The thing that makes this card different from other DAQ & Control cards for the PC serial port is the way it communicates with the computer. Many DAQ cards you may have seen include a microcontroller or a UART chip for serial communication. The present card does not have any such chips because it employs direct accessing of UART registers to enable serial to parallel conversion.

Design by George Vastianos,
Student at Electronics Department, Technological Educational Institute of Piraeus, Greece

32-channel digital input card for PC serial port

link your PC to the real world

Main specifications

Number of Inputs:	32
Type of Inputs:	TTL compatible ("0" = 0 V, "1" = 5 V)
Connection:	Serial Port
Communication:	Direct access of UART registers
Power Supply:	9-15 V DC

The field of Data Acquisition (DAQ) & Control is not something new for Electronics Science. The field really started out just after Semiconductor Technology succeeded in making the first microprocessor in an Integrated Circuit (IC) package. These were the days when computers started to take a secure position in industry, and after that, in our homes.

Data Acquisition & Control is all about processing of data and controlling through computers. So the goal of all DAQ & Control systems is to allow the computer to communicate with, and to some extent control, the 'outside world'.

The applications of DAQ & Control systems have to do with industry, where the use of the right input/output cards enables automation and control systems to solve complex tasks.

A DAQ & Control Card may have inputs only, outputs only or a combination of these. Each input or output may be analogue or digital. Another characteristic of these cards is the way they are connected with the computer or computer system. So, many cards have been designed for installation in one of the local buses of the main board (ISA, EISA, PCI etc), or for connection to the available ports of the computer (parallel, serial, game, keyboard ports).

The project described in this article is a card with 32 digital inputs (32 Channel D/I Card) for external connection to the serial (RS232) port on your IBM PC or compatible.

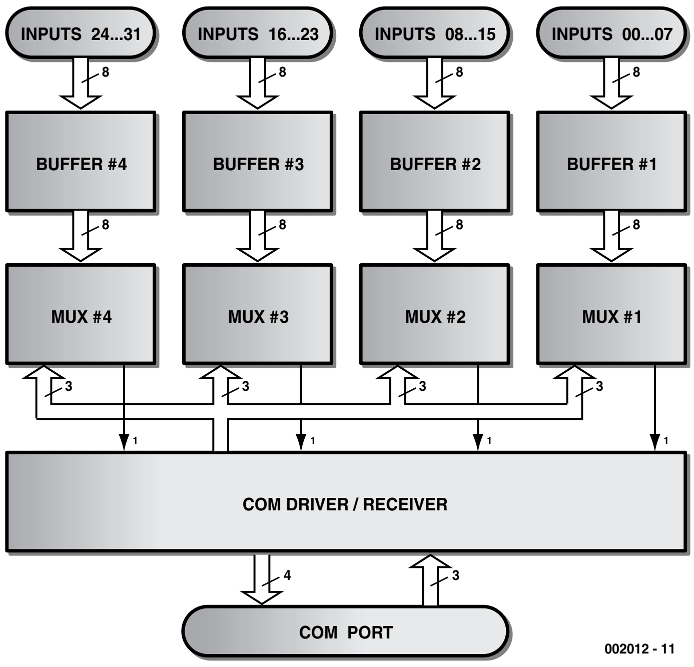


Figure 1. 32-channel D/I card block diagram.

About the serial port

Serial ports are used mainly for communication between computers, or for communication between a computer and peripherals like a modem or a mouse. The controller at the heart of this port is almost invariably a UART (Universal Asynchronous Receiver Transmitter) chip found on the main board. This chip works as a serial-to-parallel and parallel-to-serial adapter.

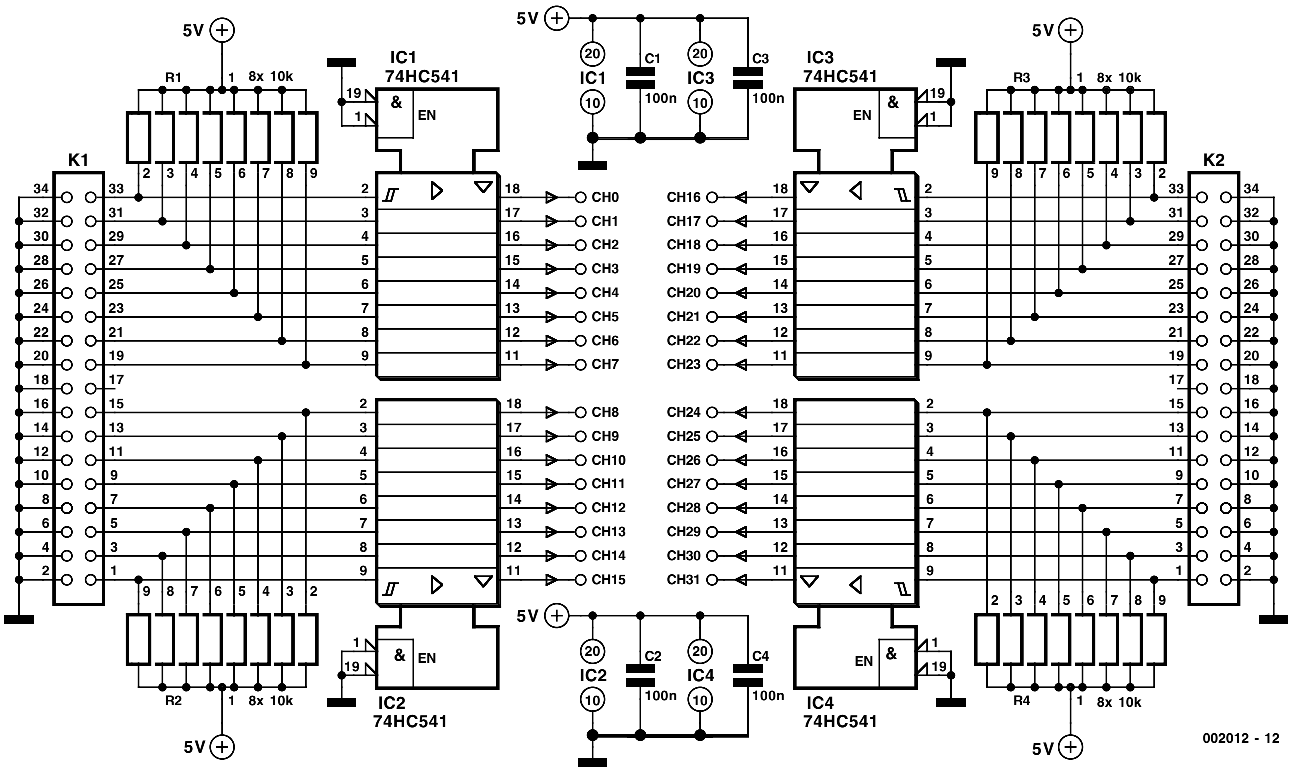


Figure 2. Buffers Unit circuit diagram.

A computer may have one to four serial ports (COM1 to COM4), where each port occupies eight locations in its I/O memory area. See **Table 1** for the relevant details.
The basic lines used by a UART in serial communication for transmission and reception are called TxD and RxD. Also a group of extra lines (DCD, DSR, RTS, CTS, DTR, RI) is used to establish different types of serial communication. Although some of these extra lines work as inputs and others as outputs, each one (except RxD) may controlled through a bit of a UART register. **Table 2** summarizes the interface pin connections and system I/O addresses.
The voltage levels used on the serial port (RS232 levels) are different from TTL levels. In RS232 lingo, a logic '1' is represented by a voltage of –12V, and a logic '0' by +12V.

Table 1.

Transmit/Receive Buffer	3F8h	2F8h	3E8h	2E8h
Interrupt Enable Register	3F9h	2F9h	3E9h	2E9h
Interrupt Identification Register	3FAh	2FAh	3EAh	2EAh
Line Control Register	3FBh	2FBh	3EBh	2EBh
Modem Control Register	3FCh	2FCh	3ECh	2ECh
Line Status Register	3FDh	2FDh	3EDh	2EDh
Modem Status Register	3FEh	2FEh	3EEh	2Eeh

The hardware

In the block diagram of the circuit (**Figure 1**), the available inputs have been divided in four groups of eight inputs (00-07, 08-15, 16-23, 24-31), and they all enter the Buffers Unit. After that, all the lines leaving the Buffers Unit enter the Multiplexers Unit, where only one input of each group is selected. The four selected inputs pass through the COM Driver/Receiver Unit (where they are converted from TTL compatible to RS232 compatible) and arrive at four serial port inputs (CTS, DSR, RI, DCD). For the selection of the inputs, we use the three outputs of on the serial port (TXD, DTR, RTS). Having passed the COM Driver/Receiver Unit (and being adapted from RS232 compatible to TTL compatible), the output signals arrive on the Multiplexers address inputs.

Table 2.

Pin Name	**Pin # on 25-pin connector**	**Pin # on 9-pin connector**	**COM1**	**COM2**	**COM3**	**COM4**	**Bit**	**I/O**
TxD	2	3	3FBh	2FBh	3EBh	2EBh	6	O
DTR	20	4	3FCh	2FCh	3ECh	2ECh	0	O
RTS	4	7	3FCh	2FCh	3ECh	2ECh	1	O
CTS	5	8	3FEh	2FEh	3EEh	2EEh	4	I
DSR	6	6	3FEh	2FEh	3EEh	2EEh	5	I
RI	22	9	3FEh	2FEh	3EEh	2EEh	6	I
DCD	8	1	3FEh	2FEh	3EEh	2EEh	7	I

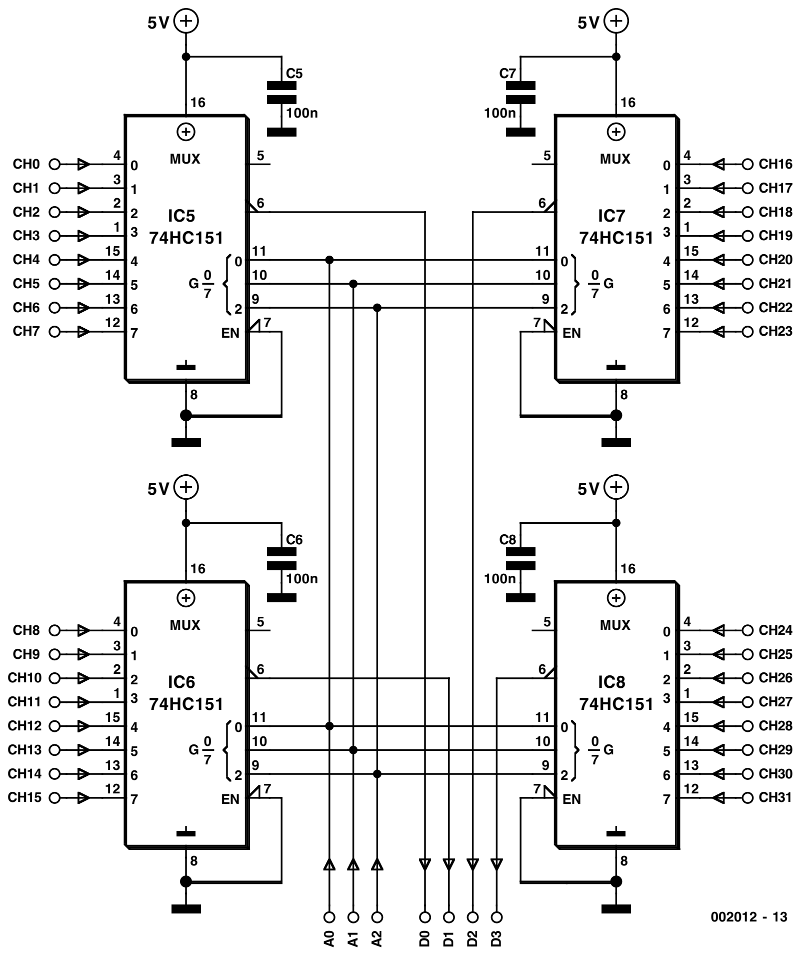


Figure 3. Multiplexers Unit circuit diagram.

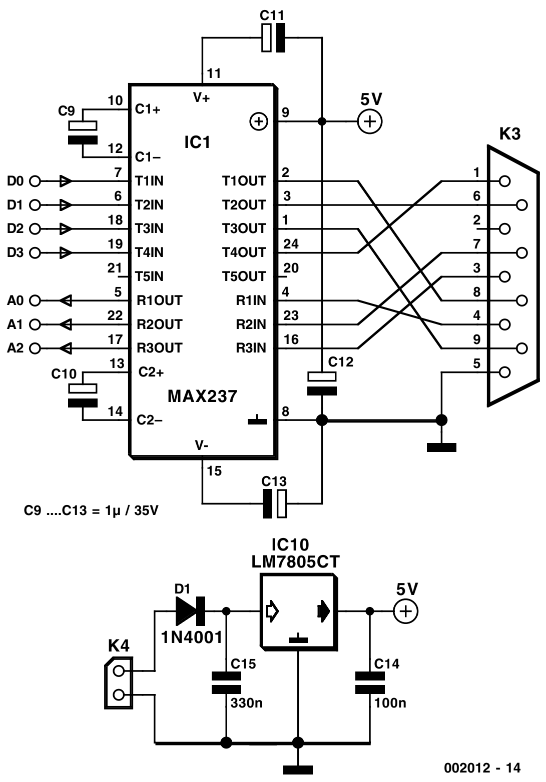


Figure 4. COM Driver/Receiver and Power Supply circuit diagram.

Buffers Unit

In the schematic, **Figure 2**, the 32 inputs have been divided in two groups of 16 (for easy PCB design) and enter the circuit through connectors K1 and K2. The correspondence between the inputs and the connectors can be seen in **Table 3**.

All inputs are fitted with pull-up resistors R1-R4 (10 kΩ) to establish termination in case where one or more inputs are not connected. The buffers are the four chips IC1-IC4 (74HC541). The four capacitors C1-C4 (100 nF) work as bypass capacitors to improve the stability of the circuit. The Output Enable control inputs (pins 1 and 19) of the 74HC541s are connected to ground to make the buffers work continuously.

Multiplexers Unit

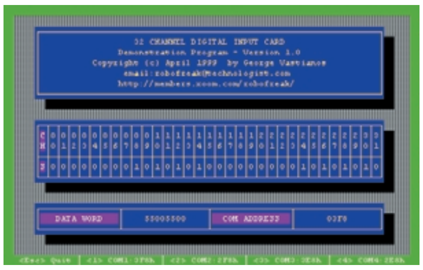
As shown in **Figure 3**, the multiplexers are in reality four 74HC151s (IC5-IC8). Four capacitors C5-C8 (100 nF) are added to ensure adequate supply decoupling. The Output Enable control input (pin 7) of each 74HC151 is connected to ground to make each multiplexer work continuously. The A, B, C address inputs of all 74HC151 are connected together to implement multiplexing of all 32 inputs.

COM Driver/Receiver Unit

The last unit includes a voltage regulator so that the card will not need a regulated power supply to work. In the schematic circuit of the COM Driver/Receiver Unit (**Figure 4**), IC9 (MAX237) works as an RS232 Driver/Receiver. It has 3 channels converting from RS232 to TTL and 5 channels converting from TTL to RS232. Five satellite capacitors (C9-C13; 1 μF, 35V max. working voltage) enable the MAX237 to perform voltage doubling (so the Driver section can produce a voltage of 10 V for the five RS232 outputs, using a 5 V supply voltage). Inside the MAX237, each channel has an inverter. To overcome this problem we use the four inverted outputs of the four 74HC151s (pin 6), so that after inverting two times we have no inverting at all. IC10, a 7805 together with two capacitors C14 and C15, steps down the supply voltage to 5 V. Diode D1 protects the circuit against damage from supply polarity reversal.

The control software

The software for the communication with the card was developed in QBasic. The communication routine is called CARD32DI and its source code



```
32 CHANNEL DIGITAL INPUT CARD
Demonstration Program - Version 1.0
Copyright (c) April 1999  by George Vastianos
email:robofreak@technologist.com
http://members.xoom.com/robofreak/

C 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 3 3
H 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
S 0 0 0 0 0 0 0 0 1 0 1 0 1 0 1 0 0 0 0 0 0 0 0 0 1 0 1 0 1 0 1 0

DATA WORD   55005500   COM ADDRESS   03F8

<Esc> Quit | <1> COM1:3F8h | <2> COM2:2F8h | <3> COM3:3E8h | <4> COM4:2E8h
```

Figure 5. Screendump of the demo program in action.

Table 3.

K1		K2	
Channel	**Pin Number**	**Channel**	**Pin Number**
00	33	16	33
01	31	17	31
02	29	18	29
03	27	19	27
04	25	20	25
05	23	21	23
06	21	22	21
07	19	23	19
08	15	24	15
09	13	25	13
10	11	26	11
11	9	27	9
12	7	28	7
13	5	29	5
14	3	30	3
15	1	31	1

may be found in **Listing 1**.
If you call this routine (from any program written in QBasic), follow this syntax:

```
CALL CARD32DI (COMADDRESS,
CHANNEL (), DATA0, DATA1,
DATA2, DATA3)
```

Where:
COMADDRESS: Integer type variable, which (before calling) must contain the base address of the serial port. Acceptable values of this variable are &H3F8 (for COM1), &H2F8 (for COM2), &H3E8 (for COM3), &H2E8 (for COM4).

CHANNEL (): Matrix Integer type variable (with pointers from 0 to 31), which (after calling) contains the logic state of each channel (values 1 or 0).

DATA0, DATA1, DATA2 & DATA3: Integer type variables that (after calling) contain the arithmetic value of each group of eight channels (00-07, 08-15, 16-23, and 24-31). The logic states of all 32 channels make a Double Word (32 bit) with Ch0 as the LS Bit and Ch31 as the MS Bit. This Double Word may be expressed though the four bytes DATA0, DATA1, DATA2 & DATA3 where LS Byte is DATA0 and MS Byte is DATA3. Employ these variables in cases where you want to save the logic states of all channels in a file. The 'compresssion' allows you to save only 4 bytes instead of a whopping 32.
A demonstration program has been developed to test the 32-channel D/I card. A screenshot of this program is shown in **Figure 5**.
To change the serial port address use the keys <1> to <4>. If you want to quit, just press <Esc>.

How to obtain the software

The source code of the communication routine (CARD32DI.SUB) and the demonstration program (32DICARD.BAS), with an executable version of the demonstration program (32DICARD.EXE) may be obtained via this website

http://members.xoom.com/robofreak/download/32dicard.htm

which for the actual downloading will take you to the author's website at

http://www.robofreak.xs3.com

Finally, the author may be reached by email on *sebastian@mail.kapatel.gr*

(002012-1)

```
REM ****************************************
REM *        32 Channel D/I Card           *
REM *   CARD32DI Communication Routine     *
REM *     Copyright (c)  April 1999        *
REM *        by George Vastianos           *
REM *  email:robofreak@technologist.com    *
REM * http://members.xoom.com/robofreak/   *
REM ****************************************
'
SUB CARD32DI (COMADDRESS, CHANNEL(), DATA0, DATA1, DATA2, DATA3)

  DATA0 = 0: DATA1 = 0: DATA2 = 0: DATA3 = 0

  FOR BIT = 0 TO 7

    IF (BIT AND 1) = (INP(COMADDRESS + 4) AND 1) THEN
      OUT (COMADDRESS + 4), INP(COMADDRESS + 4) XOR 1
    END IF
    IF (BIT AND 2) = (INP(COMADDRESS + 4) AND 2) THEN
      OUT (COMADDRESS + 4), INP(COMADDRESS + 4) XOR 2
    END IF
    IF (BIT AND 4) = (INP(COMADDRESS + 3) AND 64) / 16 THEN
      OUT (COMADDRESS + 3), INP(COMADDRESS + 3) XOR 64
    END IF

    OUT COMADDRESS + 1, 0
    OUT COMADDRESS + 2, 0

    INDATA = INP(COMADDRESS + 6) AND 240

    CHANNEL(BIT) = (INDATA AND 16) / 16
    CHANNEL(BIT + 8) = (INDATA AND 32) / 32
    CHANNEL(BIT + 16) = (INDATA AND 64) / 64
    CHANNEL(BIT + 24) = (INDATA AND 128) / 128

    DATA0 = DATA0 + CHANNEL(BIT) * 2 ^ BIT
    DATA1 = DATA1 + CHANNEL(BIT + 8) * 2 ^ BIT
    DATA2 = DATA2 + CHANNEL(BIT + 16) * 2 ^ BIT
    DATA3 = DATA3 + CHANNEL(BIT + 24) * 2 ^ BIT

  NEXT BIT

END SUB
```