Dh allows the user to select non–standard baud rates up to 57,600 on the csc. The command is issued followed by the LSB and MSB, in order, of the count to be calculated as follows:

COUNT = $^{115,200}/_{Desired\_Baud\_Rate}$

Note that the count must be greater than or equal to 2. Both bytes must be sent, even if one is zero. This command flushes both queues on the csc.

It is not possible to set different non–standard rates on pairs of channels such as 0–1, 2–3, etc. The second channel would have to use either the first one's rate or a standard baud rate, eg. channel 0 set to 57,600 would cause you to use either 57,600 for channel 1 or a standard baud rate such as 1200, 2400, etc. for channel 1.

usage:

```
;Set the baud rate to 1920 baud
;we calculated the count for the above data and formula
;to be 003Ch.
        mov     al,1Dh   ;command to set non standard baud
        call    xcom     ;issue the command
        mov     al, 3Ch  ;the low byte of the count
        call    xdat     ;send the low byte
        mov     al, 0    ;the high byte of the count
        call    xdat     ;send the high byte
;now the baud rate is 1920 on the csc, RX and TX
;the data for 2400 baud would be 0030h
```

Command 1Eh
Block commands

This command is issued followed by the function number on the data port as follows:

Command 1Eh, Data 00h
Return Free Memory Blocks

Issuing command 1Eh followed by a zero to the data port will cause the operating system to return the number of blocks of buffer memory that are currently not in use to the data port. The low byte is returned first, followed by the high byte on successive reads.

Command 1Eh, Data 01h
Return Buffer Memory Size

Issuing command 1Eh followed by a one to the data port will cause the operating system to return the number of blocks that can be used by the buffers in this system. The low byte is returned first, followed by the high byte on successive reads.

Command 1Eh, Data 02h
Return Reserved Code Size

Issuing command 1Eh followed by a two to the data port will cause the operating system to return the number of blocks that are reserved for code. The low byte is returned first, followed by the high byte on successive reads.

Command 1Eh, Data 03h
Return number of TX Blocks in Use

Issuing command 1Eh followed by a three to the data port will cause the operating system to return the number of blocks that are in use in the transmit queue on the currently selected channel. The low byte is returned first, followed by the high byte on successive reads.

Command 1Eh, Data 04h
Return number of RX Blocks in Use

Issuing command 1Eh followed by a four to the data port will cause the operating system to return the number of blocks that are in use in the receive queue on the currently selected channel. The low byte is returned first, followed by the high byte on successive reads.

Command 1Eh, Data 05h
Return TX Limit

Issuing command 1Eh followed by a five to the data port will cause the operating system to return the maximum number of blocks of memory that the transmit queue on the currently selected channel can use. The low byte is returned first, followed by the high byte on successive reads.

Command 1Eh, Data 06h
Return RX Limit

Issuing command 1Eh followed by a six to the data port will cause the operating system to return the maximum number of blocks of memory that the receive queue on the currently selected channel can use. If hand shaking is enabled, the remote transmitter will be inhibited from transmitting when the receive queue reaches this limit. The remote transmitter will also be inhibited when there is no available memory even though the receive queue has not reached the limit. If this happens, the remote transmitter will be allowed to transmit as soon as memory becomes available. The low byte is returned first, followed by the high byte on successive reads.

Command 1Eh, Data 07h
Return RX Low Limit

Issuing command 1Eh followed by a seven to the data port will cause the operating system to return the "low water mark." This is the size in blocks that the receive queue has to go down to in order for the remote transmitter to continue transmitting if it had been inhibited from transmitting because the receive queue had reached the limit. If the low limit is set such that it is within one block of the receive limit, the remote transmitter will be inhibited/enabled as the last block is allocated/de–allocated to the receive queue. This will result in no spread between the high water and low water limits and could slow operation. The low byte is returned first, followed by the high byte on successive reads.

Command 1Eh, Data 08h
Return Total Number of Blocks in Use

Issuing command 1Eh followed by a eight to the data port will cause the operating system to return the total number of blocks of memory that are in use in all the queues. This number includes the eight blocks held in reserve by the receive queues. The low byte is returned first, followed by the high byte on successive reads.

Command 1Eh, Data 09h
Return System Memory Size

Issuing command 1Eh followed by a nine to the data port will cause the operating system to return the number of 32k "segments" of memory that are installed on the board. The low byte is returned first, followed by the high byte on successive reads.

Command 1Eh, Data 0Ah
Set TX Limit

This function will set the maximum number of blocks available for the transmit queue on the currently selected channel. This is accomplished by first issuing command 1Eh and then writing 0Ah to the data latch followed by the limit that you would like to use. Write the low byte first followed by the high byte even if the high byte is zero. The default transmit limit is 6 times the number of 32K memory blocks. Remember there is a minimum of one.

Command 1Eh, Data 0Bh
Set RX Limit

This function will set the maximum number of blocks available for the receive queue on the currently selected channel. This is accomplished by first issuing command 1Eh and then writing 0Bh to the data latch followed by the limit that you would like to use. Write the low byte first followed by the high byte even if the high byte is zero. The default receive limit is 10 times the number of 32K memory blocks. Remember there is a minimum of two because each channel holds one block in reserve to use if there are no free blocks available as the queue grows. When the queue uses this or its reserved block, the remote transmitter is XOFFed or DTR is negated if either type of handshaking is enabled. As soon as memory becomes available it is allocated to the receive queue and the remote transmitter is allowed to begin sending again.

Command 1Eh, Data 0Ch
Set RX Low Water Mark

This function will set the "low water mark" in blocks for the receive queue on the currently selected channel. This is accomplished by first issuing command 1Eh and then writing 0Ch to the data latch followed by the lower limit that you would like to use. Write the low byte first followed by the high byte even if the high byte is zero. The default lower limit is 8 times the number of 32K memory blocks. Remember there is a minimum of one. If the lower limit is set to a number one lower than the RX limit or greater, the remote transmitter will be allowed to continue sending as soon as the queue drops below the high water mark.

Command 1Eh, Data 0Dh
Enable Irq line

This function will enable the IRQ line on the board. The IRQ number is selected by jumper as discussed in chapter 3. The selected IRQ is enabled by issuing command 1Eh and then writing 0Dh to the data latch followed by data 02h. If this command is not issued or if it is issued with a zero for the data byte, the board will not interrupt the host PC at all. Events that are enabled by command 30 will still be stored in the event queue and command 2Fh can be used to detect their occurrence.

Command 1Eh, Data 0Eh
Enable Timed interrupt

Command 1Eh sub 0Eh enables interrupt type 8, which is a timed interrupt. Write 1Eh to the command port followed byt 0Eh to the data port. Follow this with the low and then the high byte of the number of milliseconds that you wish to wait before an interrupt occurs. This timed intterupt must be enabled each time that you want the interrupt to occur.

usage:

```
;Enable an interrupt after 520 milliseconds.
        mov     al,1Eh     ;major command
        call    xcom       ;issue the command
        mov     al,0Eh     ;sub command for timed interrupt
        call    xdat       ;send the sub command number
        mov     al, 8      ;the low byte of the count
        call    xdat       ;send the low byte
        mov     al, 2      ;the high byte of the count
        call    xdat       ;send the high byte
;interrupt type 8 will occur in 208h or 520 milliseconds.
```

Command 1Fh
Poll Event ID Queue

Command 1Fh has been added to check for an interrupt ID in the interrupt ID queue. This command will return the oldest interrupt ID or 0FFh if no events have occurred. These interrupt events are enabled with command 30h. This command, unlike command 2Fh, does not remove the interrupt ID from the queue or cause the board to interrupt the PC if more interrupts are pending. This may be useful in your interrupt service routine in order to decide what to process without having to disable interrupts. You should still do command 2Fh at the end of your interrupt service routine so that the board can interrupt the PC on the next event.

Command 20h
Flush Tx Queue On CSC.

A write of 20h to the command register immediately empties the currently selected channel's transmit queue. It has no effect on any other channel.

usage:

```
mov     al, 20h
call    xcom;that's all there is to it.
```

Command 21h
Flush Rx Queue On CSC.

A write of 21h to the command register immediately empties the currently selected channel's receive queue. It has no effect on any other channel.

usage:

```
mov     al, 21h
call    xcom      ;that's all there is to it.
```

Command 22h
Flush Tx/Rx Queues.

A write of 22h to base address + 1 immediately empties the currently selected channel's receive and transmit queue. It has no effect on any other channel.

usage:

```
mov     al, 22h
call    xcom      ;that's all there is to it.
```

Command 23h
Get Rx Queue Count

This command will give you the number of characters in the csc receive queue . The data returned to the data port (base + 0) is the low byte of the count followed by the high byte of the count.

usage:

```
mov     al, 23h   ;command
call    xcom
call    rdat
mov     ah, al    ;save low byte
call    rdat      ;get msb
xchg    al, ah    ;now have count in ax
```

The default value of 10 blocks on a 32K "segment" would allow for a maximum of 1680 characters.

Command 24h
Get Tx Queue Count

This command will give you the number of characters in the csc transmit queue that are waiting to be transmitted. The format is identical to command 23H.

Command 25h
Get Global Rxrdy Status.

This command allows you to check to see if any queue has received characters available. Bit 7 of the byte corresponds to channel 7, bit 6 is channel 6, etc.

usage:

```
        mov     al, 25h
        call    xcom
        call    rdat        ;get bit pattern
```

If the data byte read is FFh, that would mean that there are characters in every channel. You could then determine how many in a particular channel by performing a channel select command (0–7h for channels 0–7) and then a command 23h.

Command 26h
Get Global Txrdy Status.

This command returns a bit pattern corresponding to the transmitter queue ready bits. A bit that is set indicates the transmit queue for that channel can accept more data (ie bit 7 set means that channel 7 transmit queue is not full).

usage:

```
        mov     al, 26h
        call    xcom
        call    rdat        ;get bit pattern
```

Command 27h
Get Global Txemp Status.

This command returns a bit pattern corresponding to the transmitter queue empty bits. This actually is telling you there are no characters at all waiting to be sent in the transmit queue of each channel. Bit 7 is channel 7, bit 6 is channel 6, etc.

usage:

```
mov     al, 27h
call    xcom
call    rdat        ;get bit pattern
```

Command 28h
Pre–emptive Transmit.

This command allows you to put a character directly into the transmit holding register of the uart on the csc. More than likely, the character in the uart's transmit holding register will be over–written and thus that character will never be sent. Use this command with that in mind. It could even over–write an Xoff or Xon character in the tx holding buffer.

usage:

```
xchg    al, ah      ;save char in ah
mov     al, 28h
call    xcom
xchg    al, ah      ;get data
call    xdat        ;give pcss8fa/fx the data
```

Command 29h
Sample Most Recently Received Character.

This command allows you to sample the last character and status put into the receive queue. Status is returned first followed by the data. NOTE: This command reads the latest received character and does not remove the character from the queue. The command is thus non–destructive.

usage:

```
mov     al, 29h
call    xcom
call    rdat     ;get status
mov     ah, al   ;save it
call    rdat     ;now have stat in ah, data in al
```

Command 2Ah
Send Xon.

This command sends an Xon character on the csc. This command is used to send an xon character if you have auto xon/xoff rx handshaking in effect and you think that the remote device may have missed his xon. You can tell that the remote device is not supposed to be xoffed by examining bit 5 in the modem status.(see command 16h)

usage:

```
mov     al, 2ah
call    xcom     ;do it
```

Command 2Bh
Receive Xon.

This command allows you to simulate reception of Xon in the receive buffer. This will allow the resumption of the transmission of data if you have Auto Xon/Xoff Tx Handshaking enabled. It is acted on immediately, as if an Xon had been received. You can tell if you are in an Xoffed condition by examining bit 1 in the modem status (command 16h).

usage:

```
mov     al, 2bh
call    xcom    ;ok
```

Command 2Ch
Reserved

Command 2Dh
Reserved

Command 2Eh
Scan Receive Queue

This command is used to search the receive queue without removing any characters from the queue. Issue the command followed by the low and then the high bytes of the number **N** of the character that you wish to examine. Byte number 1 is the oldest byte in the queue. Be sure to use command 23h first to determine how many characters are in the receive queue. This command will return first the status and then the data that was in the queue at position **N**.

Examples

The following example is an assembly language program to initialize a channel on the PCSS-8FA/FX.

```
csc:    db      0         ;current com channel selected 0-7
IUART:  mov     al, csc   ;comc= current com channel
        call    xcom      ;give command to select channel
        mov     al, 18h   ;initialize baud/parity/stop
        call    xcom      ;send command
        mov     al, 23h   ;19,200/8data/1stop/no parity
        call    xdat      ;send data to finish
        mov     al, 1Ah   ;set new baud of 57,600
        call    xcom      ;send command
        mov     al, 0DDh  ;parameter for 57,600 tx/rx
        call    xdat      ;send data to finish
        mov     al, 19h   ;set handshaking command
        call    xcom      ;give command
        mov     al, 14h   ;data to set hardware handshaking
        call    xdat      ;send data to finish
        ret     ;we are finished initializing the channel
```

Command 2Fh
Interrupt Acknowledge

This command should be issued after an interrupt has occurred and you have acknowledged the 8259 interrupt controller. It will return a data byte indicating what type of interrupt has occurred as outlined in the documentation of command 30h. If more interrupts are pending when this command is issued, the selected interrupt will be activated again. You should disable PC interrupts while in your interrupt service routine to keep from having nested interrupts.

Command 30h
Enable Interrupt on Event

Command 30h is used to enable the PCSS–8FA/FX to interrupt the Host PC on one of four events. You can select the type of event for the PCSS–8FA/FX to interrupt the PC with the ID that you write to the data port after the command.

Event	ID for channel 0—7
Transmitter Empty	00h—07h
Received data available	10h—17h
Receive buffer at limit	20h—27h
Transmitter Ready	30h—37h
Receive character match	40h—47h
Receive queue at level	50h—57h

For example if you wish to have the PC interrupted when there is received data on channel 3, the data byte to send would be 13h. Each of these interrupts is a one time event marker. If you wish to have the interrupt again on a similar event, you must enable it again. Command 2Fh is the interrupt acknowledge and it will return to you the identification number of the interrupt that has occurred in the same format as outlined here. The "Receive buffer at limit" interrupt depends on some type of receive handshaking being enabled with command 19h first. If you do not have receive hand shaking enabled, this event will never occur. The "Transmitter ready" interrupt can be used to mark the change from transmit buffer full (not ready) to ready. If you enable it before the buffer is full, the condition is true! Enabling any of the interrupts while the condition is true will cause the interrupt to occur right away. When this interrupt occurs you are assured of being able to transmit 252 more characters.

Multiple types of events on multiple channels can be enabled at any time. As the events occur the associated IDs are stored in a first in first out buffer so that they can be serviced in the order that they occur. If a condition is true when that event is enabled with this command the ID is immediately put into the event buffer, except type 40–47h.

Event Types 40h–47h occur when a received character matches the compare character stored with command 3ch. Event types 40h–47h , unlike the other event types, do not have to be enabled after each time the matched character is received. This command toggles the enabling of this type of interrupt. You simply enable this type of interrupt during the initialization of the board and each time a match receive character comes in, the interrupt ID will be stored in the interrupt ID queue. If you ever want to disable this interrupt type after it is enabled, simple issue command 30h followed by data 40-47h again to turn it back off.

Event Types 50–57 signal the receive count has reached the count supplied for the appropriate channel by command 3ch. The PCSS–8F/FX will continue to buffer receive characters even after the count is reached.

Command 3Bh
Return the Number of Pending Interrupts

Command 3Bh will return a data byte on the data port indicating how many events have been detected and are awaiting service. The count that is returned is the sum of events that have been enabled with command 30h that have occurred and have not been acknowledged with command 2Fh.

Command 3Ch
Store Compare Character

Command 3Ch is used to store the compare character or the receive level count on the currently selected channnel. Issue 3Ch to the command port at the base address plus one, followed by the byte you wish to look for or the number of received characters to the data port at the base address.

Command 3Dh
Clear Parity Bit On The Csc.

Command 3Dh forces the transmitter on the csc to transmit with the parity bit cleared. This command is sometimes used with RS–485 communication to indicate data bytes.

Command 3Eh
Set Parity Bit On The CSC.

Command 3Eh forces the transmitter on the csc to transmit with the parity bit set. This command is sometimes used with RS–485 communications to indicate address bytes for slaves.

*Technical information and specifications provided in this document are **SUBJECT TO CHANGE WITHOUT NOTICE.***

—Notes—

APPENDIX—B

Table of Contents

Using RES8F.COM

RES8F.COM is a terminate and stay resident program to extend the BIOS INT14 handler. Once installed, any program that uses INT14 for communications will work with the PCSS–8FA/FX. After RES8F has been installed, SSI.com can be run to attach DOS ports COM1: through COM4: to any of the intelligent card ports (or simply use function call 18h). The DOS MODE command will work with a channel attached in this manner except that 19.2k baud is available instead of 150 baud. See function call zero for details.

The driver is installed with the following syntax:

C>**RES8F 2e0 2e4 aaa aaa<enter>**

Where the addresses which follow the command are the hexadecimal board base addresses of all PCSS–8FA/FX's (address=aaa) present in the system that you want to be recognized by RES8F. The intelligent channels are addressed with DL=80h–FFh (indicating channels 0–128) when making the INT14 function call. In the example above, channels 80h–87h would be on the card at address 2E0h (channels 0–7) since it is first in the list. Channels 88h–8Fh would be on the card at address 2E4h (channels 8–15).

If you use multiple cards, you should enter the base addresses in order on the RES8F command line so the channels will be organized neatly on the rear panel of the computer. An "N" on the command line before the addresses would keep RES8F from aborting if it finds a non–working card (or an empty slot). An example is:

```
RES8F  n 2e4 2ec 2e8 2e0
```

This is the way the hardware is configured for previous example:

board	@2e4	@2ec	@2e8	@2e0	other boards
	0(80h)	8	16	24(98h)	etc.
	1	9	17	25	channel #'s in order
	2	10	18	26	of appearance
	3	11	19	27	
	4	12	20	28	
	5	13	21	29	
	6	14	22	30	
	7(87h)	15	23	31(9fh)	

Channel numbers are in decimal!

The following error level codes are returned when RES8F is executed:

error level=5 if de–installation is successful
error level=4 if de–installation fails
error level=3 if driver already installed
error level=1 if driver installs successfully
(note—if "n" on command line, boards not installed will not cause an error level=0)
error level=0 if driver installation fails.

These error levels can be used by a batch file to recover from errors. An autoexec batch file example might be:

```
PROMPT $P$G
PATH C:\;C:\DOSCOMS;C:\COMM;
RES8F 2C0 2E0 2D0 2F0
IF ERRORLEVEL=3 GOTO PDONE
IF ERRORLEVEL=1 GOTO PASS
IF ERRORLEVEL=0 GOTO FAIL
GOTO FINISH
:PDONE
ECHO DRIVER WAS ALREADY INSTALLED.
GOTO FINISH
:PASS
ECHO DRIVER WAS SUCCESSFULLY INSTALLED
GOTO FINISH
```

```
:FAIL
ECHO DRIVER COULD NOT BE INSTALLED
:FINISH
ECHO DONE
```

To remove RES8F from memory simply enter:
C>**RES8F u<enter>**
This will free the memory occupied by RES8F and also allow you to enter RES8F again with different parameters. If this is done from a "shell" or when you have loaded other TSRs "on top of" RES8F, the memory is freed up, but can't be used by the computer.

The function calls are executed by loading AH with the number of the call that you want to execute and then executing an INT14. AL and Dh are used to pass information to the function call. DL must contain the channel number (80h–FFh) as outlined above.

INT 14h

This routine provides byte stream i/o to the communications ports according to the following parameters. Note that all registers are preserved, except ax:

On entry, DL must contain the port number as follows: DL=communications channel 80h–FFh (indicating channels 0–127). This is 128 channels, or 16 cards.

On entry, AH contains the function number and AL contains parameters.

On entry, DH may also contain parameters (function call 6) or 0 if you are using a standard communications port or 0–7 if you are using a PCSS–8 card port.

On return, AX contains answers (if applicable):
Generally, if bit 7 of AH is set upon return, an error has occurred. See function 40h

Function Calls for INT 14h

AH=00h
Initialize the Communications Port.

AL has parameters for initialization as follows:

```
7       6       5       4       3       2               1       0
—baud rate—             -parity-        stop            word length
000- 110                x0-none         0-1 stop        10-7 bits
001-19200*              01-odd          1-2 stop        11-8 bits
010-300                 11-even
011-600
100-1200        (*)150 bps in IBM bios. (Who needs that?)
101-2400        you can use the dos MODE command to
110-4800        set 19.2k by using 150 in the command
111-9600        MODE COM1:150 will yield 19.2K baud
```

On return, conditions set as in call to communications status (ah=3). See function 40 for possible error return codes. No error if ah bit 7 is clear. See function 2Ch for setting extended baud rates.

```
Extended Baud rates: See function 2Ch for details:
Bits    765                     43210-the rest are the same as above
        000          880 Baud
        001       19,200
        010       28,800
        011       57,600
        100      115,200
        101       57,600
        110        4,800
        111        9,600
```

AH=01h

Send Character in AL Over the Comm Line.

AL contains the character to be sent and is preserved.

On exit, bit 7 of AH is set if the routine was unable to transmit the byte over the line. If bit 7 is not set, the remainder of AH is set as in a status request, reflecting the current status of the line. (AH=3). Note: The character won't be sent unless the transmit character buffer is empty and CTS is high if in the compatible mode. See function call 5, bit 1.

Also in the "compatability mode" DTR will be asserted the first time this call is made. The transmit time-out is approximately a second. See function 40 for possible error return codes. No error if AH bit 7 is clear.

AH=02h

Receive Character in AL From Comm Line Before Returning to Caller.

No parameter required for AL on entry. On exit, AH has the current line status, as set by the status routine, except that the only bits left on are the error bits (7,4,3,2,1). If AH has bit 7=1 (time–out), the remaining bits are not predictable. This means AH is non–zero only when an error has occurred.

DTR will be set high the first time you use this function call for IBM bios compatibility, as long as function call 5 has not been called. If function call 5 has been performed with other than 0 as a parameter, this function call will not affect DTR. See function 40 for possible error return codes. No error has ocurred if AH bit 7 is clear.

AH=03h
Return the Comm Port Status in AX.

No parameter required for AL on entry. On exit, AH contains the line control status:
Bit 7 = time out
Bit 6 = transmit shift register empty
Bit 5 = transmit holding register empty
Bit 4 = Break detect
Bit 3 = Framing error
Bit 2 = Parity error
Bit 1 = Overrun error
Bit 0 = Data ready

The information concerning bits 1–4 comes from the receive queue. If bit 0, the RXRDY bit = 1, there is data in the receive queue. AL contains the current (real time) modem status:
Bit 7 = CD. Set if CD=+12V
Bit 6 = (see function 5)
Bit 5 = (see function 5)
Bit 4 = CTS. Set if CTS=+12
Bit 3 = ΔCD. CD changed since last fc 3 if set.
Bit 2 = (see function 5)
Bit 1 = (see function 5)
Bit 0 = ΔCTS. CTS changed since last fc 3 if set.

See function 40 for possible error return codes. No error if ah bit 7 is clear.

The following functions are enhancements to the standard INT 14h functions.

AH=04h
Determine if GTEK RES8F Driver Installed.

On entry, Al=00h. On exit, Al is the high byte of RES8F version number with bit 7 forced to 1. If the version number =1.00A, then this byte would be 90h. AH is the low byte of the RES8F driver version. In the example above, AH would equal 0Ah. AL will be equal to 0 if the RES8F driver is not present.

AH=05h
Set Handshaking and Compatibility Mode.

This function call enables or disables the handshake and 8250 compatibility features. On entry, AL is set as in the following table:

Bit	Clear(0)	Set(1)	
7	disable	enable	CTS (rx) flow control
6	disable	enable	Xon/Xoff (tx) control
5	disable	enable	DTR (tx) flow control
4	disable	enable	Xon/Xoff (rx) control
3	disable	enable	Set handshake as above
2	disable	enable	Extended Status in AL
1	disable	enable	Tx Queue status meaning
0	Reserved		

Bit 0=Reserved, used internally.

Bit 1=1—allows queueing of transmit data.

NOTES: If bit 1 is set to 1, the txrdy status bit of function call 3 will indicate that the transmit queue is not full when it is a 1. That means txrdy=1 (see function call ah=03h) if tx queue can hold more data. This allows for queuing of transmitted data. This allows you to transmit until the transmit queue is full. You can use the TX empty bit to tell when the transmit queue is empty.

When bit 1 = 0, (which is the default) the txrdy status bit in function call 3 will be set to indicate the transmit queue is empty. This provides compatibility with a standard 8250 type communication channel in that the 8250 does not queue transmitted data. **IMPORTANT**: With this bit 0, which is the default, function call 1 (transmit data) cannot transmit unless the CTS line is high (+12v). This is compatible with the IBM bios. If you set bit 1 to a 1, and don't check CTS yourself or enable one of the auto handshake modes, the transmitter can transmit regardless of the state of CTS. Set bit 1=1 in this function call if you want to use the transmit queue.

Bit 2=1 Enable extended status return in AL when you use function calls 3 and 0 of RES8F. The modem status which is returned in AL in functions 0 and 3 is as follows:

Bit	default AL status	extended AL status
7	as fc 03h	
6	always 0 (RI)	reserved
5	always 1 (DSR)	1 if DTR −12v and auto DTR Rx flow control enabled or if Xoff sent and auto Xon/Xoff Rx flow control enabled.
4	as fc 03h	
3	as fc 03h	
2	always 0 (ΔRI)	1 if Xoff received and Xon/Xoff tx flow control is enabled
1	always 0(Δ DSR)	reserved
0	as fc 03h	

Note that since the PCSS–8FA/FX does not have RI or DSR inputs, the DSR, RI, delta DSR and delta RI are set to 1 for DSR and 0 for the rest in the default mode for maximum compatibility reasons.

Bit 3 =1 to enable handshake mode set via the next 4 bits. The following four bits (4–7)are the handshake control bits. They have no effect unless bit 3 is set when you make this function call. That allows you to make changes to the functions in bits 0–2 without changing the handshake mode. The reason you might wish to do this is that setting the handshake mode on the PCSS–8FA/FX causes the queues to be flushed and the remote transmitter (ie. the remote equipment that is connected to the PCSS–8FA/FX port) to be enabled if it had been disabled by automatic handshaking.

Bit 4 = 1 to enable auto rx xon/xoff flow control. If the receive queue reaches the high water level, an Xoff will be sent (providing one has not already been sent.) When the receive queue reaches the low water level, the PCSS–8FA/FX will automatically transmit an xon in this mode. Bit 5 of AL in function calls 3 and 0 will read a 1 if the PCSS–8FA/FX has transmitted an Xoff and if bit 2 is set in this function call, there by allowing extended status to be read.

Bit 5 = 1 to enable auto rx DTR type flow control. If the receive queue reaches the high water level, dtr will be negated. When the receive queue reaches the low water level, DTR will be re–asserted (+12v). If bit 5 is set in this fc, bit 4 is meaningless as the DTR type handshaking takes priority over the Xon/Xoff method. Bit 5 of AL in function calls 3 and 0 will read a 1 if DTR is low and bit 2 is set in this function call.

Bit 6 = 1 to enable tx Xon/Xoff flow control. If bit 6=1 and the PCSS–8FA/FX has received an xoff character, then the PCSS–8FA/FX will not transmit anything until an xon is received. This of course, does not prevent you from transmitting as long as there is room in the transmit queue and bit 1 of this function is set so that Txrdy from fc 03h indicates that there is room in the queue. If bit 2=1 of this fc, then bit 2 from fc 03h and 00h returned in AL will indicate xoff received.

Bit 7 = 1 to enable tx CTS handshaking. If the cts input is low (–12v), the PCSS–8FA/FX's transmitter will not be able to transmit. The condition of CTS is returned in bit 4 of AL in functions 00h and 03h. See function 40 for possible error return codes. No error if ah bit 7 is clear.

AH=06h
Handshaking Level Control.

See also function 1Eh in this appendix. This function is used to set the queue levels at which handshaking flow control is activated. On entry, DL=channel number with high bit set (80h–0FFh). On entry, DH=LSB of level and AL=MSB of level.

AL also has bit 7 set to indicate the number in DH and the rest of AL is the Low Water Mark. This is the receive queue level at which DTR goes active (+12) or Xon gets transmitted if one of these auto handshake modes is enabled. The default level is 1344 characters.

If AL has bit 7 set to 0, that means the DH and AL is to set the High Water Mark. This is the receive queue level at which DTR goes inactive (–12) or Xoff gets transmitted if one of these auto handshake modes is enabled. The default level is 1680 characters. See function 40 for possible error return codes. No error if AH bit 7 is clear.

AH=09h
Transmit BREAK Control.

This function is provided so that the user can send a break on the line if he so desires. Enter with AL=1 to set break. TXD goes to +12v. Enter with AL=0 to clear it –12v. Returns nothing. See function 40 for possible error return codes. No error if AH bit 7 is clear.

AH=0Bh
Request queue count/size.

Returns the number of characters currently in the queue or the size of the queue in AX. Enter with DL=channel no. and AL=0 for tx queue count, AL=1 for rx queue count, AL=2 for tx queue size, Al=3 for rx queue size. See function 40 for possible error return codes. No error if AH bit 7 is clear.

AH=0Ch
Queue Flush.

Emptys the specified queue or queues. Returns nothing. On entry if AL=0, flush tx queue, AL=1, flush rx queue. Returns DTR to ready (+12v) if DTR is low and auto DTR mode is enabled. Sends Xon if Xoff was sent and auto

Xon/Xoff mode is enabled. If AL=2, flush both. See DTR, Xon/Xoff side effects above. See function 40 for possible error return codes. No error if AH bit 7 is clear.

AH=0Dh
Control DTR line.

This function is provided so that the user has control of DTR. Enter with AL=1 to set it (+12v), AL=0 to clear it (–12v). Returns AH bit 7 set if error.

IMPORTANT CONSIDERATIONS:
DTR is NOT set active when the port is initialized by function call 00h. DTR is set active by initializing the channel with DTR handshaking. It also gets set high the first time you call function 01h or 02h (for IBM bios compatibility), as long as you don't use function 05h. If you call function call 05h (with anything other than 0 in AL), you have left the world of compatibility. Then you must use this function call to set it high if you want to. This function should not be used if the auto DTR handshake mode is in use.

See function 40 for possible error return codes. No error if AH bit 7 is clear.

AH=0Eh
Sample Most Recently Received Character.

Returns the most recent received status and data in AX. This is non–destructive and the queue pointers are not updated. This function call will return meaningless data if there was no received data in the queue. You should check the status byte to be sure there is some received data before you attempt this function call. See function 40 for possible error return codes. No error if AH bit 7 is clear.

AH=18h
Redirect Standard COM Port to Intelligent Port.

Enter with AL=0, 1, 2, 3 for COM1, COM2, COM3, or COM4 respectively and DL=80h–0FFh, the intelligent port you wish redirection to. After this function call, INT14 function calls to the standard com channel will be directed to this intelligent port. Redirection may be canceled by calling with AL=80h and DX=0–3, the redirected port. See function call 1ah to get redirection status. See function 40 for possible error return codes. No error if AH bit 7 is clear.

AH=19h
Global Poll.

Returns the bit pattern of requested status bits for the entire board. Enter with AL=0 for Receive character ready, AL=1 for Transmit ready, AL=2 for Transmit empty, AL=3 for CTS –12v, AL=4 for CD=–12v. DL contains the channel number. On return, AL contains the status bits requested. Bit 0 represents channel 0 on the board, Bit 1 for channel 1 and so on. See function 40 for possible error return codes. No error if AH bit 7 is clear.

AH=1Ah
Get the CSC # or Board Base Address.

Enter with AL=0 and DX = 0, 1, 2, 3 for com1, com2, com3, or com4. Returns AL = 80h–0FFh, the intelligent port that this DOS com port is attached to. Returns 0 if it's unattached, or if fc 18h has not been used.

Enter with AL=1 and DL=80h–0FFh (any channel on the board in question). This returns the board's base address. The base address is returned in the AX register. Example: AX=02E0h=intelligent board base address. Your program could use this function to find the base address of the board and then operate the board in the command driven mode as explained in Appendix A. See function 40 for possible error return codes. No error if AH bit 7 is clear.

AH=1Eh
Memory Block Status.

Enter with **DL** equal to the channel on the board you are interested in and **AL** equal to the sub–function number that you wish to call. NOTE: All sub functions of function 1eh return results in **AX**. Bit 7 set in **AH** means that an error occurred. Note this Function Call looks almost the same as COMMAND 1Eh—But it is not accessed the same way!

Command 1Eh Sub Functions:

Command 1Eh, Data 00h
Get # of Available Buffer Memory Blocks

AL=0 returns free buffer memory blocks in system in **AX**. (Those not currently in use but which are available for use as receive or transmit buffers). This is a global value and is board specific, not channel specific.

Command 1Eh, Data 01h
Get Total # Buffer Memory Blocks in System

AL=1 returns total number of buffer memory blocks in system. This is the number of free blocks plus all those in use as buffers. For example, this number is 125 (7dh) on a PCSS–8FA/FX with 32K of Dynamemory. This is a global value and is board specific, not channel specific.

Command 1Eh, Data 02h
Get # Blocks Reserved for Code Memory

AL=2 returns the number of blocks reserved for code memory. This is a global value and is board specific, not channel specific.

Command 1Eh, Data 03h
Get # TX Blocks in Use on CSC

AL=3 returns the number of transmit blocks in use on the CSC (Currently Selected Channel).

Command 1Eh, Data 04h
Get # RX Blocks in Use on CSC

AL=4 returns the number of receive blocks in use on the CSC.

NOTE: the following limits can always be reached if the sum of all the limits which you set do not exceed that returned in sub function 1 above. If the sum of your limits exceeds the number of buffer memory blocks in the system, then the system is operating in the DYNAMIC allocation mode. Memory is allocated to the buffers as needed. They are returned to the free memory pool as characters are removed. So you might not be able to get to the limit you assigned. The nice thing though is that the condition is only temporary. You can always be assured of 1 transmit block and 2 receive blocks.

Example: suppose you set the transmit limit to 25 blocks on channel 2. Assume so far that you have 22 blocks filled. Also the same condition exists on other channels to the extent that all memory has been allocated. You go to transmit on channel 2 and find that the transmitter is not ready, i.e. txrdy is low. As soon as any memory block in the system is returned to the free pool, this condition will clear itself.

The same is true of the Receiving subsystem. When a character comes in, it is put into the queue. If a new block must be allocated and there is no free memory, the Reserve block for this channel is allocated. DTR is negated or xoff sent if the handshaking is enabled. As soon as a block of free memory is available in the system, it will be allocated to the channel and the reserve block restored and handshake enabled.

Receive memory blocks are released to the free pool as you receive characters or do a queue flush. Transmit memory blocks are released to the free pool as the characters in them are transmitted or if you do a queue flush.

Command 1Eh, Data 05h
Get TX Blocks Limit

AL=5 returns the transmit limit on the CSC. The default is 6 on the PCSS–8FA/FX with 32K of Dynamemory.

Command 1Eh, Data 06h
Get RX Blocks Limit

AL=6 returns the receive limit on the CSC. This is also the highwater mark (HWLB). The default is 10 on the PCSS–8FA/FX with 32K of Dynamemory.

Command 1Eh, Data 07h
Get RX Low Water Block Limit

AL=7 returns the receive low water block limit on the CSC. (LWLB). The default limit is 8 on the PCSS–8FA/FX with 32K of Dynamemory.

Command 1Eh, Data 08h
Get Total # of Blocks Allocated

AL=8 returns the global total number of blocks in use. (i.e. the sum of all allocated blocks.)

Command 1Eh, Data 09h
Get System Memory Size in 32K Blocks

AL=9 returns the system memory size in 32k blocks. (i.e. 1=32k)

AH=1Fh
Set Transmit Limit on CSC.

DH=MSB of # of blocks and AL=LSB of # of blocks to set for the transmit limit on the csc. The default level is 10 and the minimum value is one on the PCSS–8FA/FX with 32K of Dynamemory.

AH=20h

Set Receive Limit on CSC.

DH=MSB of # of blocks and AL=LSB of # of blocks to set the receive limit on the csc. The minimum value is 2. This is also the high water limit (HWLB) . If handshaking is enabled, it will be asserted (DTR goes low or XOFF sent) when HWLB is reached. At this point, what is known as the RESERVE block for this channel is allocated. When this happens, 168 more characters can be received before an overflow will occur. This is true whether or not handshaking is enabled. The RESERVE block is allocated to this channel any time there are no free blocks available or the receive limit will be reached as a result of the block allocation.

AH=21h

Set Receive Low Water Mark.

DH=MSB of # of blocks and AL=LSB of # of blocks to set the receive low water mark on the csc (LWLB). There is a minimum value of one. If some form of handshaking is enabled and if HWLB is reached (upon which DTR goes low or XOFF is transmitted), then the re–enabling of the handshake signal (raising DTR or sending XON) will not occur until LWLB is reached.

IF the sum of all the transmit and receive limits exceeds the available system memory then there is the possibility that when a queue attempts to allocate a memory block, there will be none available. If this happens, the handshake flow control will take place (dtr low or xoff sent) and the RESERVE block will be allocated to the channel. Each receive channel keeps a reserved block in case this condition occurs. If this happens, then the flow control will be re-enabled as soon as system memory becomes available. That is the dtr high or XON will occur as soon as the memory shortage clears rather than wait until the low water mark is reached.

AH=2Ch
Set Extended Baud Rates.

This function call allows for more flexibility in selecting baud rates than does function call 0. You must execute function call 00h before function 2ch. Otherwise the function call 00h will change the baud rate to what is specified in it. Bits 7–4 of AL sets receive, bits 3–0 sets transmit baud rates. Enter with AL high and low nibbles with as follows:

Nibble Data (hex)	Baud Rate Standard	Baud Rate Extended	
0	75	7,200	
1	110	880	see #2 below
3	150	14,400	
4	300	28,800	
5	600	57,600	
6	1,200	115,200	
7	2,000	2000	
8	2,400	57,600	
9	4,800	4,800	
A	1,800	14,400	
B	9,600	9,600	
C	19,200	19,200	
2	38,400	38,400	Note order of Nibble...
D	57,600	57,600	
E	reserved	reserved	

Special Case AL=0FFh means to TOGGLE between the Standard and Extended sets...

The PCSS–8FA/FX defaults to the standard baud rate set. Function call 2Ch with AL=0FFh will toggle between the Extended Baud Rate Set and the Standard Baud Rate Set.

AH=2Dh
Unlock Xon/Xoff Condition.

This function may be useful if auto Xon/Xoff mode is being used. On entry, AL=0 to force Xon to be sent, allowing sender to transmit to us. This clears any pending Xon which would be sent when the low water mark is reached if an Xoff had been sent. On entry, with AL=1 clears Xoffed condition. This will allow us to transmit if we have been Xoffed by some remote party. The remote party may not like us however. See function 40 for possible error return codes. No error if AH bit 7 is clear.

AH=2Fh
Event Queue Read.

AL=don't care (on entry). This function returns to the user the oldest of the events which the board was told to detect with function call 30. The event ID is returned in **AL**. If **AL** is 0FFh upon return, then no events have occurred. If an event has occurred, then it is removed from the event first in first out buffer and returned in **AL**. The events are described previously in command 2Fh. Typical events are the occurrence of TXRDY, RXRDY, TXEMP, RXHSO (receive level reaches high water mark.) If one of the above events was detected, then the bottom three bits of the ID indicate the channel upon this board in which the event occurred.

Using events allows the PCSS–8FA/FX to drive your application software rather than your having to poll for them. For example, if you are receiving on several channels and storing information in different places, you could use the global rxrdy poll or enable the event marker system on all of the channels upon which you are waiting for data. Using the event queue instead of the global rxrdy poll will allow you to service the ports in the order in which the rxrdys became ready. Also if you went to transmit one of these received characters and found that the transmitter was not ready, you could enable the TXRDY event for that channel, save the character you want to transmit and go back to your receive routines. When that transmitter becomes ready, the event ID will be put into the event queue. This could cause your software to come back and finish the task of transmitting the character.

AH=30h
Enable Event Detection.

AL=event ID, **DL**=channel. Enable Event detection. Enter with **DL** being any one of the channels upon this board and **AL** with the event ID of the event you wish to detect.

Event	ID for channel 0–7
Transmitter Empty	00—07h
Receive Data Available	10—17h
Receive Buffer at Limit	20—27h
Transmitter Ready	30—37h
Received Character Match	40—47h
Receive Queue Reached Set Level	50—57h

Refer to the table above or Command 30h in Appendix A for the event ID to use.

NOTE: Once the event occurs and is detected, it is then disabled, except for types 40–47h. Call this function again to re–enable the event. Types 40–47 need only be enabled once.

AH=3Bh
Return Number of Events Pending.

AL=don't care, **DL**=any channel on a particular board (to select the board in question). Upon return **AL** equals the number of events which are pending. On entry, **DL** must be set to any port upon the PCSS–8FA/FX board being interrogated for this information.

AH=40h
Get Extended Error Code.

If AH bit 7 is set upon the return from any function call, an error has occurred. If you want more information about it, make a function call 40h before any other INT14h function call. DX must be set the same as on the function call where the error occurred. AL will be returned as follows:

Bit 7 = 1 means	time–out error from function call 1 or 2.
Bit 6 = 1 means	illegal, non–existent function call.
Bit 5 = 1 means	port (board) does not exist.
Bit 4 = 1 means	board not ready fault. (Call GTEK)
Bit 3 = 1 means	illegal parameter passed to function call
Bit 2	reserved
Bit 1	reserved
Bit 0	reserved

AH=41h
Un–install RES8F.

If you execute this function call, RES8F is removed from memory. The function call must be made with AL=0abh and DX=0cdefh. This function is used by RES8F with the U option. It is not one that you will normally use.

—NOTES—

Appendix—C

Typical Rs-232 Hook-ups

The PCSS-8FA/FX is a DTE type device. Hook-ups to DCE devices run straight through, name to name: TXD—TXD, RXD—RXD, etc.

Modems like Hayes Stack, Novation, etc.

Wires run straight through (to DCE):
TXD–TXD or 2–2
RXD–RXD or 5–3
CTS–CTS or 1–5
CD–CD or 3–8
DTR–DTR or 6–20
SG–SG or 4–7

Serial Printers like Epson MX 100, NEC 7700, Brother, Okidata, and Anadex.

Wires run crossed (to DTE)
TXD–RXD or 2—3
RXD–TXD or 5—2
DTR–CTS or 6—5
CTS–11 or 1—11 (CTS—19 or 1—19)
CD–DTR or 3—20
SG–SG or 4—7

Serial Printers like Qume:

Wires run crossed (to DTE)
same as above except
CTS and CD–DTR (5 and 3—20)

Slow CRT and printers (old):

Wires run crossed (to DTE)
TXD–RXD or 2—3
RXD–TXD or 5—2
DTR–CTS or 6—5
CTS and CD–DTR or 1 and 3—6
SG–SG or 4—7

Diablo 620 Printer:

Wires run crossed (to DTE):
same as slow printer except
CD–DTR or 3—20
DTR–DSR or 6—6

Diablo 630 printer:

Wires run crossed (to DTE):
same as slow printer except
CTS–11 or 1—11
CD–DTR or 3—20 (CD–RTS or 3—4)

MX-80, IDS:

Wires run crossed (to DTE):
same as slow printer

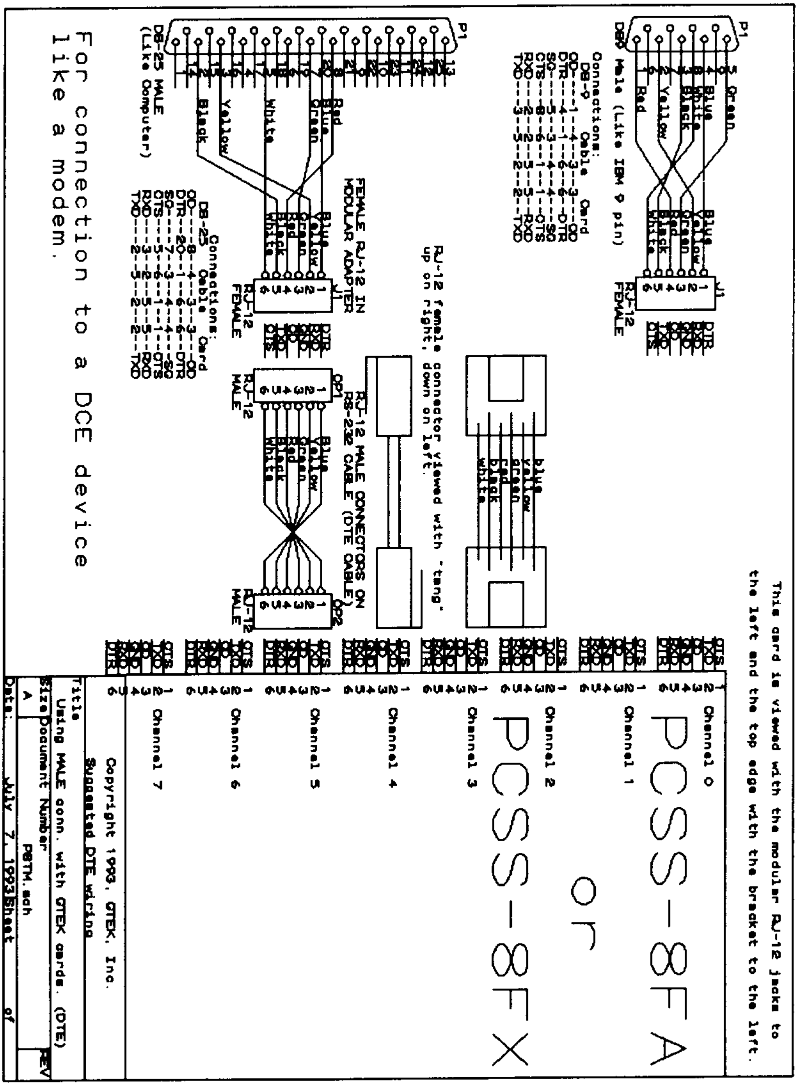
SCTP–1:

Wires run crossed (to DTE):
same as slow printer except
CTS–RTS or 1—4
CD–DTR or 3—20

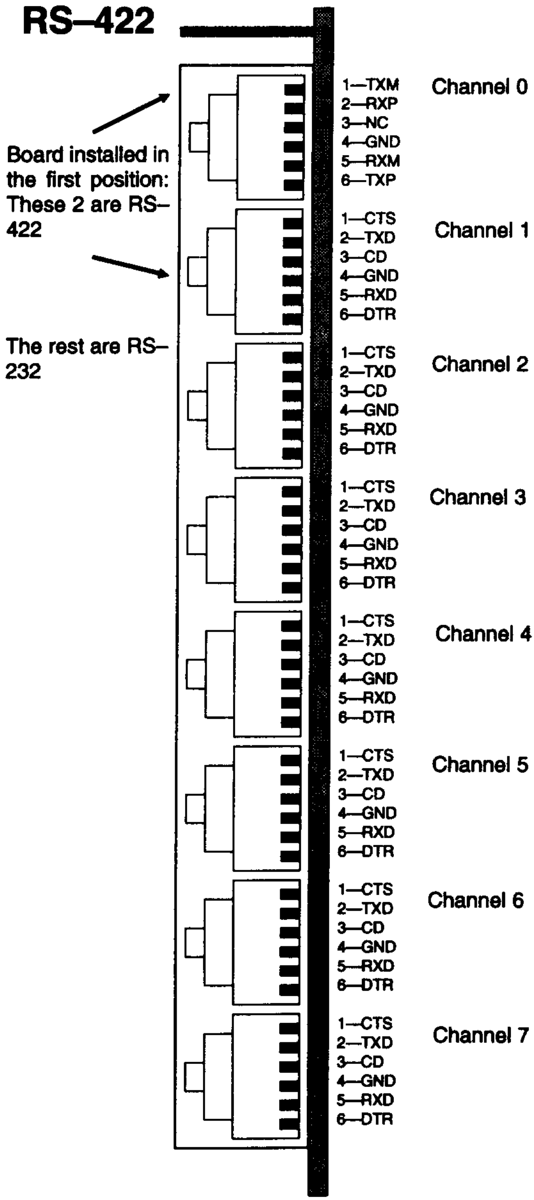
Hewlett-Packard, Houston Instruments Plotters:

Wires run same as slow printer (to DTE)

—Notes—



C.1 Wiring of P8TM (male DB-25 at end of wire)



On a PCSS–8FA/FX type card, the chips from top to bottom are 1488, 1489, 1488. To install a RS–422–2 card, remove a pair (1488/89) **BEGINNING AT THE TOP** and insert the RS–422 card where the **1488/1489** (not 1489/88) was. This converts 2 channels at a time to RS–422.

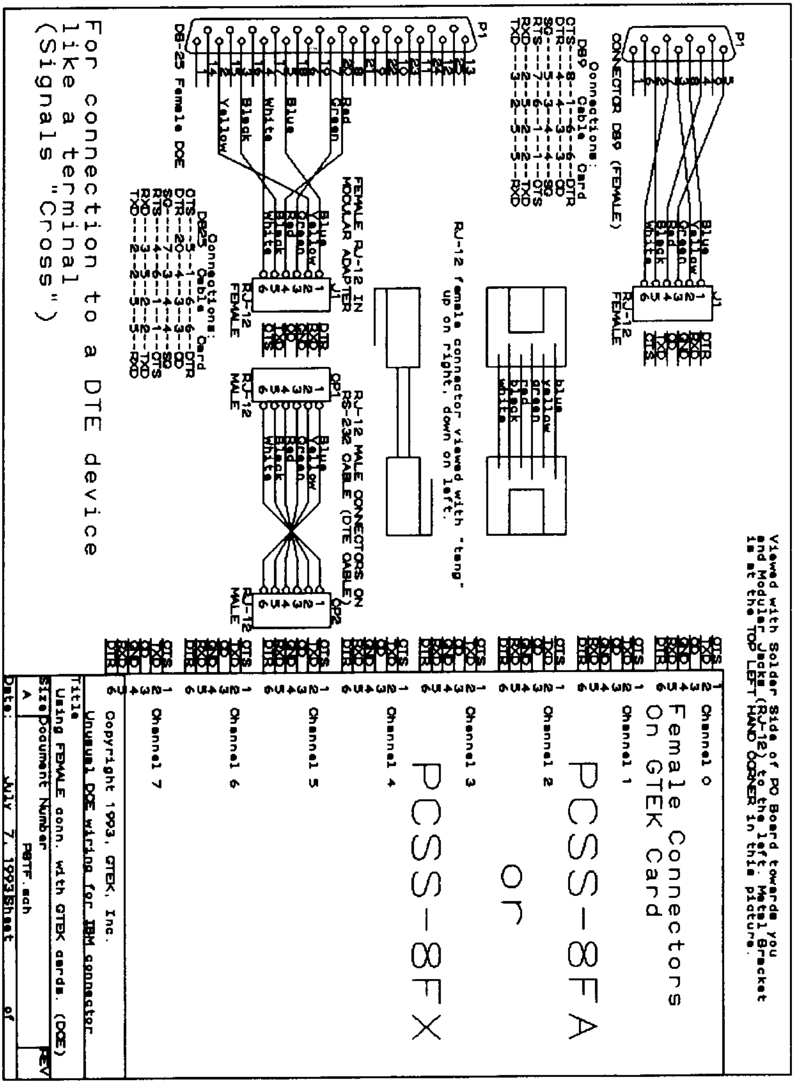
There is only one possible way to insert the RS–422 card due to the way it's manufactured. Just make sure that the correct "pair" of 1488/89 is selected. The 1488/89 pair controls 2 channels at a time, so if you insert the RS–422–2 card in place of the top 2 chips, this converts channel 0 and 1 to RS–422.

The pin numbers are:

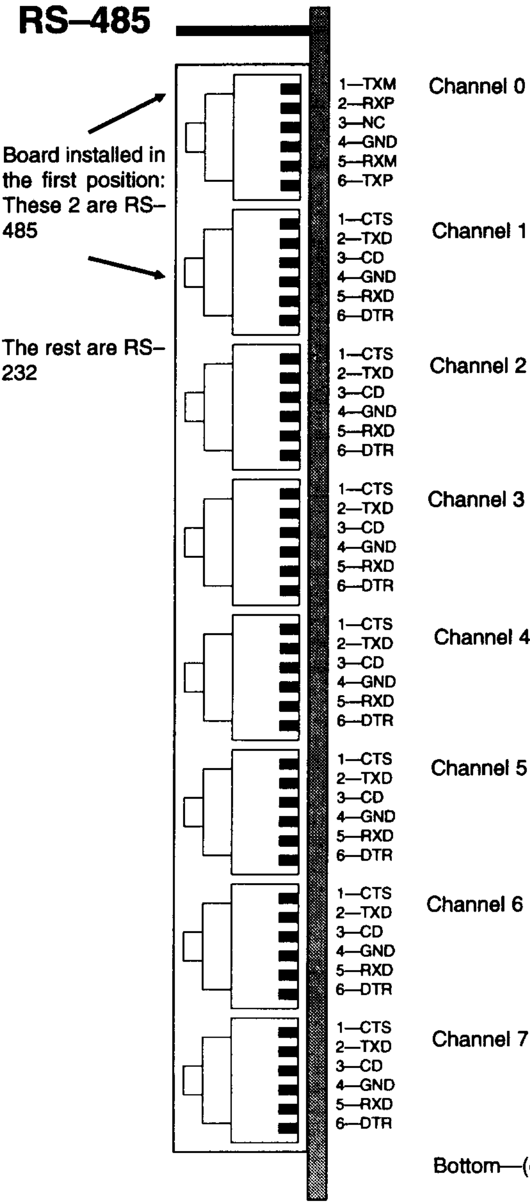
1—Transmit MINUS
2—Receive PLUS
3—No Connection
4—Ground
5—Receive MINUS

6—Transmit PLUS

Figure C.2 Installation of RS–422–2 board.



C.3 Wiring of P8TF (female connector at end)



On a PCSS-8FA/FX type card, the chips from top to bottom are 1488, 1489, 1488. To install a RS-485 card, remove a pair (1488/89) **BEGINNING AT THE TOP** and insert the RS-485 card where the **1488/1489** (not 1489/88) was. This converts 2 channels at a time to RS-485.

There is only one possible way to insert the RS-485 card due to the way it's manufactured. Just make sure that the correct "pair" of 1488/89 is selected. The 1488/89 pair controls 2 channels at a time, so if you insert the RS-485 card in place of the top 2 chips, this converts channel 0 and 1 to RS-485.

The pin numbers are:

1—No connection
2—Transmit/Receive PLUS
3—No Connection
4—Ground
5—Transmit/Receive MINUS

6—Transmit PLUS

On the PCSS-8FA/FX, automatic DTR control is enabled with command 19h followed by data 20h.

Figure C.4 Installation of RS–485 adapter

—NOTES—

Appendix—D

General Interfacing Examples

You can use as many PCSS–8FA/FX's as you have room for in a computer so possibly as many as 130 channels can be used simultaneously with RES8F.COM. Put the command "RES8F" in your autoexec.bat file.

RES8F adds many new function calls to BIOS INT14H. Access to INT14h from Quick Basic Versions 2.0 and 3.0 is gained thru USR.OBJ, which must be linked to your Quick Basic program. The 4.xx interface is different only in the way that it is linked to the main program. Visual Basic for DOS as well as PDS V7 etc. are similar to QuickBASIC version 4.5 in the way that QLB and LIB files are made. OBJ as well as QLB and LIB files may already be provided on disk.

USR.OBJ contains a function which can be called from your Quick Basic program. Within your basic program you "Call res(ax,dx)" where ax and dx are integer variables. They are passed to and from BIOS INT 14h in the AX and DX registers.

Simple Basic Example:

```
REM See the disk provided with this card for newer versions of
REM this REM program
A$="A"
TRANSMIT_FUNC = 1 * 256 : REM AH=1 TO XMIT
AX% = TRANSMIT_FUNC + ASC(A$)
REM load AH with 1, AL with the ascii code for 'A'
CHANNEL = 5 * 256         'DH gets the channel #
XCOM2=&HE4                'DL gets base address
DX% = CHANNEL + XCOM2     'channel 5 on COM2
CALL RES(AX%,DX%)         'Transmit the character 'A'
```

See the QUICK BASIC example program EXAMPLE.BAS for more information.

The following is an example of what you can do with int 14 in machine code. In this case we wanted to run at a baud rate not available using function call 0, so we programmed the baud rate divisor registers directly.

```
setup:                          ;this uses bios 14h fc
      mov   dx,0501h            ;channel 5, com2
      mov   al,00101010b        ;9600 baud, even parity ,1s, 7b
      mov   ah,0
                                ;function 0 - initialize (IBM style)
      int   14h                 ;Next modify baud rate
      mov   ah,17h              ;read register
      mov   al,3                ;for xFBh to read LCR
      int   14h
      or    al,80h              ;set dlab
      mov   ah,13h              ;for xFBh to write LCR
      int   14h                 ;now dlab is high
      mov   al,1                ;115,200 bits per second!
      mov   ah,10h              ;for xF8h, write LSB of baud.
      int   14h                 ;we know that MSB of baud
                                ;is already 0. now  set
                                ;DLAB low for norm operation
      mov   ah,17h              ;read register function
      mov   al,3                ;for xFBh, LCR
      int   14h
      and   al,7Fh              ;clear dlab
      mov   ah,13h              ;write register funct for xFBh
      int   14h                 ;now dlab is LOW
      mov   al,1                ;enable interrupt drive.
      mov   ah,5          ;function call 5
      int   14h
```

D

```
    ret
```

—USR.OBJ Program—

```
;     COPYRIGHT 1986, 1987, GTEK, INCORPORATED
;           ALL RIGHTS RESERVED, WORLDWIDE
;                 USR.ASM
;A program to enable QuickBasic to interface with the Serial
;Port through the use of the RES14.COM TSR program
;to replace the INT14 IBM Bios driver. See EXAMPLE.BAS for
;usage with a machine language program through QuickBasic
CODE  SEGMENT     PARA PUBLIC 'CODE'
      ASSUME      CS:CODE,DS:CODE
      PUBLIC      RES
RES   PROC  FAR
      PUSH  BP          ;save frame reference
      MOV   BP,SP       ;this has to be done first
      PUSH  AX          ;save the registers used.
      PUSH  BX
      PUSH  DX
      MOV   BX,[BP]+6   ;Point to DX% from basic
      MOV   DX,[BX]     ;Store Index register
      CMP   DL,0C0H     ;Check for illegal base addr
      JNC   RSX
;here if using base address 0 or 1
;assumes standard I/O address at 2E0 or 2E4
      CMP   DL,0        ;see if should be 2E0
      JZ    RSX0        ;brif, should be 2E0
      MOV   DL,0E4H     ;com=1 make base=2E4
      JMP   RSX         ;Branch to continue
RSX0: MOV   DL,0E0H     ;com=0 make base =2E0
RSX:  MOV   BX,[BP]+8   ;Point to AX% from basic
      MOV   AX,[BX]     ;Store in AX
      INT   14H         ;Int 14 bios function call
      MOV   [BX],AX     ;get from AX to basic AX%
      POP   DX          ;dx doesn't change
      POP   BX
      POP   AX
      POP   BP
      RET   4           ;FIX STACK TO RETURN TO QB
RES   ENDP
CODE  ENDS
      END
```

QuickBasic example

```
'* * * * * * READ THE FOLLOWING FIRST * * * * * *
'You must first build a library file to use USR from the
'debug mode of QuickBasic. Use BUILDLIB to create
'a USERLIB.EXE FILE that contains USERLIB.OBJ
```

```
'and USR.OBJ
'QuickBasic V2 and V3
'      C>BUILDLIB USERLIB.OBJ USR.OBJ;↵
'then to invoke the program to run in debug mode:
'      C>QB example /L userlib.exe↵
'QuickBasic V4.xx (V4.0 has bugs, be careful)
'To prepare a QuickBasic library to use in debug
'mode. You can't use the userlib.exe file created in
'the previous example since QB4 is an entirely new
'version of Quickbasic.
'To make a qb library file to use in debug mode of
'QB assuming you use our USR.OBJ and QB V4.0b
'make sure your commas are all there and in proper order
'If they aren't, you will probably destroy your bqlb41.lib file
'In the example below, substitue the correct library name for
'your version of QB, PDS or VBDOS, or if you have a LIB=path
'variable set in your environment, just put a semicolon:
'C>LINK /Q USR.OBJ, USRA.QLB, , bqlb41.lib;↵
' or C>link /q usr.obj,usra.qlb;
'Then if you don't already have a library named usra.lib:
'C>LIB USRA.LIB+USR.OBJ;↵

'You make the LIB file also so that when you compile
'your program to run stand alone, the LIB file will
'be present to compile with. Erase any USRA.LIB before doing
the
'example above...
'Finally

'C>qb example /L usra.qlb↵

'to run.
'To make stand alone EXE file from within QB4,
'select the RUN menu and make a stand alone EXE
'by selection of the right menu options. Remember
'this won't work if you didn't make the USR.LIB file.

'You can also make a stand alone program from DOS
'Read chapter 8 in the Learning and Using Microsoft QuickBASIC
manual.
'You can then run QuickBasic to run the example or your
'own program You have to load the user library before you
'can run QB in the debug mode
'Study what this program does before you run it.
'Simple dumb terminal demonstrates use of pcss8i
'in a simple situation

DEFINT a-z  'all variables used in SUB must be short integer!
DECLARE SUB res(AxRegister,DxRegister)
```

```
CONST false = 0, true = NOT false
CLS
'set up channel. use channel 2 with auto xon/xoff handshake
'for now
AxRegister = &H23        '19,200 8 n 1     binary=0010 0011b
DxRegister = &H82        'channel 2        binary=1000 0010b
HsRegister = &H55A 'Xon/xoff rx and tx.    binary=0101 1010b
PRINT "Setting Parameters"
CALL res(AxRegister,DxRegister)     'set baud/params
CALL res(HsRegister,DxRegister)     'set handshakes
'remove next section to ***** if you aren't using a
'GTEK programmer
'On GTEK programmers, break sets the baud rate. When you
'send the 80h character
BrkonRegister = &H901   'see function calls set break
BrkoffRegister = &H900  'restore break
SetbaudRegister = &H180 'transmit 80h character
PRINT "Sending BREAK to GTEK programmer."
CALL res(BrkonRegister, DxRegister)
FOR BrkTimer = 1 to 20000: NEXT BrkTimer  '100 ms
PRINT "Break Off."
CALL res(BrkoffRegister, DxRegister)
FOR BrkTimer = 1 to 2000: NEXT BrkTimer   '10 ms
PRINT "Sending 80h."
CALL res(SetbaudRegister, DxRegister)          'Set baud rate
'Remove to here if not using GTEK programmer
'**********
PRINT "Dumb Terminal."
DO
      a$=INKEY$                    'get a character
      a$=UCASE$(a$)                'uncase it
      IF a$="" THEN                'receive a character
            AxRegister = &HB01     'get # chars
            CALL res(AxRegister, DxRegister)
            IF AxRegister >0 THEN   'something ready
                  'get all available characters
                  AxRegisterold = AxRegister
                  FOR i=1 to AxRegisterold
                        'get 1 available character
                        AxRegister = &H200
                        CALL res(AxRegister,AxRegister)
                        'extract character from AxRegister
                        al = AxRegister - (AxRegister\256)*256
                        lcr$ = chr$(al)
                  'if it is line feed, don't need to print it
                        IF al<>10 THEN
                              PRINT lcr$;
                        END IF
                  NEXT i
            'notice that characters are received from 1 check
```

```
            'of the buffer. Other characters could still be
            'coming in at the same time.
            END IF
      ELSE IF a$ = CHR$(1) THEN      'control-A switches channel
            PRINT: PRINT "Old channel = ";
            IF DxRegister = &h88 THEN      'switch to channel 0
                  DxRegister=&H80
            END IF
            PRINT "New Channel=";DxRegister AND 7
      ELSE
            IF A$<>CHR$(3) THEN      'transmit character
                  DxRegister = &H100 + ASC(a$)
                  CALL res(AxRegister,DxRegister)
            END IF
      END IF
LOOP WHILE a$<>CHR$(3)  'abort on control-C from keyboard
PRINT
PRINT "DONE..."
AxRegister = &HE3
HsRegister = &H500
CALL res(AxRegister, DxRegister)
CALL res(HsRegister, DxRegister)
END
```

Index

A

B

C

D

E

F

G

P

R

S

T

W

Printed July 1993

GTEK®, INC.

Development Hardware and Software
P. O. Box 2310
399 Highway 90
Bay St. Louis, MS 39521–2310 USA