BBC A3000

I/O Card

User Guide

Watford Electronics

Watford Electronics

Distributors of: Electronic Components, Micro Computers & Peripherals

250 High Street, Watford, WD1 2AN, England. Tel: Watford (09231237774. Telex 8956095 WATFRD Fax: (0923) 233642

If you have any comments on this guide or the product then please address your correspondence to:

Watford Electronics
Jesse House
250 High Street
WATFORD
Hertfordshire
WD1 2AN

Telephone	:	0923 **237774**
Telex	:	**8956095 WATFRD**
Fax	:	**0923 233642**

Watford Electronics have now been established for twenty years. We are one of the major electronics distributors and retailers in the country, supplying thousands of different electronics components and computer peripherals, by mail order, and through our retail outlet at Watford. We specialise in the BBC range of microcomputers, including the Archimedes series and associated software and hardware products.

Hardware and Software design by
Watford Electronics Research & Development Department

Published by Watford Electronics
Issue 1.0. December 1991

Contents

Chapter 1 - Introduction

The original BBC Micro has been a classroom favourite for almost a decade. One of the strengths of the BBC Model B and its derivatives was undoubtedly the versatile range of expansion ports it offered. These included an 8-bit user port and a four-channel joystick interface based around a four-channel analogue to digital converter (ADC). Countless educational and leisure hardware projects have been devised with these ports in mind, and therefore a BBC I/O expansion is a must for anybody hoping to undertake electronics projects which require computerised control by an Archimedes.

With the arrival of the Archimedes and BBC A3000, users have been faced with the problem that the user port and analogue interfaces are absent. Watford Electronics has recognised the requirement for an add-on interface board to make the Archimedes range compatible with software and hardware designed specifically for these ports on the original BBC range.

In addition, a new expansion bus standard, the Philips developed IIC (Inter IC) standard, which is likely to become popular in education and DIY electronics projects, has been included on the A3000 version. IIC is an inexpensive and simple to use serial interface which can accommodate a number of different devices attached to the same port.

From the software point of view there is maximum compatibility with the old BBC Micro. Original software can run under the Acorn BBC emulator as a ROM on the card contains full OSBYTE calls and ADVAL support.

User Port

The BBC Microcomputer has an 8-bit user-port which can be connected to a wide range of devices like graphpads, keyboards, EPROM programmers and other interface circuits. The Watford BBC I/O Card uses the same chip as the BBC and allows you to perform similar tasks with the Archimedes.

The user port consists of a set of programmable byte wide data lines and two control lines made available by connection to a 20 pin IDC connector. The connector has the same pin-out as the BBC micro user port and also includes a 5V supply output to power small expansion projects. Full details of the user port connector can be found in *Appendix 1 — Specification.*

The user port is designed around one half of a 65C22 Versatile Interface Adapter (VIA). The VIA is the heart of the interface and is programmed entirely by use of its 16 registers which are accessed using two OSBYTE calls, numbered 150 to read, 151 to write, these are detailed fully in *Chapter 3 — Interface Software.*

Analogue to Digital Converter

The BBC Microcomputer is also fitted with a socket marked *'Analogue In* to which you can connect games paddles, joysticks and other circuits with voltages which the computer can measure. This is now made possible on the Archimedes by the Watford BBC I/O card utilising the same chip as the BBC.

The Analogue to Digital Converter (ADC) is a 12 bit integrating type which can operate in either 12 or 8 bit mode. The converter is based around the NEC µPD7002 chip and works by comparing the voltage from one of the four input channels to its reference voltage $Vref$. The output from the converter appears as two bytes and these are merged to form a 16 bit word with a range from 0 to 65520. The lower 4 bits of the output are always zero, so in 12 bit mode the number changes by 16, and in 8 bit mode the number changes in steps of 256 (only the most significant byte is valid).

The ADC can be accessed using the BASIC keyword ADVAL, or by using the OSBYTE calls 16 17 128 189 and 190. Direct access to the registers can also be gained using the two OSBYTE calls which read and write BBC I/O address space, numbers 150 and 151.

The connector has the same pin-out as the BBC Micro and includes a 5V supply to power joysticks and other low power circuits.

Inter IC Expansion

This expansion connector is only fitted on the A3000 version of the interface and provides a connection to the A3000 IIC interface, as used to control the Archimedes real time clock and CMOS RAM.

A five pin DIN socket is used to connect the IIC bus and a 5V output is also provided to power small circuits.

A simple SWI call is already provided in RISC OS to communicate with IIC devices and the use of this is described in *Chapter 3 – Interface Software.*

Chapter 2 – Installation

The I/O interface software requires the RISC OS operating system to run. If you do not have RISC OS already fitted, then you will need to upgrade before installing the card. (A540, A5000 and A3000 are supplied with RISC OS).

There are different fitting procedures for the A3000, A5000 and A300/400/500 series machines and you should follow the section that applies to your machine.

Please read all the installation text before starting to install your card.

If for any reason you are not able, or do not feel confident about fitting the interface card (especially on the A3000) then you should consult Watford or any Acorn Authorised Dealer who will arrange to fit the card for you. Remember to take your computer in its original packaging. The Dealer may make a charge for installing the interface card.

What you will need

All you you need to install the card is a No. 1 Pozidriv or Philips screwdriver to remove the screws retaining the computer's cover and blanking plates.

Handling Precautions

In order to ensure long life and reliability from your interface card you should take care when handling it, especially to avoid damage by static electricity.

1 Always switch off the computer and unplug it from the mains.
2 Disconnect any peripherals such as printers or modems.
3 DO NOT remove the interface card from the protective anti-static bag until you are ready to fit it.
4 Avoid touching the components on the card; always handle the card by holding the rear panel.

To install the interface card the computer must be switched off and disconnected from the mains. All peripherals should be disconnected.

There are three simple stages required to fit the card:

Stage 1 - Disassembly - remove the cover from the computer
2 - Installation - fit the interface card.
3 - Reassembly - replace the cover.

Please follow the steps for your particular machine from the following sections.

A3000 Installation

First ensure your machine is switched off and disconnected from the mains supply, then switch off and disconnect all peripherals. If your machine has any expansion boxes or an Ultimum fitted these too should be removed.

Stage 1 Disassembly

1 Loosen the two screws at the rear of the machine until the small square metal plates they retain can be slid out.

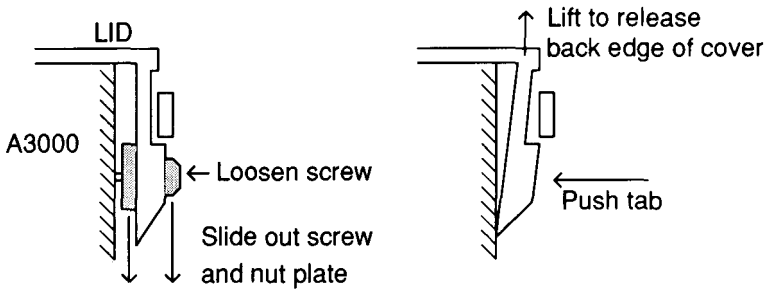
2 Place the computer on a work surface covered by a clean soft covering (eg. a blanket) and turn it over so that it rests on its top cover.

3 Remove the screw from the deep hole in the centre of the base.

4 Turn the machine back onto its base, then press the two rear retaining fingers toward the body of the machine while lifting the lid upwards. This will release the rear clips.

5 Turn the A3000 on its side, then release each of the three retaining clips (underneath the front edge of the machine) by inserting a wide flat-blade screwdriver into the slot, and twisting slightly. You must insert the screwdriver blade into the rear half of the slot to get it behind the locking tab.

6 Place the machine back on its base. The lid should now lift off.



Removal of fixing clips on the A3000

Stage 2 Installation

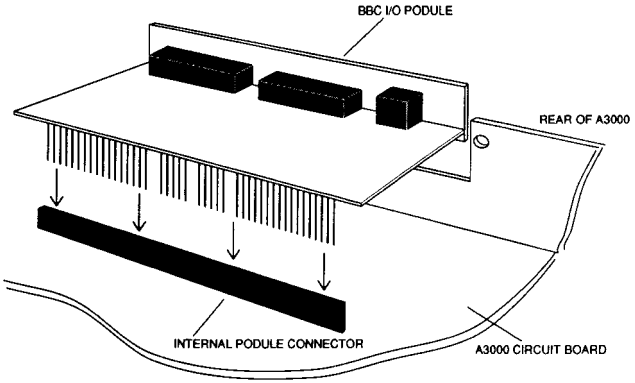
The interface card is designed to fit into the internal podule expansion slot and the blanking panel must be removed first. The blanking panel is located at the rear above the *Parallel Printer* connector and is removed by unscrewing the two fixing screws either side.

1 Remove the blanking plate from the rear of the machine.

2 Unpack the card from the protective anti-static bag.

3 Install the card by first aligning the header pins on the interface card with the header sockets on the motherboard, making sure that the panel is aligned correctly with the back of the A3000. Press the pins firmly into the headers until they are fully located whilst making sure the connectors are aligned at all times and that the card is at right angles to the motherboard.

A little force may be required to fit the card, however if it does not fit easily then remove it and try again, otherwise damage may be done to the card or motherboard.



Insertion of interface card pins into the A3000

4 Screw the panel of the card into the rear of the computer using the screws that held the original blanking plate.

Stage 3 Reassembly

Replace the cover, using the reverse process for disassembly, by following steps I to 6 for *'Stage 1 Disassembly'* in reverse.

Make sure that the screws are securely tightened.

This completes installation.

A300, A400, A500 Installation

The interface card is a single half-width Eurocard designed to occupy a chosen slot in your expansion card backplane. A backplane *and* cooling fan must be installed in your computer before the interface card can be fitted.

In some early A300 machines the backplane and fan were optional, if you have one of these machines then you should first obtain a backplane and fan from Watford Electronics.

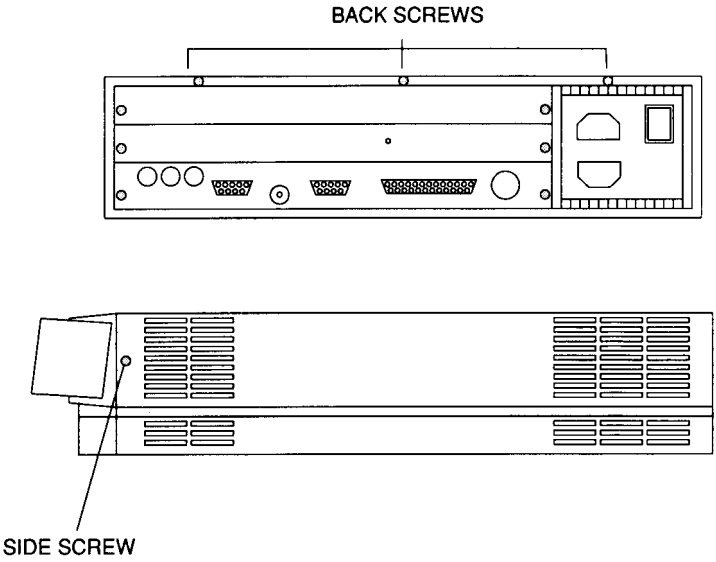
The interface card is supplied with a short blanking plate, a T piece and two screws. These are used to extend the interface card panel up to the full width required if an adjacent slot has no installed card.

Stage 1 Disassembly

To remove the top cover of the computer from A300/400/500 series machines:

1 Remove the five retaining screws for the cover, there are three at the top of the back panel, and a further two on either side.

2 The top cover is removed by sliding it carefully towards the the back until it can be lifted up and away. If the sides are held slightly apart then it will be easier to slide off.



Location of cover retaining screws (A300, A400, A500 series)

Stage 2 Installation

1 Locate the backplane and select a free slot for the card.

2 Remove the corresponding blanking plate from the rear of the machine.

3 Unpack the card from the protective anti-static bag.

4 Either: use the T-piece and screws to extend the card's rear panel to cover the full width of the blanking plate that was removed.

Or: if another card is adjacent, use the T-piece to attach the rear panels together in the middle.

5 Support the top of the backplane with one hand and carefully pass the card through the hole in the rear of the machine and push the connector home onto the backplane socket. Ensure the connectors are aligned and that the card is at right angles to the backplane.

Very little force is required to fit the card, if it does not fit easily then remove it and try again, otherwise damage may be done to the card or backplane.

6 Screw the rear panel of the card (and any blanking plate) into the rear of the computer from where the original full width blanking plate was removed.

When properly installed the rear panel should be flush with the rear of the computer, if not then check that the card is plugged fully into the backplane.

(On some early machines the card may protrude by a few millimetres, since the backplane on these machines was located incorrectly.)

Stage 3 Reassembly

Replace the cover, using the reverse process for disassembly, by following steps 1 to 2 for *'Stage 1 Disassembly'* in reverse.

Make sure that the screws are securely tightened.

This completes installation.

A5000 Installation

The interface card is a single half-width Eurocard designed to occupy a slot of the expansion card backplane.

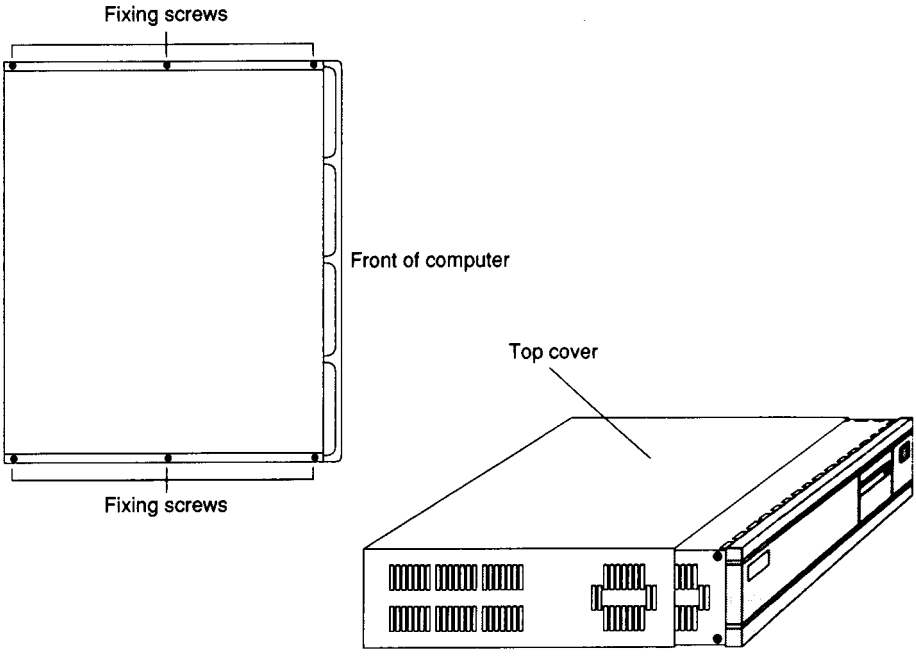
The interface card is supplied with a short blanking plate, a T piece and two screws. These are used to extend the interface card panel up to the full width required if an adjacent slot has no installed card.

Stage 1 Disassembly

To remove the top cover of the computer from A5000 series machines:

1 Switch off and ensure all leads and peripherals etc. are disconnected.

2 Place the computer on a work surface covered by a clean soft covering (eg. a blanket) and turn it over so that it rests on its top cover.



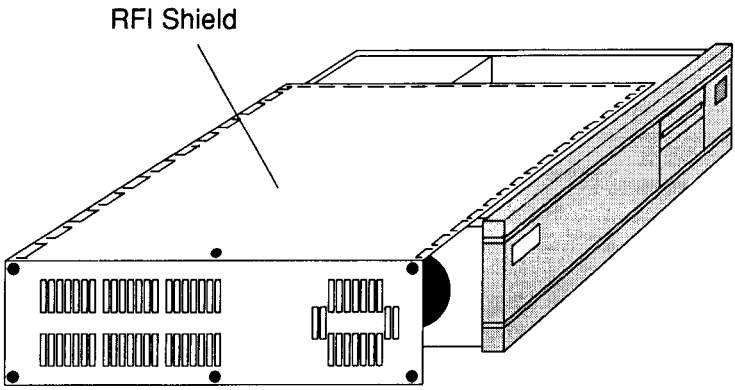
Top cover removal on the A5000

3 There are six cover retaining screws, three along each of the short sides, which should be removed.

4 After removal of the retaining screws turn the computer over, being sure to hold it along the longest sides to ensure the cover does not slide off.

5 The top cover can now be removed by sliding it carefully towards the the back until it can be lifted up and away. Remove the lid completely.

6 Some A5000 models incorporate an RFI shield over the case which must be removed. To remove the shield unscrew the five fixing screws on the right hand side of the computer (as viewed from the rear). Slide the shield off carefully, using a screwdriver in the top hole if required (see below).



RFI shield removal on the A5000

Stage 2 Installation

1 Locate the backplane and select a free slot for the card.

2 Remove the corresponding blanking plate from the rear of the machine

3 Unpack the card from the protective anti-static bag.

4 Either: use the T-piece and screws to extend the card's rear panel to cover the full width of the blanking plate that was removed.

Or: if another card is adjacent, use the T-piece to attach the rear panels together in the middle.

5 Support the top of the backplane with one hand and carefully pass the card through the hole in the rear of the machine and push the connector home *onto* the backplane socket. Ensure the connectors are aligned and that the card is at right-angles to the backplane.

Very little force is required to fit the card, if it does not fit easily then remove it and try again, otherwise damage may be done to the card or backplane.

6 Screw the rear panel of the card (and any blanking plate) into the rear of the computer from where the original full width blanking plate was removed.

When properly installed the rear panel should be flush with the rear of the computer, if not then check that the card is plugged fully into the backplane.

Stage 3 Reassembly

Replace the cover, using the reverse process for disassembly, by following steps to 6 for *'Stage 1 Disassembly'* in reverse.

Make sure that the screws are securely tightened.

This completes installation.

Chapter 3 - Interface Software

The user port and ADC can be read from and written to in BASIC, C, or ARM assembler by using the OSBYTE SWI calls provided by the card's ROM software.

The user port 65C22 VIA is a complicated chip and a data sheet is essential to understand and utilise this chip to the full!

This chapter provides the information you need to talk to the hardware and is not intended as a full introduction on programming the chips. This information can be found by obtaining the data sheets given in *Appendix 3 — References.*

OSBYTE The BBC and Master series of computers use a multi-purpose call known as OSBYTE to perform the software interfacing of the user port and ADC hardware.

RISC OS maintains compatibility with the BBC OSBYTE calls by providing a SWI called OS_Byte.

If you are familiar with the BBC OSBYTE calls then all you need to know to use the RISC OS equivalent is that;

the value that was passed in A on the BBC Microcomputer is passed in the least significant byte of R0,

X is now passed in the LSB of RI, and

Y is passed in the LSB of R2.

The OS_Byte SWI must be called with the requred OSBYTE number in R0, with the other registers, such as R1 and R2, setup as required.

***FX** Some OSBYTE calls just set variables and do not return values.

For these calls the equivalent *FX command can also be used.

User Port OSBYTE Calls

The user port is addressed as sixteen registers, and is mapped into the old BBC SHEILA address space at an offset of &60 to &6F (96 to 111 decimal).

The address offset is passed in R1 on OSBYTEs 150 and 151, and R2 is used to read or write the byte.

The SHEILA address offset and the register descriptions are shown below

OSBYTE SHEILA OFFSETS FOR USER PORT VIA

Register	Write	Read (if different)
&60 ORB/IRB	Output register B	Input register B
&61 ORA/IRA	Output register A	Input register A
&62 DDRB	Data direction register B	
&63 DDRA	Data direction register A	
&64 T1 C-L	T1 low order latches	T1 low order counter
&65 T1 C-H	T1 high order counter	
&66 T1 L-L	T1 low order latches	
&67 T1 L-H	T1 high order latches	
&68 T2C-L	T2 low order latches	T2 low order counter
&69 T2C-H	T2 high order counter	
&6A SR	Shift register	
&6B ACR	Auxiliary control register	
&6C PCR	Peripheral control register	
&6D IFR	Interrupt flag register	
&6E IER	Interrupt enable register	
&6F ORA/IRA as &61 except no handshake performed		

Please refer to the 65C22 data sheet for further information on the registers.

OSBYTE 150 (&96)

Read from a user port register.

Entry R0 = 150
R1 = Sheila address offset

Exit R2 = Byte read

Use The byte returned in R2 is read from the 65C22 register specified by the SHEILA address offset given in R1.

OSBYTE 151 (&97)

Write to a user port register.

Entry R0 = 151
R1 = Sheila address offset
R2 = Byte to write

Exit all preserved

Use The byte passed in R2 is written to the 65C22 register specified by the SHEILA address offset given in R1.

Example Write &FF to data direction register B (&62).

```
*FX 151, &62, &FF
```

Analogue to Digital Converter OSBYTE Calls

Like the original BBC, the analogue to digital converter software returns all results as a sixteen bit number to ensure future converters with higher resolution can be supported. This sometimes causes a little confusion because although the result from the ADC fits the 16 bit range, it is not accurate to 16 bits.

8 bit In 8 bit mode there are only 256 possible levels which start at 0 and end at 65280, incrementing by 256 each level. In fact in 8 bit mode only the most significant byte (MSB) holds the data, the LSB should be ignored. The original 256 levels can be obtained from the 16 bit value by dividing by 256.

10 bit In 10 bit mode the MSB and the top two bits of the LSB are valid. The output range is from 0 to 65472, in steps of 64. The 10 bit data can be obtained by dividing the 16 bit value by 64, to yield 1024 levels.

It is best to use the ADC in 8 bit mode, unless high resolution is required, because the conversion times are three times shorter and therefore more samples per second can be obtained. With more samples the movement of a pointer on the screen will be smoother.

OSBYTEs 188-190

OS_Bytes 188, 189 and 190 can be used to read or alter the OSBYTE variable and the contents of R1 and R2 determine what happens to the status variable:

```
New value = ( Old value AND R2 ) FOR RI
```

On exit, the value of the old variable is returned in R1.

To read a variable without affecting it, set `R2=255` and `R1=0`.

To change the variable to a new value, set `R2=0` and `RI= new value`.

OSBYTE 16 (&10)

Select the number of ADC channels which are to be sampled.

Entry R0 = 16
R1 = number of channels to be sampled (0 to 4)

Exit R1 = number of channels being sampled (0 to 4)

Use Up to four channels can be chosen for sampling with this call.

Each ADC channel takes an average of 10ms to convert an analogue voltage into a digital value. If all four channels are enabled then new samples for a given channel will only be updated every 40ms. It is therefore sensible to enable only the number of channels actually required.

Four sampling channels are set by default when the computer is reset.

If the number of channels to be sampled is set to zero then sampling is disabled, and the ADC will generate no system interrupts.

Example *FX 16 , 0 Sampling disabled

*FX 16, n Number of channels to be sampled is n. (n must be in the range 0 to 4, if it is greater than 4 then it is set to 4, and the value 4 is returned.)

The equivalent ARM assembler routine is:

```
MOV       RO, #16   ; OS_Byte number
MOV       RI, #n    ; set n channels
SWI       "OS_Byte"
```

OSBYTE 17 (&11)

Force an ADC conversion.

Entry R0 = 17
R1 = channel to be forced (0 to 4)

Exit R1 = channel forced

Use The given channel number is used as the next channel for conversion and takes effect after the current channel (if any) has been converted.

For example, if four channels are enabled then they are sampled in the consecutive order 1, 2 ,3, 4, 1, 2, 3 etc. If the current sample is from channel 2. then ***FX** 17, 1 will force a conversion on channel 1 instead of proceeding to channel 3, and the sequence would then be: 1, 2, 3, 4, 1, 2 , **1*FX** 17,1 here 1, 2, 3, 4 etc.

If a value of 0 is specified for the channel then no channel is forced. Instead the last used ADC conversion channel (as read with OSBYTE 128) is set to zero.

Example ***FX** 17, n Start a new ADC conversion on channel n. (n must be in the range 0 to 4, if it is greater than 4 then it is set to 4, and the value 4 is returned.)

The equivalent ARM assembler routine is:

```
MOV       RO, #17    ; OS_Byte number
MOV       R1, #n     ; force conversion on channel n
SWI       "OS_Byte"
```

Related Calls

OSBYTE &80 (128) can be used to determine when the conversion is complete.

OSBYTE 128 (&80)

Read ADC channel or get ADC buffer status.

Entry R0 =128
R1 = 0 read the last channel used for ADC conversion
R1 = 1 to 4 read the 16 bit result for the conversion of given channel 1..4

Exit If RI = 0 on entry then
R1 = fire button status
R2 = last channel to complete an ADC conversion (1 to 4), or 0 if none

If R1 = 1 to 4 on entry then
R1 = low byte of conversion (ignore for 8 bit mode)
R2 = high byte of conversion
R1 and R2 hold the 16 bit value of the conversion.

Use This OSBYTE returns the result of the most recent ADC conversion of a particular channel. It may also be used to determine an end of conversion and to find out which fire buttons are pressed.

On entry R1 holds the channel to be read. If R1 is in the range 1 to 4 then the specified channel will be read and on exit the 16 bit result will be in R1 and R2.

If on entry R1 = 0 then R2 holds a channel number from 1 to 4 indicating which channel was the last to complete. Both OSBYTEs 16 and 17 clear this value to 0. A value of 0 indicates that no channel has completed. Also on exit the two least significant bits of R1 will indicate the status of the two fire buttons, as follows:

`bit 0 FIRE 0` left side fire button
`bit 1 FIRE 1` right side fire button

Example Force a conversion on channel 1 and wait for it to convert, then read result.

```
   MOV R0, #17       ; OS_Byte number
   MOV RI, #1        ; force conversion on channel 1
   SWI   "OS_Byte"

   MOV R0, #128 ; OS_Byte number
.not_done
   MOV R1, #0
   SWI   "OS_Byte" ; read last channel converted
   TEQ R2, #0        ; has a conversion completed?
   BEQ not_done ; if not wait

   MOV RI, #1        ; read channel 1
   SWI   "OS_Byte"
   ORR RI, RI, R2, LSL #8 ; 16 bit result in RI
```

OSBYTE 188 (&BC)

Alter or read current ADC channel.

Entry R0 = 188
R1 = EOR value
R2 = AND value

Exit R1 = old ADC channel number
R2 = maximum ADC channel number

Use This OSBYTE accesses the variable location containing the number of the ADC channel currently being converted.

The current ADC channel number is altered:

<new current channel> = (<current channel> AND R2) EOR R1

The old value is returned in R1 and the maximum ADC channel number (variable 189) is returned in R2.

The call should not be used to force ADC conversions, use OSBYTE 17 instead.

Example Read the current ADC channel number into chan% without altering the value:

```
SYS "OS_Byte" , 188, 0, 255 TO , chan%
```

OSBYTE 189 (&BD)

Alter or read maximum ADC channel number.

Entry R0 = 189
R1 = EOR value
R2 = AND value

Exit R1 = maximum channel number 1 to 4, or 0 if sampling disabled
R2 = ADC conversion type

Use This OSBYTE accesses the variable location containing the maximum ADC channel number to be used for sampling.

The maximum ADC channel number is altered:

```
<new max channel>      ( <max channel> AND R2 ) EOR R1
```

The old value is returned in R1 and the ADC conversion type (variable 190) is returned in R2.

The maximum channel number to be used for ADC conversions must be in the range 0 to 4 and can also be set using OSBYTE &16.

Example Read the maximum ADC channel number into max% without altering the value:

```
SYS "OS_Byte",189, 0, 255 TO , max%
```

OSBYTE 190 (&BE)

Alter or read ADC conversion type.

Entry R0 = 190
R1= EOR value
R2 = AND value

Exit R1 = 8 or 12 bit conversion type:
0 for default (12 bit) conversion.
8 for 8 bit conversion
12 for 12 bit conversion
R2 corrupted

Use This OSBYTE accesses the variable location containing the ADC conversion type.

The ADC conversion type is altered:

```
<new type>     ( <old type> AND R2 ) EOR RI
```

The old value is returned in R1 and R2 is corrupted. Note

that the 12 bit mode is only accurate to 10 bits.

Example Set conversion to 8 bit:

```
SYS "OS_Byte",190,.8, 0
```

Set conversion to 12 bit

```
SYS "OS_Byte",190, 12, 0
```

Read conversion type into `type%` without affecting it:

```
SYS "OS_Byte",190, 0, 255 TO, type%
```

ADC BASIC Keyword ADVAL

The BASIC function ADVAL is used to obtain information and data from the analogue to digital converter.

A single parameter between 0 and the maximum ADC channel number can be passed to determine the action to be performed and is detailed below.

ADVAL (1)	reads the value of the last conversion on channel 1
ADVAL (2)	reads the value of the last conversion on channel 2
ADVAL (3)	reads the value of the last conversion on channel 3
ADVAL (4)	reads the value of the last conversion on channel 4

ADVAL (0) performs a special function in that it can be used to test to see which of the fire buttons is pressed. The value returned also indicates which ADC channel was the last one to be updated.

To determine the fire button status use:

```
X=ADVAL(0) AND 3
```

The value of X can then be used to find which fire buttons are pressed. x
=0 no buttons pressed
x=1 left side fire button pressed
X= 2 right side fire button pressed
X= 3 both fire buttons pressed

X=ADVAL (0) DIV 256 will give the last ADC channel to complete conversion. If the value returned is zero, then no channel has yet completed conversion.

Example Read the value of the last conversion on channel 2 into the variable Chan2 %.

```
Chan2 % = ADVAL ( 2 )
```

If this conversion is required to 8 bits then the value can be shifted to remove the lower 8 bits, like this:

```
Chan2% = Chan2% >> > 8
```

or, alternatively:

```
Chan 2 % = ADVAL ( 2 ) >> 8
```

Inter IC SWI Calls

There is one SWI call provided by RISC OS to control IIC devices. This SWI can read or write to any device given its IIC address and a block of memory.

SWI IIC_Control (&240)

Control IIC devices.

Entry R0 = device address (bit 0 clear => write, bit 0 set => read)
R1 = pointer to memory block
R2 = length of block in bytes

Exit R0 - R2 preserved

Use On a write operation, each byte from the memory block is sent using the IIC serial protocols.

On a read operation, R2 bytes are read from the IIC device and placed into the memory block.

The error `No Acknowledge from IIC device` (&20300) may be returned if an IIC device does not respond, due to wrong addressing or bad connection etc.

The following IIC addresses are reserved and should not be used by other IIC devices: &40, &70, &A0.

Hardware SWI Call

There is a SWI call provided which returns the absolute location of the expansion card in the podule address map.

SWI I/O_Podule_Hardware (&40500)

Entry -

Exit R1 = synchronous base address of podule

Use This SWI can be used to find out the base address of the card so that addresses can be determined for interrupt handlers etc.

The VIA registers start at an offset of &2000. See *Appendix I - Specification.* for further hardware details.

Programming Examples

Here are some further examples on how to program the user port and ADC. Other examples are given with the OSBYTE descriptions.

Examples in both ARM Assembler and BASIC are shown.

User Port

This example will write &FF to data direction register B (DDRB) of the VIA.

BASIC
```
10 ddrb% = &62          : REM Sheila address offset for DDRB
20 byte% = &FF          : REM Byte to write = &FF
30 SYS "OS_Byte", 151, ddrb%, byte%
```

Alternatively, an FX command could be used:

```
10 *FX 151, &62, &FF
```

ARM The equivalent ARM code would be:

```
MOV     R0, #151      ; OS_Byte call 151
MOV     R1, #&62      ; Address offset for DDRB
MOV     R2, #&FF      ; Byte to write
SWI     "OS_Byte"     ; do it
```

Analogue to Digital Converter

This example will read the value of ADC channel 2, the fire button status and the last channel used.

BASIC

```
10 AtoD% = ADVAL(2)
20 firebutton%   ADVAL(0)
30 PRINT "Channel 2 value      ";AtoD%
40 PRINT "Fire button status ";firebutton% AND 3
50 channel% = firebutton% >> 8
60 PRINT "Last converted chan ";channel%
```

Or alternatively, using SYS calls:

```
10 SYS "OS_Byte", 128, 2 TO ,low%, high% :REM read value
20 AtoD% = low% + high% * 256 :REM make 16 bit result
30 SYS "OS_Byte", 128, 0 TO ,firebutton%, channel%
40 PRINT "Channel 2 value      ";AtoD%
50 PRINT "Fire button status ";firebutton% AND 3
60 PRINT "Last converted chan ";channel%
```

ARM

In this ARM example the results are returned in separate registers.

```
STMFD    STACK !, {R0, LINK} ; Save registers
MOV      R0, #128            ; OS_Byte call 128
MOV      RI, #2              ; Channel 2
SWI      "OS_Byte"           ; Read value of chan 2
ORR      R3, R1, R2, LSL #8 ; form 16 bit result in R3

; now read last channel and fire button status

MOV      R1, #0
SWI      "OS_Byte"           ; Read status

; RI = fire button status
; R2 = last used channel
; R3 = 16 bit value from channel 2

LDMFD    STACK !, {R0, PC}   ; Restore registers, exit
```

Appendix 1 - Specification

This section gives details on the hardware and software specification and is intended to be read by users familiar with the Archimedes expansion bus software interface.

Hardware Addresses

The synchronous base address of the podule hardware can be found by using the SWI call `I /O_Podule_Hardware`, which is documented in *Chapter 3 — Software.*

User Port VIA 65C22

The VIA hardware uses an addressing scheme compatible with the Acorn I/O Podule. The VIA is mapped into synchronous podule space at an offset of &2000 and each of the 16 registers are one word (4 bytes) apart.

VIA Interrupt Request

A separate IRQ request register is implemented to allow easy interfacing to RISC OS interrupt code and is mapped into synchronous podule space at an offset of &3000. Bit 7 is set if the VIA is generating a podule IRQ interrupt.

Analogue to Digital Converter µPD7002

The ADC hardware is not address compatible with the Acorn I/O Podule because each register is 64 bytes apart and not four. The ADC is mapped into MEMC podule space at an offset of &1000 and each of the four registers are 16 words (64 bytes) apart.

ADC Interrupt Request

A separate IRQ request register is implemented to allow easy interfacing to RISC OS interrupt code and is mapped into synchronous podule space at an offset of &3100. Bit 7 is set if the ADC is generating a podule IRQ interrupt due to an end of conversion.

Podule Base Offsets

Device	Cycle Type	Address Offset
ROM	Sync	+0000 to +1FFF in steps of 4 bytes
ADC	MEMC	+1000 to +1000 in steps of 64 bytes
VIA	Sync	+2000 to +203C in steps of 4 bytes
VIA IRQ	Sync	+3000 bit 7
ADC IRQ	Sync	+3100 bit 7

Podule Slot Offsets

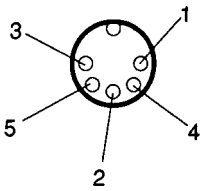
Podule	**Device**	**Address Space**
0	ADC	0300 1000 to 0300 1000 step 64
	EOC IRQ	033C 3100 bit 7
	VIA	033C 2000 to 033C 203C step 4
	VIA IRQ	033C 3000 bit 7
1	ADC	0300 5000 to 0300 5000 step 64
	EOC IRQ	033C 7100 bit 7
	VIA	033C 6000 to 033C 603C step 4
	VIA IRQ	033C 7000 bit 7
2	ADC	0300 9000 to 0300 9000 step 64
	EOC IRQ	033C B100 bit 7
	VIA	033C A000 to 033C A03C step 4
	VIA IRQ	033C B000 bit 7
3	ADC	0300 D000 to 0300 D0C0 step 64
	EOC IRQ	033C F100 bit 7
	VIA	033C E000 to 033C E03C step 4
	VIA IRQ	033C F000 bit 7

These hardware addresses are provided for reference only and software should not assume or access the locations directly.

Always use the legal OSBYTE or ADVAL calls to access the I/O ports.

Connector Details

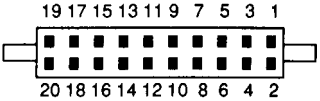
Key DGND Digital ground
AGND Analogue ground



Inter IC 5 pin DIN Socket (A3000 only)

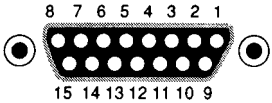
1 DGND
2 +5V
3 DGND
4 IIC CLOCK
5 IIC DATA

User Port 20 pin IDC header



1 +5V	2 CB1
3 +5V	4 CB2
5 DGND	6 PBO
7 DGND	8 PB1
9 DGND	10 PB2
11 DGND	12 PB3
13 DGND	14 PB4
15 DGND	16 PB5
17 DGND	18 P86
19 DGND	20 PB7

ADC 15 Pin D-type Socket



1 +5V	9 LPEN (connected to VIA CA1)
2 DGND	10 FIRE 1 (connected to VIA PA7)
3 DGND	11 Vref
4 CH3	12 CH2
5 AGND	13 FIRE 0 (connected to VIA PA6)
6 DGND	14 VREF
7 CH1	15 CH0
8 AGND	

Note:
Fire buttons should use DGND.

Circuits using V_{ref} or connecting to the ADC inputs should connect to AGND to minimise noise.

Technical Specification

Interface Card		Min	Typ	Max	Units
Supply voltage	Vcc	4.5	5.0	5.5	V
Supply current		-	-	100	mA
Fused current supply to User, ADC, IIC ports		-	-	500	mA

Note

If more than 500mA in total are drawn from the 5V outputs on the User, ADC and IIC ports then the on-board fuse will blow. If this occurs then the card will have to be returned to Watford Electronics for fuse replacement and a charge will be made for this. (If no fuse is fitted damage may result to your machine or interface card.)

µPD7002 ADC		Min	Typ	Max	Units
Input voltage range		0	-	Vref	V
Voltage Reference	Vref		1.8	-	V
Resolution		8	-	12	bits
Full scale error		-	-	±0.2	% FS R
Conversion speed (8 bit)		2.4	4	5	ms
Conversion speed (12 bit)		8.5	10	15	ms
Input impedance		-	1000	-	Mohm

Note

The internal resolution of the ADC is 12 bit but the two least significant bits contain internal chip noise, noise from the diode Vref source, and noise coupled by the connecting wires due to bad screening and analogue ground connections. Although these bits will also vary with the analogue input it is best to use the ADC as a 10 bit converter and ignore the least significant 6 bits.

65C22 VIA User Port		Min	Typ	Max	Units
Input high voltage	VIH	+2.0	-	Vcc	V
Input low voltage	VIL	-0.3	-	+0.8	V
Input high current	IIH	-200	-400	-	µA at V_{IH} = 2.4V
Input low current	IIL	-	-2	-2.6	mA at V_{IL} = 0.4V
Output high voltage	VOH	1.5	-	-	V at -3.2mA source
Output low voltage	VOL	-	-	+0.4	V at 3.2mA sink
Output high current	IOH	-3.2	-6	-	mA source
Output low current	logy	3.2	-	-	mA sink

Note

The user port can drive many CMOS loads or two LS TTL loads or one standard TTL load. A -ve sign indicates current flow out of VIA, +ve indicates inward flow

Inter IC	Min	Typ	Max	Units
Output drive			1	TTL load

Appendix 2 - Compatibility

This appendix details the differences between the ADC and user port implementations on the BBC Microcomputer and the Archimedes version.

User Port

The 65C22 VIA is operated at 2MHz, twice the rate of the BBC Microcomputer's 1MHz device. *If* the timers are used with values designed for the BBC then they will run twice as fast. This also applies if the shift registers are being used under control of the timers.

There is a limit to the current that may be taken from the user port. The maximum is 500mA, less any current taken by the ADC and IIC power connections.

The interrupt from the VIA is supported and may be used if a suitable interrupt driver is written. A separate interrupt request bit is provided (independent of the VIA) which may be used to support a proper RISC OS interrupt handler (see *Appendix A).*

Analogue to Digital Converter

The µPD7002 ADC has a faster clock which results in faster conversion times but is otherwise compatible if the legal software calls are used.

Appendix 3 - References

65C22 VIA data sheet from Rockwell

µPD7002 Analogue to Digital Converter data sheet from NEC

BBC Microcomputer System User Guide from Acorn

Glossary

µPD7002 Numeric identification of the ADC chip.

6522 Numeric identification of the VIA.

ADC Analogue to Digital Converter.

address A number representing a specific memory location.

analogue Continuous and non-discrete. For example, an analogue voltage may have any value. See digital.

Analogue to Digital Convertor
A specialised chip, or connection to one, which converts analogue voltages to a digital representation.

ARM Acorn RISC Machine.

binary A numbering system which counts using just zeroes and ones. For example, counting from 0 to 3 is represented as: 00, 01, 10, 11.

bit A single binary digit, either 0 or 1.

bus A group of signals used to communicate electronic information. A computer always has one main bus to allow the microprocessor to communicate with other components. For example the data bus.

digital Having discrete values. For example, a digital voltage is usually one of two values (binary), which are combined into groups which are able to represent digital values using the binary system.

D-type A D-shaped type of connector. **FOR**

Boolean Exclusive-OR operation.

expansion bus
A collection of signals which enable the facilities of a computer to be extended by attaching other devices.

I/O Input/Output

IDC A connector which uses uses an insulation displacement method to make connection with ribbon cables.

IIC Inter IC (sometimes PC), an expansion bus standard to enable integrated circuits to communicate between one another.

IRO Interrupt Request. A hardware signal which interrupts the current program or process.

latch A register which retains its value.

most significant bit
The bit which represents the highest value in binary numbers. For example, in the binary value 100, the number 1 equivalent to decimal 4 is most significant.

peripheral
A device connected to a computer such as a printer or disc drive.

pin-out The identification of individual pins in a port, socket, connector, etc.

port A connector or socket which enables a device to be plugged into a computer.

RAM Random Access Memory. A memory device which can have any address accessed in any order.

register A single address in memory which transfers data.

RFI Radio Frequency Interference.

RISC Reduced Instruction Set Computer.

RISC OS Acorn's RISC Operating System.

ROM Read Only Memory. A memory device that holds operating software that cannot be altered, even by removing power.

serial One bit at a time.

SWI SoftWare Interrupt. An Acorn Archimedes software interface.

user-port A socket which enables users to connect a variety of peripherals to the computer.

VIA Versatile Interface Adapter. An integrated circuit which provides several low-level features.

Vref A reference voltage against which others are compared.

word A collection of bits, usually 8 or 16.

Jessa House, 250 High Street, Watford, WD1 2AN, England
Tel: Watford (0923) 237774, Telex: 8956095 WATFRD, Fax: 0923 233642