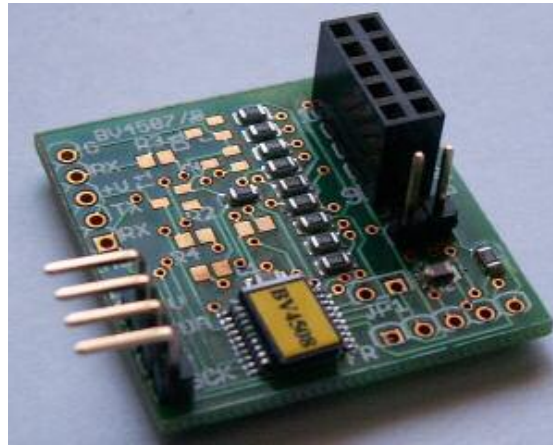

I2C-9 Channel ADC

BV4508



BV4508

I2C-9 Channel ADC

Product specification

January 2009 V0.a

I2C-9 Channel ADC**BV4508****Contents**

1.	Introduction	5
2.	Features	5
3.	Electrical Specification	5
4.	ADC Input.....	5
5.	Factory Reset	5
6.	Operation	6
6.1.	Normal Mode	6
6.2.	Autoscan Mode.....	6
6.3.	Differential Readings	6
7.	IASI2 Command set.....	6
7.1.	Command 1	7
7.2.	Command 2	7
7.3.	Command 3	7
7.4.	Command 4	7
7.5.	Command 5	7
7.6.	Command 6	7
7.7.	Command 7	7
7.8.	Command 8	7
7.9.	Command 9	8
7.10.	Command 0x10.....	8
7.11.	Command 0x11.....	8
7.12.	Command 0x12.....	8
8.	System Commands	9
8.1.	0x55	9
8.2.	Command 0x90.....	9
8.3.	Command 0x91.....	9
8.4.	Command 0x93.....	9
8.5.	Command 0x94.....	9
8.6.	Command 0x95.....	10
8.7.	Command 0x98.....	10
8.8.	Command 0x99.....	10
8.9.	Command 0xA0.....	10
9.	Hardware Reset.....	10
10.	Command Diagrams.....	10
10.1.	Sending a single command	10
10.2.	Sending a command with a parameter byte/s	11
10.3.	Receiving bytes from the salve	11
11.	Trouble Shooting	11
11.1.	Pulse Stretching	11
11.2.	Last Read NACK	11
11.3.	Pull Up's.....	11

I2C-9 Channel ADC

BV4508

I2C-9 Channel ADC

BV4508

I2C-9 Channel ADC

BV4508

Rev	Change
Jan 2009	Preliminary
July 2009	Description error – command 7

1. Introduction

This is a 9 channel Analogue to Digital Converter (ADC) with an I2C interface that makes it easy to connect to a microcontroller.

The 9 channels are capable of 10 bit resolution and there is a feature to allow automatic conversion to take place. The device is controlled by simple text commands using a serial interface.

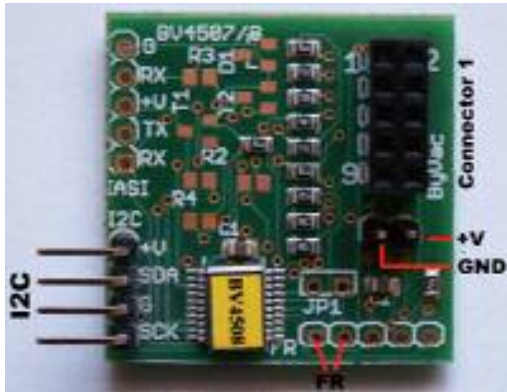
In addition the device has a user programmable address so that several of them can be connected to the same serial bus.

Default I2C address is 0x62

2. Features

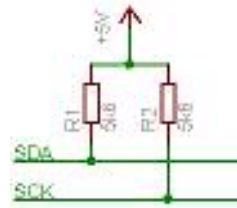
- I2C up to 400kHz
- 9 channels with 10 bit resolution
- Automatic conversion mode
- Choice of voltage reference source
- Socket output for easy interfacing
- Size 32mm x 32mm x 12mm

3. Electrical Specification



Using simple commands the I2C interface controls the 9 channel ADC. The input is taken from connector 1.

Pull up resistors are required for the I2C interface, these are not provided on the ADC board. The board is also used for the IAS12 interface and so any other holes should not be used.



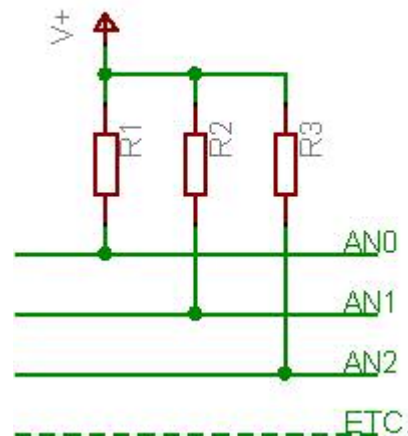
4. ADC Input

Pin	Function
1	AN0
2	AN2
3	AN3
4	AN4
5	AN5
6	AN6
7	AN7
8	AN8
9	AN9
10	AN1/Vref

Connector 1

At the bottom end of connector 1 there is also a pin connector that is connected to +V and Ground. This can be used to provide power for any external equipment.

The input to all of the channels has a 100k pull up resistor as shown.



This does not greatly effect the input impedance but does prevent spurious readings for unconnected channels.

By default the reference voltage is taken from the +V supply but if required this can be taken from pin 10. This is done by sending a command to tell the BV4507 to use pin 10 as a reference.

5. Factory Reset

As the I2C address is stored in EEPROM and is fully configurable, it is possible that the user may forget what the address is. To restore factory defaults do the following:

I2C-9 Channel ADC

BV4508

- 1) remove power
- 2) Connect the square hole marked FR to the hole next to it, this will in fact connect the square hole to +V
- 3) Apply power
- 4) Remove power

The address and any other parameters that may be stored in the first half of the EEPROM will now be reset.

In general the first 16 bytes of EEPROM are normally reserved for system use but these areas can still be overwritten by the user.

6. Operation

There are two modes of operation for this device. The normal mode and autoscan mode.

6.1. Normal Mode

The normal mode of operation is a 3 or 4 stage process:

1. Select channel
2. Do conversion
3. Check conversion has finished (optional)
4. Read conversion results

This mode of operation uses less power as conversion only takes place when requested. An alternative is to use the automatic or autoscan mode.

6.2. Autoscan Mode

This is enabled by issuing a command and once enabled each channel is scanned in turn and the result stored as a 16 bit value. The result can be read at any time. The rate at which scans are performed are set by the acquisition delay and is set by default to 10uS.

The only operation required is to read the result as steps 1 to 3 from the above list are carried out continuously.

Another advantage of using this mode is that because the results are stored, the differential of two readings can be obtained.

6.3. Differential Readings

The differential command is used in conjunction with the autoscan mode and returns the difference between 2 channels. This can effectively double the range that the ADC can produce.

The calculation is done internally and the result is returned as a signed number, the channel numbers used are as follows:

- 0 AN0 – AN1
- 1 AN2 – AN3
- 2 AN4 – AN5

- 3 AN6 – AN7
- 4 AN8 – AN9

As an example if AD0=75 and AS1=100 then the result from differential channel 0 would be $75-100 = -25$.

7. IAS12 Command set

The format used by this device consists of a command, this is a number, followed by other bytes depending on that command.

There are two type of command, those referring to the device and those referring to the system. The system commands enable changing of the device address etc.

Device default address is 0x62

Command	Device Command Set
1	Select channel
2	Do conversion
3	Conversion status
4	Fetch Result
5	Enable ADC
6	ADC autoscan
7	Read results from autoscan
8	Acquisition delay
9	Read results from autoscan – differential
0x10	Result justification
0x11	Voltage reference source
0x12	Conversion clock

Table 1 Device Command Set

Table 1 is a command summary of all of the device and system commands.

The method of writing to the device using the I2C protocol follows a consistent format, typically:

<S-Addr><Command><data..><Stop>

Where S-Addr is the start condition followed by the device address (0x62). Command is one of the commands given in the table. Data is one or more bytes and Stop is the stop condition.

Reading data requires a restart and this will be in the format:

<S-Addr><Command><R-Addr><data..><Stop>

The restart address will be one greater than the start address, thus if the start address is 0x62, the restart address will be 0x63. Again the data can be one or more bytes read from the device.

I2C-9 Channel ADC

BV4508

For more information about the command formats see section 10.

Each command will have its own format and is described in the following text. A start condition and address is always followed by a command.

7.1. Command 1

Name: **Select channel**

Format: <S-addr><1><0 to 9><Stop>

Sets a channel for later conversion.

7.2. Command 2

Name: **Do conversion**

Format: <S-addr><2><Stop>

After selecting the channel this command will instruct the ADC to carry out the conversion. The conversion itself takes approximately 2µs but this will be influenced by the serial interface.

7.3. Command 3

Name: **Conversion Status**

Format: <S-addr><3><R-Addr><byte><Stop>

This command will return 1 if the conversion is still going on otherwise it will return 0.

7.4. Command 4

Name: **Fetch Result**

Format: <S-addr><4><R-Addr><h-byte><l-byte><Stop>

This command returns the ADC value of the selected channel. It returns two bytes, high byte first.

BV4221 Example

0x62>s 4 r g-2 p

The above command will return 2 bytes, high byte followed by low byte that makes up a 16 bit value

7.5. Command 5

Name: **Enable ADC**

Format: <S-addr><5><0 or 1><Stop>

By default and at reset the ADC is enabled, however some applications may wish to turn off the ADC to save power. This is done by sending a 0 value after the command.

1=Enabled (default)

0=Disabled

7.6. Command 6

Name: **Enable Autoscan**

Format: <S-addr><6><0 or 1><Stop>

By default autoscan is switched off, see the description of what autoscan does in section 6.2.

Sending the second byte as 1 will enable autoscan. Note that in this mode the normal scanning method of select, convert and read is disabled.

7.7. Command 7

Name: **Read Autoscan Results**

Format: <S-addr><7><start><R-Addr><h-byte><l-byte><Stop>

When autoscan is enabled the results in the internal RAM are constantly being updated, this command reads the results of a particular channel as represented by that RAM location.

The result is a 16 bit number ranging from 0 to 1024 represented as two bytes.

RAM Location given by start	Channel
0	0
2	1
4	2
6	3
8	4
10	5
12	6
14	7
16	8
18	9

The table can be read starting at any point so the option is to either read the table in full and select the required bytes programmatically or to set a start address to read a particular byte.

The data is stored <high><low> and so an even start address will retrieve the high byte first.

BV4221 Example

0x62>s 7 0 r g-2 p

The above command will return 2 bytes, high byte followed by low byte that makes up a 16 bit value of channel 0.

7.8. Command 8

Name: **Set Acquisition Delay**

Format: <S-addr><8><1 to 255><Stop>

As autoscan goes from one channel to the next a short delay is introduced. The delay is necessary for the ADC to be able to select, read and store the channel. This is set at 5µs

I2C-9 Channel ADC

BV4508

by default and this command allows that time to be changed.

Each value represents 0.5uS, the default value is 10 which gives a delay of 5uS. The maximum value is 255 which represents a delay of 127.5uS. The minimum delay value is 1 which represents 0.5uS. If 0 is entered this will be converted to 1.

Second byte values:

0=2uS (default)

1=4uS

2=8uS

7.9. Command 9

Name: **Read Autoscan Results - Differential**

Format: <S-addr><9><R-Addr><h-byte><l-byte><Stop>

See section 6.3 for a description of what the differential reading is.

This command does not check for a valid channel number so any number outside the above range will give unpredictable results.

7.10. Command 0x10

Name: **Result Justification**

Format: <S-addr><0x10><0 or 1><Stop>

By default the 10 bit result is right justified in a 16 bit field, the higher bits (11 to 15) are cleared to 0. This gives a result range of 0 to 1023 (decimal).

This command can change the justification to be left of the 16 bit field occupying bits 15 to 6. This effectively multiplies the result by 64.

Second byte value:

1=right (default)

0=left.

7.11. Command 0x11

Name: **Voltage Reference Source**

Format: <S-addr><0x11><0 or 1><Stop>

By default and at reset the voltage reference is taken from +V and pin 10 of connector 1 is channel 1. This will give a full scale reading (1025) when a channel = +V.

An alternative voltage reference can be supplied to pin 10, from a voltage reference for example, and this command is used to make use of it.

Second byte value:

1=Voltage reference taken from +V (default)

0=Voltage reference taken from pin 10

7.12. Command 0x12

Name: **Conversion Clock**

Format: <S-addr><0x12><0 to 2><Stop>

The ADC conversion speed can be slowed if required using this command

I2C-9 Channel ADC

BV4508

Rev	System Section Changes
Oct 2008	Preliminary
Dec 2008	Removed command 96

8. System Commands

The following section deals with system commands that are common to all I2C devices. Note that not all of these commands are available for all devices.

Command	System Command Set
0x55	Test
0x90	Read EEPROM
0x91	Write EEPROM
0x93	End of EEPROM
0x94	Sleep
0x95	Reset
0x98	Change Device Address Temporary
0x99	Change Device Address Permanent
0xA0	Firmware Version

Most but not all devices contain an EEPROM that can store data when the power is off. The first 16 bytes of the memory is reserved for system use and should not be changed by using these commands.

If the contents of the first 16 bytes are changed then, depending on the device unpredictable results may occur. A factory reset will put the contents back to normal. In some devices not all 16 bytes are used.

The rest of the EEPROM can be used by the user for any purpose.

8.1. 0x55

Name: **Test**

Format: < S-addr><0x55><start><R-Addr><Value..><NACK><Stop>

BV4221 Example

```
0x42>s 55 r g-3 p
```

The above command will return 1,2,3 if the device is connected and working correctly.

This command simply returns an incrementing value until NACK is sent by the master prior to stop. This can be useful for testing the interface.

It can be used for testing the presence or other wise of a device at a particular address.

It is also useful during the development stage to ensure that the I2C is working for that device.

8.2. Command 0x90

Name: **Read EEPROM**

Format: <S-addr><0x90><EE-Address><R-Addr><data...><Stop>

BV4221 Example

```
0x42>s 90 0 r g-3 p
```

The above will fetch 3 bytes from the EEPROM addresses 0, 1 and 2

This command will allow a single or several bytes to be read from a specified EEPROM address.

8.3. Command 0x91

Name: **Write EEPROM**

Format: <S-addr><0x91><EE-Address><data...><Stop>

BV4221 Example

```
0x42>s 91 10 1 2 3 p
```

The above write 1,2 and 3 to EEPROM addresses 0x10, 0x11 & 0x12

This command will write one or more, up to a maximum of 30 bytes at any one time, to be written to the EEPROM. Address 0 of the EEPROM is the device address and this cannot be written to by this command. A special command 0x99 is used for this purpose.

The first 16 bytes 0 to 15 are reserved for system use.

8.4. Command 0x93

Name: **End of EEPROM**

Format: <S-addr><0x93><R-Addr><data><Stop>

BV4221 Example

```
0x42>s 93 r g-1 p
```

Returns the address of the end of the EEPROM, normally 0xff

The system only uses a small portion of the first part of the EEPROM, the rest of the EEPROM can be used for user data or other purposes depending on the device. This command returns a single byte that will determine the last writeable address of EEPROM, normally 0xFF.

8.5. Command 0x94

Name: **Sleep**

I2C-9 Channel ADC

BV4508

Format: <S-addr><0x94><Stop>

BV4221 Example

0x42>s 94 p

This will put the IC into sleep mode. Any other command will wake the IC. Depending on the device this can be a considerable power saving.

8.6. Command 0x95

Name: **Reset**

Format: <S-addr><0x95><Stop>

BV4221 Example

0x42>s 95 p

Resets the device, this is equivalent to disconnecting and then connecting the power again.

8.7. Command 0x98

Name: **Change Device Address Temporary**

Format: <S-addr><0x98><New-Addr><Stop>

BV4221 Example

0x42>s 98 62 p

Changes the device address to 0x62, the device will revert back to 0x42 at reset.

This will change the device address with immediate effect and so the next command must use the new address. The address must be a write address (even number) Odd numbers will simply be ignored. The effect will last as long as the device is switched on. Resetting the device will restore the address to its original value. The address is stored in EEPROM location 0.

8.8. Command 0x99

Name: **Change Device Address Permanent**

Format: <S-addr><0x99><New-Addr><0x55><0xaa><Current-Addr><Stop>

BV4221 Example

0x42>s 99 62 55 aa 42 p

Permanently changes the device address to 0x62.

This command changes the address immediately (the next command will need to use the new address) and permanently (see hardware reset). The address must be a write address (even number) and follow the sequence exactly.

Permanent in this case means that the device will retain this address after power down, i.e. it is stored in EEPROM. Should anything go

wrong the default address can be restored by using a hardware reset.

8.9. Command 0xA0

Name: **Firmware version**

Format: <S-addr><a0><R-Addr><byte><byte><Stop>

BV4221 Example

0x42>s a0 r g-2 p

This will return the two firmware bytes.

This simply returns two bytes that represents the firmware version.

9. Hardware Reset

A hardware reset has been provided should the device address be changed to some unknown value.

The method of restoring the factory defaults and thus the default device address is as follows:

- 1) Remove power
- 2) Hold the designated pin low or high or connect two pins together. The actual pins are device dependant and will be referenced in the sections above this text.
- 3) Apply power
- 4) Remove power
- 5) Remove shorting link

When power is now restored the device will have the default I2C address, normally 0x42.

10. Command Diagrams

To further explain the format of the commands this section has been provided. The design of the interface has been purposely kept simple and so there are only a few standard sequences required.

Key

Master

Slave

S Start condition

P Stop Condition

A Acknowledge = 1

N Not acknowledge = 0

10.1. Sending a single command

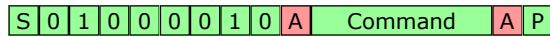
This is designated in the list as:

I2C-9 Channel ADC

BV4508

<S-addr><cmd><Stop>

This sequence is used for simple functions where no data is involved. The I2C sequence, using the default address is:

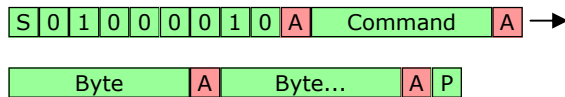


10.2. Sending a command with a parameter byte/s

This is designated in the list as:

<S-addr><cmd><data...><Stop>

Some commands expect a parameter after the command. In this case the bytes are sent one after the other up to the maximum of 31 bytes. The stop command tells the slave that there is a command ready to be executed.



10.3. Receiving bytes from the slave

<S-addr><cmd-Addr><byte><Stop>

When receiving one or a number of bytes from the device a restart is required.

The command is sent with an even address (the R/W bit 0). When the slave acknowledges the command another start condition is sent with the R/W bit set to 1. This is called a restart. After this restart the slave will continue to send bytes until the master sends a not acknowledge (N or NACK).

If the master does not send a NACK at the last byte the slave will be expecting another byte to be requested and this may cause either unpredictable results or the I2C bus to lock.

11. Trouble Shooting

This section has been added to answer frequently asked questions. The problems are usually caused because the master device has not had the I2C specification fully implemented.

11.1. Pulse Stretching

This is a method of I2C handshaking which is used in BV slave devices but it is not always supported by the master system. The symptoms are erratic behaviour, some commands will be accepted and others will not.

To explain: when the slave device is busy it holds the clock line low (normally only the master controls the clock line), the master should check that the clock line is high before sending the start condition. If it is low the master should wait until it is free.

Quite a few slave devices do not use pulse stretching and so this not being implemented in the master does not show up. However when dealing with relatively slow hardware, an LCD display for example (i.e. BV4219), this will become a problem. The work round is to make sure that the master recognises pulse stretching properly or introduce delays after each command.

11.2. Last Read NACK

When optionally multiple reads of a slave is required (the 0x55 command is a good example) the last read should send a NACK rather than a ACK. This informs the slave that no more reads from that command are required.

It has been found that some master implementations do not send a NACK on the last read. This causes the BV slave to remain in the (multi read) command effectively blocking any other commands.

The typical symptoms are that when the 0x55 command is implemented no other commands will work after that.

11.3. Pull Up's

The most common problem when trying to get a new device going is to forget to put the pull up resistors somewhere on the bus. BV Slave devices do not have pull up resistor on board so they must be provided by the master (the BV4221 has pull up's) or provided externally. A value of around 5k is okay but this is not usually critical.

