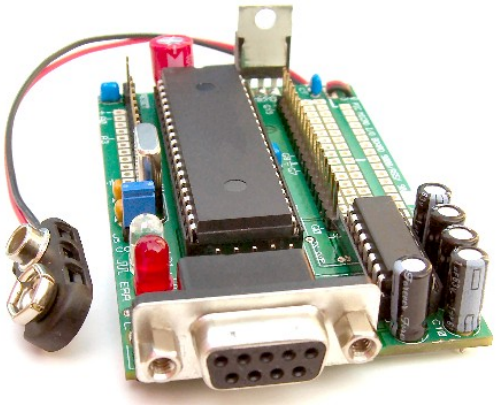
Advanced Data Capture and Control Systems

Tronisoft Limited
Email: sales@tronisoft.com
Web: www.tronisoft.com



DACIO 200 Series User Guide

RS232 Controlled Digital & Analogue Input / Output Modules

P/N: 9600
Document version: 1.01
Date: July 2006

Information furnished is believed to be accurate and reliable. However, Tronisoft Limited assumes no responsibility, consequential or otherwise of use of such information.
Email support@tronisoft.com with suggestions or to report document inaccuracies, omissions and errors.

Specifications mentioned in this publication are subject to change without notice. This publication supersedes and replaces all information previously supplied.

© Copyright 2006 Tronisoft Limited - Printed in England, United Kingdom - All Rights Reserved.
http://www.tronisoft.com

1 Important Information, Notices and Acknowledgements

No parts of this work may be reproduced in any form or by any means - graphic, electronic, or mechanical, including photocopying, recording, taping, or information storage and retrieval systems without the written permission of the publisher and author. All rights reserved.

1.1 Trademarks

Products that are referred to in this document may be either trademarks and/or registered trademarks of the respective owners. The publisher and the author make no claim to these trademarks.

1.2 Disclaimer

Information furnished is believed to be accurate and reliable. However, the publisher and the author assume no responsibility, consequential or otherwise, for errors or omissions or use of such information contained in this document or from the use of programs and source code that may accompany it. In no event shall the publisher and the author be liable for any loss of profit or any other commercial damage caused or alleged to have been caused directly or indirectly by this document.

1.3 Safety and Regulatory Compliance

Tronisoft Limited takes considerable care in designing products with safety and ease of use in mind. However, Tronisoft's products are not authorized for use as critical components in life support or safety critical devices or systems. The product(s) contained within this User Guide are OEM products. These may be suitable only for system integrators and electronics enthusiasts. By using, testing or integrating the product(s) described in this User Guide, you accept responsibility for ensuring that relevant local regulatory compliancy requirements are met depending on the country of use. Follow any recommendations and take note of any cautions or warnings stated within this User Guide. Generally, do not use Tronisoft products without first reading their respective User Guide.

2 Introduction

The DACIO 200 series modules are powerful new microcontroller (MCU) based computer interface boards. A single board provides two 8-bit digital input/output ports and up to 8 analogue input channels to computers equipped with a spare RS232 COM port. The advanced design features unmatched power and ease of use at a budget price. It is useful for applications requiring a serial to parallel (and parallel to serial) converter. It is ideal for use in computer-controlled systems, for digital/analogue data acquisition, robotics, automatic testing equipment, process control or for experimental and educational purposes. This document is applicable to the DACIO 210 and DACIO 280 versions of the interface boards. Table 1 highlights the main differences.

Table 1: DACIO 2XX Variations

Model No.	Digital I/O Lines	A/D Channels	A/D Resolution
DACIO 210	16	1	8-bit
DACIO 280	16	8 (or 7 + 1 ref. input)	8-bit

2.1 Features List

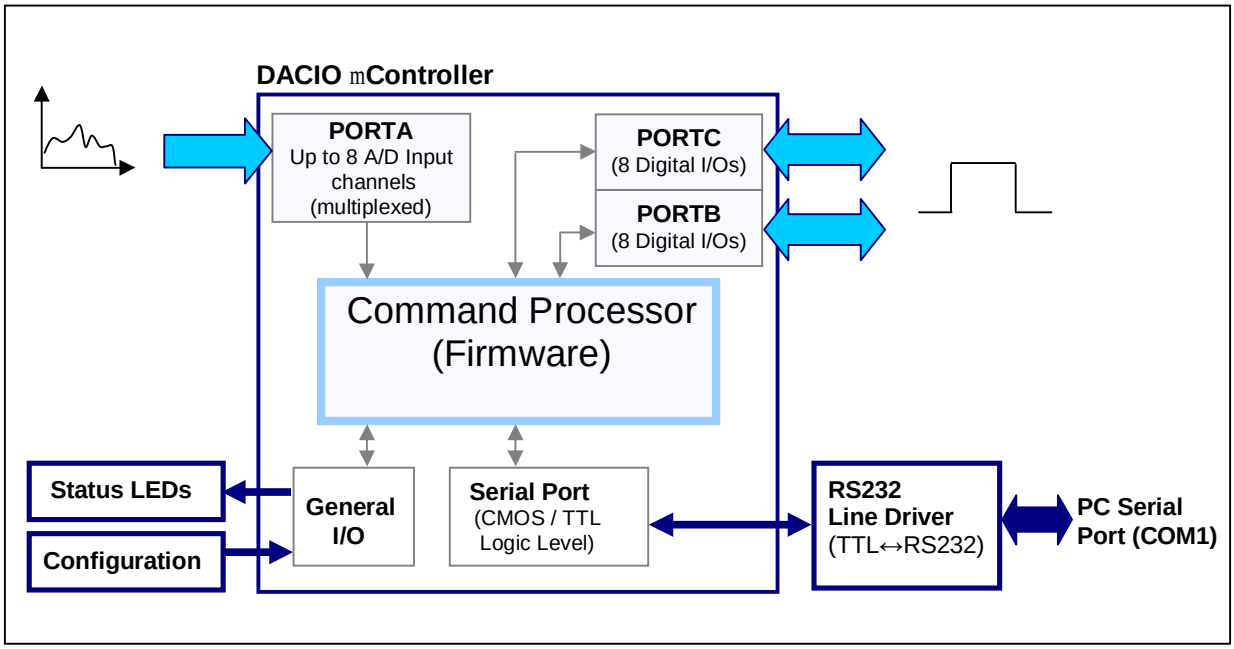
A brief list of some of the features included in the DACIO module is given below:

- Two 8-bit digital ports (CMOS/TTL compatible). Each input / output line is individually software configurable.
- The DACIO 210 features 1 A/D input channel with VDD reference and the DACIO 280 features 8 A/D input channels or 7 A/D + 1 reference channel.
- Single DC supply: 6 to 10V wide input (or 12V with heatsink).
- Very low power consumption - approx. 10mA no load.
- Micro based design with small PCB footprint (80×50mm).
- High performance PC interface – Industry standard RS232 line drivers. Baud rates: 9600 and 19200 bps - others available on request.
- Full (but suppressible) error reporting. Two status LEDs.
- Quickstart guide, default settings, LED demo and powerful trace feature gets you up and running in minutes! Intelligent on-board micro communicates with a Terminal program on the host computer allowing you to run demos or try out all the features by typing commands manually.
- Flexible and easy to remember instruction set. Bit, byte or 16-bit operations.

 © Copyright 2006 Tronisoft Limited. All Rights Reserved.

2.2 Block Diagram - DACIO Module (200 Series)

Figure 1: Module Block Diagram



The module provides two digital I/O ports called PORTB and PORTC and one analogue input port called PORTA.

The PC/computer interface is a serial port connection via RS232 line drivers into the module's microcontroller providing the input/output lines. Computers equipped with an RS232 COM port can control the connected module by sending commands in ASCII which are processed by the module's internal Command Processor (firmware).

Status LEDs indicate communication to the unit and are also used to indicate errors.

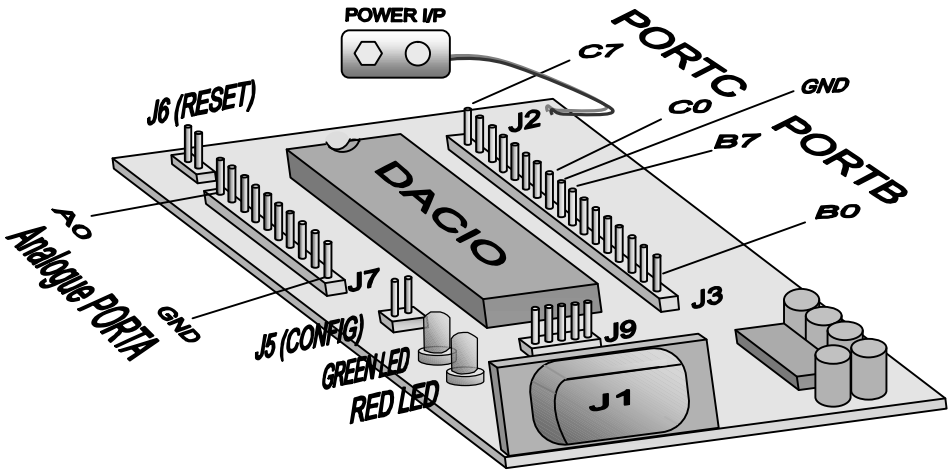
The module can be configured for baud rate selection via configuration jumper pins.

 © Copyright 2006 Tronisoft Limited. All Rights Reserved.

3 Hardware Description

Figure 2 below is a simplified diagram depicting the main hardware components and input / output ports. A hardware description of the features is provided here followed by the software control description in Section 4.

Figure 2: Simplified Module Diagram.



3.1 Power Supply

The module requires an external power supply or battery to be connected via the attached “PP3” connector. The input voltage may be 6 to 10V. Higher voltages require an adequate heatsink to be fitted to the on-board voltage regulator. The input voltage Vin is regulated using a precision voltage regulator to the voltage used internally by the module called VDD. VDD is 5V (+/- 1%).

Current consumption may be as low as 10mA (no load) and as high as 250mA if all 16 digital I/O lines are powering LEDs at 15mA each.

3.2 Analogue to Digital PORTA

The module has up to 8 (DACIO 280) analogue input channels accessible via connector J7. The pin numbers and signal descriptions are as follows:

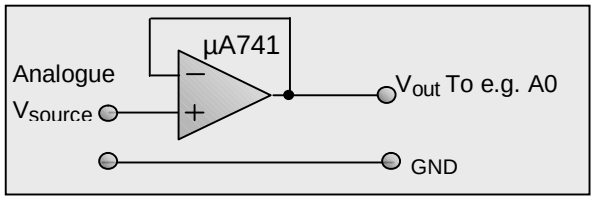
J7 Pin Number	Signal Name	Description
1	A0	Analogue Channel 0
2	A1	Analogue Channel 1
3	A2	Analogue Channel 2
4	A3/Ref	Analogue Channel 3 / Optional Reference Input
5	N/C	Reserved – do not use
6	A4	Analogue Channel 4
7	A5	Analogue Channel 5
8	A6	Analogue Channel 6
9	A7	Analogue Channel 7
10	GND	Signal ground

The on-board Analogue to Digital Converter (ADC) allows conversion of an analogue input signal to a corresponding 8-bit digital number. The 8 channels are multiplexed. Any one of the 8 analogue input channels when read give a value of 0-255 where 0 represents 0V and 255 represents the reference voltage (e.g. VDD) of the module.

The analogue port may be configured under software control as 8 analogue input channels with VDD (+5V) as the voltage reference or as 7 analogue channel input channels with channel A3 (J7, pin 4) as the reference input signal. If channel A3 is used as the reference signal, ensure the voltage on this line is between 3V and VDD.

Analogue to Digital (A/D) resolution is 8-bit. Accuracy is within +/- 1 bit, but that can be made worse depending on the analogue source. Keep the source impedance low (<< 10 kΩ). If required, pass the analogue source through a voltage follower. Figure 3 below shows the use of a unity gain buffer (as a voltage follower) to provide a high input impedance to an analogue voltage source.

Figure 3: Unity Gain Buffer



Note that the DACIO 210 module only has channel A0 available.

3.3 Digital PORTB and PORTC

There are two 8-bit digital ports called PORTB and PORTC.

PORTC is an 8-bit port accessible via connector J2. The pin numbers and signal descriptions are as follows:

J2 Pin Number	Signal Name	Description
8	C7	PORTC Bit 7
7	C6	PORTC Bit 6
6	C5	PORTC Bit 5
5	C4	PORTC Bit 4
4	C3	PORTC Bit 3
3	C2	PORTC Bit 2
2	C1	PORTC Bit 1
1	C0	PORTC Bit 0

PORTB is an 8-bit port accessible via connector J3. The pin numbers and signal descriptions are as follows:

J3 Pin Number	Signal Name	Description
8	B7	PORTB Bit 7
7	B6	PORTB Bit 6
6	B5	PORTB Bit 5
5	B4	PORTB Bit 4
4	B3	PORTB Bit 3
3	B2	PORTB Bit 2
2	B1	PORTB Bit 1
1	B0	PORTB Bit 0

 © Copyright 2006 Tronisoft Limited. All Rights Reserved.

Note that each bit of PORTB or PORTC is individually configurable under software control as either an input or an output line. In addition, the module features an internal programmable weak pull-up to the +5V supply rail on PORTC. See Figure 4 below. This feature may be used to read the state of switches without requiring the use of external pull-up resistors. Assuming PORTC is configured as an input port the switches can be directly connected to the I/O pins on PORTC with the other end to ground. This is shown in Figure 5 below. Reading the input pins will read as '1' when the switch is open and '0' when the switch is closed. The internal pull-ups on all lines may be enabled or disabled under software control (via commands).

Figure 4: Simplified Diagram of PORTC with Programmable Internal Pull-ups

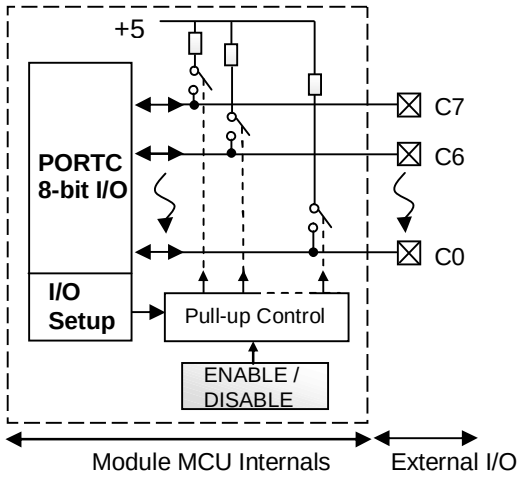
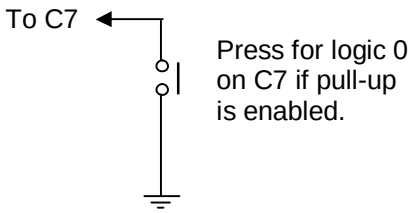


Figure 5: Connecting Switches to Input Pins



Note: Internal pull-ups are disabled automatically for I/O lines configured as outputs. So set relevant bits to inputs, then enable the internal pull-ups.

3.4 Reset Pins

The module may be reset by shorting the jumper pins on J6 together for at least 1ms.

J6 (Reset)	Usage
	Normally open. Short pins to initiate hardware reset to the module.

3.5 Other Digital I/O Ports

An additional general purpose port is accessible via connector J9 for future expandability. This is currently unimplemented.

3.6 RS232 Interface

The module connection to a host computer is via J1. J1 is a 9 way D-type socket connector. A standard serial extension cable (pin 1 to pin1, pin 2 to pin 2 etc.) is required to connect to a host computer. The module supports two baud rates: 9600 and 19200. The data format is 8 data bits, no parity and 1 stop bits.

3.6.1 Baud Rate Configuration

The baud rate of the module is 19200 by default. To set to 9600 place a jumper over connector J5 (CONFIG) on the module. Table 2 below shows the different baud rate options and how to set the jumper pin settings.

Table 2: Baud Rate Setup Options

J5 (Config) Settings	Serial port baud rate (bps)
	19200 (Default)
	9600

Notes:

a) Communication speeds 9600 to 19200 bps have all been tested reliably with shielded cable lengths up to 20m. Longer cable lengths may be possible but may result in unreliable, degraded signal performance.

b) The module only reads the J5 jumper pin settings on power up or when the module is reset. Therefore, if changing J5 configuration options, reset the module for the change to take effect.

3.7 Status LEDs

Two status LEDs are provided, one for indicating valid communication with the module and the other for notifying errors:

Table 3: Status LED Descriptions

LED Colour	Description
Green	This flashes during successful communication e.g. on receiving valid command sequences.
Red	This is turned on for a hard communication error such as a framing or baud rate error.

Note that the “Red” LED should never light up in normal running mode unless the wrong communication settings are being used with the module or the module is being used in a very noisy environment such that communication to the module is being corrupted.

 © Copyright 2006 Tronisoft Limited. All Rights Reserved.

3.8 Module Specification

3.8.1 Power Supply

Input Supply Voltage: 6-10V or 12V with optional heatsink.
Input Current: approximately 10mA on no load and 250mA on full load.
Internal Regulated Voltage (VDD): Linear 5V, 1% accuracy.

3.8.2 Host Connection / RS232 Line Drivers

Connection: RS232 / serial port, 9 Way D-type Connector.
Line settings: 9600 or 19200 (default) baud, 8 data bits, no parity and 1 stop bits.
Line Drivers: Industry standard RS232 line drivers - meets full EIA/TIA-232D specifications.

3.8.3 Input / Output Ports

Analogue PORTA

A/D Channels: DACIO 210 has one A/D channel (A0). The DACIO 280 has eight multiplexed A/D channels (A0-A7) or 7 A/D + 1 reference channel.
Resolution: 8-bit.
Accuracy: +/- 1 bit (assuming source impedance is low i.e. << 10 kΩ).
Reference Voltage: software configurable as VDD or channel A3.
Reference Voltage Range: +2.5V minimum, +5V maximum.

Digital PORTB

Number of lines: 8-bits wide, bi-directional port.
Data direction: input by default - each line individually configurable.
Output current: source / sink up to 25mA per line, however see note 1 below.

Digital PORTC

Number of lines: 8-bits wide, bi-directional port.
Data direction: output by default - each line individually configurable.
Output current: source / sink up to 25mA per line, however see notes below.

Notes:

a) Ensure that the total current sourced or sunk by PORTB is less than 200mA.
b) Ensure that the total current sourced or sunk by PORTC is less than 200mA.
c) Ensure that the total current sourced or sunk by PORTB and PORTC combined is less than 240mA.

In other words, PORTB and PORTC combined can source or sink safely a maximum of 240mA in total. Do not exceed this amount. As an example, a total of 16 LEDs may be driven at 15mA via PORTB and PORTC using current limiting resistors. Alternatively up to 8 LEDs can be driven at 25mA from say PORTB only. PORTB and PORTC outputs have direct LED drive capability but ensure that the limits described above are not exceeded.

© Copyright 2006 Tronisoft Limited. All Rights Reserved.

3.8.4 Electrical Characteristics

DC Characteristics

Symbol - Description	Min	Typ	Max	Units	Conditions
VIL – Input Low Voltage	GND	-	0.8	V	
VIH – Input High Voltage	2.0	-	5	V	
VOL – Output Low Voltage	-	-	0.7	V	Current sink = 1-10 mA, VDD = 5V
VOH – Output High Voltage	4.2	-	-	V	Current source = 1-10 mA, VDD = 5V
IPURC - PORTC Weak Pull-up Current	50	250	400	µA	

Notes:
a) VDD is 5V +/- 1%.
b) Data in “Typ” column is at 25°C
c) Figures are quoted for operating temperature range 0°C to +85°C

Absolute Maximum Ratings †
Ambient temperature under use..... 0 to +85°C
Storage temperature ..... -10°C to +85°C
Voltage on any pin with respect to GND (except VDD)..... -0.3V to (VDD + 0.3V)
Voltage on VDD with respect to GND..... -0.3 to +6.0V
Maximum current into VDD pin..... 250 mA
Maximum output current sunk by any I/O..... 25 mA
Maximum output current sourced by any I/O pin..... 25 mA
Maximum current sunk by PORTB (combined)..... 200 mA
Maximum current sourced by PORTB (combined)..... 200 mA
Maximum current sunk by PORTC (combined)..... 200 mA
Maximum current sourced by PORTC (combined)..... 200 mA

† **NOTICE:** Stresses above those listed under “Absolute Maximum Ratings” may cause permanent damage to the device. This is a stress rating only and functional operation of the device at those or any other conditions above those indicated in the operation listings of this specification is not implied. Exposure to maximum rating conditions for extended periods may affect device reliability.

4 Module Protocol and Control Commands

There are two digital I/O ports called PORTB and PORTC and one analogue input port called PORTA. All bits of PORTB & PORTC are individually configurable as input or output. On powerup, by default PORTB is configured as an input port and PORTC as output port. Computers equipped with an RS232 COM port can control a connected module by sending command strings i.e. a recognised sequence of ASCII characters. The command string consists of three parts as shown below:

The whole command string can consist of 4-9 ASCII characters depending on the function being performed. The first character is the *start character* to inform the module of the type of command expression coming up. The *command expression* characters specify the expression to be processed by the module. It is similar to a high level programming language expression and is easy to remember. All command expressions must be terminated with a *termination/end character*.

The general command string is formed as follows:

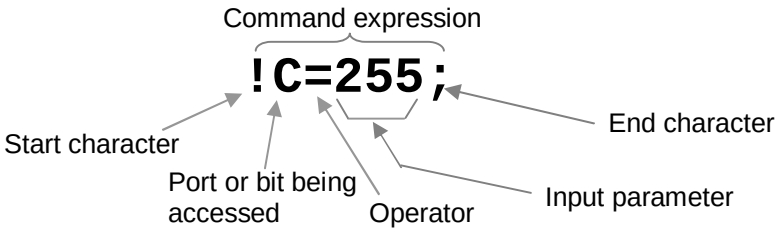
- **Start Character:** All commands to the module must begin with either an exclamation mark (!) or a hash (#). The start character is also used to specify the radix (number base) of input parameters: '!' for decimal and '#' for hexadecimal. The examples given below will demonstrate which to use.

- **Command Expression:**
  a) With all except the configuration and miscellaneous commands, the next one or two character(s) specifies the port (e.g A for PORTA, B for PORTB etc) or bit (e.g. A0 for bit 0 of PORTA).
  b) An operator symbol is then required ('=' to write, '?' to read, etc).
  c) The next character(s), if required, is used to provide an input parameter.

- **End Character:** All commands must end with a semicolon (;).

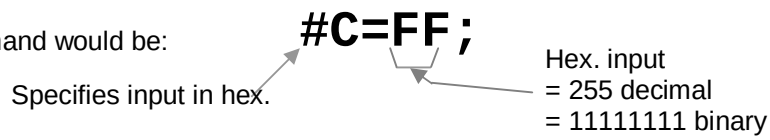
If a valid command is received, the module will normally reply with an exclamation character (!). The module will then output the data requested (if any) and end with a carriage return <0D>. On error, the module will normally reply with a question mark (?) and end with a carriage return <0D>.

Example command string:



In the command shown above the module is being instructed to write a decimal byte value of 255 to PORTC.

An equivalent (hex) command would be:



 © Copyright 2006 Tronisoft Limited. All Rights Reserved.

There are two main groups of commands accepted by the module: decimal radix & hexadecimal radix commands. Commands starting with '!' are the decimal radix commands. Commands starting with '#' are the hexadecimal radix commands.

The two groups have similar commands, however, in hex radix mode all 16-bit operations are implemented. The main difference is that the input parameters and the data from the module replies are given in either decimal or hex.

The instruction set is described below. Most instructions/commands are described for PORTB only but may be equally applied to the other available ports in a similar fashion. Boldface lower capitals 'x' denote a required input parameter. Remember that all characters sent to the module must be in uppercase. For clarity, commands may sometimes be shown in quotation marks e.g. "!B=15;" but the quotation marks are not part of the actual ASCII command string sent to the module. Similarly, the same is true for replies from the module. A carriage return char (hex 0D) is shown as: <0D>.

4.1 Basic Instruction Set

4.1.1 Digital Read Commands

!Bx?; Read bit x (0-7) from PORTB. Reply is: "!x<0D>" where x is 1 or 0.

!B?; Read byte from PORTB. Reply is: "!xxx<0D>" where xxx is 000-255.

Example 1:	"!B0?;"	(Read bit 0 of PORTB)
	"!1<0D>"	(Reply: logic high on bit 0)
Example 2:	"!B?;"	(Read PORTB)
	"!045<0D>"	(Reply: PORTB = 45)
Example 3:	"#B?;"	(Read PORTB)
	"2D<0D>"	(Reply: PORTB = 45)

Equivalent hex commands: "#Bx?;" and "#B?;"
Available ports: PORTB, PORTC

4.1.2 Analogue Read Command

!Ax?; Read channel x (0-7) from the analogue port. Reply is: "!xxx<0D>" where xxx is 000-255.

To work out the voltage Va on the input channel use the following formula:

Va = (X)(Vr/255)

where X is the number returned by the module and Vr is the reference voltage.

Note:
a) As the analogue port is read only, the module will accept a command even if the read operator (?) is not used. The commands "!A2?;" and "!A2;" achieve the same result.
b) Maximum sample rate is approximately 218 sps (samples per second) using hexadecimal commands. Conversion time is 20us.

Example 1:	"!A2?;"	(Read PORTA channel 2)

 © Copyright 2006 Tronisoft Limited. All Rights Reserved.

"!127<0D>" (Reply: A/D channel 2 = 127)

Assuming analogue source impedance is < 10 kΩ and reference voltage used is 5 volts then the voltage on channel 2 is:
127×(5/255) = 2.5 volts approx.

Equivalent hex command: "#Ax?;"
Available ports: PORTA only

4.1.3 Write Commands

!B=xxx; Write the decimal byte value xxx to PORTB where xxx is a 1 to 3 digit number ranging from 0-255.

!Bx=1/0; Write 1/0 to bit x (0-7) of PORTB.

Example 1: "!B=15;"(Write 15 to PORTB)
"!<0D>" (Reply: acknowledged)

Example 2: "!B4=1;"(Write logic 1 to bit 4 of PORTB)
"!<0D>" (Reply: acknowledged)

Note: commands "!B=015;" and "!B=15;" have the same effect.

Equivalent hex commands: "#B=xx;" and "#Bx=1/0;" (where xx are hex digits 0-FF).
Available ports: PORTB, PORTC

4.1.4 Invert/Toggle Commands

!B~; Invert all bits in PORTB (change all 1s to 0s and vice versa).

!Bx~; Invert bit x (0-7) of PORTB

Example 1: "!B~;" (Invert PORTB)
"!<0D>" (Reply: acknowledged)

Example 2: "!B3~;" (Invert bit 3 of PORTB)
"!<0D>" (Reply: acknowledged)

Equivalent hex commands: "#B~;" and "#Bx~;"
Available ports: PORTB, PORTC

4.1.5 Shift Commands

!B>; Shift right PORTB (towards LSB)

!B<; Shift left PORTB (towards MSB)

Example 1: !B>; (shift right PORTB)
!<0D> (Reply: acknowledged)

Equivalent hex commands: "#B>;" and "#B<;"
Available ports: PORTB, PORTC

 © Copyright 2006 Tronisoft Limited. All Rights Reserved.

4.2 Setup / Configuration Commands

There are various setup options available online. The module can be configured for baud rate, input/output direction, analogue setup etc. The full list of configuration commands is given below. On power-up the module is set to a default configuration setting and this is given after the commands have been described. There are a lot of configuration options so practice using the module with the default settings at first then use the commands in this section when the need arises.

4.2.1 Input/Output Direction Setup

!SBx=I/O; Set bit x (0-7) of PORTB as an input/output line.
I = input, O = output

!SB=xxx; Set PORTB input/output configuration byte (0-255).
Bit = 1 = Input, Bit = 0 = Output.

Note: by default, PORTB is an input port and PORTC is an output port on module powering up or after a reset.

Example 1: "!SB3=I;" (Set B3 of PORTB as an input line).
"!<0D>" (Reply: acknowledged)

Example 2: "!SB=15;" (Set B0-B3 as inputs and B4-B7 as outputs).
"!<0D>" (Reply: acknowledged)

Equivalent hex commands: "#SBx=I/O;" and "#SB=xx;" (xx is a 1 to 2 digit hex no.).
Available Ports: PORTB, PORTC.

4.2.2 Analogue Configuration

There are 8 analogue input channels on PORTA. The on-board ADC allows conversion of an analogue input signal to a corresponding 8-bit digital number. The 8 channels are multiplexed. Any one of the 8 analogue input channels when read give a value of 0-255 where 0 represents 0V and 255 represents the positive supply voltage (VDD) of the module. VDD (+5V) is the default reference voltage. The reference voltage can be changed to anything between 3V and VDD only if PORTA is configured to 7-channel mode where the reference voltage is applied to channel A3.

!SA=7/8; Configure analogue to digital port to 7 channel (with channel A3 as ref.) or 8 channel mode (with VDD as ref.).

Equivalent Hex Command: "#SA=7/8;"

Example: "!SA=8;" Set to 8 channel A/D - use all channels as analogue inputs.

Note a) If PORTA is set to 7 channel mode then reading channel A3 will not give a meaningful result. This is because channel A3 is the reference input. Reading channel A3 with itself as reference will always return the value 255.

4.2.3 Internal Pull-ups

The DACIO features an internal programmable weak pull-up to the +Ve supply rail on PORTC. See Figure 4 in section 3.3 "Digital PORTB and PORTC". This feature may be used to read the

state of switches without requiring the use of external pull-up resistors. To enable/disable internal pull-ups on all lines use the following command:

!SCPU=E/D; Enable/Disable internal pull-ups

Equivalent Hex Command: “#SCPU=E/D;”

That's it! Those are all the basic commands used to control the module. Read the QuickStart guide and use the demo and the powerful trace feature to quickly learn the instruction set and understand how the module operates.

The following sections cover the more advanced features of the module. Again the trace feature will help you to quickly understand and use all the advanced features.

4.3 Miscellaneous Commands

PORTB and PORTC are both 8-bit I/O ports and these can be combined and thought of as a “general” 16-bit port or PORTG. The low 8-bits of PORTG map directly onto PORTB and the most significant 8-bits map onto PORTC. Instructions for operating on PORTG are similar to those already shown. The only difference is that reading from or writing to both PORTB and PORTC can be done using one instruction.

Note: For the decimal radix commands shown below the parameter x is a single decimal digit (0-9) and therefore limits access to only the 10 bits of PORTG. Only the equivalent hexadecimal commands will allow access to all 16 bits of PORTG as x can then be 0-9 or A-F.

4.3.1 Pseudo 16-bit Read Commands

!Gx?; Read bit x (0-9) from PORTG. Reply is: “!x<0D>” where x is 1 or 0.

!G?; Read 16-bit decimal word value from PORTG (i.e. read both PORTB and PORTC). Reply is: “!xxxxx<0D>” where xxxxx is 00000-65535.

Example 1: "!G?;" (The module will reply with a 16-bit number indicating the status of the bits of both PORTB and PORTC. An example reply may be "!00002<0D>" which would mean that PORTB = 2 and PORTC = 0).

Equivalent hex commands: “#Gx?;” and “#G?;” (where x is a hex digit 0-9 or A-F).

4.3.2 Pseudo 16-bit Write Commands

!G=xxxxx; Write the decimal word value xxxxx to PORTB (low byte) and PORTC (high byte) where xxxxx is a 1 to 5 digit decimal number ranging from 0-65535.

!Gx=1/0; Write 1/0 to bit x (0-9) of PORTG.

Example 1: "!G=65535;" (Write 255 to both PORTB and PORTC).

Example 2: "#G=2D;" (Write hex. 2D to PORTB and 0 to PORTC.).

Example 3: "!G7=1;" (Write '1' - logic high, to bit 7 of PORTB).

Example 4: "!G10=1;" (Error! The parameter x in "!Gx=1/0" must be a single digit. Attempting to write to bit 3 of PORTC. This is only possible using the equivalent hexadecimal command.)

 © Copyright 2006 Tronisoft Limited. All Rights Reserved.

Example 5: "#GA=1;" (Hex command. Write '1' to bit 3 of PORTC).

Equivalent hex commands: “#G=xxxx;” and “#Gx=1/0;”

4.3.3 Pseudo 16-bit Invert/Toggle Commands

!G~; Invert all bits in PORTG (change all 1s to 0s and vice versa).

!Gx~; Invert bit x (0-9) of PORTG.

Equivalent hex commands: “#G~;” and “#Gx~;”

4.3.4 Pseudo 16-bit Shift Commands

!G>; Shift right PORTG (towards LSB)

!G<; Shift left PORTG (towards MSB)

Equivalent Hex Commands: “#G>;” and “#G<;”

4.3.5 Pseudo 16-bit Input/Output Direction Setup

!SGx=I/O; Set bit x (0-7) of PORTB as an input/output line. I = input, O = output

!SG=xxxxx; Set PORTG input/output configuration word (0-65535).
Bit = 1 = input.
Bit = 0 = output.

Example 1: "#SG=FFFF;" (Configure all lines of PORTB and PORTC as inputs.).

Equivalent hex commands: “#SGx=I/O;” and “#SG=xxxx;” (xxxx are hex digits 0-FFFF).

4.3.6 Radix Mode Setup

!SRM=x; Set radix mode to either decimal, hexadecimal or both where x is the character D,H or B. Setting the radix mode to D (decimal) allows the module to respond only to the decimal radix commands i.e. commands starting with ‘!’. Similarly only hex commands are accepted when in radix mode H. Set radix mode to B (both) if the module is to respond to all commands. Default: radix mode is set to B.

Example 1: “!SRM=D;” Set radix mode to decimal. Only decimal radix commands will now be recognised by the module

Equivalent Hex Command: “#SRM=x;”

4.3.7 Output Response Level

This sets the level of response the module provides to indicate a good or a bad command. Normally if a command other than a data read command is executed then the module just replies with a simple acknowledgement e.g. for the command “!B=255;” the module will reply

 © Copyright 2006 Tronisoft Limited. All Rights Reserved.

with either “!<0D>” or “?<0D>” to indicate success or failure respectively. If the module did not accept the command then an error code can be given to indicate the reason for the error. Alternatively, the module can even be prevented from replying at all. Set the “Response Level” to define how the module will respond.

!SRL=x; Set response level to x where x is ASCII ‘0’, ‘1’, ‘2’, ‘D’ or ‘E’. See descriptions in Table 1 below.

Equivalent Hex Command: “#SRL=x;”

Table 2: Output Response / Error Checking Levels.

Response Level x	Action and Description
‘0’	SILENT MODE The module will execute all valid commands but will not reply to any (valid or not). This mode allows easy use of the module where only output/write commands are used and feedback is not required. It will also reliably allow multiple write commands to be sent as one long string. E.g. "!B=1;!B>;B>;!B;!B>;" is a 19 character long command string consisting of five individual commands. Notice that the fourth command is not valid - the module will just ignore it and execute the next command. The final value written to PORTB (assuming it is configured as an output port) will be decimal 8.
‘1’	1 CHARACTER This is the default mode. Basic one character + <0D> based responses. Use this mode for basic feedback on whether commands sent to the module have been accepted. Example response: "!<0D>".
‘2’	2 CHARACTERS Two characters + <0D> command responses. Example responses: “!x<0D>” or “?x<0D>” where x is a response/error code. See Appendix A – “Response/Error Codes” for a full list. This level may be useful for debugging communication problems or syntax errors.
‘D’	DISABLE I/O MISMATCH DETECTION Default mode. Disable detection of input/output configuration mismatch errors. If an I/O mismatch does occur, an error will not be generated though the command will be safely (albeit partially) executed. For example, the command "!SB=15;" followed with "!B=255;" will result in PORTB written with the decimal value of 240 because bits B0-B3 have been configured as inputs.
‘E’	ENABLE I/O MISMATCH DETECTION Enable detection of input/output mismatch errors. See above for more information.

 © Copyright 2006 Tronisoft Limited. All Rights Reserved.

4.3.8 Module ID

Other DACIO modules with different feature sets but using similar hardware may exist in the future. For internal use and for assisting software configuration, modules will store a module type ID to differentiate between them and can be read using the following command.

!SMID?; Reply is !ABC<0D> where A, B and C are ASCII digits '0'-'9' or 'A'-'Z'.

Example 1: "!210<0D>" indicates a "DACIO 210" module.

Equivalent Hex Command: "#SMID?;"

4.3.9 Software Version

The DACIO modules are based on microcontrollers and run firmware to achieve specific functionality. To obtain the current version number of the DACIO firmware, use the following command:

!SVER?; Reply is !XY<0D> where X and Y are ASCII digits '0'-'9'. X is the major release number and Y is the minor number. The minor number changes with minor improvements or bug fixes. Version numbers are usually written in the form X.Y.

Note: This command may be used to automatically check for version changes if modules are replaced. Software on the computer could alert the user if the new module has a different firmware version than the replaced module. The user should then read about the changes associated with the new module and decide if any action has to be taken.

Example 1: "!10<0D>" indicates software version 1.0.

Equivalent Hex Command: "#SVER?;"

 © Copyright 2006 Tronisoft Limited. All Rights Reserved.

5 Demonstration and Trace Mode

The module is normally in “Running” mode where commands are sent to the module to control or read the state of the I/O lines. The module may be placed into a special “Demonstration Mode”. This will allow the use of any terminal emulator to activate demonstration modes and test the features of the module using the ‘trace’ option all without writing a single line of code!

5.1 Enabling Demonstration Mode

The module can be placed in “Demonstration Mode” using the following steps:

1) Connect the module using a serial cable to a computer’s serial port e.g. COM1.

2) Ensure a jumper is placed across J5 (CONFIG) (for baud rate 9600).

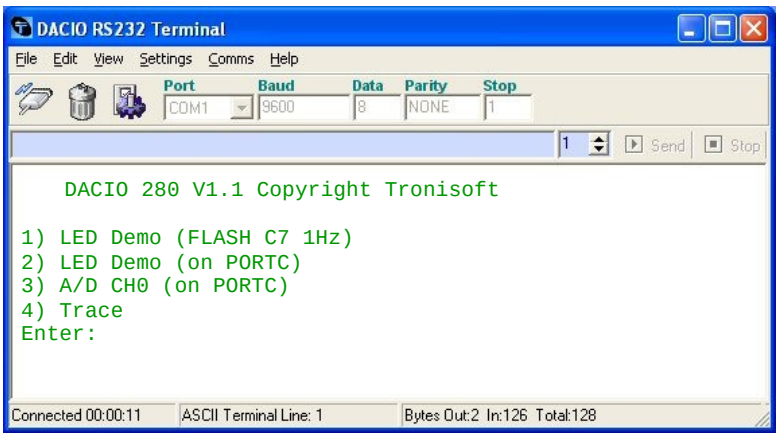
3) Run the DACIO RS232 Terminal (or a terminal emulator such as HyperTerminal configured to use e.g. COM1 and with line settings of 9600 baud, 8 data bits, no parity and no handshaking).

4) If the module is powered up and running then reset it by either shorting J6 (RESET) e.g. by placing a metal tipped flat blade screwdriver across the pins or remove power from the module for at least 2 seconds and then reapply.

5) Within 10 seconds of step (4), press the space bar twice in DACIO RS232 Terminal (or HyperTerminal).

   The LEDs on the module should start to flash indicating that the module is in demonstration mode.

   You will be presented with a menu option for selecting simple LED based demonstrations or choosing the trace feature.



   The trace option (press 4) allows you to enter commands into the module and see the responses. You will also get logic level port dump for digital I/O ports.

Trace output / port dump:

```
1) LED Demo (FLASH C7 1Hz)
2) LED Demo (on PORTC)
3) A/D CH0 (on PORTC)
4) Trace
Enter: 44
Running option 4...

Radix Mode          : Hex&Dec
Analog Mode         : 8 ch
PORTC Pull-up       : Off
I/O Mismatch Det    : Off
O/P Supp Level      : 1 char
_____________________________________________________
                    PORTC               PORTB
Logic Level:  0 0 0 0 0 0 0 0     0 0 0 0 0 0 0 0
              = 000               = 000
_____________________________________________________

I/O Setup:    0 0 0 0 0 0 0 0     1 1 1 1 1 1 1 1
              = 000               = 255
_____________________________________________________

Enter Command:
```

5.2 Enabling Running Mode

Resetting the module puts it back into “Running” mode by either shorting J6 (RESET) or removing and re-applying power.

Appendix A – Response / Error Codes

The following only applies if the module has been configured with the “Output Response Level” configuration setting set to “2 characters”. See section 4.3.7.

Normally if a command is not recognised by the module it will reply with a question mark and a carriage return: “?<0D>”. If more information is required then the module can be configured to report an error code as described in section 4.3.7 “Output Response Level”. The format is as follows: ?x<0D> where x is an error code. The module responds only after it receives the end character (;) or if a timeout occurs. Valid commands are acknowledged with “!A<0D>”.

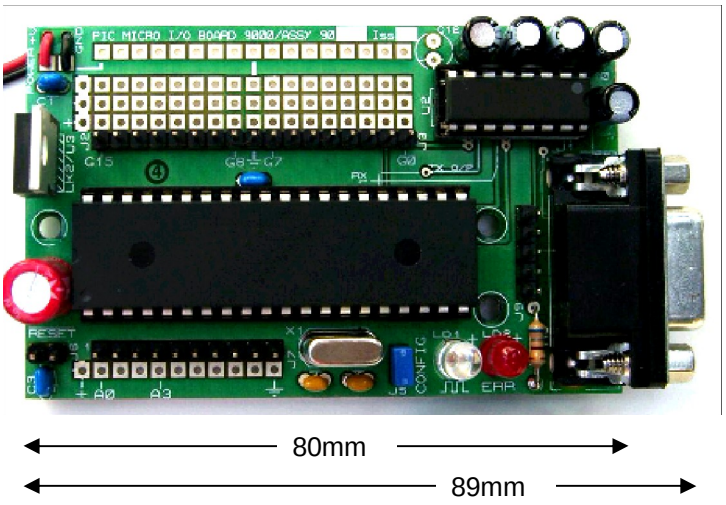
Note: If an error code is returned then the command will not have been executed. This means that the logic states on the module’s I/O pins will not have changed.

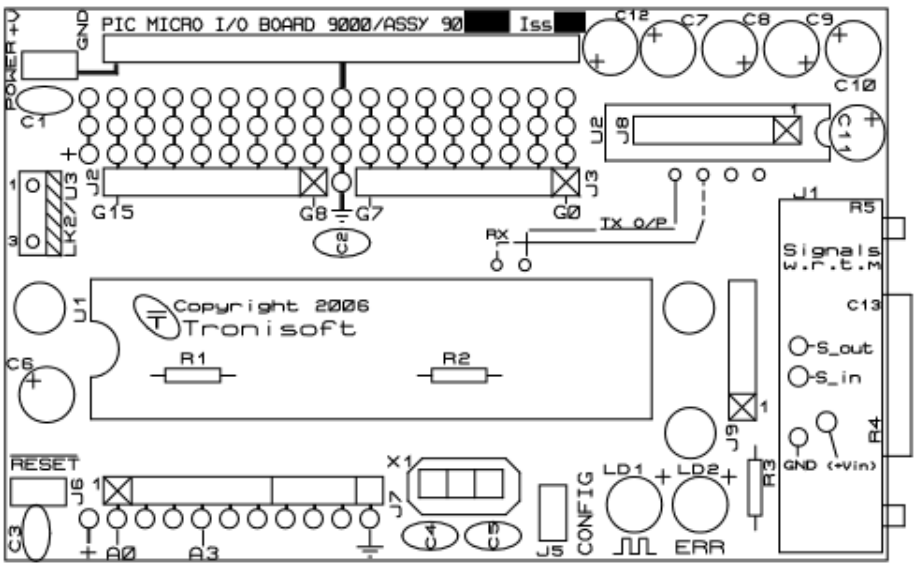
Table 3: Response/Error Codes.

‘A’	Response code - valid command acknowledged. The module will output "!A<0D>" and the command will have been executed successfully.
‘E’	Error code - expected x before or after '=', but received y. E.g. expected a number x in the command “!Bx=1;” to be 0-7 but received “!B8=1;”. The module will output "?E<0D>".
‘V’	Error code - overflow condition. E.g. writing 256 to (an 8-bit) PORTB.
‘M’	Error code – I/O configuration mismatch. E.g. writing a ‘1’ (logic high) to any bits that are defined as inputs. Example: Take the first command to be “!SB=15;” (set bits B0-B3 as inputs and B4-B7 as outputs). The following three commands illustrate its use. a) If followed with “!B3=1;” an I/O mismatch error will be generated because an attempt was made to write a ‘1’ to B3 (which is defined as an input line). b) If followed with “!B=3;” an I/O mismatch error will be generated as B0 and B1 are defined as inputs. c) The command “!B=0;” will not generate an error as writing 0 has no effect on an input line.
‘U’	Error code - unrecognised command. This is returned when commands are too badly formed to given an error code. The command string may be too long e.g. “!B=25090;”.

 © Copyright 2006 Tronisoft Limited. All Rights Reserved.

Appendix B – Board Image and Legend





Appendix C – ASCII Chart

ASCII, the American Standard Code for Information Interchange, was developed in the 1960's as a standard 7-bit code for identifying letters, numbers, symbols, and special characters in the English language. It was later expanded (as Extended ASCII) to include additional symbols and foreign language characters. Standard ASCII consists of 128 characters, ranging from 0 to 127 which can be broken down into the following subgroups:

0 to 31, 127: Control codes (includes null, backspace, line feed and others)
32 to 47, 58 to 64, 81 to 86, 123 to 126: Punctuation marks, mathematical (and other) symbols
48 to 57: Numbers 0 through 9
65 to 80: Capital letters A through Z
87 to 122: Lower case letters a through z

Standard ASCII Chart

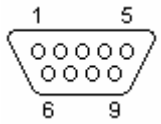
The standard chart includes the ASCII character or control and their related decimal and hexadecimal values. Also shown are the control key sequences for the control codes. Note, to obtain codes 0 to 31, console Control Key is pressed while simultaneously pressing a Letter Key, e.g. ^J is the line feed character. Control Key subtracts decimal 64 (40h) from Letter Key pressed.

Dec	Hex	^Key	ASCII	Dec	Hex	ASCII	Dec	Hex	ASCII	Dec	Hex	ASCII
0	0	^@	NUL (Null)	32	20	(Space)	64	40	@	96	60	`
1	1	^A	SOH (Start of Heading)	33	21	!	65	41	A	97	61	a
2	2	^B	STX (Start of Text)	34	22	"	66	42	B	98	62	b
3	3	^C	ETX (End of Text)	35	23	#	67	43	C	99	63	c
4	4	^D	EOT (End of Transmission)	36	24	$	68	44	D	100	64	d
5	5	^E	ENQ (Enquiry)	37	25	%	69	45	E	101	65	e
6	6	^F	ACK (Acknowledgement)	38	26	&	70	46	F	102	66	f
7	7	^G	BEL (Bell)	39	27	'	71	47	G	103	67	g
8	8	^H	BS (Backspace)	40	28	(	72	48	H	104	68	h
9	9	^I	HT (Horizontal Tab)	41	29	)	73	49	I	105	69	i
10	A	^J	LF (Line Feed)	42	2A	*	74	4A	J	106	6A	j
11	B	^K	VT (Vertical Tab)	43	2B	+	75	4B	K	107	6B	k
12	C	^L	FF (Form Feed)	44	2C	,	76	4C	L	108	6C	l
13	D	^M	CR (Carriage Return)	45	2D	-	77	4D	M	109	6D	m
14	E	^N	SO (Shift Out)	46	2E	.	78	4E	N	110	6E	n
15	F	^O	SI (Shift In	47	2F	/	79	4F	O	111	6F	o
16	10	^P	DLE (Data Line Escape)	48	30	0	80	50	P	112	70	p
17	11	^Q	DC1 (Device Control 1)	49	31	1	81	51	Q	113	71	q
18	12	^R	DC2 (Device Control 2)	50	32	2	82	52	R	114	72	r
19	13	^S	DC3 (Device Control 3)	51	33	3	83	53	S	115	73	s
20	14	^T	DC4 (Device Control 4)	52	34	4	84	54	T	116	74	t
21	15	^U	NAK (Negative Ack.)	53	35	5	85	55	U	117	75	u
22	16	^V	SYN (Synchronous Idle)	54	36	6	86	56	V	118	76	v
23	17	^W	ETB (End of Transmit Block)	55	37	7	87	57	W	119	77	w
24	18	^X	CAN (Cancel)	56	38	8	88	58	X	120	78	x
25	19	^Y	EM (End of Medium)	57	39	9	89	59	Y	121	79	y
26	1A	^Z	SUB (Substitute)	58	3A	:	90	5A	Z	122	7A	z
27	1B	^[	ESC (Escape)	59	3B	;	91	5B	[	123	7B	{
28	1C	^\	FS (File Separator)	60	3C	<	92	5C	\	124	7C	|
29	1D	^]	GS (Group Separator)	61	3D	=	93	5D	]	125	7D	}
30	1E	^^	RS (Record Separator)	62	3E	>	94	5E	^	126	7E	~
31	1F	^_	US (Unit Separator)	63	3F	?	95	5F	_	127	7F	(Delete)

Legend: ^*Key* - Control Key + Letter Key combination except 30 (1E hex) = Control +caret.

Appendix D – Pinout and Signals for the PC RS232 Connector

DB9 RS232 Port (IBM PC XT/AT)

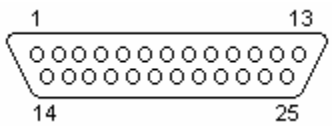


DB9 pin D-SUB male

Pin	Signal Name	Direction (DTE ← DCE)
1	CD (Carrier Detect)	←
2	RXD (Receive Data)	←
3	TXD (Transmit Data)	→
4	DTR (Data Terminal Ready)	→
5	GND (System Ground)	-
6	DSR (Data Set Ready)	←
7	RTS (Request to Send)	→
8	CTS (Clear to Send)	←
9	RI (Ring Indicator)	←

Note: a) Signal names are with respect to the computer/PC. b) Direction is from peripheral/modem (DCE) to the computer (DTE) e.g. RXD is the computer's input pin.

DB25 RS232 Port



DB25 pin D-SUB male

Pin	Signal Name	Direction (DTE ← DCE)
1	SHIELD (Shield/Protective Ground)	-
2	TXD (Transmit Data)	→
3	RXD (Receive Data)	←
4	RTS (Request to Send)	→
5	CTS (Clear to Send)	←
6	DSR (Data Set Ready)	←
7	GND (System Ground)	-
8	CD (Carrier Detect)	←
9	n/c	-
10	n/c	-
11	n/c	-
12	n/c	-
13	n/c	-
14	n/c	-
15	n/c	-
16	n/c	-
17	n/c	-
18	n/c	-
19	n/c	-
20	DTR (Data Terminal Ready)	→
21	n/c	-
22	RI (Ring Indicator)	←
23	n/c	-
24	n/c	-
25	n/c	-

Note: a) Signal names are with respect to the computer/PC. b) Direction is from peripheral/modem (DCE) to the computer (DTE) e.g. RXD is the computer's input pin.
c) Do not connect SHIELD(1) to GND(7).

 © Copyright 2006 Tronisoft Limited. All Rights Reserved.

Appendix E – Decimal- Hexadecimal Conversion Chart

Converting Hexadecimal to Decimal
This chart shows the conversion between hex and decimal.

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	000	001	002	003	004	005	006	007	008	009	010	011	012	013	014	015
1	016	017	018	019	020	021	022	023	024	025	026	027	028	029	030	031
2	032	033	034	035	036	037	038	039	040	041	042	043	044	045	046	047
3	048	049	050	051	052	053	054	055	056	057	058	059	060	061	062	063
4	064	065	066	067	068	069	070	071	072	073	074	075	076	077	078	079
5	080	081	082	083	084	085	086	087	088	089	090	091	092	093	094	095
6	096	097	098	099	100	101	102	103	104	105	106	107	108	109	110	111
7	112	113	114	115	116	117	118	119	120	121	122	123	124	125	126	127
8	128	129	130	131	132	133	134	135	136	137	138	139	140	141	142	143
9	144	145	146	147	148	149	150	151	152	153	154	155	156	157	158	159
A	160	161	162	163	164	165	166	167	168	169	170	171	172	173	174	175
B	176	177	178	179	180	181	182	183	184	185	186	187	188	189	190	191
C	192	193	194	195	196	197	198	199	200	201	202	203	204	205	206	207
D	208	209	210	211	212	213	214	215	216	217	218	219	220	221	222	223
E	224	225	226	227	228	229	230	231	232	233	234	235	236	237	238	239
F	240	241	242	243	244	245	246	247	248	249	250	251	252	253	254	255

Usage example:
Hex 31 is equivalent to decimal 49. Alternatively decimal 255 is equivalent to hex FF.

 © Copyright 2006 Tronisoft Limited. All Rights Reserved.