

---

---

# Interfacing IASI

---

---

# AN2



## AN2

### Interfacing IASI

Application note 2

April 2007

---

**Interfacing IASI**

---

**AN1**

---

## Contents

1.	Introduction.....	3
2.	RS232 Mode .....	3
2.1.	Electrical Interface .....	3
2.2.	Software Setup .....	3
3.	Microcontroller Mode.....	3
3.1.	Electrical Interface .....	4
3.2.	Software Setup .....	4
4.	Physical Connections.....	4
5.	Software Considerations.....	5
5.1.	Line End Charatcer .....	5
5.2.	Timing .....	6

# Interfacing IASI

# AN2

## 1. Introduction

This application note updates and supersedes application note AN1 and adds section 5 to deal with the software requirements for the correct operation of IASI devices.

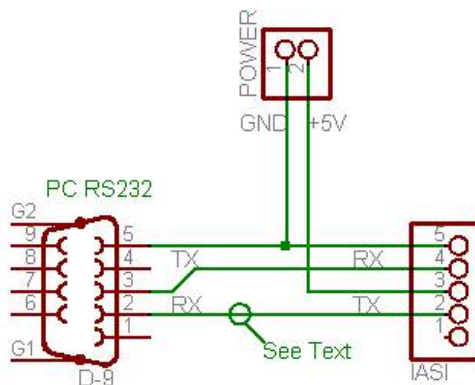
There are two distinct electrical possible ways of connecting an IASI device, RS232 Mode and Microcontroller Mode. Both are presented.

## 2. RS232 Mode

The RS232 Mode is referred to in software as the inverted mode. This is because the signals are in fact inverted when they leave the RS232 interface.

The main restriction of this mode is that multiple devices cannot feedback information. This is unimportant for some applications but if this is required than see the Microcontroller mode.

### 2.1. Electrical Interface



**Figure 1 RS232-IASI**

Figure 1 shows the basic connectivity from an RS232 9 pin connector to the IASI interface. The TX line from the IASI marked with a circle is only required for setting each individual device up. It is important to realise that in this mode the IASI will not transmit any information when using multiple devices.

The RS232 mode (inverted) is only intended for:

- Setting up (configuring) the device.
- Using devices that do not need to send any information, i.e. a display device.

### 2.2. Software Setup

Set up is done on single devices, it is assumed that you have an L> prompt, if not follow the instructions for returning to factory defaults.

1. Set the terminal Baud rate to the one you will be using say 9,600. Re-establish the L> prompt on the device if this is a new Baud rate by cycling the power on the device and pressing return 3 times.

2. Set the address of the device to any two characters, say D1 (**ZAD1**)
3. Set the Baud rate to 9,600 (**ZB0**)
4. Create a configuration slot as follows: (**ZC7 - w NNNYNNN** ). This will also reset the device. Note this just creates the slot it does not set the device to that configuration yet.
5. Optionally turn off error reporting (**ZD0**)
6. Now set the device to configuration 7 (**ZC7 -r**)

From now on in the device will only respond to its own address which must be preceded by a colon thus: (**:D1<command>**) where <command> is appropriate for the device.

#### NOTES:

- The device will be ready at switch on, there is no need to establish a Baud rate by pressing (sending) 3 carriage returns.
- The letters chosen for the address is case sensitive so in the above example using address D1 will work but d1 will be ignored.
- Each device is configured as above (with different addresses) **before connecting them together.**
- All devices will respond to address 00 (zero zero).
- Because the invert flag was set in configuration 7 above (step 4) then individually the device will still feedback information although there will be no prompt.
- To get the device back to factory defaults type :00ZF

## 3. Microcontroller Mode

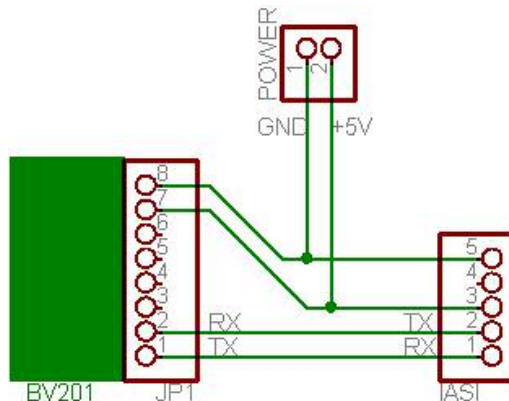
In this mode the TX and RX are available (except the BV4102). It is intended for connection to a microcontroller but this mode can still be used with RS232 providing a converter is used. Typically this will be a MAX232 type device. Two ready built devices are available from ByVac: the BV101 for connecting to USB and the BV201 for connecting to RS232. The circuit diagram is available for the BV201 at [www.byvac.co.uk](http://www.byvac.co.uk) in the datasheet.

For this example a BV201 will be used connected to a PC COM port.

# Interfacing IASI

# AN2

## 3.1. Electrical Interface



**Figure 2 PC-Converter – IASI**

Figure 2 shows the set up for connecting a PC to a RS232 converter and then to the IASI interface. The converter of course could easily be replaced by a microcontroller UART.

Both the TX and RX lines are relevant for individual set up and for multiple devices.

**Jumper:** refer to the actual device as to where this is but all jumpers should be removed (this may involve cutting the PCB track between the jumper holes) except one for optimal results. If only two or three devices are used, depending on other factors the jumpers may be left in place.

## 3.2. Software Setup

As the factory default is inverted (RS232) mode it is quite likely that you will set the devices up in inverted mode using a standard RS232 connection and then transfer to the arrangement in Figure 2 when the device is configured.

It is not absolutely necessary to do this as the IASI device will listen inverted and non-inverted regardless of the software set up. The arrangement in Figure 2 can be used for set up but there will be no feedback and so there will be no indication if mistakes have been made in the typing of the set up commands.

The starting point for the instructions that follow are factory defaults and connected to an RS232 (inverted) connection as Figure 1.

1. Set the terminal Baud rate to the one you will be using say 9,600. Re-establish the L> prompt on the device if this is a new Baud rate by cycling the power on the device and pressing return 3 times.
2. Set the address of the device to any two characters, say D1 ( **ZAD1** )
3. Set the Baud rate to 9,600 ( **ZB0** )
4. Create a configuration slot as follows: ( **ZC7 – w NNNNNNN** ). This will also reset the device. Note this just creates

the slot it does not set the device to that configuration yet.

5. Optionally turn off error reporting ( **ZD0** )
6. Now set the device to configuration 7 ( **ZC7 –r** )

From now on in the device will only respond to its own address which must be preceded by a colon thus: ( **:D1<command>** ) where <command> is appropriate for the device.

Plug in the device using arrangement as in Figure 2 and type **:D1ZV** and this will return the version information.

### NOTES:

- The device will be ready at switch on, there is no need to establish a Baud rate by pressing (sending) 3 carriage returns.
- The letters chosen for the address is case sensitive so in the above example using address D1 will work but d1 will be ignored.
- Each device is configured as above (with different addresses) before connecting them together.
- All devices will respond to address 00 (zero zero).
- The configuration in the example has ACK switched off, use ZC7 –w YNNNNNN if you require an acknowledge at the end of each command.
- To get the device back to factory defaults type :00ZF

## 4. Physical Connections

There are many ways to achieve this from simply soldering the pins together or to use some kind of connector. The method shown here is probably the simplest and most versatile.

It uses IDC (Insulation Displacement Connectors) which means there is no soldering or special crimp tools involved.

5 way IDC connectors are not widely available and so a 10 way is used but as it turns out this has some advantages.

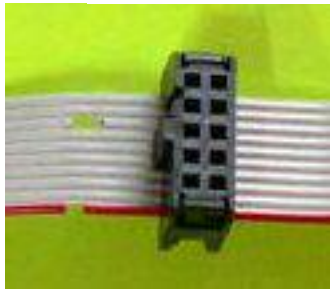
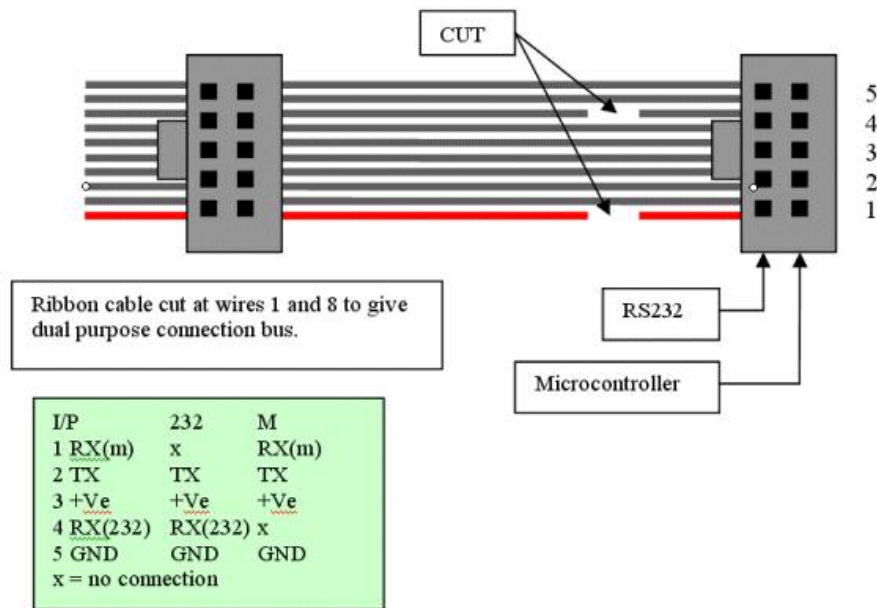


Here are 3, 10 way IDC connectors crimped onto 10 way ribbon cable. This is sufficient for connecting both RS232 and Microcontroller type interfaces.

To facilitate this the wire is cut after the installation of the connectors.

# Interfacing IASI

# AN2

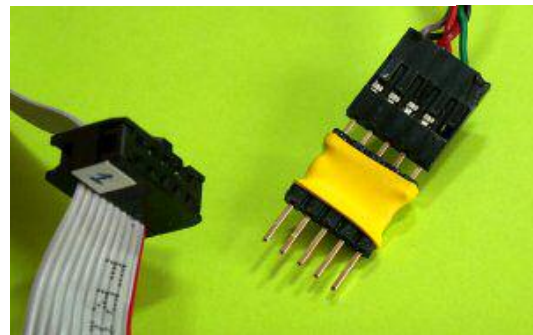


This is shown better in this close up. One row of 5 connectors can be used for RS232 and the other row can be used for Microcontroller connection. Thus one cable can be used for both applications.

The diagram shows how to connect and cut the cable. Using this method there is no limit physical to the size of the bus. On the front cover are 10 IASI devices driven by an RS232 interface. The back of this is shown here:



The connection to the standard RS232 connector is simply 2 pin connectors back to back.



## 5. Software Considerations

This section has been added as a result of feedback from users and deals with connecting multiple or a single device to a PC or microcontroller and controlling the device with software, say C++ or C#

### 5.1. Line End Character

The IASI protocol works by interpreting a 2 character command and then acting upon it but how does IASI know when the command has been sent?

The key to this is that the IASI will continue to receive information until it receives a line terminator character, this by default is set to CR (carriage return) ASCII 13. The device will do nothing other than receive test until this character is received.

Microcontroller and PC software is not consistent in the way line end is handled, all combinations of CR and LF (line feed ASCII 10) may be sent and this may or may not cause problems with the IASI device.

For consistent results the software should send only one terminating character, this can usually be arranged by using the "\r" switch in the printf() function. If this can't be arranged then

---

## Interfacing IASI

---

## AN2

---

the ZL command is provided to enable the line end character to be changed. As an example using % as the terminating character the ZL command would be:

```
ZL37
```

37 is the ASCII code for %. The IASI device will now only respond when this character is received so to change the address of a device to say 01 the following would now be needed:

```
Printf("ZA01%");
```

Any LF or CR sent along with the printf(0 function would be ignored because the terminating character is now %.

### 5.2. Timing

A second problem that may be encountered in a microcontroller set up is timing. The IASI device will accept all input until the input buffer is full at which point the buffer is cleared. If this happens an error will be reported.

When a line end character is received the buffer is cleared and the command interpreted and acted upon, whilst the command is in progress, characters will still be accepted into the buffer but if the commands are sent faster then the command can be processed the buffer will eventually become full and an error will occur.

This is particularly noticeable in inherently slow devices such as an LCD display that takes a long time ( several ms ) to clear the display. The actual timings are best found by trial and error.

There are several ways to handle this; one is simply to allow enough time for the worst case. A second more elegant way is to get feedback from the device.

An IASI device is capable of presenting feedback when a command has been completed, this can be used to determine when to send the next command. By default the IASI puts out a prompt L> which is in fact the following ASCII string:

```
76 62 13
```

Note the CR at the end, a very simple way is for the software to look for the 62 or 13. A more elegant way but takes more setting up is to use the ACK mechanism. This will need the configuration changing but is fully explained in the datasheet for the device. Basically the IASI can be configured to give no prompt but just an ACK character (can be any chosen character). The software can then simply look for this character.