
Keypad Interface

BV4107



BV4107 Keypad Interface

Product specification

Jul 2006 V0.a

Keypad Interface

BV4107

Contents

1.	Corrections to this section	4
2.	Introduction.....	4
3.	Specification	4
4.	Electrical Interface.....	4
4.1.	IASI Connector	4
4.2.	The 14 pin Connector.....	5
5.	Electrical Specifications	5
6.	Commands	5
6.1.	KB.....	5
6.2.	KC.....	6
6.3.	KD.....	6
6.4.	KG.....	6
6.5.	KI.....	6
6.6.	KM	6
7.	Setting Up a Keypad	6
8.	Error codes.....	6
9.	Factory Reset.....	6
10.	The Key Map.....	6
11.	Revisions to the IASI Section	8
12.	Introduction to IASI.....	8
13.	IASI Electrical Interface.....	8
14.	Serial Connections	8
15.	Factory Configuration.....	9
16.	Non-Inverted Mode.....	9
17.	Commands	9
17.1.	Addressing	10
17.2.	Command Format.....	10
17.3.	Command Parameters.....	10
17.4.	Command Line (Buffer).....	10
18.	Command Set.....	10
18.1.	The Star *	10
19.	Common Command Set (Z)	11
19.1.	Summary	11
19.2.	ZA.....	11
19.3.	ZB.....	11
19.4.	ZC.....	12
19.5.	ZD.....	13
19.6.	ZF	13
19.7.	ZK.....	13
19.8.	ZL	13
19.9.	ZM	14

Keypad Interface

BV4107

19.10.	ZR.....	14
19.11.	ZT.....	14
19.12.	ZV.....	14
19.13.	ZW.....	14
20.	Z Command Error Codes	15
21.	Connecting and Configuration	15
21.1.	Reconfiguring	15
22.	Microcontroller Use	16
22.1.	Simple	16
22.2.	With feedback.....	16
22.3.	Baud rate	16
22.4.	Multiple Devices	16
23.	Restoring Factory Defaults.....	17
23.1.	Software	17
23.2.	Hardware	17
24.	Watchdog	18
25.	Troubleshooting	18

Keypad Interface

BV4107

1. Corrections to this section

Rev	Change
Jul 2006	Preliminary

2. Introduction

The BV4107 is a keypad interface designed to accept up to 16 keys in a 4 x 4 matrix. The board will interpret key presses and translate this into text values.

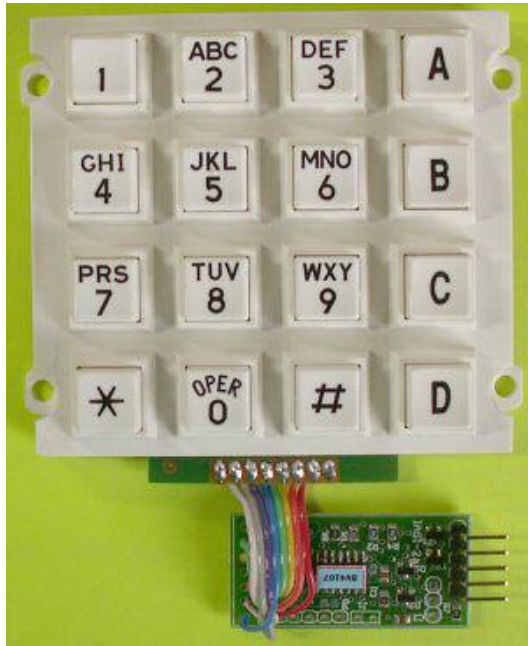


Figure 1 The BV4107 connected to a 4 x 4 keypad

Key presses are debounced and buffered into a first in first out (FIFO) key buffer that will hold 16 keys. Reading the key will decrement the buffer count.

There is a 'key down' facility that can detect when the key is actually pressed down. This can be useful for interactive applications.

In addition to the above there is a single line input which can be read as logic 1 or 0 depending on the value presented to it.

All commands can be entered as simple text and the results seen immediately. There is no complex programming to do, the device is very easy to use.

3. Specification

- up to 16 keys 4 x 4 matrix
- 16 key FILO buffer
- Key down detect
- additional single bit input
- +5V supply

- Low power 2.4mA
- IASI interface

4. Electrical Interface

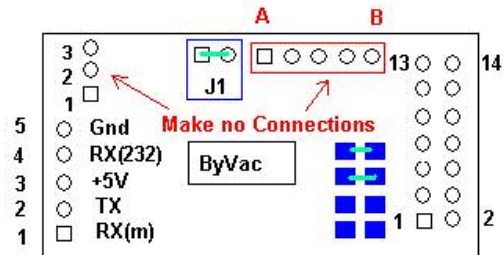


Figure 2 Connections

The interface comprises of the 5 pin IASI and the 14 pin connection to the keypad. Not all of the 14 pins are used in this application.

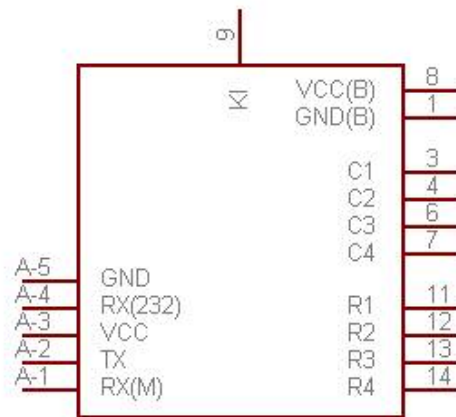
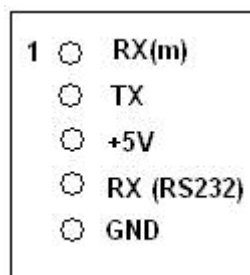


Figure 3 Schematic

Not all of the pins are used on the 14 pin connector, and some are duplicated on the IASI connector. The schematic shows the pins that are used for each of the 2 connectors. C1 to C4 should be connected to the column inputs and R1 to R4 should be connected to the row inputs, although as can be seen later it does not really matter which way round the columns and rows are connected.

For quick reference the IASI electrical connections are shown here:

4.1. IASI Connector



Keypad Interface

BV4107

This is the standard connector and the interface is fully described in section 13.

4.2. The 14 pin Connector

Pin	Cmd	Description
1		Ground
2		+Ve [1]
3		C1 [2]
4		C2
5		Ground
6		C3
7		C4
8		+Ve Power (also pin 3 IASI)
9	KI	Digital input [3]
10		n/c
11		R1 [4]
12		R2
13		R3
14		R4

Table 1 14 Pin connector

NOTES

[1] This is a filtered output to power the microcontroller and offers some immunity from noise to the circuit. It is the supply fed through a 100R resistor so if something is connected to this it should not draw more than 1 or 2 mA.

[2] Columns are inputs with internal pull up resistors. If less columns are required then the column input should be left disconnected, there is no need to tie it high

[3] This input does not have an internal pull up resistor. If this is being used for say a switched input then an external resistor should be provided.

[4]. Rows are active low outputs and will scan the keypad.

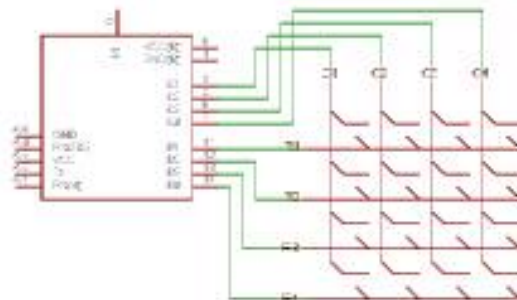


Figure 4 Typical connection

Connection to a keypad is very simple as shown in Figure 4. It does not matter if the columns and rows are the wrong way round as there is

an opportunity of changing the key values in software, see the **KM** command.

There is a single input on pin 9 that can be used as an extra switch input, this is accessed via the **KI** command.

Keypads with less rows and columns can be used, simply leave the row or column disconnected but row 1 column 1 should be used first.

5. Electrical Specifications

Supply voltage 4.5V to 6.0V

Digital o/p 20mA

Current Consumption 2.4mA @ 5V

6. Commands

The following command descriptions refer specifically to this board. See also section 17 for the IASI specific commands.

Command	BV4105 Analogue
KB	Key buffer
KC	Clear key buffer
KD	Key down
KG	Get key from buffer
KI	Get input from pin 9
KM	Key map
KS	Get key scan code from buffer

The star *

Most commands can have a * following the command as an option. A command by default will return the value in text. **KG** for example on an empty key buffer will return 255. This is 3 bytes of information. It is not always convenient to read 3 bytes when used as part of another program. Work may be needed to convert this back to a number. Also **KG** may return a single or double byte depending on the user input, this equally applies to other commands. Using the * option will direct the command to return the value in binary which may be easier to handle.

6.1. KB

Name: **Keypad Buffer Indicator**

Command Parameters: **[*]**

Typical Use **KB**

KB is used to determine how many keys there are in the keypad buffer. Each time a key is pressed the buffer pointer is incremented and the key stored.

The key buffer is a first in first out type and so when reading the keys, they will be in the same

Keypad Interface

BV4107

order as input. The buffer will hold 16 key presses and then reset to 0 so it is up to the user to read the keys before the buffer fills.

When the buffer resets to 0 on overflow any key information that may have been stored in the buffer is lost.

KB will return 0 if there are no keys in the buffer.

6.2. KC

Name: **Clear key buffer**

Command Parameters: **None**

Typical Use **KC**

Resets the key buffer to 0 and removes keys from the buffer. This is in effect a key buffer reset.

6.3. KD

Name: **Key Down**

Command Parameters: **[*]**

Typical Use **KD**

This command will return either a 0 or 1. If the command is issued at the same time that a key is being pressed it will return 1.

This command is useful for interactive type input, a volume control for example where the user will keep a key down until the desired value is reached. The value of the key can be determined by reading the key buffer with the **KG** command.

6.4. KG

Name: **Get key**

Command Parameters: **[*]**

Typical Use **KG**

Gets one key from the keyboard buffer, if there are no keys in the buffer 255 (ff) is returned by default but this value can be changed using the **KM** command.

Key values are returned in the same order as they were entered so if the user presses 1,2 and 3 then subsequent calls to KG would result in the values 1,2 and 3 in that order.

6.5. KI

Name: **Input from pin 9**

Command Parameters: **[*]**

Typical Use **KI**

There is a single input line on pin 9 that can be read. If this is high then a 1 is returned if low then a 0 is returned.

An extra switch or sensor can be added to this pin and read with the **KI** command. Note that this input does not have an internal pull up resistor as the row inputs do so it is up to the

user to take care of this. Leaving the line unconnected will produce indeterminate results when using the **KI** command.

6.6. KM

Name: **Key Map**

Command Parameters: **[n][n]**

Typical Use **KG4**

When a key is pressed a scan code is produced, this code is an address from 0 to 15. This address is used to access values in an EEPROM table. The key map command allows the alteration of these values.

The key map command is useful in that it allows any value (0-256) to be assigned to any key, it also enables the rows and columns to be wired in any order because the KM command will be able to remap to the correct values.

As well as the 16 values (0-15) there is an additional value at address 16. This value is the return value used by the KG command to indicate that the key buffer is empty. By default this is set to 256 (0ffh).

To read an address simply follow the command with an address:

KM7

To change the value of an address follow the address with the value:

KM7 9

The above will set the address 7 to value 9. NOTE there must be a space between the address (7) and the value (9).

See the example at the end of section 10

7. Setting Up a Keypad

See the quick start guide for an example of setting up a keypad from scratch and also the key map in section 10.

8. Error codes

There are no error codes specific to this board. Standard IASI error codes apply.

9. Factory Reset

Because this is a purely input device there is no physical indication that a factory reset has occurred. For more information about this see section 23.

10. The Key Map

The key map is an area in EEPROM that allows the key scan codes to be mapped to a user value. This allows the use of odd values but more importantly it enables great flexibility for wiring the rows and columns. They do not need to be in any particular order.

Keypad Interface

BV4107

When using the KS command it will return the scan code from the keypad and the values will be as Figure 5.

R1	14	30	46	62
R2	13	29	45	61
R3	11	27	43	59
R4	7	23	39	55
	C1	C2	C3	C4

Figure 5 Decimal scan code values

If a scan code of 46 is returned then row 1 and column 3 was connected together to produce this. Likewise a scan code of 7 must mean that row 4 and column 1 was connected. The KS command is used to discover this.

The KG command translates the scan codes to an address of between 0 and 15 as Figure 6.

R1	0	4	8	12
R2	1	5	9	13
R3	2	6	10	14
R4	3	7	11	15
	C1	C2	C3	C4

Figure 6 Address Translation

A scan code of 46 will produce an address of 8 and a scan code of 7 will produce address 3. The addresses are not directly accessible but their values in EEPROM are.

The address in Figure 6 is used to look up a value in the EEPROM table

A	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
V	1	4	7	15	2	5	8	0	3	6	9	14	10	11	12	13	15

Figure 7 EEPROM Values

Where A is address in yellow and V is the value in pink. By using a table any value can be assigned to any key (the pink row is user configurable using the KM command). The default values have been set up for a standard keypad shown in Figure 1, where the top left (R1, C1) is marked as 1 and the bottom right is marked as D.

Looking at Figure 6 the top left address is 0 which is associated with a value of 1 (hence KG will return 1) in the EEPROM table.

The last address (16) is used by the KG command. If the command is activated and there are no keys in the key buffer KG will return this value. All of the 16 EEPROM values can be viewed and changed using the KM command.

Example

When the keypad key '1' is pressed this returns a scan code of 46, this is the default for key 3 and so needs to be changed. Note the position of 46 in Figure 5, this corresponds to a value of 8 in Figure 6. The keymap command therefore needs to be:

KM8 1

The procedure is:

1. Discover the scan code of the key by using the KS command
2. Look up the position of this code using Figure 6, use this value as the first part of the KM command
3. Use the value of the physical key for the second parameter of the KM command.

Keypad Interface

BV4107

11. Revisions to the IASI Section

Rev	Change
May 2006	Additional information about connecting multiple devices and hardware factory reset.
June 2006	Troubleshooting guide added
V1.01	

12. Introduction to IASI

The Intelligent Asynchronous Serial Interface (IASI) is a common standard that makes it much easier to control and use hardware from either a standard communication interface (terminal) or a microcontroller.

It is based on a very simple text command set and a flexible hardware interface. The 'Intelligent' aspect is derived from the fact that each particular IASI knows about the connected hardware so a simple command can make the hardware perform a reasonably complex function. Scroll text on an LCD display for example.

when used in a microcontroller system this enables the controller and designer to concentrate on the important aspects of the design and control rather than the mundane job of controlling the hardware. It also means that the task of driving common peripherals is not being constantly re-invented.

13. IASI Electrical Interface

The device has very simple requirements. A power supply, transmit and receive lines as shown in table E1.

The interface is specifically designed so that it can be connected to either a standard com port (on a PC for example) or directly to a microcontroller UART or even a microcontroller port pin with a software generated UART (Universal Asynchronous Receiver and Transmitter). A five pin connector is used with normally only 3 or four pins being connected at any one time.

There are **TWO** receive lines, pin 1 receive line will accept normal 5V logic as presented by a microcontroller pin or UART and pin 4 will accept positive and negative voltages up to 15V that are normally present on a standard RS232 interface. Pin 4 will also invert the logic which is also normal for this interface.

The Baud rate is automatically detected at start up or it can be configured in software to a fixed, default baud rate. The device detects the baud rate on receiving a CR character (0Dh). Other received characters will be ignored until the Baud rate has been established.

The transmit pin has an open collector output that has a pull-up resistor on board connected through a jumper. Where more than one device is used on the same serial line, only one jumper should be shorted. See the section on multiple devices for further information.

14. Serial Connections

The device is designed to work in either of **two** modes: an **INVERTED** mode for connecting directly to an RS232 port (factory default) or a **NON-INVERTED** mode for connecting to a microcontroller UART.

As previously described there are two inputs, one for each alternative interface. On the transmit side (output from the interface) there is only one pin that takes care of inverted and non-inverted logic, this is configured in software. The output is 0 to +5V only, rather than the RS232 specification requiring positive and negative signals.

On most RS232 specification interfaces this will work although it is not within the actual RS232 specification.

Keypad Interface

BV4107

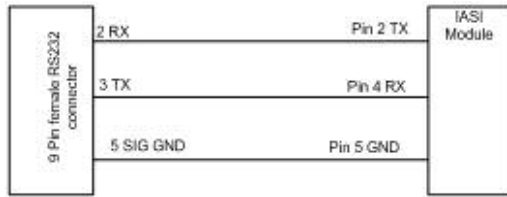


Figure 8 Connection to a PC

Figure E1 shows the connections to a 9 pin D type connector found on most PC's.

The Baud rate is automatically determined by detecting the first Carriage return it receives or it can be configured to a fixed baud rate in software. See the section on commands (**ZC**) for further details.

15. Factory Configuration

When an IASIM (Intelligent Asynchronous Serial Interface Module) leaves the factory it is configured to automatically detect the baud rate and interface with a standard PC com port. Regardless of the configuration the device will receive both forms of electrical interface (inverted and non-inverted) depending on which pin is used. So even if it may not be possible to see the output from the device, it will normally accept the input.

The transmit is configured by software and this is configured to invert. The interface does not need this pin to be connected to receive commands.

Factory settings can be restored both in software and hardware. The command ZF (see later) will restore the factory configuration. To restore factory defaults using hardware see Appendix A

16. Non-Inverted Mode

As previously mentioned the device is capable of operating with a standard RS232 communication port and this is the factory default and is useful for configuring and learning the device capabilities.

Most microcontrollers however operate on non-

inverted logic and output 0V for zero and +5V for logic 1. To operate in this mode issue the command:

ZC3 -r

See Section 21 or the command set for more information.

17. Commands

The interface is completely software driven, all commands and configuration are done through a serial interface. The only exception to this is the hardware factory default restore, see Section 23.

A very simple command protocol is used that essentially operates in two modes. An interactive mode and an addressing mode. The general format of the command is:

Interactive

<command><command-parameters><CR or ,>

e.g. **ZA**

Addressing

:<address><command><command-parameters><CR or ,>

e.g. **:00ZA**

The purpose of the interactive mode is to enable the user to learn the device capabilities and also to be used in a system where only one device is connected to the bus.

This is the default protocol the prompt for this is:

L>

The 'L' indicates that the device is in 'Listen' mode and will react to any command given and also produce error messages if non-existent commands are given.

An alternative protocol is available that allows multiple devices to be connected to the same serial bus. This is configured using the **ZC** command (see later).

In non-listen mode the prompt is:

Pin	Name	Description
1	RX	Receive data in non-inverted form at +5V logic levels. Use this pin for connecting to MAX232 devices or directly to microcontrollers.
2	TX	Transmit (output) data. This is 0V and +5V, RS232 levels are not used. Devices will work without this connected but no feedback can be received. This pin is configurable in software to transmit either normal or inverted logic. (see multiple devices section 22.4)
3	+5V	Standard 5V power to the device
4	RX-Invert	Receive data (input) this will accept -12V to +15V volts in inverted logic as is normally available on a PC Com port. The format is RS232 1 start bit 8 data bits and 2 stop bits.
5	GND	Ground

Table E2 Serial Connection Details

Keypad Interface

BV4107

:>

This indicates that a colon and address is required. In this mode the device will only accept commands with its own address and ignore everything else. In this mode the previous **ZA** command would be:

:00ZA

Assuming that the device address is 00.

Note that all addresses must be preceded by a colon. The prompt can be turned off by the **ZC** command.

17.1. Addressing

All devices have a two **character** address, this enables several devices to be connected to the same serial lines.

The address can be any two ASCII characters and it is case sensitive. Typical examples are **L1 AA 25**, etc. note that **aa** will be a different address to **AA**.

There is a special address **00** (zero, zero) that all devices will listen to regardless of their own address. There is nothing to stop more than one device having the same address, this can be useful if both devices always require the same commands.

As mentioned earlier if the device has been set to interactive mode and address is not required.

17.2. Command Format

The command is a two character string **ZA** for example and is **not** case sensitive, so **zc** would be the same as **ZC**. All commands have two characters and in general the first character is a class of command and the following letter is a subclass of that command. All commands beginning with **Z** are common to all devices and usually deal with configuration and communication.

Other two letter commands deal with the device direct and the same commands may mean different things for different devices.

17.3. Command Parameters

The command parameters can be anything and are specific to that command. In most circumstances spaces are ignored and can be used freely. Spaces do however serve to separate parameters (space is the delimiter) where more than one is required. For example given the command below that is used on the LCD IASM

L>WT3 "Hello"

The '3' requires a space after it so that the number can be distinguished from the text. In general though spaces are ignored and can be used freely.

17.4. Command Line (Buffer)

All commands strings are buffered in a 64 byte buffer and devices will store any text received on the bus. Any lines greater than 64 characters will be ignored and generate an error so it is important not to exceed this figure on an automatic system.

Providing the debug level is set to greater than zero (see **ZD** command) then this will be reported as an error.

When a Carriage Return (ASCII 0dh) is received (configurable) the text string is passed to the command processor. In non-listen mode only devices with a matching address will respond, two or more devices can have the same address if required.

The line end character is set at the factory to be 13 (0dh) and this can be changed using the **ZL** command. Some systems send only this others send a Line Feed (10, 0ah) and others send both, either way round. The IASI will only accept the configured value and ignore or skip over anything else.

A comma (,) can be used to allow more than one command in a single line.

This is especially useful in non-listen (address) mode as the device only needs addressing once:

:>:00ZAL2

:>:L2ZD1 or :>:00ZD1

can be sent as:

:>:00ZAL2,ZD1

(prompt)

(command)

The above will set the device address to L2 and the debug level to 1.

18. Command Set

Command sets consist of a two character command. The first character usually refers to the general type and the second later is specific to that command. The Z command set for example is the common command set for all devices. The Z commands configure the device interface and communications.

18.1. The Star *

As a general rule most commands issued on their own will return their value in printable format for example **ZD** will return the current debug level 0,1 or 2. This will be 'printed' on the terminal screen.

In absolute terms the values returned are in fact the values 30h, 31h or 32h which correspond to the printable values 1,2 or 3 (the ASCII codes for these characters).

When used with a microcontroller for example it may be more convenient to know the actual

Keypad Interface

BV4107

BINARY value. By using a * this will return the actual value thus.

ZD*

The above will return the actual binary value 1,2 or 3 which will not be printable on the terminal screen. This may be particularly important for numbers greater than 9 as the printed version of 10 is in fact 1 (31h) and 0 (30h), two bytes. Whereas the actual value 10 is only one byte and much easier to handle.

19. Common Command Set (Z)

A summary of the complete common command set is given in the table. All common commands begin with Z, as previously mentioned commands are grouped by the first letter so any command beginning with a Z is a common command.

A full description of each command is given after the summary.

19.1. Summary

Command	Description
ZA	Address
ZB	Baud Rate
ZC	Device configuration
ZD	Debug level
ZF	Factory reset
ZK	Ack character
ZL	Line end character
ZM	Record Macro
ZR	Reset
ZT	Test macro
ZV	Version
ZW	Wait

Note that any example given will assume that the device is in interactive mode (listen flag on) so no address is required to precede it.

19.2. ZA

Name: **Address**

Command Parameters: **[*address]**

Typical use **:00ZA or ZA (listen on)**

This command will show, confirm or set the address. To see the address of the current device simply use the command without any parameters:

ZA

This will return the actual address of the device, this will be 00 after factory reset.

ZAK1

This will set a new address for the device, K1. Addresses consist of two characters (including numbers) and are case sensitive so setting the address to AA will be different from aa.

:K1ZA*K1

This command is used for confirming that a device is on the bus and working okay. In the above if device K1 is on the bus it will respond with ACK (see the ZK command for ACK). The single ACK character is usually easier to detect by a microcontroller than the two byte address returned without the * parameter.

19.3. ZB

Name: **Baud rate**

Command Parameters: **[*][rate]**

Typical use **:00ZB or ZB (listen on)**

At factory default the Baud rate is detected by receiving a carriage return character (CR value 13, 0dh) from the sending device. A minimum of two are required, one for the detection and the other for confirmation. The quality and length of the serial bus may also effect the detection process.

The detection will chooses from a fixed set of baud rates:

0. 9600
1. 14400
2. 19200
3. 38400
4. 57600
5. 115200

This has been found to be very reliable depending on the quality of the electrical interface.

The command enables viewing and setting of any of the above values 0-5 that correspond to the Baud rate. If the Baud Detect flag (see the ZC command) is set to off (N) then the Baud rate will be determined by whatever this command is set to.

As an example if ZB is set to 3 and Baud detect is off, the device will communicate at 38400 Baud when switched on. If ZB is set to 3 and Baud detect is off then the Baud rate will be determined by the first acceptable Carriage Return character received.

ZB	Baud Detect	Baud Rate
3	Off	38400
3	On	Set by receiving CR character from host device.

Keypad Interface

BV4107

Config	Byte	Ack	Macro	Baud-d	Invert	Listen	Prompt	Echo	
0	1E	n	n	y	y	y	y	n	Inv-interact
1	1A	n	n	y	y	n	y	n	Inv-colon
2	58	y	n	y	y	n	n	n	Inv-auto
3	16	n	n	y	n	y	y	n	Interact
4	12	n	n	y	n	n	y	n	colon
5	50	y	n	y	n	n	n	n	Auto

The Baud rate can also be reported in binary, in the above example:

BR

will return 3 (ASCII value 33h)

BR*

will return the value 3 which will not be a printable character, this is useful for microcontroller input, see the section on the star command extension.

19.4. ZC

Name: **Configure**

Command Parameters: **n [*][-w aaaaaa][-r]**

Typical Use **:00ZC3 -r**

This is probably the most complex command but will alter the way the device behaves and is only likely to be seldom used. There are 7 'flags' that will effect the device and these are:

1. ACK
2. Macro
3. Baud detect
4. Invert
5. Listen
6. Prompt
7. Echo

The above flags can either be on or off (Y or N) and they have the following meaning:

ACK

When the line end character (ZL command) is received by the device there is an option to send the ACK character (ZK command). If this is set to on (Y) then the ACK character will be sent. See the ZK command for details on why you would want to do this.

Macro

When the device is first powered on, optionally a set of commands can be run, see the ZM, ZT commands. If this flag is set to Y then the macro commands will be run at start up.

Baud detect

The device can operate at a fixed Baud rate or detect the host Baud rate at start up. When this is set to Y the device will wait for input from the controller Terminal to determine the Baud rate, see the ZB command)

Invert

This only applies to the transmit line of the device. When set to Y all transmitted data will be inverted, this is suitable for connection to normal RS232 type interfaces, a PC com port for example.

Listen

If this is set to Y a command will be accepted without addressing, this is useful for single devices connected to the bus and saves having to prefix all commands with colon, address.

Prompt

When set to Y will output on the transmit line a prompt. This is useful for experimenting with the device when using a standard terminal. There are two types of prompt depending on how the listen flag is set, see the section on addressing.

Echo

When this is set to Y all characters received will be transmitted to the transmit line. It is not advisable to have this on when used with automated systems. It may also slow down the command reception process.

The technique for setting these consists of storing a configuration in non-volatile memory and then restoring this configuration later.

There are eight configuration locations that can all be used. When restoring factory defaults the first 6 locations are overwritten with the values shown in the table.

To observe a configuration use:

ZCn or ZCn*

where n is a number between 0 and 8 using ZCn without a * will produce a sting of Y and N. As an example:

ZC0 will produce **NNYYYYN** given the factory defaults.

ZC0* will return 1Eh which will not be printable using a Terminal but will be more understandable to a microcontroller.

To change a configuration **-w** is used to indicate 'write'. As an example to set a new configuration in slot 6 the following command can be issued:

ZC6 -w NNYYYYN

This will write a new configuration to slot 6 having the same values as the factory default, configuration 0. Any slot (configuration) can be overwritten at any time but using the **ZF**

Keypad Interface

BV4107

command, factory defaults, will reset the first 6 configurations to that in the above table.

To set the device to a configuration **-r** is used (read), so:

ZC0 -r

will set the device to configuration 0.

NOTES

The purpose of the 8 configuration slots are to enable quick changing of device configuration. In most cases one of the default configurations will suffice and so the user may not be concerned with the individual aspects of each flag.

The first 3 (0-2) configurations are intended for using on PC terminals that have an inverted logic. The next three are intended for use with a microcontroller output that does not have an inverted logic.

There is nothing to stop the user from specifying illogical configurations or even using a slot that has not been previously configured. This can make the device very difficult to communicate with. If this should happen than a hard reset will probably be required, this is described in Section 23.

NOTE The use of this command with -r or -w will cause the device to reset, similar to using the ZR command.

19.5. ZD

Name: **Debug**

Command Parameters: **[*][n]**

Typical Use **:00ZD0**

There are three debug levels available. level 0 will not report any errors at all. This is used where multiple devices are on the same bus and the error reporting may interrupt normal command flow.

Level 1 will report to the normal transmit line and level 2 will report to the device if this is possible, LCD displays for example. Note if the device is not capable of displaying the error then level 2 will be equivalent to level 0.

As in the previous commands ZD on its own will report to the terminal in ASCII form and ZD* will report in binary form.

A new level can be set by following the command with the new level, e.g.

ZD1 set debug to level 1.

19.6. ZF

Name: **Reset to Factory Defaults**

Command Parameters: **none**

Typical Use **ZF**

This will set the device to the factory default configuration, see section 23 for what this is.

Providing there is communication with the device this command can be useful for establishing a known configuration automatically. For example:

```
ZF
CR (carriage return)
CR
:00ZF
CR
CR
```

Should set the factory defaults even if the device is in interactive mode or not. This can then be followed by a desired configuration, **ZC3 -r** for example.

19.7. ZK

Name: **ACK (acknowledge)**

Command Parameters: **[*][n]**

Typical Use **ZK6**

When a line end (CR) is sent to the device the contents of the buffer are interpreted for a command and then acted upon. This takes a finite time to complete. In an automated system a delay can be used to allow for this. However the delay must be set for the longest time to ensure that all commands can be dealt with.

Optionally (see the ZC command) an acknowledge (ACK) mechanism can be switched on. If this mechanism is switched on, when the command has completed an ACK will be sent to indicate to the controller that another line is ready to be accepted.

This forms a simple but reliable handshake method that can be used for sending text files to a device or for using automated devices.

The ACK character can be set or displayed using this command.

ZK displays the character represented as a decimal printable number, **ZK*** will return the ACK character as a binary value and **ZK6** will set the ACK character to the factory default value of 6

19.8. ZL

Name: **Line end character**

Command Parameters: **[*][n]**

Typical Use **ZL13**

The line end character (EOL) is important as it informs the IASI to interpret the contents of the buffer. Unfortunately this character is not in universal use, some systems send 13, other send 10 and others send a combination of both which can be either way round.

The ZL command allows the specification of the character that will mark the end of line. This also gives the opportunity for any special requirements, e.g. 0 can be specified.

Examples:

Keypad Interface

BV4107

ZL returns the value as a decimal number in text form.

ZL* returns the value in binary form (1 byte)

ZL2 sets the end of line character to 2. **WARNING** if you do this you will not be able to communicate with the device unless you end the command line with <CTRL>B (02)

NOTE:

The system will ignore or skip over any received values that are below a value of 32 (20h) except the following characters:

Escape	17, 1bh (interpreted as EOL)
Back space	8, 8h
ZL character	13, 0dh (by default)

19.9. ZM

Name: **Macro**

Command Parameters: **none**

Typical Use **ZM <see below>**

A macro is a common computer term to allow a sequence of commands to be recorded and played back at a later time. This command allows just that. The playback is normally at start up and allows programming of logos etc. on appropriate devices.

The command is used on its own; enter the command **ZM** and the following prompt appears:

175>

The number 175 (differing devices may have a different number) in front of the prompt shows the space available in characters. This will reduce as commands are entered.

Any valid commands can be used but the syntax is not checked until run time, the macro sequence can be checked with ZT. To exit this command use **Escape**.

Macros cannot be edited, the only way is to enter them from the beginning again. They can of course be sent as a text file from a terminal.

Note that if the macro flag is set, see the ZC command, the commands will be run at start up so care should be taken what commands are entered.

The factory defaults will not change the commands you have entered but it will set the macro flag to 0 so the macro will not run at start up. A brand new device has the macro flag set to 1 so it may be worth while setting this to 0 before experimenting. See the ZC command on how to do this.

The macro can be tested using the ZT command.

19.10. ZR

Name: **Reset**

Command Parameters: **none**.

Typical use **ZR**

This is the reset command and will reset the hardware and put the communication mode as if it had just been switched on.

19.11. ZT

Name: **Test Macro**

Command Parameters: **none**

Typical Use **:00ZT**

This will test any stored macro and should be used before setting the macro flag.

19.12. ZV

Name: **Version information**

Command Parameters: **none**

Typical Use **ZV**

This simply returns a string that contains the firmware and device version information.

Additional information is also displayed but this does not apply to all devices:

The ZV command is also used for detecting errors and indicating if the factory defaults have been set. The format is:

[WnFn] <version information>

Where n is either 0 or 1. Under normal circumstances this would read [W0F0]. For information about this see the separate headings under watchdog and factory reset.

NOTE that reading the data clears it so if the first read was:

[W1F0] <version information>

The second call of this command would read:

[W0F0] <version information>

19.13. ZW

Name: **Wait**

Command Parameters: **n (1-65534)**

Typical Use **ZW110**

This command will simply suspend the device for a length of time. The time is entered in approximately 10ms intervals, so 100 is 1000 ms which is 1 second.

A typical use is to 'hold back' a text file when downloading or in a macro so that animated displays or actions can be implemented. At any time during the delay if a character is received the delay will abort.

This command is not suitable for highly accurate delays as the 10ms interval may vary, particularly from device to device.

Keypad Interface

BV4107

Example:
ZW 200

The above example will delay approximately 2 seconds.

20.Z Command Error Codes

Error codes will be displayed if the debug level (ZD) is set to greater than 0.

Code	Description
1	End of input buffer reached, buffer full. If this happens the whole of the input line will be ignored.
2	Unknown command, the command issued is not in the command table for this device.
4	Non-valid hex, where a command is expecting a hex value and something is entered outside 0-9, a-f, A-F. NOTE This is rarely used as most numbers values are in decimal.
5	Incorrect parameter following command. This refers to commands that expect a particular value to follow. This error will indicate that a command has been entered that is not in the correct / expected format.
6	Bad decimal number. This is because the command is expecting a number and something else has been entered. Note that a space should always follow a number.

Start HyperTerminal or some other terminal software, BV Terminal is ideal and can be obtained from www.byvac.com **Error! Reference source not found.** The following settings should be used:

Baud rate 9600
Start bits 1
Stop bits 2
Handshake none
Local echo on

(The Baud rate is not that important as the IASI will adjust to the terminals Baud rate)

Power up the device and press 'return' a few times making sure that the terminal is active (mouse cursor in the terminal window). The device should respond with:

L>

when it is ready to accept commands. If this is not the case check the power supply and terminal settings.

The device is now ready to accept commands, try **za**. This will return 00 which is the default device address.

After this prompt the device is now ready to be configured but it may be worthwhile spending some time looking at the command options available.

21.1. Reconfiguring

It may be that you want to use the device through a line driver device (MAX232) or microprocessor UART without bothering with the PC com port cable. This is also possible.

21. Connecting and Configuration

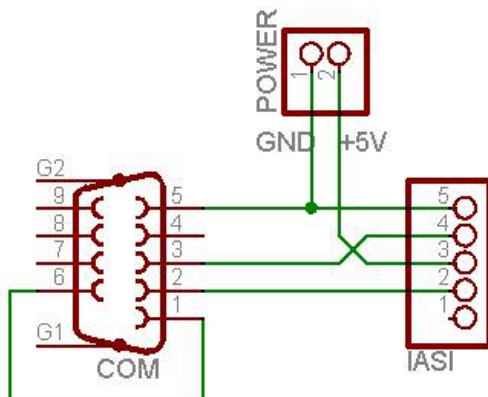


Figure 9 Connection wiring

The above wiring diagram shows the connections to a standard PC 9 way com port (RS232 connector). Pin 1 of the IASI has no connection as this is used to connect to a microcontroller UART.

The factory defaults will work with the above configuration.

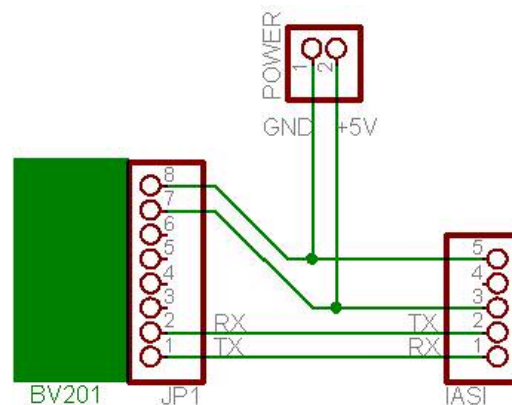


Figure 10 Using non-inverted

The above illustrates the connections used for, in this example a BV201 board that simply translates the PC com port to non-inverted 5V logic levels.

Using a BV101 a USB solution could be provided and there would be no need for a separate power supply.

See www.byvac.co.uk for these products.

Start up HyperTerminal or BV Terminal as before and press enter two or three times. In

Keypad Interface

BV4107

this instance the terminal will not show the prompt as the output is the wrong polarity, simply ignore any characters returned and type:

ZC3 -r

Now press return two or three times to establish the Baud rate and the **L>** prompt should appear. This illustrates that it is not necessary to receive anything from the device in order to reconfigure it.

The command ZC3 -r, loads configuration 'slot' 3 into the device. See the ZC command for standard configurations.

22. Microcontroller Use

22.1. Simple

There are several ways the device can be used with a microcontroller. The simplest way is to use it in the default mode:

ZC0 -r (inverted)

ZC3 -r (non-inverted - most likely)

For an output device, and LCD for example no return value is necessary but sufficient time should be given for the command and device to operate. The TX line (from the IASI) does not even need to be connected.

22.2. With feedback

A more sophisticated method would be to have some feedback from the device. There is also no need for a prompt.

The table shows a configuration that will do this for a single device. To place this is say, slot 6 requires the following command:

ZC6 -w YNYNYNN

To load this into the device type:

ZC6 -r

and issue two or three CR to establish the Baud rate.

ACK	Y
Macro	N
Baud-d	Y
Invert	N
Listen	Y
Prompt	N
Echo	N

As described in the ACK section of the ZC command the purpose of this is to tell the master controller (microcontroller) that another command can be issued. This can greatly speed up the transfer process.

22.3. Baud rate

It may be convenient to fix a Baud rate rather than having to establish it each time by sending CR. Two things need to be done for this:

ACK	Y
Macro	N
Baud-d	N
Invert	N
Listen	Y
Prompt	N
Echo	N

First set the desired baud rate according to the table illustrated in the ZB command. As an example, to set the baud rate to 38400 do this:

ZB3

Now configure the device so that the automatic Baud rate detection is switched off as in the table above.

The summary of commands would be as follows:

ZB3

ZC6 -w YNYNYNN

CR (carriage return)

CR

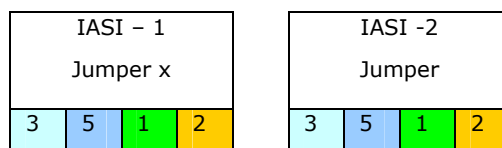
The device will retain the above configuration until either reconfigured or the factory defaults have been reset.

22.4. Multiple Devices

In the above examples the configuration assumed that only a single device was connected to the serial bus and so no addressing was required, commands such as ZA and ZL would be readily accepted by the device (called listen mode).

If more than one device is connected then, unless both are not required to respond to the same command, each device needs its own address.

Four (three if no feedback is required) connections are required. The diagram shows two devices but more devices can be added.



3 - Connect +5V together

5 - Connect Ground together

1 - Connect RX together

Keypad Interface

BV4107

2 – Connect TX together

Note: the electrical interface is designed to accept multiple devices on the RX side for either input from RS232 (inverted) or UART (non-inverted). The TX side however will only transmit to multiple UART devices in non-invert mode.

Jumper: remove all of the jumpers except one

ZAD1

Will set the device to address D1 (note that d1, lower case would be different). It is also a good idea to stop error messages:

ZD0

Finally configure the device to stop listening and no prompt.

ACK	Y
Macro	N
Baud-d	N
Invert	N
Listen	N
Prompt	N
Echo	N

To illustrate this as a series of commands the following would be required. Using slot 7 to hold the configuration data.

ZAD1
ZD0
ZC7 -w ynnnnnn
CR (carriage return)
CR

The device can now be commanded as follows:

:D1za

This will return 3 bytes 'D' '1' and 6 (if using the default ACK).

23. Restoring Factory Defaults

Factory defaults can be restored either by software or hardware. The factory default condition is:

ZA00 (address 00)
 ZC configuration 0 (see the ZC command)
 ZK6 (ack character)
 ZL13 (end of line character)
 ZB0 (baud rate 9600) **

** Note that the Baud rate will be ignored as the default Baud detect, set by configuration slot 0 is set to on.

The first 6 configuration slots will be set as the table given in the ZC command. Any user

information placed in those slots will be overwritten.

Applicable only to some devices:

Where a factory reset has occurred for whatever reason the ZV command will indicate this. A typical read out would be:

[W0F1] <version information>

The F1 in the square brackets indicates that a factory reset has taken place, this data is stored in EEPROM so even if the device is switched off the data still remains. The act of reading the data using the ZV command will clear the F1 back to F0, so this can only be seen once per factory reset.

23.1. Software

To set the above condition in software use the ZF command, once the ZF command has been issued the device will reset and wait for at least two CR (carriage returns) from the host device.

23.2. Hardware

It is unlikely that this will ever be used but has been provided in the unlikely event that a combination of configurations has rendered the device unable to communicate. It will require a short piece of wire and the holes will vary slightly from device to device.

The technique is as follows:

1. Power down the device.
2. Temporarily connect the two holes on the device together as shown. If the picture does not match exactly, then look for 5 holes in a row, at one end there will be a square hole, this is hole 1. Connect together holes 1 and 5.
3. Power up the device, this will restore the factory settings. On display devices this is shown as *F
4. Power down the device.
5. Remove the shorting link.

The device is now restored to the factory settings. NOTE that if a macro was programmed at the factory this will no longer show. On an LCD device for example it will not show the ByVac screen as it did when it left the factory, just the cursor will show.

Keypad Interface

BV4107

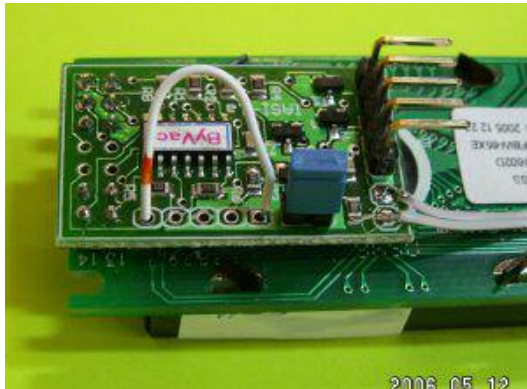


Figure 11 Shorting link

24. Watchdog

*** Not applicable to all devices ***

Some devices have additional error protection in the form of a watchdog timer. The timer is set internally and is constantly kept up to date by the firmware. Should anything unforeseen go wrong and the firmware is unable to update the timer a reset will occur.

The most likely scenario for this happening is if the serial interface is overwhelmed by input that the built in overrun protection cannot cope with.

If a watchdog time out occurs, a flag is written to the EEPROM. This can be read using the ZV command:

[W1F0] <version information>

The output from the ZV command will indicate that a watchdog time out has occurred by a '1'

following W in the example above. The act of reading the information will clear the flag, so the next read using ZV will show:

[W0F0] <version information>

Use this information to debug the external driving software.

25. Troubleshooting

The device does not show an L>prompt anymore and does not appear to respond to any commands.

The IASI device has a very comprehensive set of input options and it is quite possible to set a combination of input options that are not compatible with what is connected. It will appear that the device has stopped working when in fact it is simply operating in a different mode.

What's required is to get back to the factory default and take it from there. To do this follow the methods used in section **23 Restoring Factory Defaults**

Note that any factory macro which displays a start up sequence, the "ByVac" in the case of the LCD display for example will not be displayed. This is because the default setting is to turn the macro run off, this is a safety feature in case there is some erroneous code in the macro which prevents the display from starting.