PCM-3641
4-port RS-232 High-Speed Module

Introduction

The PCM-3641 is a PC/104-compatible 4-port High-Speed RS-232 serial interface module. It works with PC/104 CPU modules or CPU cards which accept PC/104 expansion modules. It provides four independent serial interfaces, accessed through male DB-9 connectors.

The module's industry-standard 16C550 asynchronous communication chip is fully programmable. The module requires no special commands or control codes if you use the standard COM1 ~ COM4 port addresses.

Features

- Four independent RS-232 serial ports
- Transmission speeds up to 460 Kbps
- Independent/Shared IRQ settings for each of 4 serial ports
- Wide IRQ selection: 3, 4, 5, 6, 7, 9, 10, 11, 12, 15
- Supports Standard DOS COM1, COM2, COM3, and COM4
- Supports DOS/Windows 3.1 (PC-ComLib), Windows 95/98/NT
- LED indicators: TX, RX
- Compatible with standard PC ports: COM1, COM2, COM3, COM4
- On-board interrupt status register for greater throughput
- Complete RS-232 Modem-control signals

Specification:

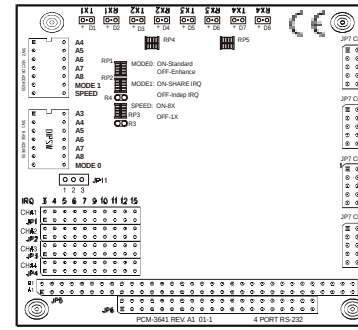
- Bus interface: ISA
- Number of ports: 4
- I/O address: 0x0200 ~ 0x03F8
- IRQ: 3, 4, 5, 6, 7, 9, 10, 11, 12, 15
- Data bits: 5, 6, 7, 8
- Stop bits: 1, 1.5, 2
- Parity: none, even, odd
- Speed (bps): 50 ~ 460 K
- Connectors: Four DB-9 male
- Data signals: TxD, RxD, RTS, CTS, DTR, DSR, DCD, RI, GND
- Power requirement: +5V@200 mA (Typical) ;
  +5V@ 250 mA (Max.)
- Clock input: 14.7456 MHz
- Operating Temperature.: 0 ~ 65° C (refer to IEC-68-1-1, 2)
- Storage Temperature: -25 ~ 80° C

Initial inspection

We carefully inspected the PCM-3641 both mechanically and electrically before we shipped it. It should be free of marks and scratches and in perfect electrical order on receipt.

Handle the board only by its edges. The static charge on your body may damage its integrated circuits. Keep the card in its anti-static package whenever it is not installed. You can use this package to return the card if it should need repair.

Board Layout



Card Configuration

Each port on the PCM-3641 card has jumpers and 2 DIP switches which require configuring prior to use. The DIP switches set the port to the appropriate I/O address and different modes. The jumpers set the port's IRQ.

Part no. 2000364100 1st Edition Printed in Taiwan August 2000

Default Settings

The board is shipped with default settings. If you need to change these settings, however, see the following sections. Otherwise, you can simply install the card.

PCM-3641 Default Configuration	
Setting	**Default Function**
JP1	IRQ 5
Speed Mode	1 x
IRQ mode	Share
Base Address	Address 300H
Vector Address	Interrupt 280H
Standard/ Enhanced	Enhance
Operating System	Windows 95/98/NT

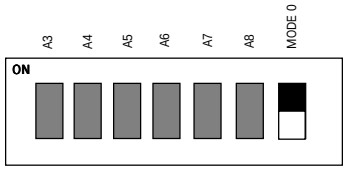
Jumper and switch settings

The PCM-3641 can function in many different modes according to your application needs. These modes include Standard/Enhance Mode, Independent/Shared IRQ Mode, Speed Mode and Operating System Mode. Details of these specific modes are described as follows:

Standard/Enhanced Mode (MODE0)

The PCM-3641 can be used in standard or enhanced mode. In standard mode the I/O addresses are compatible with the standard PC communication ports, COM1~COM4. In enhanced mode you can select a different base address. The offset of each port from the base address is fixed. You can use MODE0 of DIP switch SW1 to select standard or enhanced mode.

Standard mode



SW1

In standard mode, the I/O address of the ports are as follows:

Port	I/O address	Interrupt No.
Port 1	3F8	Selectable
Port 2	2F8	Selectable
Port 3	3E8	Selectable
Port 4	2E8	Selectable

Enhanced mode

SW1

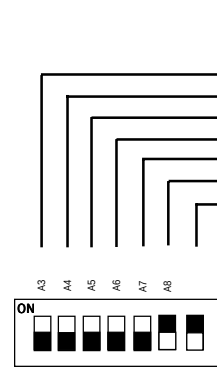
In enhanced mode, you can select a different base address. The base address determines the address for each of the four ports.

The I/O addresses for the four ports are as follows:

Port	I/O address
Port 1	Base + 00H
Port 2	Base + 08H
Port 3	Base + 10H
Port 4	Base + 18H

You use switches 1~6 of DIP switch SW1, a 7-position DIP switch, to set the base address. You can set the base address anywhere from hex 200 to 3F8.

To set the base address, you have to calculate the base address as follows:



Note: On the PCM-3641 the address line A9 does not appear on the DIP switch as it is permanently hard-wired to HEX 200 on the card.

The following table shows different base address settings.

Port base address (SW1)						
Base Address	**A3**	**A4**	**A5**	**A6**	**A7**	**A8**
200 - 207	●	●	●	●	●	●
208 - 20F	○	●	●	●	●	●
*300 - 307	●	●	●	●	●	○
3E8 - 3EF	○	●	○	○	○	○
3F8 - 3FF	○	○	○	○	○	○

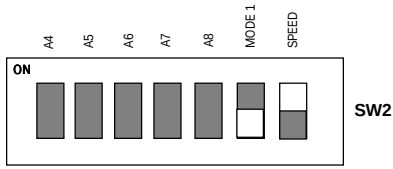
●: ON ○: OFF ***** : Default

Note: If your CPU module or card has serial interface ports, you will need to adjust the I/O port addresses (or disable the ports) to avoid conflicts.

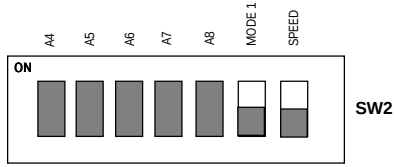
Independent/Shared IRQ mode (MODE1, JP1, JP2, JP3, JP4)

The card's IRQ can be set up using MODE1 of SW2. Please note that the DIP switch is for setting the mode as shown below.

Shared IRQ Mode



Independent IRQ Mode



Independent IRQ Mode (JP1-JP4)

In this mode, each of the four ports can have IRQ channels set individually. For each port, select an IRQ which is not already in use by another card in the system. The map of jumpers and ports is shown below.

Port 1 —> JP1

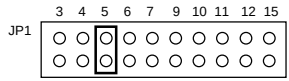
Port 2 —> JP2

Port 3 —> JP3

Port 4 —> JP4

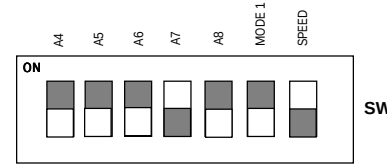
Shared IRQ Mode (JP1)

Select an IRQ which is not already in use by another card in the system. If you are installing more than one PCM-3641, set them to different IRQ numbers. Jumper Bank JP1 controls the card IRQ. Simply place the jumper on the required interrupt level as shown in the following figure.



Interrupt Status Register Setup (SW2, Vector address)

This feature on the PCM-3641 is utilized in the shared IRQ mode. When data arrives at one of the four ports, it will generate an interrupt in the interrupt register. The PC software can read this, and identify immediately which port generated the interrupt. This saves time, and makes programming easier. When a data bit of the interrupt status register is set to 0, the corresponding channel is selected to generate an interrupt. If the bit is 1, then no interrupt is generated. DIP switch SW2 controls the card's interrupt status register, as shown in the following figure and table.



Interrupt Status Register SW2	
Bit	**Function**
0	Port 1
1	Port 2
2	Port 3
3	Port 4
4	Not Used
5	Not Used
6	Not Used
7	Not Used

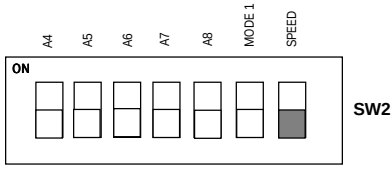
The user may change the interrupt status address via SW2. Please note that the address decoder will occupy a continuous, 16-byte area related to the switch setting. For example, if you set the switch to 210H, then the address 210H to 21FH will all be decoded. The various DIP switch settings (SW2) for the interrupt status register are as shown in the table opposite.

A4	A5	A6	A7	A8	Interrupt Register
ON	ON	ON	ON	ON	200H
OFF	ON	ON	ON	ON	210H
ON	OFF	ON	ON	ON	220H
OFF	OFF	ON	ON	ON	230H
ON	ON	OFF	ON	ON	240H
OFF	ON	OFF	ON	ON	250H
ON	OFF	OFF	ON	ON	260H
OFF	OFF	OFF	ON	ON	270H
ON	ON	ON	OFF	ON	280H
OFF	ON	ON	OFF	ON	290H
ON	OFF	ON	OFF	ON	2A0H
OFF	OFF	ON	OFF	ON	2B0H
ON	ON	OFF	OFF	ON	2C0H
OFF	ON	OFF	OFF	ON	2D0H
ON	OFF	OFF	OFF	ON	2E0H
OFF	OFF	OFF	OFF	ON	2F0H
ON	ON	ON	ON	OFF	300H
OFF	ON	ON	ON	OFF	310H
ON	OFF	ON	ON	OFF	320H
OFF	OFF	ON	ON	OFF	330H
ON	ON	OFF	ON	OFF	340H
OFF	ON	OFF	ON	OFF	350H
ON	OFF	OFF	ON	OFF	360H
OFF	OFF	OFF	ON	OFF	370H
ON	ON	ON	OFF	OFF	380H
OFF	ON	ON	OFF	OFF	390H
ON	OFF	ON	OFF	OFF	3A0H
OFF	OFF	ON	OFF	OFF	3B0H
ON	ON	OFF	OFF	OFF	3C0H
OFF	ON	OFF	OFF	OFF	3D0H
ON	OFF	OFF	OFF	OFF	3E0H
OFF	OFF	OFF	OFF	OFF	3F0H

Speed Mode Selection

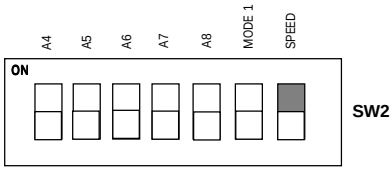
The PCM-3641 employs a unique speed option that allows the user to choose either normal speed mode (1x) or high speed mode (8x). This high speed mode is selected at SPEED of SW2.

Normal Speed Mode



To select the baud rate commonly associated with COM ports , such as 2400, 4800, 9600. . .115.2 Kbps, place the switch as follows.

High Speed Mode



To increase normal mode rates up to eight times, (e.g. if 115.2 Kbps is selected, the rate can be increased up to 921.6 Kbps), place the switches as follows.

Operating Environment Selection

Set jumper 11 (JP11) to correspond with your desired software operating environment. Connect the left two pins of JP11 to operate in DOS or Windows 3.1 mode, as shown below. Connect the right two pins to operate in Windows 95/98/NT mode.

DOS and Windows3.1 Windows 95/98/NT

Driver Installation for DOS Users

Make a duplicate copy of the driver diskette in case the original disk becomes lost or damaged. Copy the files to a subdirectory on your hard disk if you wish.

Card setup

The PCM-3641's driver determines the configuration of the installed cards by reading a data file, GEN-DRV.CNF. When you first install the PCM-3641, and each time you change the card's address and IRQ, you will need to run the card setup program to save the settings to the configuration file. Program files should be installed to the hard disk. Insert the driver disk in your computer, type DOSINST from the A: (or B:) prompt and press enter. Once the files have been installed, type SETUP from the \COMLIB\BIN prompt and press ENTER. You will then see the screen on the following page.

After the screen shows up, move the cursor bar (using the arrow keys or the mouse) to the general serial board field and press ENTER. The screen shown below will appear.

When you are finished setting up the ports, press the ESC key to return to the previous windows. Press F10 to save the new configuration or ESC to quit without saving. The setup program will then create a new configuration data file GEN-DRV.CNF.

communications under Windows 3.x . It is a simple example program capable of sending and receiving data after each port is opened with selected communication parameters. As Windows 3.x features multitasking, multiple windows for the ports can appear simultaneously under TTY. However, Terminal, the application provided by Windows is limited for the use of COM1 to COM4.

After completing the installation, restart Windows. An additional line, "comm.drv=sercomm.drv", will appear for the PCM-3641 in the [boot] section of the Windows SYSTEM.INI file. In addition, a Windows group "PC-ComLIB Standard COMM Driver" will be generated for reconfiguration, driver removal, etc. At this point, you are ready to execute applications that support Windows COMM API calls.

ICOM Utility Setup for Windows 95/98/ NT Environments

This section discusses the ICOM utility software package installation, configuration and upgrade/ removal procedure for the Windows 95/98 and NT environments.

Utility Installation

Follow the installation procedure below to install the PCM-3641 under Windows 95/98/NT:

1. Run **Setup.exe** on the driver diskette.
2. Select "**Advantech Icom Utility**" to install and configure the board, following the on-line instructions.
3. After the Advantech Icom Utility configuration panel pops up, please refer to the software help file for more details.
4. Following completion of the installation, restart Windows 95.

Following completion of installation, please restart your system as prompted.

Once the board and driver have been installed and the system restarts successfully, users can execute any ready-made applications, such as HyperTerminal to transmit/receive data, or Remote Access Service to provide dial-up networking capability.

Configuration:

Enter the configuration program to install the device driver, or click the Taskbar [**Start**] button, then select the [**Programs**] menu, then the [**Advantech Icom Utility**] menu and then [**Icom**].

When the configuration panel pops up,

click the [**Add Board**] button to add a board.

Click the [**Delete**] button to remove a board.

Board Type: PCM-3641

Base COM: Specifies the COM number of the first port. Subsequent ports are mapped to subsequent COM numbers. For instance, if the first port is mapped to COM10, then the second port is mapped to COM11 sequentially.

Base Address (200H~3F8H): Specifies the base address of the first port. Subsequent base addresses are mapped to subsequent COM numbers. For instance, if the first port is mapped to 300H, then the second port is mapped to 308H sequentially.

PCM-3641 series cards can be installed together in a single system as long as the system memory resources are sufficient and available in a system. Different boards should be assigned different IRQs.

Click the [**Share IRQ Enable**] button to set the share IRQ function.

Share IRQ: 3, 4, 5, 6, 7, 9, 10, 11, 12, 15

Vector Address: 200H ~ 3F0H

After you finish the installation, you can click [**Exit**] and restart your system. Unless the system is restarted, the latest configuration will not take effect.

Jumper Setting

STANDARD/ ENHANCED Mode

MODE 0 of **SW1** is used to set the Standard/enhanced mode of this card.

MODE 0: ON (Upper) position —> **STANDARD** mode

MODE 0: OFF (Lower) position —> **ENHANCED** mode

STANDARD Mode:

In this mode, The I/O Addresses and its IRQ Level for each port are set to default as shown below, (Disable BIOS setting of on-board COM1 ~ COM4).

Port No.	I/O Address	COM Port No.	IRQ Level (*)	
			Independent IRQ	Shared IRQ
Port 1	3F8h	COM1	JP1	JP1
Port 2	2F8h	COM2	JP2	JP1
Port 3	3E8h	COM3	JP3	JP1
Port 4	2E8h	COM4	JP4	JP1

ENHANCED Mode:

In this mode, The I/O Addresses and its IRQ Level for each port are set to default as shown below, (Make sure that the I/O address on BIOS setting of on-board COM1 ~ COM4 will never conflict with [Base Address] ~ [Base Address + 20h])

Port No.	I/O Address	COM Port No.	IRQ Level (*)	
			Independent IRQ	Shared IRQ
Port 1	Base Address + 00h	COM1	JP1	JP1
Port 2	Base Address + 08h	COM2	JP2	JP1
Port 3	Base Address + 10h	COM3	JP3	JP1
Port 4	Base Address + 18h	COM4	JP4	JP1

- **IRQ Mode**

DIP 6 **(MODE 1)** of **SW2** is used to set the IRQ mode of this card.

MODE 1: **ON** (Upper) position →**Shared IRQ** mode

MODE 1: **OFF** (Lower) position → **Independent IRQ** mode

- **SPEED Mode**

SPEED of **SW2** is used to decide the speed mode of this card.

SPEED: **ON** (Upper) position → High Speed Mode

SPEED: **OFF** (Lower) position → **Normal Speed Mode**

- **Operating System Mode**

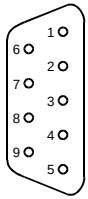
Connect the left two pins of **JP11** to use DOS, Windows 3.1

Connect the right two pins of **JP11** to use Windows 95/98/NT

Signal Wiring

Connector pin assignments

You access the PCM-3641's ports through four external male DB-9 connectors. RS-232 connector pin assignments are as follows :



Pin	description
1 DCD	receive line signal detector
2 RD	received data
3 TD	transmitted data
4 DTR	data terminal ready
5 GND	ground
6 DSR	data set ready
7 RTS	request to send
8 CTS	clear to send
9 RI	ring indicator

RS-232 Signal wiring

Since the RS-232 interface is not strictly defined, many devices have their own connection methods, which may ignore some signal lines or define reserved lines to other functions. It is best to refer to the user's manual for your device for installation instructions. You may find the following helpful.

In general, DTE (Data Terminal Equipment) refers to the device that is leading the communication. Examples include PC's, terminals and some printers. DCE refers to the device being communicated with or controlled. Examples include modems, DSU's (digital service units), printers and lab/factory equipment.

In some situations you may be able to get by with just three lines: data on TxD, a Signal Ground and a handshaking line. Examples are printer or plotter connections, troubleshooting and situations where you require only one-wire communication.

Terminal or PC (DTE) connections

PCM-3641 (DTE): (DB-9)		Terminal (DTE) : DB-25	
Pin	Signal	Pin	Signal
3	TxD	3	RxD
2	RxD	2	TxD
7	RTS	5	CTS
8	CTS	4	RTS
6	DSR	20	DTR
5	GND	7	GND
4	DTR	6	DSR
1	DCD	8	DCD

Modem connections

PCM-3641 (DTE): DB-9 male		Modem (DCE)	
Pin	Signal	Pin	Signal
3	**TxD**	**2**	**RxD**
2	**RxD**	**3**	**TxD**
7	**RTS**	**4**	**CTS**
8	**CTS**	**5**	**RTS**
6	**DSR**	**6**	**DTR**
5	**GND**	**7**	**GND**
4	**DTR**	**20**	**DSR**
1	**DCD**	**8**	**DCD**

For DTE to DCE connection, use straight through cable connections, i.e. you don't have to reverse lines 2 and 3, lines 4 and 5, and lines 6 and 20. Because in general DCE RS-232 interfaces are reversed themselves.

Terminal without handshake

PCM-3641 (DTE): DB-9 male		Terminal (DTE)	
Pin	Signal	Pin	Signal
3	**TxD**	**3**	**RxD**
2	**RxD**	**2**	**TxD**
7	**RTS**		
8	**CTS**		
6	**DSR**		
5	**GND**	**7**	**GND**
4	**DTR**		
1	**DCD**		

This completes the hardware installation.

Programming

The maximum length of a RS-232 cable is 100 ft. If you need to connect over longer distances, (longer than 100 ft), you will have to use another standard (like RS-422 or RS-485).

If you do not use CTS, RTS, DSR, DTR signals, please loop them back, otherwise the PC-ComLIB software will not function correctly. PC-ComLIB always checks for handshake signals.

Hardware Installation

TURN OFF your PC power supply whenever you install or remove the PCM-3641 or connect and disconnect cables.

Installing the module on a CPU card

1. Turn the PC's power off. Turn the power off to any peripheral devices such as printers and monitors.
2. Disconnect the power cord and any other cables from the back of the computer.
3. Remove the system unit cover (see the user's guide for your chassis if necessary).
4. Remove the CPU card from the chassis (if necessary) to gain access to the card's PC/104 connector.
5. Screw the brass spacer (included with the module) into the threaded hole on the CPU card. Do not tighten too much, or the threads may be damaged.
6. Carefully align the pins of the PCM-3641 with the PC/104 connector. Slide the module into the connector. The module pins may not slide all the way into the connector; do not push too hard or the module may be damaged.
7. Secure the module to the CPU card to the threaded hole in the CPU card using the included screw.

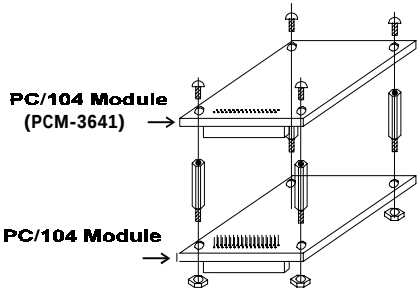
8. Attach any accessories to the PCM-3641.

9. Reinstall the CPU card and replace the system unit cover. Reconnect the cables you removed in step 2.

Turn the power on.

Connecting to another PC/104 module.

1. Insert the pins of connector JP6 (on the end of the PCM-3641 module) into the piggyback connector on the other PC/104 module.



2. Screw the PCM-3641 to the brass spacer. This completes the hardware installation.

Programming with COM1 or COM2

If you set the PCM-3641's ports as COM1 and COM2, you can send and receive data using the normal communication functions found in high-level languages. The following examples use BASIC to demonstrate PCM-3641 programming.

The BASIC communication process starts with the OPEN "COMn: , , ..." statement. This statement assigns a buffer for communication purposes and sets up the communication parameters.

Command format

```
OPEN "COMn: [speed][,parity][,data][,stop]
[,RS][,CS[n]][,DS[n]][,CD[n]][,LF][,PE]"
AS [#]filenum
```

Example:

```
OPEN "COM1:9600,N,8,,CS,DS,CD" AS #1
```

Where:

COMn: n is 1 or 2, indicating either COM1 or COM2

speed: An integer constant specifying the baud rate in bits per second

parity: One of the following characters:

S: space

O: odd

M: mark

E: even

N: none

data: An integer constant indicating the number of data bits. Valid values are 4, 5, 6, 7 and 8. The default is 7.

stop: The number of stop bits. Valid values are 1 and 2. The default is 2 for 75 and 110 bps, 1 for all others.

RS: Suppresses RTS

CS: Controls CTS

DS: Controls DSR

CD: Controls CD

LF: Sends a line feed following each carriage return

PE: Enables parity checking

filenum: filenum is an integer expression which evaluates to a valid file number

You must put the speed, parity, data and stop parameters in this position and order, but you can put the RS, CS, DS, CD, LF and PE parameters in any order. The n argument in the CS, DS and CD parameters specifies the number of milliseconds to wait for the signal before returning a "device timeout" error. n may range from 0 to 65535. If you omit n or set it equal to 0, then the line status is not checked at all.

Refer to the IBM BASIC reference manual for more detailed information.

Programming example - standard COM ports

You can use the following BASIC program to test the PCM-3641's send and receive functions.

```
10   \*****************************************
20   \Program: DEMO01.BAS                      *
30   \Description: This demo program transmits a  *
40   \string through COM1 and receives it through  *
50   \COM2                                     *
70   \*****************************************
160  \Set the proper parameters
170  \COM1 & COM2: baud rate=9600 ; no parity check;
180  \Data bit=8; stop bit=1
190  \Ignore the CTS, RTS and DSR signals.
200  OPEN "COM1:9600,N,8,1,RS,CS,DS,CD" FOR RANDOM AS #1
210  OPEN "COM2:9600,N,8,1,RS,CS,DS,CD" FOR RANDOM AS #1
220  INPUT "INPUT COMMAND:";CMD$
230  IF CMD$="Q" OR "q" THEN CLOSE:END ELSE GOSUB 250
240  GOSUB 300:GOTO 220
250  \****** Transmit data sub-routine ******
260  PRINT #1,CMD$
270  RETURN
300  \****** Receive data sub-routine ******
310  T=TIMER:TEMP$="":RX$=""
320  IF TIMER>T+.5 THEN PRINT "TIMEOUT ERROR":RETURN
330  IF LOC(2)>0 THEN TEMP$=INPUT$(1,#2) ELSE GOTO 320
340  RX$=RX$+TEMP$
350  IF TEMP$=CHR$(13) THEN GOTO 360 ELSE GOTO 320
360  PRINT "RECEIVE DATA:";RX$:RETURN
380  'See if the data is available by checking
390  'the Data Ready bit.
400  IF (INP(PORT2%+5) AND 1)=0 GOTO 400
410  J=INP(PORT2%)
420  'Print out the data byte received
430  PRINT "port ";HEX$(PORT2%)" = ";HEX$(J)
440  'If the value sent <> the received value then error
450  IF J<>BYTE GOTO 620
460  NEXT BYTE
470  'Loop over data (0-255) and send it
480  'from port2 to port1.
490  FOR BYTE=0 TO 255
500  'See if the transmitter buffer is empty.
510  IF (INP(PORT2%+5) AND 32)=0 GOTO 510
520  OUT PORT2%,BYTE
```

```
530  'See if the data is available by
540  'checking the Data Ready bit.
550  IF (INP(PORT1%+5) AND 1)=0 GOTO 550
560  J=INP(PORT1%)
570  PRINT "port ";HEX$(PORT1%)" = ";HEX$(J)
580  IF J<>BYTE GOTO 620
590  NEXT BYTE
600  'If everything is OK, then stop.
610  END
620  PRINT "Data transmission error!":BEEP:END
```

Programming example-communication

The following pair of example programs show how you can set up communication between two computers. The first program sends data then receives data. The second program receives data then sends data. Run the first program on one computer and the second on another.

```
10   \*************STEP 1: INITIALIZATION *********
20   \Clear screen
30   CLS
40   \Define variables A to Z as integer
50   DEFINT A-Z
60   \Set port base address (must match hardware)
70   PORT = &H3F8
80   \Set baud rate to 300
90   OUT PORT + 3, &H80
100  OUT PORT, &H80
110  OUT PORT, 1
120  OUT PORT + 3, &H1F
130  \******** STEP 2: SEND DATA ********
150  FOR I = 65 TO 90
160  \
170  GOSUB 200
180  NEXT I
190  GOTO 260
200  STATUS = INP(PORT + 5) AND &H20
210  IF STATUS = 0 THEN 200
220  OUT PORT, I
230  FOR J = 0 TO 1200: NEXT J
240  RETURN
250  \****** STEP 3: RECEIVE DATA *****
260  FOR I = 65 TO 90: GOSUB 280: NEXT I
270  END
280  STATUS = INP(PORT + 5)
290  IF (STATUS AND &H1E) THEN 280
300  IF (STATUS AND &H1) = 0 THEN 280
310  D = INP(PORT)
320  PRINT "DATA= "; CHR$(D)
330  RETURN
```

Program for second computer

```
10   \*************STEP 1: INITIALIZATION *********
20   \Clear screen
30   CLS
40   \Define variables A TO Z as integer
50   DEFINT A-Z
60   \Set port base address (must match hardware)
70   PORT = &H2F8
80   \Set baud rate to 300
90   OUT PORT + 3, &H80
100  OUT PORT, &H80
110  OUT PORT, 1
120  OUT PORT + 3, &H1F
130  \***STEP 2: RECEIVE DATA FROM ANOTHER PC *****
140  FOR I = 65 TO 90: GOSUB 190: NEXT I
150  PRINT: PRINT: PRINT
160  PRINT"DATA RECEIVES END, THEN DATA SEND BEGINNING."
170  PRINT: PRINT "PRESS ANY KEY..."
180  IF INKEY$ = "" THEN 180 ELSE 260
190  STATUS = INP(PORT + 5)
200  IF STATUS AND &H1E THEN GOTO 190
210  IF (STATUS AND &H1) = 0 THEN 190
220  d = INP(PORT)
230  PRINT "DATA= "; CHR$(d)
240  RETURN
250  \********** STEP 3: SEND DATA ***************
260  FOR I = 65 TO 90
270  d = I
280  GOSUB 310
290  NEXT I
300  END
310  STATUS = INP(PORT + 5) AND &H20
320  IF STATUS = 0 THEN 310
330  OUT PORT, d
340  FOR J = 0 TO 1200: NEXT J
350  RETURN
```

C language test program

You can use the following C program to test the PCM-3641's send and receive functions.

```
/*************************************************/
/* Program: DEMO01.C                             */
/* Description: This demo program transmits a string */
/* to COM1 and receives a string from COM2       */
/* Compiler: Turbo C 2.0 */
/*************************************************/

#include <dos.h>
#include <io.h>
#include <stdio.h>
#include <conio.h>

#define TIME_OUT 10000

static int base0 = 0x3f8; /* Base address of port 0 */
static int base1 = 0x2f8; /* Base address of port 1 */
static char rec[16]; /* Buffer for received string */
static char cmd[16]; /* Buffer for transmitted string /

void main()
{
   int i; /* Counter for character being sent/received */
   char flag; /* Flag for end of output/input data */
   int timeout; /* Timeout counter */

   outport((base0+2), 0xc9); /* enable port 0 FIFO */
   outport((base1+2), 0xc9); /* enable port 1 FIFO */

   /* Set communication parameters for port 0 */
   outp(base0+3, 0x80); /* Set DLAB=1 */
   /* Set baud = 115200 */
   outp(base0, 0x01);
   outp(base1+1, 0);
   /* Set data=8, stop=1, no parity */
   outp(base0+3, 0x03);
   /* Disable port 0 interrupt */
   outp(base0+1, 0x00);

   /* Set communication parameters for port 1 */
   outp(base1+3, 0x80); /* Set DLAB=1 */
   /* Set baud = 115200 */
   outp(base1, 0x01);
   outp(base1+1, 0);
/* Set data=8, stop=1, no parity */
   outp(base1+3, 0x03);
/* Disable port 1 interrupt */
outp(base1+1, 0x00);

printf("\nEnter a string to be transmitted "
       "(15 characters or less) or Q to quit:");
gets(cmd);
while (cmd[0] != 'q' && cmd[0] != 'Q')
{
   i=0;
   cmd[strlen(cmd)] = 0x0d;
   flag=1;
   while (flag)
   {
           outportb(base0, cmd[i]); /* Send data */
           if (cmd[i] == 0x0d)
               flag=0;
           i++;
   }
   i=0;
   flag=1;
   timeout=TIME_OUT;
   while (flag)
   {
```

```
        /* Check if receiver data is ready */
        if ((inportb(base1+5) & 1) !=0)
  {

        rec[i]=inportb(base1); /* Receive data */
        if (rec[i] == 0x0d)
        {
          rec[i+1]='\0';
          flag=0;
          printf("\nReceived data: %s\n", rec);
        }
        i++;
  }
  else
  { /* Check timeout */
        timeout--;
        if (timeout == 0)
        {
          flag = 0;
          printf("\nTimeout error\n");
        }
    }
  }
  printf("\nEnter a string to be transmitted "
       "(15 characters or less) or Q to quit:");
  gets(cmd);
  }
}
```

Register Structure and Format

This section gives short description of each of the module's registers. For more information please refer to the data book for the TEXAS INSTRUMENTS 16C554 UART chip.

All registers are one byte. Bit 0 is the least significant bit, and bit 7 is the most significant bit. The address of each register is specified as an offset from the port base address (BASE), selected with DIP switch SW1.

DLAB is the "Divisor Latch Access Bit", bit 7 of BASE+3.

BASE+0 Receiver buffer register when DLAB=0 and the operation is a read.

BASE+0 Transmitter holding register when DLAB=0 and the operation is a write.

BASE+0 Divisor latch bits 0 - 7 when DLAB=1.

BASE+1 Divisor latch bits 8 - 15 when DLAB=1.

The two bytes BASE+0 and BASE+1 together form a 16-bit number, the divisor, which determines the baud rate. Set the divisor as follows:

Baud rate	Divisor
50	2304
75	1536
110	1047
133.5	857
150	768
300	384
600	192
1200	96
1800	64
2000	58
2400	48
3600	32
4800	24
7200	16
9600	12
19200	6
38400	3
56000	2
115200	1

BASE+1 Interrupt Enable Register (IER) when DLAB=0

bit 0 Enable received-data-available interrupt

bit 1 Enable transmitter-holding-register-empty interrupt

bit 2 Enable receiver-line-status interrupt

bit 3 Enable modem-status interrupt

BASE+2 Interrupt Status Register (ISR)

bit 0 Logic 0= An interrupt is pending and the ISR contents may be used as a pointer to the appropriate interrupt service routine.

Logic 1= no interrupt pending.

bit 1-3 These bits indicate the source for a pending interrupt at interrupt priority levels 1, 2, and 3 (see interrupt source table).

bit 6-7 These bits are set to a logic 0 when the FIFO is not being used. They are set to a logic 1 when the FIFOs are enabled.

Interrupt Source Table

Priority Level	Bit-3	Bit-2	Bit-1	Bit-0	FIFO trigger level
1	0	1	1	0	LSR
2	0	1	0	0	RXRDY
2	1	1	0	0	RXRDY
3	0	0	1	0	TXRDY
4	0	0	0	0	MSR

BASE+2 FIFO Control Register (FCR)

bit 0 Enable transmit and receive FIFOs

bit 1 Clear contents of receive FIFO

bit 2 Clear contents of transmit FIFO

bit 3 Change RXRDY and TXRDY from mode 0 to mode 1.

bits 6-7 Set trigger level for receiver FIFO interrupt.

Bit 7	Bit 6	FIFO trigger level
0	**0**	**01**
0	**1**	**04**
1	**0**	**08**
1	**1**	**14**

BASE+3 Line Control Register (LCR)

bit 0 Word length select bit 0

bit 1 Word length select bit 1

Bit 1	Bit 0	Word length (bits)
0	**0**	**5**
0	**1**	**6**
1	**0**	**7**
1	**1**	**8**

bit 2 Number of stop bits

bit 3 Parity enable

bit 4 Even parity select

bit 5 Stick parity

bit 6 Set break

bit 7 Divisor Latch Access Bit (DLAB)

BASE+4 Modem Control Register (MCR)

bit 0 DTR

bit 1 RTS

BASE+5 Line Status Register (LSR)

bit 0 Receiver data ready

bit 1 Overrun error

bit 2 Parity error

bit 3 Framing error

bit 4 Break interrupt

bit 5 Transmitter holding register empty

bit 6 Transmitter shift register empty

bit 7 At least one parity error, framing rror or break indication in the FIFO

BASE+6 Modem Status Register (MSR)

bit 0 Delta CTS

bit 1 Delta DSR

bit 2 Trailing edge ring indicator

bit 3 Delta received line signal detect

bit 4 CTS

bit 5 DSR

bit 6 RI

bit 7 Received line signal detect

BASE+7 Temporary data register

PC104 Bus Signal Assignment

Pin	J1/P1 Row A	J1/P1 Row B	J2/P2 Row C	J2/P2 Row D
0	**-**	**-**	**0V**	**0V**
1	**IOCHCHK***	**0V**	**SBHE***	**MEMCS16***
2	**SD7**	**RESETDRV**	**LA23**	**IOCS16***
3	**SD6**	**+5V**	**LA22**	**IRQ10**
4	**SD5**	**IRQ9**	**LA21**	**IRQ11**
5	**SD4**	**-5V**	**LA20**	**IRQ12**
6	**SD3**	**DRQ2**	**LA19**	**IRQ15**
7	**SD2**	**-12V**	**LA18**	**IRQ14**
8	**SD1**	**ENDXFR***	**LA17***	**DACK0***
9	**SD0**	**+12v**	**MEMR***	**DRQ0***
10	**IOCHRDY**	**(KEY)2**	**MEMW***	**DACK5***
11	**AEN**	**SMEMW***	**SD8**	**DRQ5**
12	**SA19**	**SMEMR***	**SD9**	**DACK6***
13	**SA18**	**IOW***	**SD10**	**DRQ6**
14	**SA17**	**IOR***	**SD11**	**DACK7***
15	**SA16**	**DACK3***	**SD12**	**DRQ7**
16	**SA15**	**DRQ3**	**SD13**	**+5V**
17	**SA14**	**DACK1***	**SD14**	**MASTER***
18	**SA13**	**DRQ1**	**SD15**	**0V**
19	**SA12**	**REFRESH***	**(KEY)2**	**0V**
20	**SA11**	**SYSCLK**	**-**	**-**
21	**SA10**	**IRQ7**	**-**	**-**
22	**SA9**	**IRQ6**	**-**	**-**
23	**SA8**	**IRQ5**	**-**	**-**
24	**SA7**	**IRQ4**	**-**	**-**
25	**SA6**	**IRQ3**	**-**	**-**
26	**SA5**	**DACK2***	**-**	**-**
27	**SA4**	**TC**	**-**	**-**
28	**SA3**	**BALE**	**-**	**-**
29	**SA2**	**+5V**	**-**	**-**
30	**SA1**	**OSC**	**-**	**-**
31	**SA0**	**0V**	**-**	**-**
32	**0V**	**0V**	**-**	**-**

Standard PC I/O Port Assignment

The following chart shows the I/O addresses used by standard PC peripheral devices.

I/O address (hex)	Assignment
000-1FF	**used by base system board**
200	**not used**
201	**game control**
202-277	**not used**
278-27F	**second printer port**
280-2F7	**not used**
2F8-2FF	**COM2**
300-377	**not used**
378-37F	**printer port**
380-3AF	**not used**
3B0-3BF	**Monochrome adapter and printer**
3C0-3CF	**not used**
3D0-3DF	**colors and graphics adapter**
3E0-3EF	**not used**
3F0-3F7	**floppy diskette drive**
3F8-3FF	**COM1**