The product described in this manual and products for use with it are subject to continuous development and improvement: All information of a technical nature and particulars of the product and its use (including the information and particulars in this manual) are given by Acorn Computers in good faith: However it is acknowledged that there may be errors or omissions in this manual or in the products it describes. Acorn Computers welcomes comments and suggestions relating to the product and this manual. A list of details of any amendments or revisions to this manual can be obtained upon request from Acorn Computers Technical Enquiries or via the Acorn Support Information Database (SID)*.

All correspondence should be addressed to:-

Technical Enquiries
Acorn Computers Limited
Fulbourn Road
Cherry Hinton
Cambridge CB1 4JN

All maintenance and service on the product must be carried out by Acorn Computers' authorised agents. Acorn Computers can accept no liability whatsoever for any loss or damage caused by service, maintenance or repair by unauthorised personnel.

This manual is intended only to assist the reader in the use of this product, and therefore Acorn Computers shall not be liable for any loss or damage whatsoever arising from the use of any information or particulars in, or any error or omission in, this manual, or anyincorrect use of the product.

Published by Acorn Computers Limited

Part Number: 0474,905
Issue B - July 1988

*SID is a direct dial viewdata system available to registered users. You can gain access to SID on (0223) 243642, which will permit you to inspect the system and use a response frame for registration.

About this manual

Introduction

This manual describes the technical detail of the Acorn Technical Publishing System. The information in this manual is primarily for reference, and is intended for -hardware and software designers, programmers and engineers who need to understand the design and operation of the system. You should be familiar with the use of the Acorn Technical Publishing System and you should understand. the concepts of computer architecture and programming.

How this manual is organised

This manual gives a general overview of the Acorn Technical Publishing System and then describes, in detail, each functional part of the system. The main system is divided as follows:

- memory and processor system
- input/output system
- SCSI interface
- video and sound system
- peripheral devices
- optional expansions.

The following Appendices are included at the end of this manual:

Appendix A	Circuit diagrams
Appendix B	Assembly drawings
Appendix C	System PCB links
Appendix D	Parts list/bill of materials
Appendix E	Device data sheets
Appendix F	Connectors
Appendix G	Mechanical specification and standards
Appendix H	Bibliography
Appendix I	Silk-Screen diagrams

Other documents

Further information on the ARM chip set is available in the ARM Chipset DataBooks from Acorn Computers. Additional information on the use and programming of the Acorn Technical Publishing System is contained in the Acorn Technical Publishing System *User Guide* and Acorn Technical Publishing System *Programmer's Reference Manual.* The Acorn Technical Publishing System *Service Manual* gives hardware service and test procedures. See Appendix H for a bibliography.

Conventions and hints

Where a signal is suffixed by a '*' it indicates that this signal is active low. That is, it is TRUE when at a low voltage and FALSE at a high voltage.

TRUE and FALSE apply to the logical state of a signal. For example, the DBAGT signal is used to indicate that a DBA cycle is in progress (DBA GranT). Thus when DBAGT is TRUE DBA is occurring. As it does not have an asterisk '*' when it is TRUE it is positive (ie TTL logic '1') and when FALSE it is close to 0V (ie TTL logic 0). When a signal is said to be 'asserted' this means that it is in its TRUE state. The magnitude of the voltage representing this is determined by whether it is Active Low or Active High.

Hexadecimal numbers are shown with a 0x preceding the number itself.

Bit numbering is such that bit 0 is the least significant.

Circuit references

All circuit references to the main PCB are taken from the Acorn Technical Publishing System schematic diagrams. The references apply to the Issue 1 PCB.

Table of Contents

Chapter 1 Product overview

1.1 Introduction

The Acorn Technical Publishing System is a high performance Acorn UNIX workstation specifically designed for the Computer Assisted Technical Publishing (CATP) market. The Acorn Technical Publishing System provides a complete system solution including computer, screen and laser beam printer together with systems and applications software. Expansions and options are provided for higher performance or capacity, or for integration with other systems.

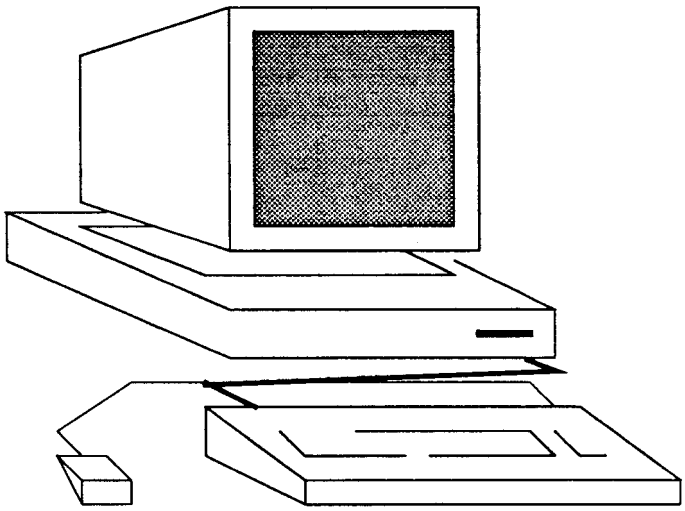

Figure 1.1 : Acorn Technical Publishing System

1:2 Hardware

The main functional units of the hardware are as follows:

- computer unit
- keyboard
- mouse
- monitor
- internal storage devices
- backplane for expansion cards
- laser beam printer (LBP) expansion card
- laser beam printer
- optional external storage devices.

Computer unit

The main computer unit comprises a metal case with plastic front and rear mouldings. This unit contains the main printed circuit board (PCB), the power supply unit (PSU), a backplane supporting up to four expansion cards and, as standard, two 3.5 inch magnetic storage devices. These internal storage devices are a 70 Mbyte hard disc and a choice of either a 2 Mbyte floppy disc drive or a 40 Mbyte cartridge tape drive.

Keyboard

The keyboard unit is in a plastic case. The layout is similar to the IBM-PC 'enhanced' or ` 101' style. Connection to the computer is via a coiled serial cable with a six pin miniature circular plug which connects to the lower front of the computer unit. Function keystrips can be fitted into a transparent holder on the keyboard. A microcontroller is contained within the keyboard. The keyboard electronics includes an identity code which enables the computer to establish the keyboard type. National keyboard variants are available; these have different keycaps together with a unique electronic identification code.

Mouse

The mouse is a three button, mechanical device and is connected to the keyboard via a nine pin, miniature circular connector. The mouse resolution is 10 edges/mm.

Monitor

The monitor is a 19 inch high resolution monochrome display with 1152 x 900 pixels at 66 Hz refresh rate. The phosphor is medium to short persistence, paper white in colour.

Hard disc drive

The main internal storage is provided by a SCSI (Small Computer Systems Interface), 3.5 inch, 70 Mbyte hard disc drive which operates with an on-board controller. The hard disc drive has an average access seek time of less than 30 ms from command issue to command complete, and a data transfer rate of greater than 0.8 Mbytes/second. An indicator light on the front of the main computer unit shows when the hard disc is being accessed by the system.

Cartridge tape drive

The Acorn Technical Publishing System is fitted with a 3.5 inch, 40 Mbyte cartridge tape drive, as standard.

Floppy disc drive

Some Acorn Technical Publishing System models are fitted with an internal 3.5 inch, 2 Mbyte floppy disc drive. This drive is capable of reading 1 Mbyte format discs. A disc indicator light on the front of the main computer unit shows when the disc drive is being accessed by the system. A disc eject button releases the floppy disc from the drive for removal.

Backplane

A four socket backplane is fitted as standard in the Acorn Technical Publishing System. The backplane is attached to the main PCB via a 96 way DIN 41612 connector and has three, 64 way standard expansion sockets and one 96 way coprocessor socket. Up to four expansion cards can be fitted which conform to the Acorn Podule Specification (see Chapter 10, *Podules and backplane).*

LBP expansion card

The LBP expansion card is fitted in one of the four backplane expansion sockets. This allows a laser beam printer to be driven directly from the Acorn Technical Publishing System.

Laser beam printer

The Acorn Technical Publishing System can support either a PostScript printer (via the serial line) or a low cost laser printer (via the laser beam printer expansion card). The laser beam printer resolution is 300 dots per inch. The printer runs at a maximum rate of 8 pages per minute.

External storage system

Optional storage devices available for the Acorn Technical Publishing System include a 5.25 inch, 280 Mbyte hard disc drive and a 5.25 inch, 150 Mbyte streaming tape cartridge. An external storage system houses one or other of these storage options together with its power supply unit. The external storage system is connected to the Acorn Technical Publishing System via the SCSI bus which allows up to six external peripherals to be daisy-chained onto the system.

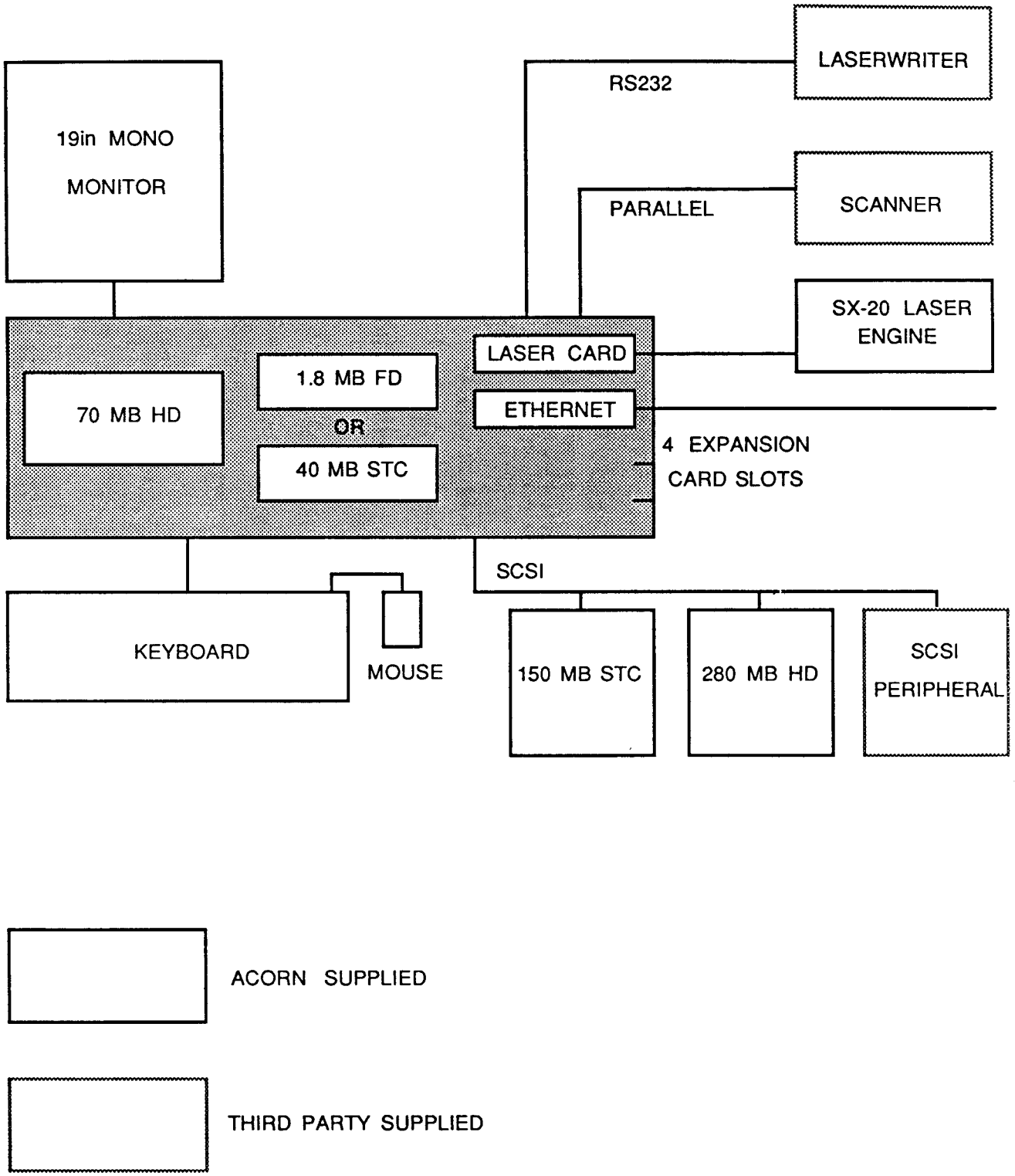

Figure 1.2 : Block diagram of the Acorn Technical Publishing System

1.3 Software

The Acorn Technical Publishing System software comprises Frame Maker, Frame Technology's Technical Publishing System. This runs under the Acorn UNIX operating system. The Acorn Technical Publishing System also supports Sun Microsystems NeWS window manager, Network Filing System (NFS) and Ethernet TCP/IP/UDP, to enable it to interact with Acorn and third party servers.

C, FORTRAN, PASCAL, BASIC compilers and LISP are also available, together with EMACS (Gnumacs version), DBX, porting tools and the standard UNIX libraries for software engineers.

For further information about the Acorn Technical Publishing System software, see the Acorn Technical Publishing System *User Guide* and the Acorn Technical Publishing System *Programmer's Reference Manual.*

Chapter 2 Main PCB system overview

2:1 Subsystem blocks

The main board contains the following subsystem blocks, illustrated in the figure below.

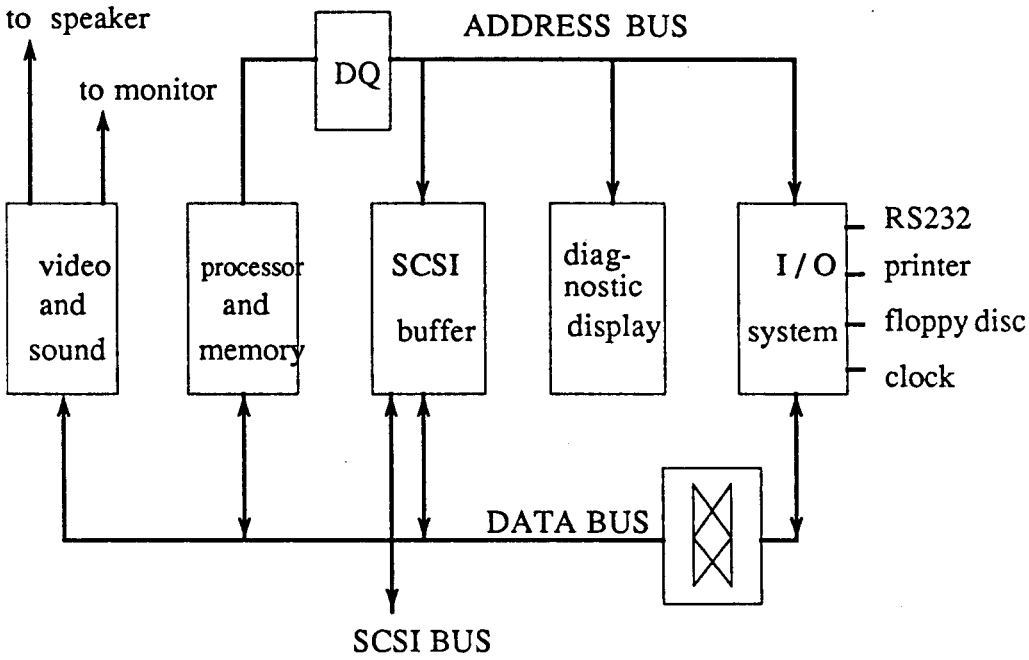

Figure 2.1 : Main PCB system block diagram

Processor and memory

This subsystem is the central processing unit. It is the basic processing engine in the Acorn Technical Publishing System. The main elements within this block are the Acorn RISC Processor (ARM), the Memory Manager (MEMC), the Random Access Memory (RAM) array and Read Only Memory (ROM) containing the operating system loader.

Input and output

This provides a mechanism by which the processor communicates with peripherals. The main elements within this block are; the Input/Output Controller (IOC), the Floppy Disc Controller (FDC), the Serial Communications Controller (SCC), the printer port, the battery backed configuration memory (CMOS RAM) and the keyboard interface. The podule bus is also part of the I/O subsystem but this warrants special attention and has been allocated its own chapter, Chapter 10, *Podules and backplane.*

SCSI buffer

This is a high speed RAM buffer and Small Computer Systems Interface (SCSI). Together these provide a high speed interface to SCSI based peripherals such as the internal hard disc drive and external storage units.

Video and sound

This subsystem is the display driver. It produces the video and synchronisation signals necessary to drive the high resolution monochrome monitor. The main element within this block is the Acorn Video Controller (VIDC). The sound subsystem is a basic sound channel capable of producing a range of sounds from a simple tone to more complex sounds such as digitised speech. This subsystem shares circuitry with the video subsystem and uses functions within the video controller (VIDC).

Built-in test hardware

This is a hexadecimal display used by test software contained in the system loader ROMs to indicate fault conditions. This display is visible only when the chassis is withdrawn from the cover.

2:2 Main system address map

The system address map is illustrated in Table 2.1 below. All devices are memory mapped and accessible to the main processor using the full instruction set. There are no special input/output instructions.

The address map is divided into a number of regions, each of which roughly corresponds to one of the subsystems which is described above, and in more detail in the following chapters.

The ROM, MEMC, logical and physical RAM are part of the processor and memory subsystem. The read only memory (ROM) and the hexadecimal display (TEST) share part of the address map with the memory manager (MEMC). This works because MEMC is a write only device and the other two are read only.

The area labelled input/output modules is part of the I/O subsystem and is described in Chapter 5, *The input/output system*.

The areas marked * are accessible only to processes running in the supervisor mode.

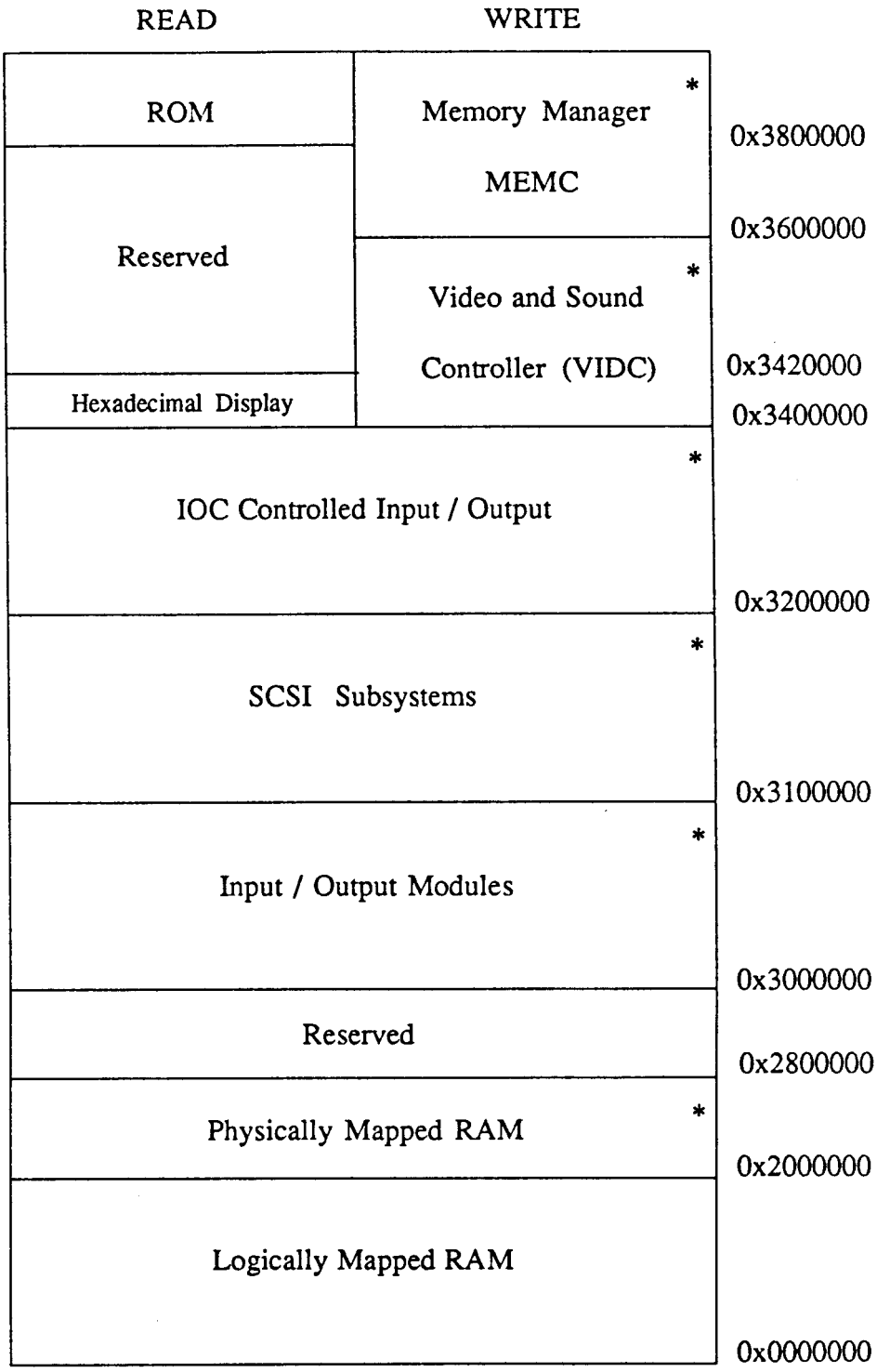

Table 2.1 : System memory map

Chapter 3 Processor and memory subsystem

This subsystem is the central processing unit. It is the basic processing engine in the Acorn Technical Publishing System. The main elements within this block are:

- Acorn RISC processor (ARM)
- memory manager (MEMC)
- random access memory (RAM) array
- read only memory (ROM) containing the operating system loader.

The figure below is a block diagram of this subsystem.

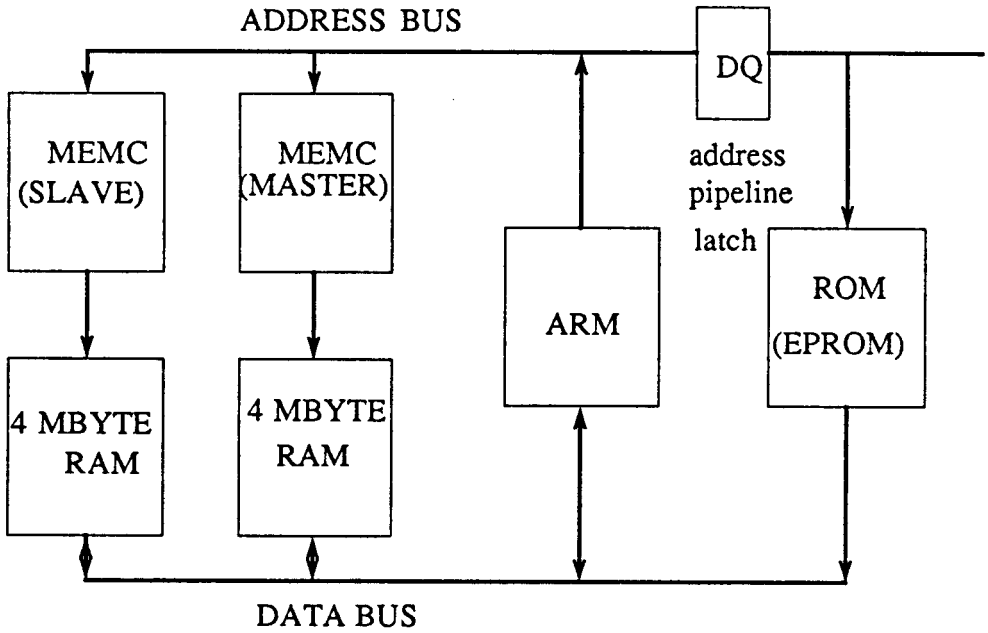

Figure 3.1 : The processor and memory block diagram

3:1 The ARM processor

The ARM is a low cost, pipelined 32 bit reduced instruction set processor (RISC). It accepts instructions and manipulates data via a high speed 32 bit data bus and 26 bit address bus. All instructions are restartable allowing virtual memory support.

The ARM data sheet (Bibliography, reference 1) describes the ARM processor in greater detail and specifies the instruction set. All functions described in the data sheet are supported by the Acorn Technical Publishing System.

3:2 The memory manager pair

Two memory managers are used in this design. The memory manager pair acts as the interface between the ARM processor, the memory and the other subsystems, providing all the critical system timing signals and clocks.

The MEMC data sheet (Bibliography, reference 4) describes the MEMC chip in detail. Two memory sizes are supported: 4 Mbyte (single MEMC), or 8 Mbyte (dual MEMC). The dual MEMC system is described by considering how physical and logical memory is addressed.

Physically mapped RAM

The dual MEMC architecture of the Acorn Technical Publishing System extends the physical RAM to 8 Mbytes. This results in an increase in the number of physical pages from 128 (for a 4 Mbyte system) to 256. Each page is 32 kbytes.

This is achieved by configuring (automatically during reset) each MEMC to map different physical pages. During reset the state of byte*/word pin determines which bank of physical pages is addressed by each device.

The MEMC chip which addresses pages 0 to 127 is referred to as the master MEMC. The MEMC chip which addresses pages 128 to 255 is referred to as the slave MEMC.

In a single MEMC chip system, where the MEMC is configured as a master, the 128 physical pages appear sequentially from address 0x2000000. The rest of the 16 Mbyte area is undefined and should not be used.

In a dual MEMC chip system, the 128 physical pages of the master appear sequentially from address 0x2000000, and the 128 physical pages of the slave MEMC appear sequentially from address 0x2400000. Both master and slave MEMCs control 4 Mbytes of DRAM, and the processor will see 256 contiguous physical pages.

Physically mapped RAM may only be accessed when supervisor mode is selected.

Logically mapped RAM

Within each MEMC a logical to physical address translator maps the physical memory into a 32 Mbyte logical address space (with three levels of protection) allowing virtual memory and multi-tasking to be implemented. Fast 'page mode' DRAM accesses are used to maximise memory bandwidth.

The bottom 32 Mbytes of the memory map consists of logically mapped RAM. MEMC treats this area of the map as a set of contiguous logical pages. Since there is a minimum of 4 Mbytes physical memory in an Acorn Technical Publishing System, the page size should be programmed to suit 1024 * 32 k logical pages. Refer to the MEMC data sheet (Bibliography, reference 4) for details of how to do this.

When a logical page is accessed, the logical to physical address translator attempts to convert the logical page number to a physical page number. Provided that the mapping exists, and the request is being made in a sufficiently privileged mode, the appropriate physical page will be accessed. If the mapping does not exist, or the access is made with insufficient privilege, MEMC will signal the processor by generating an ABORT and the memory will not be accessed.

The logical to physical mapping and protection status of each logical page is undefined at power on, but may be programmed at any time by writing to the logical to physical address translator.

The combined translator is programmed in the same manner as a single MEMC system. That is by encoding data onto the address bus. Now however Bit [7] is significant as there are 256 entries in the combined translator. The translator in the master will be updated if Bit [7] is a zero, and the translator in the slave if a one. This is transparent to software which should now treat the physical page number as an eight bit value.

A logical page must only appear in the combined translator once.

An abort will be generated only if both translators signal an abort.

3:3 ROM

The read only memory contains the system loader and a limited amount of test code. This provides a mechanism for loading the operating system (UNIX) from one of a number of peripherals including the SCSI based hard disc drive.

3:4 Detailed circuit description

The processor and memory subsystem contains the following blocks.

- ARM processor
- memory manager (MEMC) pair
- system clock generator
- memory (DRAM) array
- address pipeline latch
- read only memory (ROM)

These are now described in detail.

The ARM processor

The ARM processor is IC68. The data sheet for this device is reference 1 in the Bibliography. The following is a summary of local signals and a brief description of their function. A full description of the operation of the ARM processor is contained in the data sheet.

Name	Description
PH2m	Phase Two Clock. This is generated by the master MEMC.
PH1m	Phase One Clock. This is generated by the master MEMC:
R*/W	Not Read / Write. When high this signal indicates a processor write cycle; when low, a read cycle. It becomes valid during phase 2 of the cycle before that to which it refers, and remains valid to the end of phase 1 of the referenced cycle. The timing of this signal is similar to an address line.
OPC*	Not Op-Code Fetch. When low this signal indicates that the processor is fetching an instruction from memory; when high data (if anything) is being transferred. The signal becomes valid during phase 2 of the previous cycle, remaining valid through phase I of the referenced cycle.
MREQ*	Not Memory Request. This signal, when low, indicates that the processor requires memory access during the following cycle. The signal becomes valid during phase I, remaining valid through phase 2 of the cycle preceding that to which it refers.
ABORT	Memory Abort. This is an input which allows the memory system to tell the processor that a requested access is not allowed. It is asserted whenever the MEMC pair detects an access that is illegal for the current processor mode, or when both translation units agree that the logical page being accessed has no physical mapping. The signal is valid before the end of phase I of the cycle during which the memory transfer is attempted.

IRQ*	Not Interrupt Request. This is an asynchronous interrupt request to the processor which causes it to be interrupted if taken low when the appropriate enable in the processor is active. This signal is generated by the input/output controller (IOC) which also acts as an interrupt controller. The signal is level sensitive and is held low until a suitable response is received from the processor.
FIQ*	Not Fast Interrupt Request. As IRQ* but with higher priority.
RESET	Reset. This is a level sensitive input signal which is used to start the processor from a known address. A high level will cause the instruction being executed to terminate abnormally. When RESET becomes low for at least one clock cycle, the processor will restart from address 0. RESET is generated by circuitry in the I/O subsystem. When high the processor will perform dummy instruction fetches with the address incrementing from the point where reset was activated. The address value will overflow to zero if RESET is held beyond the maximum address limit.
TRANS*	Not Memory Translate. When this signal is low it indicates that the processor is in user mode, or that the supervisor is using a single transfer instruction with the force translate bit active. It is used to tell memory management hardware when translation of the addresses should be turned on, or as an indicator of supervisor mode activity.
M[1,01*	Not Processor Mode. These are output signals which are the inverses of the internal status bits indicating the processor operation mode. These signals are not used externally to ARM in this system.
SEQ	Sequential Address. This is an output signal. It will become high when either: - the address for the next cycle is being generated in the address incrementer, so will be equal to the present address (in bytes) plus 4, *or* - during a cycle which did not use memory (MREQ* inactive), when the next cycle will use memory and the address will be the same as the current address. The signal becomes valid during phase 1 and remains so through phase 2 of the cycle before the cycle whose address it anticipates. It is used by the MEMC pair, in combination with the low-order address lines, to deduce that the next RAM cycle can omit the row address phase (page mode) and also by-pass the address translation system.
ALE	Address Latch Enable. This input to the processor can be used to control transparent latches on the address bus outputs. Refer to the MEMC data sheet, (Bibliography, reference 4). This pin is not used in this design and is pulled high (transparent) by R125.

A[25:0]	Addresses. This is the processor address bus. The addresses become valid during phase 2 of the cycle before the one to which they refer and remain so during phase 1 of the referenced cycle. The address bus is latched in the address latch (sheet 6) to provide a stable address for the ROMs and I/O system until the end of the referenced cycle.
ABE	Address Bus Enable. This is an input signal which, when low, puts the ARM address bus drivers into a high impedance state. ABE is pulled high by R123 as there is no system requirement to turn off the address drivers. This pin can be pulled low to facilitate ATE testing.
D[0:31]	Data Bus. These are bi-directional signal paths which are used for data transfers between the processor and external memory, as follows: - during read cycles (when R*/W = 0), the input data must be valid before the end of phase 2 of the transfer cycle - during write cycles (when R*/W = 1), the output data will become valid during phase 1 and remain so throughout phase 2 of the transfer cycle.
DBE	Data Bus Enable. This is an input signal which, when low, forces data bus drivers into a high impedance state. The drivers will always be high impedance except during write cycles.
B*/W	Not Byte / Word. This is an output signal used by the processor to indicate to the external memory system when a data transfer of a byte length is required. The signal is high for word transfers and low for byte transfers and is valid for both read and write cycles. The signal will become valid during phase 2 of the cycle before the one during which the transfer will take place. It will remain stable throughout phase 1 of the transfer cycle. The timing of this signal is similar to an address line.
CPI*	Coprocessor Instruction. When ARM executes a coprocessor instruction, it will take this output low and wait for a response from the coprocessor. The action taken will depend on this response, which the coprocessor signals on the CPA and CPB* Cinoppurtsoc.
CPB*	Coprocessor Busy. A coprocessor which is capable of performing the operation which ARM is requesting (by asserting CPI*), but cannot commit to starting it immediately, should indicate this by letting CPB be pulled high by R166. When the coprocessor is ready to start it should drive CPB low. ARM samples CPB at the end of phase 1 of the cycle when CPI* is low.
CPA	Coprocessor Absent. A coprocessor which is capable of performing the operation which ARM is requesting (by asserting CPI*) should take CPA low immediately. If CPA is high at the end of phase 1 of the cycle when CPI* is low, ARM will abort the coprocessor handshake and take the undefined instruction trap. If CPA is low and remains low, ARM will busy-wait until CPB is low and then complete the coprocessor instruction.

The memory manager (MEMC) pair

The master MEMC is IC69 and the slave MEMC is IC70. The data sheet for these devices is reference 4 in the Bibliography. A full description of the internal operation of a MEMC is contained in the data sheet.

In this part of the circuit where there are two identical MEMC devices it is easy to confuse signal names. To reduce the risk of this occurring signals to/from the master MEMC have the suffix 'm' (REF8m, PHI1m etc) while those signals to/from the slave MEMC have the suffix 's' (REF8s, PHI1s etc). This does not apply where a signal is common to both the master and slave; in this case there is no lower case suffix.

Essentially the address and timing inputs to the two MEMC devices are connected in parallel, while the multiplexed row and column address outputs drive separate memory arrays.

The master MEMC is the system clock generator, providing 8 MHz two phase clocks (PHI1m, PHI2m) for the ARM processor, and a reference 8 MHz clock (REF8m) for the remainder of the system. The clock outputs from the slave MEMC, while in phase with the master are not used. Use of outputs from the slave would prevent the removal of the slave MEMC to produce a 4 Mbyte system, should such a product be required.

For details of the clock scheme see the section *System clock generator* below.

It does this by dividing an input 24 MHz clock (CLK24m) by three. It is important that the reference clocks produced by both the master (REF8m) and slave (REF8s) are in phase to within approximately 10 ns. This is achieved by the clock generator circuit on sheet 7 and described in the next subsection.

The following is a summary of the function of local signals. As both devices are identical most signals are common. Where these are different an explanation is given.

Name	Description
A[0:25]	This is the Processor Address Bus.
R*/W	Processor Not-Read/Write. Determines the direction of data flow during processor accesses.
B*/W	Processor Not-Byte/Word. Determines whether a processor access is byte-wide (8 bits) or word-wide (32 bits). This input is sampled as MRESET goes low to determine whether MEMC is a master (B*/W = 1) or slave (B*/W = 0) controller.
MMREQ	Modified Processor Memory Request. This signal is a modified version of the processor signal - Memory REQuest. The modification' is not required if MEMCIa devices are installed and the PAL (1C72) is tracked across between pins 2 and 19.
MSEQ	Modified Processor Sequential Access. This signal is a modified version of the processor signal SEQuential. The modification is required to support coprocessors.
SPVMD	Supervisor Mode Select. This is connected to the processors TRANS* pin. When low this indicates to MEMC that the processor is in user mode or that the supervisor is using a single transfer instruction with the force translate bit active.

PHI1m, PHI2m PHI1s, PHI2s	The master pair (PHI1 m and PHI2m) provide the processor PHI clocks. These are two phase and non-overlapping. The slave pair (PHI1s and PHI2s) are not used and are left open circuit.
DBEm, DBEs	Processor Data Bus Enable. The master (DBEm) enables the processor data bus during processor write cycles. Both DBEm and DBEs are inverted by gates in IC60 to provide an active low write enable for the respective dynamic RAM banks.
ABORTm, ABORTs	Memory Manager Abort. The logical AND (IC61) of these two signals is the processor ABORT. Each line is driven high by MEMC to inform the processor that the current memory access is illegal or that there is no physical mapping for the requested logical page. In the case of an illegal access both MEMCs will agree the access is illegal and both will signal ABORT: When a mapping exists it will be in one translator only and the other will signal ABORT.
IORQ*	Input/Output Cycle Request. The IORQ signal is driven low by the master MEMC to inform the I/O controllers that an I/O cycle is being requested by the processor. The equivalent slave signal IORQs is not used and is open circuit.
REF8Mm, REF8Ms	8 MHz Reference Clocks. The master REF8Mm signal is buffered in 1064 to provide copies for use by the I/O system (REF8M), the backplane (REF8Mbp) and the clock generator phase detector (REF8MPDm). The slave REF8Ms is only used by the clock generator phase detector but is also buffered by IC64 produce REF8MPDs. See the section *System clock generator* below.
IOGT*	Input/Output Cycle Grant. This input informs both MEMCs that the I/O controller is ready to complete the I0 cycle.
CLK24m, CLK24s	24 MHz Clock Inputs. MEMC generates the 8 MHz reference clocks from these inputs. See the section *System clock generator* below.
MRESET	Modified Reset. This signal is a synchronised version of the asynchronous reset from the I/O subsystem. It is synchronised to REF8m by IC136. This is done to ensure both MEMCs come out of the reset state together.
RAm[0:9], RAs[0:9]	RAM Address Bus. These are the multiplexed row and column address lines to the two DRAM arrays.
RASm*, RASs*	Row Address Strobe. These are the DRAM row address strobes. The high to low transition of RAS* strobes the row address on RA[0:9] into the DRAMs.
CASm*[0:3], CASs*[0:3]	Column address strobes. Each CAS line controls a byte wide column in the DRAM array. The high to low transition of a CAS* line strobes the column address on RA[0:9] into the DRAMs. During a word access CASs[0:3] or CASm[0:3] is active. During a byte access only one CAS line is active, selected by A[0:1].

ROMCS*	ROM Chip Select. This signal is driven low when the processor accesses the Read Only Memory (ROM). Only the master (ROMCS) is used, the equivalent slave signal (ROMCSs) is not connected.
VIDW*	Video Controller Write Strobe. This signal is driven low while the processor is performing a write operation to the video controller (VIDC). Only the master VIDW is used, the equivalent slave signal (VIDWs) is not connected.
MFLYBK	Modified Video Vertical Flyback. This is used to reinitialise the video and cursor DMA address pointers. This signal is a synchronised version of the asynchronous FLYBK from VIDC. It is synchronised to REF8m by IC136. This is done to ensure both MEMCs initialise their counters simultaneously.
HS*	Video Horizontal Synchronisation. This signal selects between video and cursor data buffers. DMA data requests from VIDC made while this signal is low will be returned data from the cursor data buffer. DMA data requests from VIDC made while this signal is high will be returned data from the video data buffer.
MVIDRQ*	Modified Video Data Request. This signal requests a video or cursor DMA operation (depending on the sense of HS*). This signal is a synchronised version of the asynchronous VIDRQ from VIDC. It is synchronised to REF8m by IC136. This is done to ensure both MEMCs service the DMA request together. Note that although both MEMCs service the DMA cycle only the master will generate CAS[0:3] and only data from the master DRAM array will be enabled onto the data bus.
VIDAK*	Video Data Acknowledge. This signal is driven low by MEMC to indicate that the requested video/cursor data is available. Only the master (VIDAK) is used, the equivalent slave signal (VIDAKs) is not connected.
MSNDRQ*	Sound Data Request This signal requests a sound DMA operation. This signal is a synchronised version of the asynchronous SNDRQ from VIDC. It is synchronised to REF8m by IC136. This is done to ensure both MEMCs service the DMA request together. Note that although both MEMCs service the DMA cycle, only the master will generate CAS[0:3] and only data from the master DRAM array will be enabled onto the data bus.
SNDAK*	Sound Data Acknowledge. This signal is driven low to indicate that the requested sound data is available. Only the master (SNDAK) is used, the equivalent slave signal (SNDAKs) is not connected.
SIRQ*	Sound Interrupt Request. This signal generates a processor interrupt if enabled in IOC. It signals that a sound buffer service is required. Only the master (SIRQ) is used, the equivalent slave signal (SIRQs) is not connected.

The system clock generator

The system clock scheme is illustrated below.

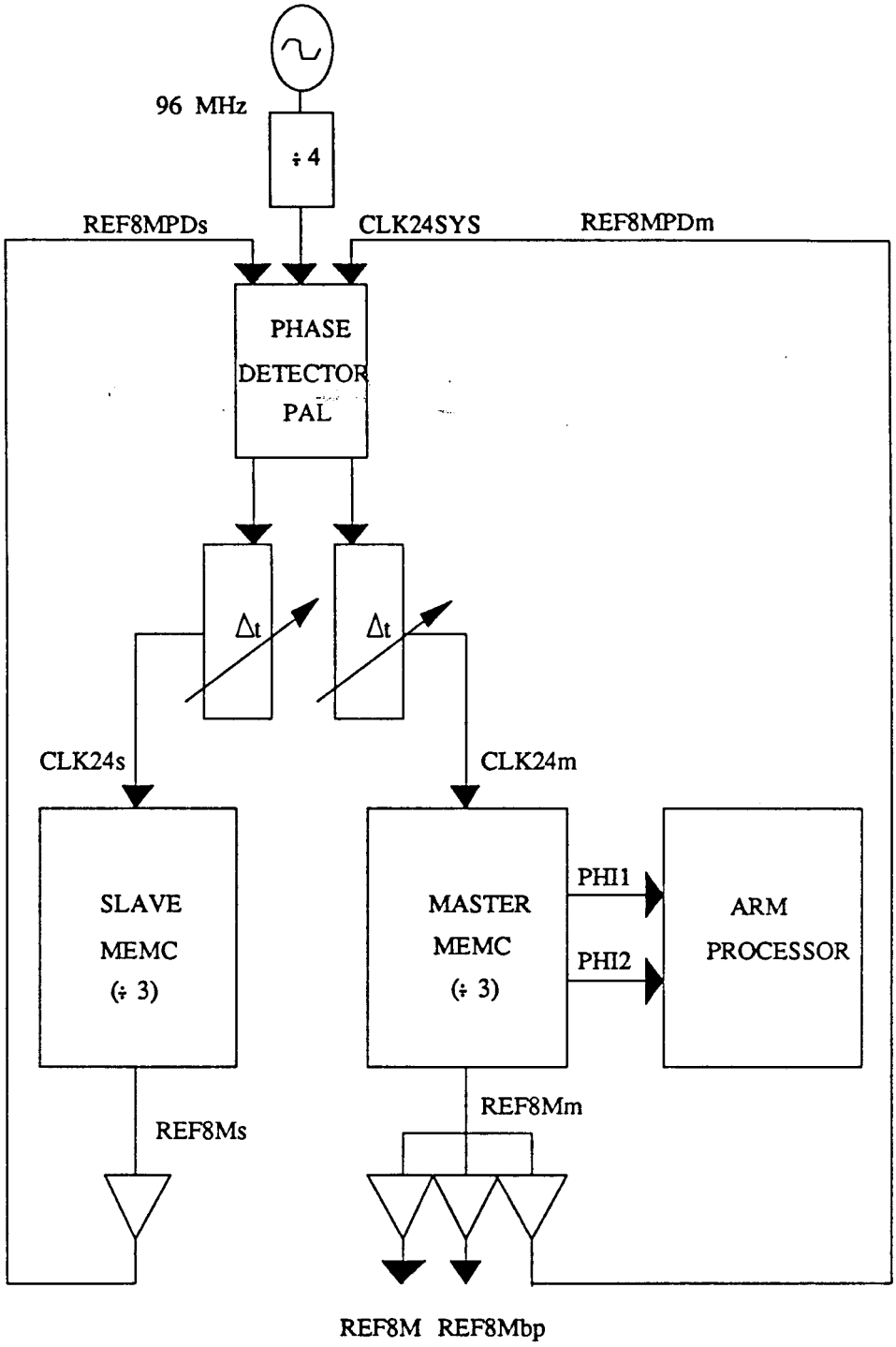

Figure 3.2 : The system clock generator

An ECL 96 MHz oscillator (sheet 2, X1) is divided by four by a quadrature divider (IC14) to produce a 24 MHz signal, CLK24SYS. This can be seen on a test point on sheet 7 (J7, pin 7).

The divide by three block in MEMC between the CLK24 input and REF8M output is not reset on system power-up. This means that the REF8M outputs from the MEMCs can be out of phase. In order for the dual MEMC system to function both master and slave, REF8M must be in phase.

While early reset (ERST*) is active the phase detector PAL (IC55) divides the 24 MHz clock by 4 to produce an asymmetric 6 MHz clock. It then drops cycles to the slave MEMC until the two REF8M clocks are in phase.

The variation in MEMC internal propagation delay due to manufacturing tolerance can also introduce phase error. This is removed by adjusting two tapped delay lines (IC36 and IC29) until the REF8M outputs are within 5 ns. In most systems both MEMCs will be from the same production batch and will have almost identical propagation delays, allowing the delay lines to be by-passed using links.

The outputs of the delay lines are buffered by IC24 and drive the CLK24 inputs of the master and slave MEMC.

The following is a summary of the functions of local signals.

Name	Description
CLK24SYS	System 24 MHz Clock. This is generated from the 96 MHz oscillator in the video circuit on sheet 2.
CLK24m, CLK24s	24 MHz MEMC Clocks. These are the phase modified versions of CLK24SYS for each MEMC.
REF8MPDm, REF8MPDs	REF8M (Phase Detector). Copies of REF8M used by the phase detector.
ERST*	Early Reset. This signal, when active, causes the phase detector PAL to output 6 MHz on the CLK24 signals. It is deasserted approximately one second *earlier* than the RST signal that releases the processor from the reset state.

The memory array

The memory array is constructed from two banks of dynamic random access memory. The master bank (IC115-118, IC120-129, IC133-138, IC141-142, IC144-147, IC149-154) is the low 4 Mbyte of physical memory, while the slave bank (IC75-76, IC79-82, IC84- 91, IC93-96, IC98-99, IC101-102, IC104-109, ICI 11-114) is the high 4 Mbyte of physical memory.

Each of the 32f data lines has a 68Ohm source terminating resistor which is shared between the two banks.

For an explanation of the signals to and from the memory array see the section *Memory manager pair* above.

Address pipeline latch

The ARM processor is a pipelined processor and the value on the address bus A[0:25] changes to the next address before the current cycle is completed. Devices like the ROMs or peripheral controllers are not pipelined and require stable addresses to be held to the end of the current cycle. The address pipeline latch resolves this by latching the current address until the beginning of the next cycle. The figure below illustrates this.

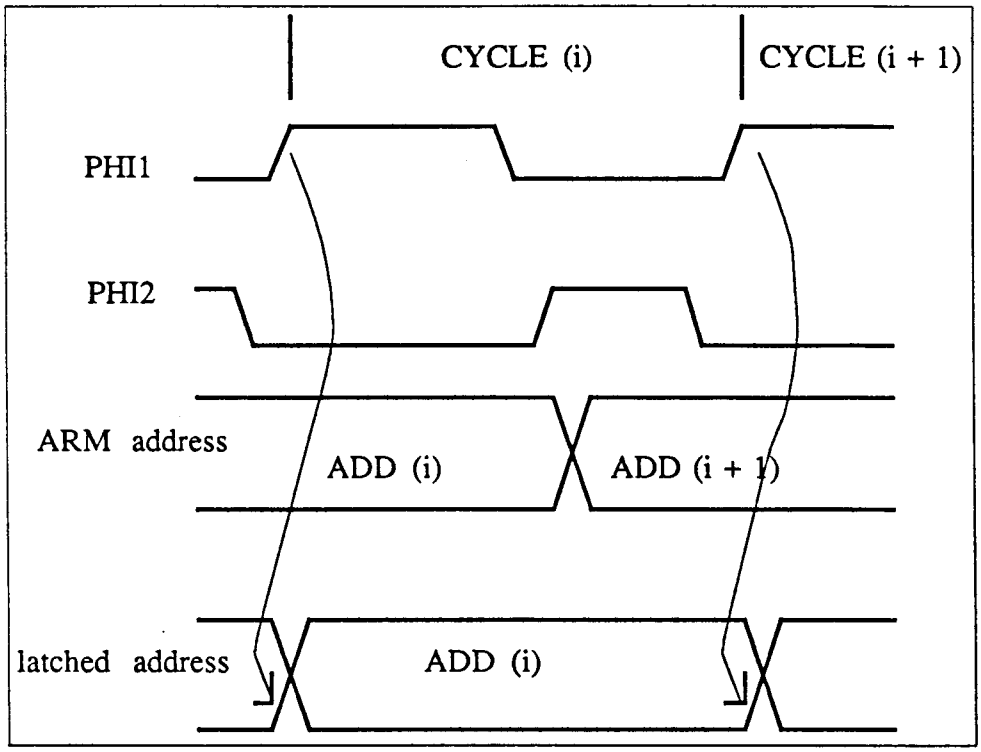

Figure 3.3 : Address pipeline latch timing

The address pipeline latch is IC46 to IC57.

The address decode PAL (IC40) extends the function of the address pipeline latch to the high order address lines (LA17:25). In addition this device performs some advance address decoding, generating various select signals for the SCSI buffer and the hexadecimal display.

The following is a summary of local signals and a brief description of their function.

Name	Description
A[2:181	The Processor Address Bus.
LA[2:18]	The Latched Address Bus.
BPHI1	The Buffered PHI1m Processor Clock. The rising edge of this clock indicates the start of a new processor cycle and allows the next address from the processor through to the non-pipelined devices in the system.
R*/W	Not Read/Write. This is the processor read/write signal. The timing and treatment in IC40 is similar to an address. Two versions of this signal are produced, (LR*/W) and LIOR*/W).
LR*/W	Latched Read/Write. This is R*/W latched on the rising edge of BPHIL
LIOR*/W	Latched Input/Output Read/Write. This is special version of LR*/W. This signal is normally equivalent to LR*/W. However, when the processor is addressing the SCSI buffer this signal is set to a logic one (write). This is done to prevent a bus clash on the processor data bus (D[0:32]) between the SCSI buffer bus drivers and the I/O subsystem bus buffers.
MODULE*	MEMC Podule Select. This signal enables MEMC podules on the podule expansion bus.
SCBSEL*	SCSI Buffer Select. This signal selects the SCSI buffer space in the address map. See Chapter 6, SCSI .
SCCSEL*	SCSI Control Select. This signal selects the SCSI control register space. See Chapter 6, SCSI.
DSPSEL*	Display Select. This signal is pulsed active when the processor reads from the hexadecimal display. This causes the display information on address bits LA[4:7] to be latched into the combined display driver (IC32).
LED*	LED Blank. This is simply LA[25] latched and buffered. It is used to blank the display.

Read only memory (ROM)

The circuit is based around four ROMs which may be 28 pin or 32 pin devices, each containing between 64 k and 1024 kbits. If 28 pin devices are to be installed, then these are fitted into the socket with the device pin 1 in the sockets pin 3. It will be necessary to set or move three links (four jumpers) according to the size of the device to be installed. A table is provided in Appendix C, *System PCB links.*

Name	Description
LA[0:18]	The Latched Processor Address Bus.
SA[16:18]	The function of these signals changes according to the position of links LK12, LK10 and LK13. For devices of 128 kbits or below LK12 and LK10 make SA[16:17] logic one. For devices above 128 kbits SA[16:17] become addresses LA[16:17]. The remaining link, LK13, allows JEDEC 1 Mbit EPROMs - LA[18] on pin 2, or MASK ROM type 1 Mbit EPROMs - LA[18] on pin 24, to be fitted.
D[0:311	The Processor Data Bus.
MROMCS	Modified ROM Chip Select. This active low signal is generated by the master MEMC. ROMCS is driven low at the beginning of every non-sequential access and removed before the end if a DRAM access is in progress. This strategy allows slower ROMs to be used than would otherwise be necessary.

Chapter 4 The hexadecimal display

A significant percentage of the main PCB must be working before faults can be reported on a monitor. In order to reduce this percentage a hexadecimal display is provided, on which test software can report faults. This display is visible when the lid of the Acorn Technical Publishing System is removed. A character is displayed by reading from the corresponding address shown in the table below.

Character	Address
0	0x3400000
1	0x3400010
2	0x3400020
3	0x3400030
4	0x3400040
5	0x3400050
6	0x3400060
7	0x3400070
8	0x3400080
9	0x3400090
A	0x34000A0
B	0x34000B0
C	0x3400000
D	0x34000D0
E	0x34000E0
F	0x34000F0
BLANK	0x3420000

Table 4.1 : Hexadecimal display

4:1 Circuit description

The hexadecimal display is IC32 (see Appendix A). This is an integral data latch, decoder and display.

The base address is decoded by IC78 which produces a strobe to latch the display value.

The character to be displayed is taken from the processor address bus .

When the processor reset is held active it outputs incrementing addresses, causing the display to flash on and off at approximately four second intervals. This can be used as a quick check that the processor is receiving the appropriate clock signals.

4:2 Test ROM LED diagnostics

Following reset the diagnostics below are displayed:

Code	Meaning
0	Should never get this
1	Failed ROM test 1. Check the test part of the ROM using minimal ARM registers and instructions.
2	Failed ARM processor test
3	Failed quick RAM check. Gives extended diagnostics; 3...address... expected data...found data
4	Failed ROM test 2. Full CRC of the ROM image.
5	Failed normal RAM test. Gives extended diagnostics; 5...address... expected data...found data
6	Failed normal MEMC CAM mapping or protection tests
7	Coprocessor failure or ARM processor undefined instruction failure
8	Failed full RAM test. Gives extended diagnostics; 8...address... expected data...found data
9	Failed full MEMC CAM mapping or protection tests
A	Failed parallel port loopback tests. A solid 'A' means the whole I/O world is suspected. Scoping patterns being generated. Scope pattern does repeatedly: Set LED to 'A' Set IOC baud rate to 45.59755 Hz (0x55AA in latch) Read back IOC baud rate (in reverse) Write 0xAA to printer data latch Write 0x55 to printer data latch Read from printer data latch Write 0x00 to printer control latch Write 0xFF to printer control latch Read from printer status latch Extended diagnostics; A...0 — Failed busy/strobe test (suspect PAL) A...1...expected...got — Failed control/status write/read back test A...2...expected...got — Failed data write/read back test A...3 — Failed interrupt operation (could be IOC or the PAL)
Flashing B	Waiting for the test station to engage in cooperative testing.
B	Engaged in cooperative test with the test station. It is up to the test station to report any faults.
C	Not used. This should never happen
D	Unexpected abort; D...0...R14contents — Branch through zero D...1...R14contents — Undefined instruction D...2...R14contents — SWI instruction D...3...R14con tents — Prefetch abort D...4.. R14contents — Data abort D...5...R14contents — Address exception D...6...R14contents — IRQ D...7...R14contents — FIQ
E	Boot started (but not finished)
F	Entered a language, all OK

Table 4.2 : ROM LED diagnostics

Chapter 5 The input/output system

The input/output (I/O) system is divided into two sections:

- The SCSI interface and buffer. This is described later in Chapter 6, *SCSI.*
- The IOC controlled peripheral subsystem (floppy disc, serial line, printer port, CMOS RAM etc) which is described in this chapter. The podule expansion bus, which is also IOC controlled, is described in Chapter 10, *Podules and backplane.*

5:1 Introduction

The input output controller (IOC) is a member of the Acorn RISC Machine (ARM) support chip set, and interfaces directly with the memory controller (MEMC) and the video controller (VIDC) to provide a unified view of interrupts and peripherals within an ARM based system.

IOC manages an I/O data bus to which peripheral controllers are connected, provides a set of internal functions, and controls access to external peripherals. The internal functions include timers, a serial keyboard interface and interrupt control logic to satisfy the basic requirements of a computer system. The peripheral cycles allow standard peripheral controllers from a wide range of manufacturers to be interfaced without any additional logic. A flexible control port offers a number of general purpose input/output pins.

5.2 Block diagram

A block diagram of the I/O system bus is shown in Figure 5.1 below.

The processor data bus is connected to the I/O data bus by a set of latches. These provide two functions. Firstly they isolate the I/O bus load from the processor data bus. Secondly they allow for the mismatch in speed between the two buses.

The buffered data bus is used on the podule expansion bus and to access IOC.

The buffered data bus is further buffered to form the peripheral data bus to which is attached the on-board peripheral controllers.

The latched address bus is the same bus that provides addresses to the system ROMs (see Chapter 3, *Processor and memory subsystem).*

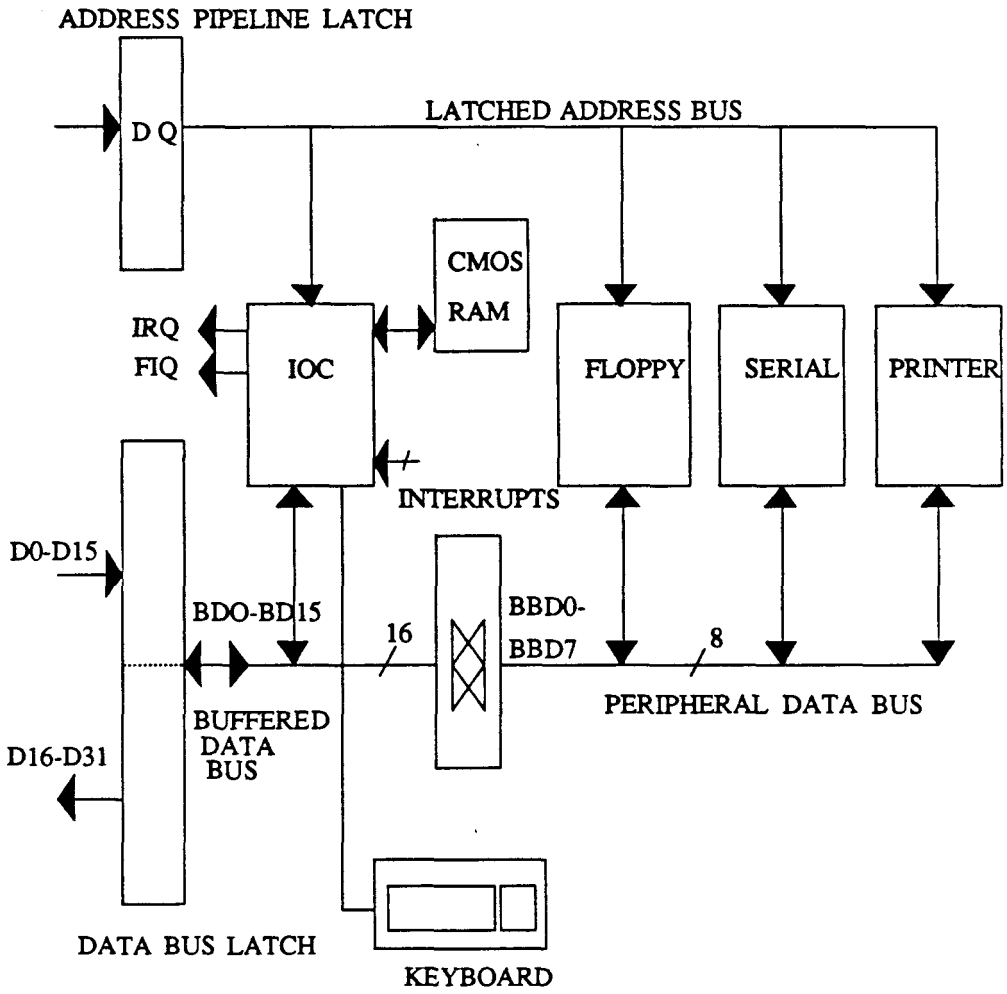

Figure 5.1 : The I/O system bus structure

The I/O system can be divided into the following subsystems:

- the IOC controller
- a bidirectional parallel port (centronics compatible)
- a dual serial line controller (RS232)
- a floppy disc controller
- a battery backed clock/RAM (PCF8583)
- the keyboard
- the podule expansion bus.

Each of the above - except the keyboard and expansion bus which have their own chapters - is described in the following sections.

5:3 I/O address map

The I/O space in the ARM address map is divided into four 1 Mbyte sections. The upper 2 Mbyte is occupied by IOC and IOC controlled peripheral devices, including podules on the expansion bus.

The lower 2 Mbyte is further decoded into two sections. The lower 1 Mbyte is occupied by MEMC podules (modules) on the expansion bus. The upper 1 Mbyte is occupied by the SCSI buffer cache. Both of these are described in their own chapters.

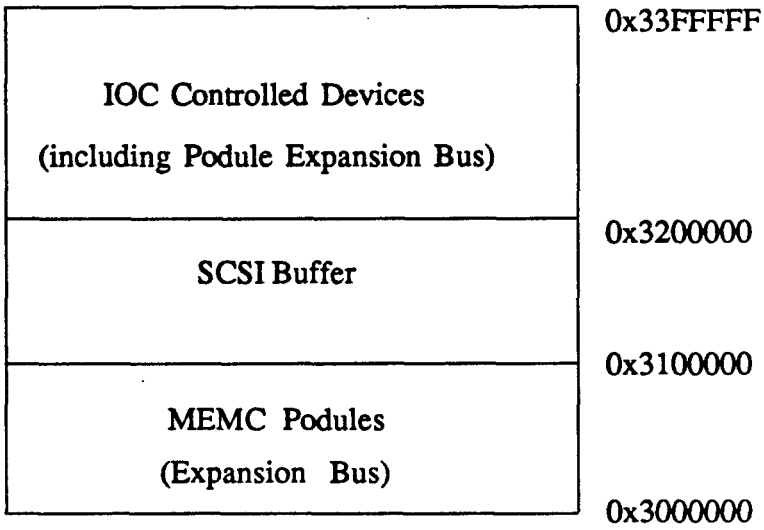

Table 5.1: I/0 address map

Table 5.1 and Table 5.2 illustrate the address map for the I/O system. The cycle type and bank fields in these tables are transparent to the programmer, as the design of IOC means these are coded into the base address of the peripheral being accessed.

Cycle Type	Bank	Base Address	Device	Function
Med	1	0x3290000	1793/9793	Floppy Disc Controller
Fast	5	0x3350040	HC Latch	Floppy Disc Control Latch 1
Fast	5	0x3350010	HC Latch	Printer Data
Fast	5	0x3350018	HC Latch	Printer Strobe
Med	3	0x32B0008	28530	Serial Line Controller
Slow	4	0x324p000	Podule(s)	Internal Podules
Med	4	0x32Cp000	Podule(s)	Internal Podules
Fast	4	0x334p000	Podule(s)	Internal Podules
Sync	4	0x33Cp000	Podule(s)	Internal Podules
Fast	6	0x3360000	PAL	Podule(s) IRQ Request (Read)
Fast	6	0x3360004	PAL	Podule(s) IRQ Mask

Table 5:2 : Peripheral address map

Byte accesses

To access byte wide peripherals byte instructions should be used. A byte store operation will place the written byte on all four bytes of the word and will therefore correctly place the desired value on the lowest byte of the buffered data bus. A byte or word load may be used to read a byte wide peripheral into the lowest byte of an ARM register.

Half-word accesses

To access half-word peripherals, word instructions should be used. When writing to a device the half-word must be placed on the upper 16 bits, D[16:31]. To maintain upwards compatibility half-word stores should replicate the written data on both half-words. When reading from a device the data is returned on the lower half word and the upper half word is undefined.

Word accesses

These are not supported on the Acorn Technical Publishing System.

5:4 The I/O controller

The IOC data sheet (Bibliography, reference 2) describes the I/O controller in detail and specifies the functions of this device.

The IOC and peripherals are memory mapped devices. This allows the programmer to specify in a single memory instruction the peripheral to be accessed and speed of access it requires.

The IOC address space is decoded into eight BANKS, bank zero through seven. The bottom bank, bank zero, selects the internal registers of IOC. The remaining seven banks control seven peripheral select lines.

The seven peripheral banks are each further decoded by IOC into four types of peripheral access. The type of the peripheral access determines the timing of the data transfer cycle.

Recommended cycle types for the peripherals in this system are given in the address map table, Table 5.2 above. The figure below illustrates how the address of a particular peripheral is calculated.

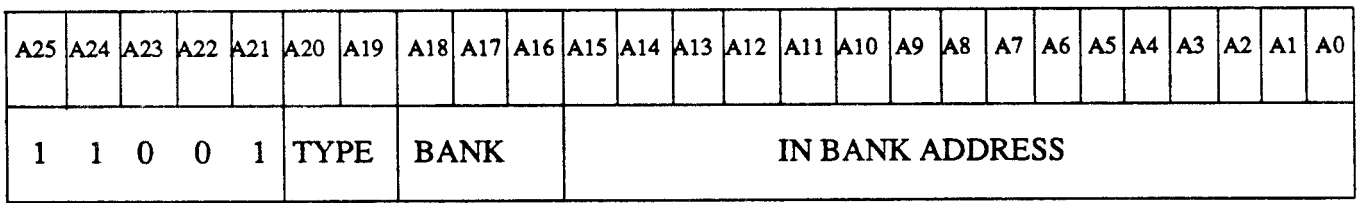

Figure 5.2 : Peripheral address calculation

IOC Internal register map

All internal registers accesses take two REF8M cycles to complete. The internal registers are decoded as bank zero. The individual registers are then selected by A[2:6]. The registers are decoded on word boundaries. The state of the T[0:1] lines is ignored. The address are shown in Table 5.3 below.

Address	Read	Write
0x3200000	Control	Control
0x3200004	KeyBd Rx Data	KeyBd Tx Data
0x3200008	-	-
0x320000C	-	-
0x3200010	IRQ status A	-
0x3200014	IRQ request A	IRQ clear
0x3200018	IRQ mask A	IRQ mask A
0x320001C	-	-
0x3200020	IRQ status B	-
0x3200024	IRQ request B	-
0x3200028	IRQ mask B	IRQ mask B
0x320002C	-	-
0x3200000	FIQ status	-
0x3200034	FIQ request	-
0x3200038	FIQ mask	FIQ mask
0x320003C	-	-
0x3200040	T0 count Low	T0 latch Low
0x3200044	T0 count High	T0 latch High
0x3200048	-	T0 go command
0x320004C	-	T0 latch command
0x3200050	T1 count Low	T1 latch Low
0x3200054	T1 count High	T1 latch High
0x3200058	-	T1 go command
0x320005C	-	T1 latch command
0x3200060	T2 count Low	T2 latch Low
0x3200064	T2 count High	T2 latch High
0x3200068	-	T2 go command
0x320006C	-	T2 latch command
0x3200070	T0 count Low	T3 latch Low
0x3200074	T3 count High	T3 latch High
0x3200078	-	T3 go command
0x320007C	-	T3 latch command

Table 5.3 : Internal register memory map

5.5 Control register

The control register allows the external control pins C[0:5] to be read and written. In addition, the current state of the two edge sensitive interrupts IR and IF inputs can be inspected.

The C[0:5] bits are bi-directional and are allocated as shown in Table 5.4 below.

Note that on reset all bits in the control register are set to '1'.

Writing to the control register

C7	C6	C5	C4	C3	C2	C1	C0
1	1	1	1	1	1	SLC	SDA

Reading from the control register

C7	C6	C5	C4	C3	C2	C1	C0
1	1	1	1	1	DC*	SLC	SDA

Table 5.4 : Control register

C[0:1], I2C SDA and SLC, Bi-directional
These bits implement I2C bus to which the real time clock and battery RAM are connected. The I2C bus is a bidirectional serial bus with just two signals. The serial data line (SDA) is C[0], and the serial clock line (SCL) is C[1]. For further details see the Mullard PCF8583 data sheet, listed in Appendix **E.**

C[2] DC•, Active Low Input
This bit is controlled by the disc changed signal from the floppy disc drive. When low it indicates that the disc has been removed. It will be reset high when a disc has been inserted and the drive heads stepped. When writing to the control register this bit must be set. See also the section *The floppy disc controller* below.C[3]**Not Used**
This bit is not used. This bit will always be set when the control register is read. When writing to the control register this bit must be set.

C[3} Not Used
This bit is not used. This bit will always be set when the control register is read. (A link option exists to make this bit controlled by the alarm signal from the I2C PCF8583 timer. This link is not normally installed.) When writing to the control register this bit must be set.

C[3] Not Used
This bit is not used. This bit will always be set when the control register is read. When writing to the control register this bit must be set.

C[6:7] Test
These bits control internal test modes and must be set.

5:6 Keyboard asynchronous receiver/ transmitter (KART)

The KART provides an asynchronous serial link to the keyboard. It is of fixed format with 8 bits to a character which is framed with one start bit and two stop bits. The least significant bit KD[0] is transmitted/received first. The KART divides into two, the receiver and the transmitter.

The ARM accesses the receiver via the serial Rx data register: A clock of 16 times the data rate is used by the KART to clock in the serial data from the KIN pin. When a data byte has been received, the SRx bit is asserted in the IRQ status B register to indicate that the byte is available for reading. False start bits of less than a half bit duration are ignored. Reading from the serial Rx data register clears any outstanding SRx interrupt and returns the currently received byte. Data is only valid while the SRx bit is set in the IRQ status B register.

The ARM accesses the transmitter via the serial Tx data register. The byte written to the serial Tx data register is transmitted serially from the KOUT pin, and the STx bit is asserted in the IRQ status B register to indicate that the transmission is finished and the serial Tx data register may be reloaded. Reloading this register loads the serial output shift register, and clears any outstanding TRx interrupt and starts transmission.

The receive and transmit speeds are the same and are programmed using counter 3.

Programming and initialisation

The keyboard protocol is described in Chapter 9, *Keyboard and mouse.*

After power-on, the KART is in an undefined state. The KART is initialised by programming the serial line speed using counter 3 and performing a read from the serial Rx data register, discarding the data byte. This will clear any outstanding receive interrupt and enable the KART for the next reception. After this the Tx data register should be written to. This will abort any transmission in progress, cause a new one to be started, and clear any STx interrupt.

Receive interrupt

The SRx interrupt is set halfway though the reception of the last data bit. Care should be taken to ensure that the last bit has been received before the serial Rx data register is read, to prevent this bit being interpreted as the start bit of the next packet.

5.7 Counters

IOC contains four identical 16 bit counters. Two are used as timers, the third as a general purpose BAUD rate output and the fourth for the keyboard line rate. They all have fully programmable start/reload values.

Each counter consists of a 16 bit down counter, a 16 bit input latch (latch low and latch high) and a 16 bit output latch (count low and count high) which contains the value of the counter when a latch command access cycle is performed. The counter decrements continuously, clocked at 2 MHz. When it decrements to zero, it is reloaded from the input latch and recommences decrementing. The reload is used to trigger different *events* depending on the use of the counter. If a counter is loaded with zero it continuously reloads and does not count. If the GO register is written at the same time as the counter reloads an extra 2 MHz clock tick is taken to reload. After power-on the state of the counters is unknown.

latch = latch low + 256 * latch high

Register actions

Latch low	Writing to this updates the low order byte of the input latch
Latch high	Writing to this updates the high order byte of the input latch
Go command	Writing to this *causes the* counter to be reloaded immediately with the latch value
Count low	This causes the low order byte of the output latch to be read
Count high	This causes the high order byte of the output latch to be read
Latch command	This causes the current value of the counter to be placed in the output latch

Counter registers

	Address for counter 0	1	2	3	read or write
Latch low	40H	50H	60H	70H	write
Latch high	44H	54H	64H	74H	write
GO command	48H	58H	68H	78H	write
Count low	40H	50H	60H	70H	read
Count high	44H	54H	64H	74H	read
Latch command	4CH	5CH	6CH	7CH	write

Table 5.5 : Counter registers

Counters 0 and 1

Counters 0 and 1 are used as timers. In the Acorn Technical Publishing System timer 0 is programmed to produce an interrupt at 10 millisecond intervals, while timer 1 *is* used by the keyboard driver software.

The reload event sets a timer interrupt, TM[0:1] in the IRQ status A register. The interrupt is cleared via the IRQ clear A register. In order to generate an interrupt after time, $T_{interval}$, the 16 bit value, (latch), to be used is calculated from the following equation:

$$T_{interval} := latch/\mu seconds$$

Counter 2 (BAUD)

The counter 2 output is used to drive the BAUD pin. In this system this is used as a baud rate source for the serial line controller when split rates are required. It should not be used as a baud rate source at or above 9600: The reload event toggles the BAUD clock line. In order to generate a clock of frequency f_{BAUD}, the 16 bit value (latch) to be used is calculated from the following equation:

$$f_{BAUD} := 1/(latch+1)\ MHz$$

Counter 3 (KART)

The counter 3 output controls the speed of the keyboard serial link. Keyboards supplied with the Acorn Technical Publishing System run at the maximum 31.25 K baud. In order to generate a baud rate $k_{BAUD,}$ the 16 bit value (latch), to be used is calculated from the following equation:

$$k_{BAUD} := 1/((latch+1)*16) \text{ MHz}$$

The maximum baud rate of 31.25 K baud is obtained by programming latch=1.

5.8 IOC interrupt allocations

The I/O system generates two independent processor interrupts, IRQ* and FIQ*. Interrupt requests can be caused by events internal to IOC **or** by external events on the interrupt or control port input pins.

The interrupts are controlled by four types of register; status, mask, request and clear. The status registers reflect the current state of the various interrupt sources. The mask registers determine which sources may generate an interrupt. The request registers are the logical AND of the status and mask registers and indicate which sources are enabled and generating interrupt requests to the processor. The clear register allows clearing of interrupt requests where appropriate. The mask registers are undefined after power up.

The IRQ events are split into two sets of registers A and B There is no hardware priority encoding of the sources.

Internal Interrupt events

- Timer interrupts *TM[0:1]*
- Power-on reset *POR*
- Keyboard Rx data available *SRx*
- Keyboard Tx data register empty *STx*
- Force interrupts *'1'*

External interrupt events

- IRQ active low inputs *IL[0:7]*
- IRQ falling-edge input *IF*
- IRQ rising-edge input *IR*
- FIQ active high inputs *FH[0:1]*
- FIQ active low input *FL*
- Control port inputs *C[3:5]*

Level triggered interrupts
The majority of external and a few of the internal interrupt sources are level sensitive. When one of these sources has caused an interrupt it is cleared by removing the source.

Latched interrupts
A few sources are edge sensitive. That is, once one of these sources has caused an interrupt, it must be cleared by an explicit write of '1' to the appropriate bit in the IRQ clear A register. One or many may be cleared in a single operation.

Synchronisation

All the interrupt sources are synchronised by the REF8M clock input. It can take up to three clock phases before a source is recognised as requesting an interrupt, and the same delay occurs between a level sensitive request going inactive at an input pin and the removal of the corresponding bit from the status register and the processor interrupt line.

Interrupt status registers

There are three interrupt status registers (IRQ status A, B and FIQ status).

Note: The 'active' level specified refers to the logic level on the input pin required to set active (logic 1) the corresponding bit in the status register. Logic within IOC takes care of any inversion required.

IRO status register 'A'

The bits within the IRQ status register A are defined below.

BD7	BD6	BD5	BD4	BD3	BD2	BD1	BD0
1	T1	T0	POR	VFLY	PINT	RII	PBSY

Table 5.6 : IRQ status register A bits

Bit [0] Printer Busy (PBUSY) Active low, LEVEL Triggered
This bit indicates that the printer is busy.

Bit [1] Ring Indicate (RII) Active low, LEVEL Triggered
This bit indicates that a ringing indication has been detected by the serial line interface. Ile ring indicate signal is derived from the bell envelope and will generate an interrupt until the call is 'answered' by the modem.

Bit [2] Printer Interrupt (PINT) NEGATIVE EDGE Triggered
If the printer port is configured as an output, this bit indicates that printer acknowledge has been received, so new data can be written to the printer port.

If the printer port is configured as an input, this bit indicates that a data strobe has been received, so new data can be read from the printer port.

Bit [3] Vertical Flyback (VFLYBK) POSITIVE EDGE Triggered
This bit indicates that a vertical flyback has commenced.

Bit [4] Power-on Reset (POR)
This bit indicates that a power-on reset has occurred.

Bit [5:6] Timer 0 and Timer 1 events
These bits indicate that timer events have occurred.
Note: These are latched interrupts.

Bit [7] Force
This bit is permanently set (1) and is used to force an IRQ interrupt.

IRO status register 'B'

The bits within IRQ status register B are defined below.

BD7	BD6	BD5	BD4	BD3	BD2	BD1	BD0
SRx	STx	PIRQ	DCIRQ	SCSI	SLCI	SIRQ	PIRQ

Table 5.7 : IRQ status register B bits

Bit [0] Podule Fast Interrupt Request (PFIQ), active low, level triggered
This bit indicates that a podule FIQ request has been received. It should usually be masked OFF.

Bit [1] Sound Buffer Swap (SIRQ), active low, level triggered
This bit indicates that the MEMC sound buffer pointer has been reloaded.

Bit [2] Serial Line Controller (SLCI), active low, level triggered
This bit indicates that a serial line controller interrupt has occurred.

Bit [3] SCSI Interface Controller (SCSII), active low, level triggered
This bit indicates that an interrupt from the SCSI controller has occurred.

Bit [4] Floppy Disc Interrupt Request (FIRQ), active low, level triggered
This bit indicates that a floppy disc has completed a command.

Bit [5] Podule Interrupt Request (PIRQ), active low, level triggered
This bit indicates that a podule IRQ request has occurred. To identify which slot has caused the interrupt, refer to Chapter 10, *Podules and backplane*.

Bit [6] Keyboard Transmission Event (STx), level triggered
This bit indicates that the keyboard transmit register is empty and may be reloaded. Cleared by writing to the KART Tx data register.

Bit [7] Keyboard Reception Event (SRx), level triggered
This bit indicates that the keyboard reception register is full and may be read. Cleared by reading from the KART Rx data register.

FIQ status register

The bits within the FIQ status register B are defined below.

BD7	BD6	BD5	BD4	BD3	BD2	BD1	BD0
1	PFIQ	0	0*	0	0	FFIQ	FFDQ

* See Text

Table 5.8 : FIQ status register bits

Bit [0] Floppy disc data request (FFDQ), Active high, LEVEL Triggered
This bit indicates that a floppy disc data request has occurred:

Bit [1] Not used
Not used.

Bit [2] Not used (EFIQ), Active low, LEVEL Triggered
This is not used and is pulled high (inactive).

Bit [3] Not used (C3), Active low, LEVEL Triggered
This is not used and is pulled high (inactive).

Bit [4] Not used (C4), Active low, LEVEL Triggered
This is not used, and is pulled high (inactive). If required the ALARM interrupt from the CMOS Clock/RAM can be linked in to this interrupt.

Bit [5] Not used (C5)
Not used.

Bit [6] Podule FIQ request (PFIQ), Active low, LEVEL Triggered
This bit indicates that a podule FIQ request has occurred. See Chapter 10, *Podules and backplane* to find out how to identify which podule has generated the FIQ request.

Bit [7] Force
This bit is permanently set (1) and is used to force an FIQ interrupt.

IRQ clear register

This register is used to clear the edge triggered interrupts described above. An interrupt may be cleared by writing to this register with the corresponding bit set to a '1'.

BD7	BD6	BD5	BD4	BD3	BD2	BD1	BD0
0	T1	T0	POR	VFLY	PACK	0	0

Table 5.9 : IRQ clear register bits

Interrupt mask registers

These three registers have bit positions which correspond to the three interrupt status registers.

Writing a '0' to a bit in a mask register disables interrupts from the corresponding bit in the status register.

Writing a '1' to a bit in a mask register enables interrupts from the corresponding bit in the status register.

Note: All unused interrupt sources should be masked off.

For the bit positions refer to the tables above detailing the status register bit positions.

The mask registers are readable to simplify the sharing of these registers between a number of interrupt handlers.

The addresses of these registers are defined in Table 5.3.

Interrupt request registers

These three registers have bit positions which correspond to the three interrupt status registers. They show which interrupts are currently enabled and active.

Reading a '0' from a bit in an interrupt request register indicates that the corresponding interrupt source is either masked or not requesting an interrupt.

Reading a '1' from a bit in an interrupt request register indicates that the corresponding interrupt source is enabled and requesting an interrupt.

For the bit positions refer to the tables above detailing the status register bit positions. The addresses of these registers are defined in Table 5.3.

Circuit description

The majority of the I/O system is contained on the circuit diagrams in Appendix A. The IOC is IC100, the data bus latch is IC78, 77, 74 and 73.

The system reset circuit is part of IOC and is in two sections. On power-up C57 charges via R146. When the threshold of IC110 pin 3 is reached, the signal ERST* - early reset -goes inactive and C81 charges via R148. When the threshold of IOC pin 28 is reached, the signal RST* goes inactive and the processor leaves the reset state. Diodes D10 and D9 protect the CMOS inputs of IC110 and IC100 (IOC) against latch-up and ensure correct operation when the power supply is removed momentarily.

A reset can also be generated by pressing the reset key on the rear of the keyboard if link LK17 is installed. However, on a UNIX system the processor should only be reset following the correct shutdown procedure and so this link will not normally be installed.

The keyboard interface is constructed from two gates within IC110 (pins 5/6 and pins 8/9). The resistor and diode network D11, D12, R157, R156 protects the keyboard output against static discharge damage. The keyboard is powered from the 5V DC rail via a protection fuse F2.

The resistor pack RN9 provides pull-up resistors for those IOC signals that require them.

R152 pulls up the buffered data bus bit BD(1) so that the processor can identify which podule expansion bus slot has a podule installed. (See Chapter 10, *Podules and backplane.)*

The following is a summary of local signals and a brief description of their function.

Name	Description
REF8M	8 MHz reference clock. The timings of all interface signals to ARM and MEMC are referenced to this clock.

CLK8 8 MHz clock for external peripherals. The timing of all interface signals to peripheral controllers are referenced to this clock. No relationship between CLK8 and REF8M should be assumed.

CLK2 2 MHz clock for external synchronous peripheral timing. When performing a synchronous type cycle the timing of all interface signals to the peripheral controller are referenced to this clock. No relationship between CLK2 and REF8M should be assumed.

IORQ* I/O cycle request. A low on this input indicates that the ARM is performing an I/O cycle.

IOGT* I/O cycle grant. An I/O cycle is complete when IOGT and IORQ are both low on a rising edge of REF8M.

BL* Buffer latch control for the ARM/ I/O data.

BD[0:15] The buffered data bus used on the podule expansion bus and for accesses to IOC internal registers.

RBE* Read buffer enable. This enables the read half of the data bus latch.

WBE* Write buffer enable. This enables the write half of the data bus latch.

LA[19:20] Type address lines. These are used to control type inputs on IOC and so control the timing characteristics of any peripheral access.

IOR*/W I/O not read/write. This line determines the direction of data transfer in an I/O cycle: low to read or high to write an I/O device. It is forced to the write state (logic 1) when a read access to the SCSI buffer occurs to prevent a bus clash on the processor data bus D[0: 31).

LA[16:18] Bank address lines. These control IOC bank select lines and access to either an internal IOC register or to a peripheral.

S*[1:7] Active low peripheral. Selects which indicate valid address and write data. They are decoded from B[0:2].

LA21 IOC chip select address line. When high allows internal register and peripheral accesses to be performed. Even when low, IOC continues to control the RBE and WBE lines. This allows MEMC podules to be located on the podule expansion bus.

LA[2:6] Address lines to select IOC internal registers.

PRE* Peripheral read enable, used to time peripheral read accesses.

WE Peripheral write enable, used to time peripheral write accesses.

IRQ* Interrupt request to ARM.

FIQ* Fast interrupt request to ARM.

PFIQ* Expansion podule FIQ request.

SIRQ*	Sound channel IRQ request.
SLCI*	Serial line controller IRQ request.
SCSII*	SCSI controller IRQ request.
PIRQ*	Expansion podule IRQ request.
PBSY*	Printer port busy IRQ request.
RII*	Serial port ring indicate IRQ request.
FLYBK*	Video flyback IRQ request.
PINT*	Printer port IRQ request.
FFDQ*	Floppy disc data transfer FIQ request.
FIRQ*	Floppy disc controller IRQ request.
EFIQ*	Not used (pulled inactive by RNI0).
C0	Bus serial data.
C1	Bus serial clock.
C2	Floppy disc changed IRQ request.
BAUD	Baud rate generator output. This is used by the serial line controller when split transmit/receive rates are required.
KIN	Keyboard serial interface input.
KOUT	Keyboard serial interface output.
POR*	Power-on reset input, usually connected to an external RC network. It is used to generate a reset pulse at power-on and to differentiate power-on from subsequent causes of reset.
RST*	Reset. The reset line is driven low by POR* at power-on and may be driven low externally at any time.
KRST*	Keyboard reset. The KRST* line is driven by the keyboard reset key.
KTX*	Keyboard transmit data. The KTX* line is a buffered version of the IOC output KOUT and is the serial data line to the keyboard.
KRX*	Keyboard receive data. The KRX* line is the serial data line from the keyboard.

5.9 The printer port

The printer port is a bidirectional parallel interface. It is designed to drive printers or document scanners that support a centronics-like interface. The printer port is configured as an output on system power-on.

In addition to the standard centronics-type strobe, busy and acknowledge flags, the interface also supports paper error, select, auto-line-feed, printer error, reset and select input. These may not always be supported by the particular printer used.

The port can be configured as an input to provide a connection mechanism for devices like document scanners. This mode also allows the printer port to be tested without a printer or loop-back cable.

The address of the printer data and control registers is shown below:

Data register

BD7	BD6	BD5	BD4	BD3	BD2	BD1	BD0
PD7	PD6	PD5	PD4	PD3	PD2	PD1	PD0

Control outputs register

BD7	BD6	BD5	BD4	BD3	BD2	BD1	BD0
0	0	PRST	CBSY	ACK	ALF	PDIR	STRB

Control inputs register

BD7	BD6	BD5	BD4	BD3	BD2	BD1	BD0
SELECT	ERROR	PRST	BSY	ACK	ALF	PDIR	PE

Table 5.10: Printer interface

The data register

The data port is a bidirectional 8 bit port. The direction of this port is controlled by the printer port direction bit (PDIR) in the control outputs register.

The control outputs register

The control outputs register provides the following control and strobe outputs:

BIT [0] STRB
Printer strobe. An active high output signal which strobes data into the printer. The timing of this signal is controlled by the printer driver. Typical timing parameters are shown in Figure 5.3.

BIT [1] PDIR
Printer port direction. This bit controls the direction of the printer data port. Writing a logic zero makes the printer port an output. This bit is cleared on power-up.

BIT [2] ALF
Auto-line feed. This active high bit puts the printer into auto-line feed mode (if this is supported by the printer). It *is* cleared on power-up.

BIT [3] ACK
Acknowledge. This active high bit will normally only be used for testing the printer acknowledge input, at other times it should be cleared. It is cleared on power-up.

BIT [4] CBSY
Clear busy. This active high bit is used to clear the printer port busy flag. It is only used when the printer port is in input mode, and at other times should be cleared. It is cleared on power-up.

BIT [5] PRST
Printer reset. This active high bit is used to reset the printer. At other times it should be cleared. It *is* set then cleared on power-up.

BIT [6,7] RSVD
Reserved write zeros.

The control Inputs register

The control inputs register provides the following control and strobe inputs:

BIT [0] PE
Paper error. An active high input indicating an error condition related to the printers paper supply, ie out of paper.

BIT [1] PDIR
Printer direction. This input is only used to test the printer direction bit (PDIR) in the output control register. It should be the same polarity.

BIT [2] ALF
Auto-line feed. This input is only used to test the auto-line feed bit (ALF) in the output control register. It should be the same polarity.

BIT [3] ACK
Printer acknowledge. This active high input is the standard printer acknowledge flag. If the printer port is configured *as an* output, then an interrupt (PINT*) will be generated when this bit *is* set active by the printer.

This bit can be tested by writing to the acknowledge (ACK) bit in the control output register. It should be the same polarity.

BIT [4] BSY
Busy. When the printer port is configured as an output, this active high input is the standard printer busy flag. When cleared by the printer an interrupt PINT* is generated.

When the printer port is configured as an input this flag can be used to test the state of the host (A680) busy flag.

BIT [5] PRST
Printer reset. This input is used to test the printer reset output bit (PRST) in the control output register. It should be the same polarity.

BIT [6] ERROR
Printer error. This active low input indicates an error condition in the printer (if the printer supports this mode).

BIT [8] SELECT
Select. This active low input informs the host that the printer is on-line if the printer supports this mode.

Most printers can be operated on a subset of the signals available at the printer port. This subset will normally be; the data bus, strobe, acknowledge and busy. These are the only signals required to support the basic output mode of operation.

The basic output sequence performed by the processor is:

1. Wait for busy inactive interrupt.
2. Write data to data register and wait for t1.
3. Assert strobe for t2.
4. Wait for Ack interrupt.

Figure 5.3 below illustrates the bus timing and specifies limits for a typical printer and Figure 5.4 for a document scanner.

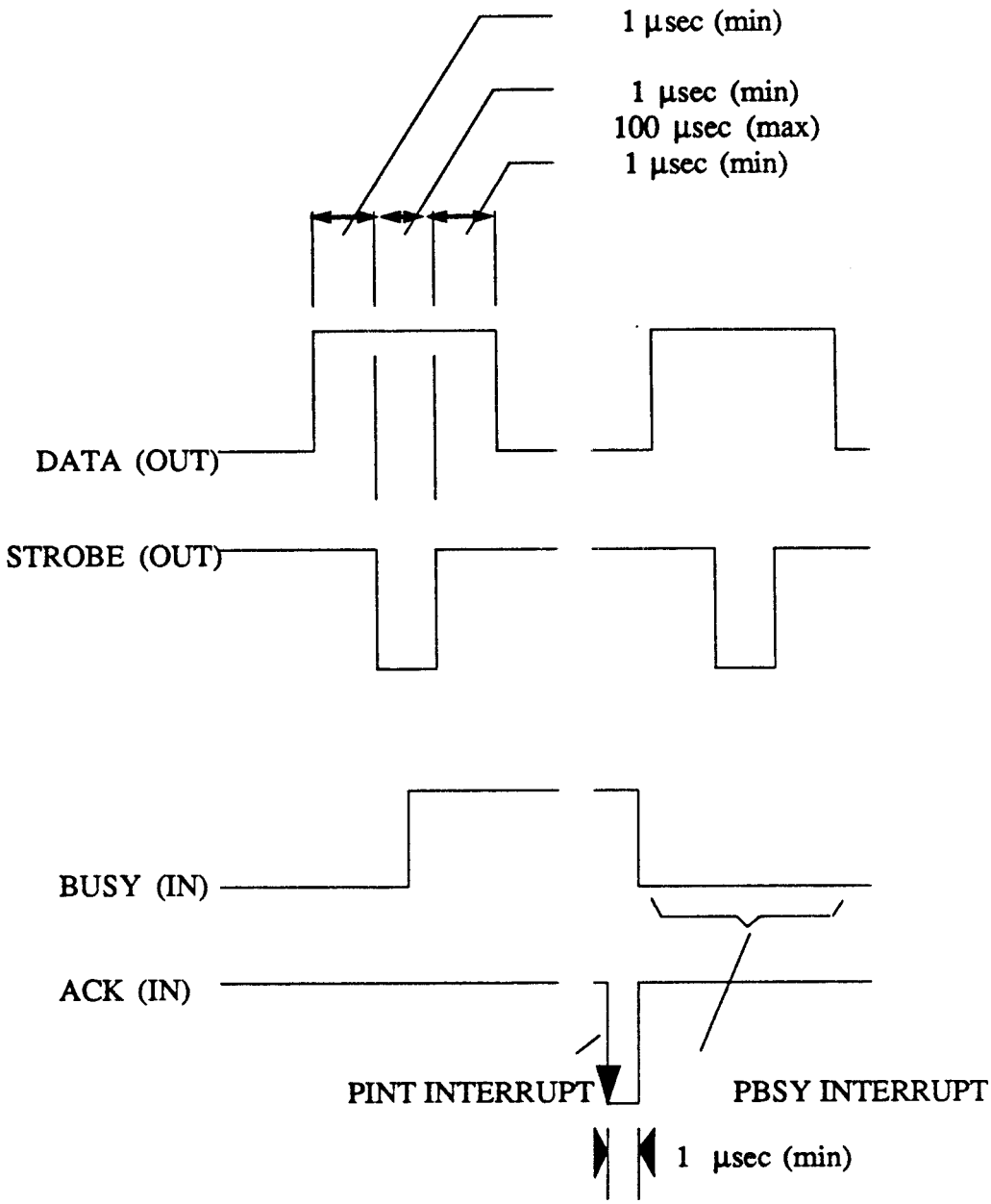

Figure 5:3 : Parallel printer port output timing

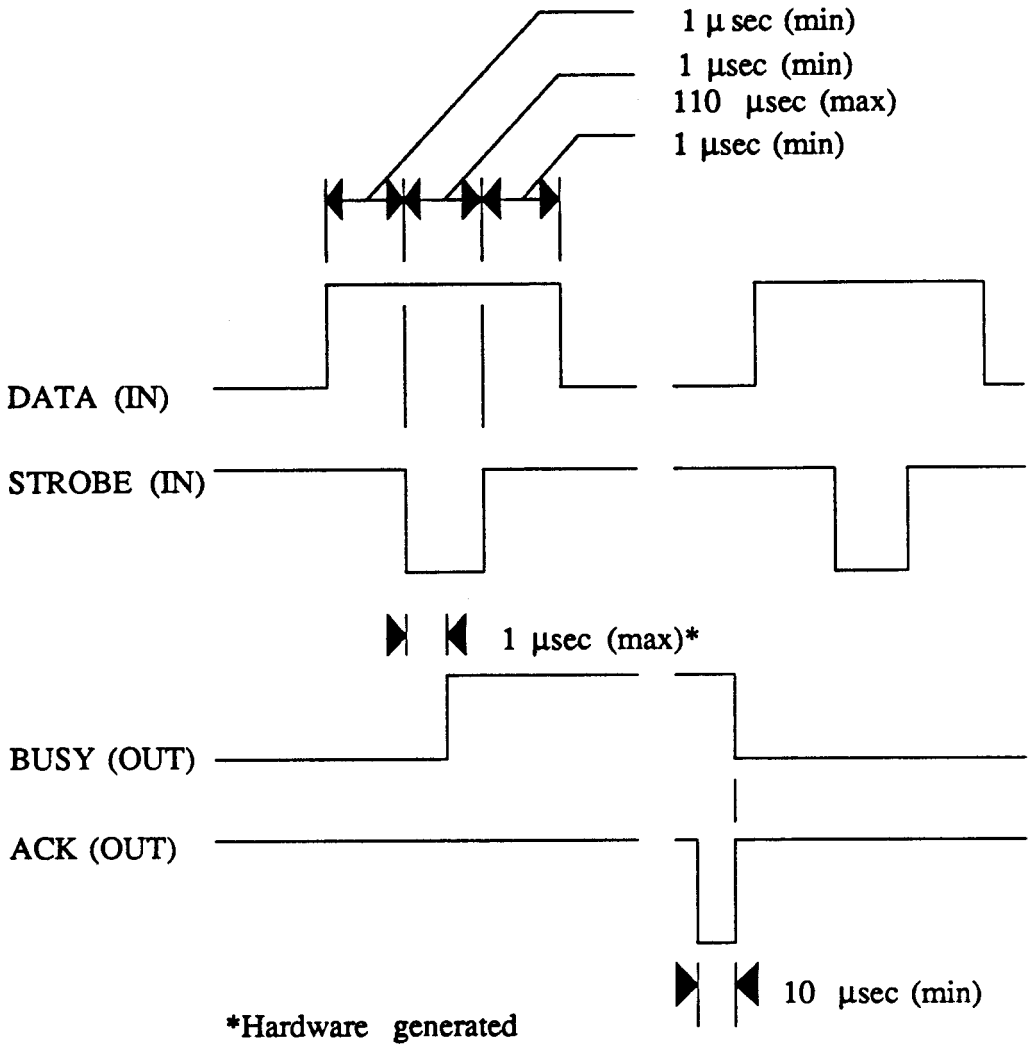

Figure 5.4 : Parallel printer port input timing

Interface characteristics

All signals are compatible with standard TTL logic levels.

Data bus

The data bus is buffered by a 74LS646 capable of sourcing -2.6 mA and sinking 14 mA. The recommended load on these signals is one LSTTL load and a 4k7 pull-up to 5 V.

Control outputs

The control outputs are buffered by open collector inverters, the outputs of these are pulled to 5 V by 4k7 pull-up resistors. Each output is capable of sinking 14 mA without exceeding the maximum TTL logic 0 voltage, (0.8 V). The recommended load on these signals is one LSTTL load and a 4k7 pull-up to 5 V.

Control inputs

The control inputs present one LS load and a 4k7 pull-up resistor. Where a signal is both a control output and a control input, only one 4k7 pull-up resistor is fitted. Input filters of 47Ohms in series with 1 nF capacitors to 0 V reject input pulses shorter than approximately 0.5 µs. The minimum input pulse width is 1 µs.

Circuit description

The printer port circuit diagram is contained in Appendix A.

The printer port data bus is latched and buffered in both directions by IC5, a 74LS646. The control outputs register is split across IC8, a 74HC174 and IC9, a 16L8 PAL. The control input register is IC12, a 74HCT573.

The operation of the circuit is different in the two modes, as described below.

Output mode
In output mode (PDIR = 0) the transceiver outputs ('A' side) are permanently enabled (POE* = 0). The output data latch is clocked on the rising edge of S510*. This is a decoded version of S5* which is a peripheral select from IOC. The decoding is done by IC83 an HC139. For more information see the circuit diagrams in Appendix A.

In this mode the strobe (STROBE), auto-line-feed (AUTOFDX), select (SELECTIN*) and printer reset (PTRRST) outputs are driven. These are all generated from bits in the control outputs register. However, select is produced indirectly from printer direction in IC9. All are inverted by IC4 and RFI filtered by an RC network before driving the printer cable.

The printer drives the acknowledge (ACK*), printer busy (BUSY), paper error (PE), selected (SELECT) and error (ERROR) inputs. These are filtered and inverted by IC16 and drive bits in the control inputs register.

The interrupt PINT* is generated by 1C9 on the falling edge of ACK* and the interrupt PBSY *is* generated when BUSY goes (and *is)* inactive. See the section *IOC interrupt allocations* above for more details.

Input mode
This mode is typically used to receive data from a document scanner.

In input mode (PDIR = 1) the transceiver outputs (`B' side) are enabled when the processor reads from the data register address (S510* is active). The signal S510* is a decoded version of 55* which is a peripheral select from IOC. The decoding is done by IC83 an HC139.

In this mode the input data latch is clocked by the rising edge of STROBEIN which is a filtered (R85, C198) and inverted (IC16) version of the input STROBE* produced by the scanner.

The interrupt PINT* is generated by IC9 on the falling edge of STROBE*. See the section *IOC interrupt allocations* above for more details.

The falling edge of STROBE* sets the PBSY latch in IC9. When the processor has responded to the STROBE* interrupt and read the data from the data register, it should assert acknowledge (ACK) and clear busy (CLRBSY) bits in the control outputs register. When clear busy is removed by the processor, the PBUSY latch is cleared.

5:10 The serial port

The serial port is based on the Zilog Z8530 Serial Communications Controller (SCC). The SCC is a dual channel, multi-protocol device. This design supports one asynchronous channel (channel A) and provides limited support for an external modem.

Transmission and reception can be accomplished independently with between five and eight bits per character and optional parity (even, odd or none):

The SCC incorporates one baud rate generator per channel. A third is provided by the I/O controller IOC BAUD: This allow the SCC to be configured for a range of baud rates including split rates.

The table below lists the common baud rates that are supported by the operating system.

50	baud
75	baud
110	baud
134.5	baud
150	baud
200	baud
300	baud
600	baud
1200	baud
1800	baud
2400	baud
4800	baud
9600	baud
19200	baud (not guaranteed)

Table 5.11 : Supported RS232 baud rates

The SCC serial transmit data, receive data, and control signals are buffered externally to RS232 levels.

The table below is a list of RS232 signals supported.

Signal	**Pin**	**Direction**	**Description**
DCD	1	Input	Data Carrier Detect
RxD	2	Input	Receive Data
TxD	3	Output	Transmit Data
DTR	4	Output	Data Terminal Ready
0V	5	---	Signal Return
DSR	6	Input	Data Set Ready
RTS	7	Output	Request To Send
CTS	8	Input	Clear To Send
RI	9	Input	Ring Indicate

Table 5.12: Supported RS232 signals

Programming the SCC

The serial line controller address map is shown below.

Cycle Type Bank	Address	Device	Function
Med 3	0x32B0000	Z8530	Channel B Control *
Med 3	0x32B0004	Z8530	Channel B Data *
Med 3	0x32B0008	Z8530	Channel A Control
Med 3	0x32B000C	Z8530	Channel A Data
			* No hardware support

Table 5.13 : Serial line controller address map

The SCC contains thirteen writeable registers (WR1-WR13) in each channel. Access to the data registers is by a direct read or write from the register concerned. Access to all others (except write register 0 and read register 0) must be preceded by a write to write register 0 to set the register pointer field. Thus writing to a register requires two write operations, and reading from a register requires one write and one read. This is a form of indirect addressing. The register pointer field is automatically reset to zero after the second access so that WR0 or RR0 is addressed again.

The data sheet listed in Appendix E provides information about the function and operation of each register.

The serial line driver software initialises the SCC to a base mode of operation and then configures specific functions such as the clock rate, character length, even or odd parity, and the number of stop bits. When this has been completed the transmitter or receiver is enabled and data supplied or removed under interrupt control.

The following is a list of the registers. For details of the bit positions within each register, refer to the data sheet.

WR0 CRC initialise, initialisation commands for the various modes, and register pointer

WR1 Transmit/receive interrupt and data transfer mode definition

WR2 Interrupt vector (accessible through either channel)

WR3 Receive parameters and control

WR4 Transmit/receive miscellaneous parameters and modes

WR5 Transmit parameters and control

WR6 Sync characters or SDLC address field

WR7 Sync character or SDLC flag

WR8 Transmit buffer

WR9 Master interrupt control and reset (accessible through either channel)

WR10 Miscellaneous transmitter/receiver control bits

WR11 Clock mode control

WR12, WR13	Baud rate generator time constant
WR14	Miscellaneous control bits
WR15	External/status interrupt control
RRO	Transmit/receive buffer status and external status
RR1	Special receive condition status
RR2	Modified interrupt vector (channel B only) Unmodified interrupt vector (channel A only)
RR3	Interrupt pending bits (channel A only)
RR8	Receive buffer
RR10	Miscellaneous status
RR12, RR13	Baud rate generator time constant
RR15	External/status interrupt information

Programming the baud rate generator

Three baud rate generators are available; one per channel internal to the 28530 and one in IOC. The BAUD output of IOC is linked into the 28530 on the TRxC input pins to provide split rate operation.

The internal baud rate generator should be used whenever possible.

Should split rates be required, the IOC baud rate generator should be used.

If split rates with one above 9600 baud is required, the higher rate should be provided by the internal baud rate generator and the lower rate provided by IOC.

Should split rates with both above 9600 be required (although this is very rare), it is possible to program the baud rate generator in channel B to be output on the TRxCB pin and configure channel A to use this as a second source.

Control over the source of the baud rate for both transmit and receive is affected by programming write register 11 .

Control over the clock source for the internal baud rate generator is affected by programming write register 14. This should be programed to select the PCLK input which is driven by an external 3.6864 MHz crystal oscillator.

Data set ready

Data set ready (DSR) is not directly supported by the 28530, though it does provide an auxiliary input pin (SYNC) that can generate an interrupt on either edge. This design uses this input for DSR and the control bit in write register 11 has to be programmed accordingly.

Interrupts

The 28530 vectored interrupt scheme is not supported by the hardware. However, the interrupt vector modified by the status information register is accessible.

The SCC generates receive, transmit and external/status interrupts. In addition the sync pin *is* used to generate an additional external/status interrupt when DSR changes. The full list of external/status interrupt sources is: Zero Count, DCD, DSR, CTS,

Transmitter underrun/End of Message and Break/Abort. These can all be enabled/disabled in write register 15. The serial line controller interrupt signal to IOC is SLCI.

To support an external modem a ring indicate signal *is* provided which generates an interrupt in IOC directly. See the section *10C interrupt allocations* above.

Time constants

The SCC clock (PCLK) *is* 3.6864 MHz. From this can be calculated the baud rate generator time constant for a given baud rate according to the formulae in the table below.

$$\text{Time Constant} = \left\{ \frac{\text{Clock Frequency}}{2 \times \text{Baud Rate} \times \text{Clock Mode}} \right\} - 2$$

$$= \left\{ \frac{3.6864 \times 10E6}{2 \times \text{Baud Rate} \times 16} \right\} - 2$$

Baud Rate	Time Constant	
	Dec.	Hex.
75	1534	05FE
150	766	02EF
300	382	017E
600	190	00BE
1200	94	005E
2400	46	002E
4800	22	0016
9600	10	000A
19200	4	0004
38400	1	0001

Table 5.14 : Baud rate time constants

Circuit description

The Z8530 serial communications controller SCC is IC18. Only a few external components are required to implement a design based on this part.

The SCC is clocked (PCLK) by a crystal oscillator running at a multiple of the standard baud rates (3.6864 MHz). This oscillator is constructed around IC139 and crystal X3.

The pull-up resistor R41 is required to inhibit the SCC vectored interrupt acknowledge cycle mechanism.

The two gates in IC3 driving the SCC RD* and WR* inputs are required to implement the SCC reset scheme. Under normal operating conditions either RD* or WR* is active. When both RD* and WR* are active the SCC interprets this as a reset.

The SCC serial transmit data, receive data, and control signals are buffered to RS232 levels by IC1 a MAXIM 239. This device produces -12 V from the +12 V DC rail and the transmitter outputs are capable of producing output swings of ± 9 V: The receiver sections of this device convert the ± 5 V to ± 15 V RS232 signals to 5 V TTL/CMOS levels. Both the transmitters and receivers are inverting. The receiver outputs will be inactive (high) if the equipment to which they are connected is switched off or the signal becomes disconnected. This is otherwise known as off-line detection .

The ring indicate input is fed direct from a receiver to IOC.

The output connector is a 9 way D-type connector (J1), mounted on the rear back panel.

5.11 The floppy disc controller

The interface provides all logic and control circuitry required to format, read and write double density and quad density formats, using MFM modulation. The formatted disc capacities are approximately 800 kbytes and 1600 kbytes.

The interface supports two drives, though only one will normally be fitted. The specification for this drive is in Chapter 14, *Floppy disc drive*.

Write precompensation is not provided as it is not required by the high quality, high density 3.5 inch disc drive installed in this system.

The floppy disc drive interface circuit is based around the WD1793 or compatible controller. The data sheet for this device is listed in Appendix E.

Programming the floppy disc controller

The address of the floppy disc controller is defined in Table 5.2.

The controller supports ten basic commands (these are described in detail in the data sheet for the WD1793):

- restore (seek track 0)
- seek
- step-in
- step-out
- read sector
- write sector
- read address
- read track
- write track
- force interrupt.

Floppy disc control latch

In addition to the floppy disc controller, an output latch is used to control drive selection. The floppy disc control latch bits are defined and described below. The base addresses are defined in Table 5.2.

BD7	BD6	BD5	BD4	BD3	BD2	BD1	BD0
FDRST*	INUSE*	ON*/OFF	SIDE*	SIZE*	NOT USED	SEL1*	SEL0*

Table 5.15 : Floppy disc control latch

BIT [0:1] SEL[0:1]*
These bits select the floppy disc unit 0 and 1 when written low: Only one bit should be active (low) at any one time. The internal drive is drive 0. The select lines are used to enable or disable all disc functions except motor-on .

BIT [2] NOT USED
Not used.

BIT [3] SIZE*
This bit selects the media density. 0 = 2 Mbyte, 1 = 1 Mbyte

BIT [4] SIDE*
Side select. This controls the side/head select line of the floppy disc interface.
0 = Side 1 (upper), 1 = Side 0 (lower).

BIT [5] ON*/OFF
This bit controls the floppy disc motor line. When low the spindle motor will start to run.

BIT [6] INUSE*
In use: This bit controls the INUSE line of the floppy disc. When this line is low and the drive is selected, the IN-USE lamp will turn on. When this line is high or the drive is not selected, the lamp will turn off.

BIT [7] **FDCRST***
Floppy disc controller reset. This bit controls the MR* (master reset) pin on the 1793/9793 controller.

Note: This bit *is* not active during the power-up sequence so must set by the FDC driver before the controller can be used. The minimum reset period for this device is 50 µs.

Basic operation

Basic command execution is initiated by the processor. The spindle motor is started and allowed to come up to speed. The drive and head required is then selected in the control register and a command issued to the controller. If the command requires the transfer of data to or from the controller, as is the case for commands like Read Sector, then this is carried out under Fast Interrupt - FIQ control. The FIQ request mode of operation replaces the DMA mode more commonly used for servicing this type of transfer. When the command has been completed by the controller an IRQ interrupt is generated. See *IOC interrupt allocations* above for more details.

Circuit description

The floppy disc controller is IC119.

Very few external components are required to implement a design based on this part.

There are many sources for 1793 compatible controllers. The principal difference between them is their power requirements. As a guide, 9793 types are normally 5 V only and pin 40 is not connected, while 1793 types require 12 V on pin 40. This is a link option in this design.

Caution: Applying 12 V to 9793 types may cause damage.

The control register is IC130. The functions of the bits in this register are described above.

The clock for the FDC is either 1 MHz or 2 MHz corresponding to operation at data rates of 250 kHz or 500 kHz. This is achieved by setting the size bit in the control register: The 1OC CLK2 is divided by 2 by one half of IC140. IC110 controls the selection of the clock for the FDC:

The read strobe produced by IOC is too narrow to be used as the read strobe for the FDC. Half of IC140 is used to generate the correct width strobe from IOR*/W.

The WD92C32 data separator IC148, is a digital device that converts the READ DATA stream from the drive into a separate clock - SEPCLK , and data - SEPDATA streams. The WD92C32 operates from a 16 MHz clock produced by a divider circuit, shown in Appendix A.

The WD92C32 data separator operates on an internal clock that is nominally 32 times the SEPCLK frequency. The internal clock is generated from the reference clock by an internal programmable divider. The divisor in this design is either 1 or 2 according to the size bit in the control latch . If the size bit is a one, corresponding to 1 Mbyte media, then the divisor is 2 and SEPCLK is 250 kHz. Otherwise the size bit is a zero, corresponding to 2 Mbyte media, then the divisor is 1 and SEPCLK is 500 kHz.

The floppy disc interface signals are buffered by the open collector driver IC131. Pull-ups are provided by RN9.

The following is a description of local signals and their function. Where a signal is derived directly from the control latch this information is not repeated.

Name	Description
BBD[0:7]	Peripheral buffered data bus.
LA[2:3]	Latched address bus. These are used to select FDC internal registers.
S1*	Peripheral select. This is the IOC bank I select strobe.
FFDQ	Floppy fast data interrupt request. This is generated by the FDC when it is ready to transfer another byte during simulated DMA data transfers.
FFIQ	Floppy interrupt request. This is raised by the FDC to indicate that a command has been completed.
DIRIN*	Step direction. When the drive is selected, this indicates the direction that the heads will step when a STEP pulse is generated.
STEP*	When the drive is selected pulse on this signal will cause the heads to step in the direction indicated on the DIRIN* signal.
WRITE DATA*	If the drive is selected and the write gate signal is enabled, data on this signal will be written onto the disc.
WRITE GATE*	If the drive is selected, this signal enables the disc drive write current circuit and data on the WR1TE DATA* signal is written on to the disc.
INDEX*	If the drive is selected, a pulse (low) is generated on this signal for each revolution. A drive that does not support the ready signal can be supported by timing the interval between these pulses *as a test* of the motor speed.
TRACK 00*	This signal is asserted when the heads are over track zero.
WRITE PROT*	Write protect. This input is asserted by the drive if a write protected disc is inserted into the drive. This signal *is* also asserted if no disc is inserted, but a better test for this condition is the disc changed signal.

READY* This signal is asserted by the drive when the following conditions are met:

- the drive is selected
- a disc is inserted
- the motor-on signal is asserted
- the spindle motor is running.

READ DATA* This is the encoded serial data stream for the drive heads.

5.12 The clock/RAM

A real time clock (RTC), and 240 bytes of CMOS RAM are provided. The RTC can be operated in either 12 or 24 hour modes and has a built-in four year calendar. The RTC also provides an alarm facility capable of generating an interrupt.

Data is maintained in the CMOS RAM while the mains power to the machine is switched off by dry cells. They may be replaced while the machine is switched on without loss of data.

Programming the real time clock

The RTC can detect if both the mains and battery power have failed, and a flag is set. The counters and registers within the RTC are programmed and arranged as shown in the data sheet for the Mullard PCF 8583 RTC chip used in this design. (This is listed in Appendix E.)

The address and data used by the RTC and RAM are transferred serially via a two line

bidirectional bus (I^2C compatible). The protocol for this serial communications channel is described in the data sheet.

Circuit description

The RTC (IC54) is powered from one of two sources; either a two cell dry battery or, when the computer is switched on, the 5 V DC rail. Diode D7 protects the battery against being charged from the 5 V rail. Diode D8 prevents the battery from being discharged through the power supply when the power supply is off. The battery voltage may be monitored on test point TP3.

The RTC has an internal oscillator, requiring a crystal (X2) and trimming capacitor (C58). The oscillator frequency can be measured on test point TP4, using a low capacitance x10 probe.

The I^2C serial clock is controlled by IOC control bit C1.

The I^2C serial data is IOC control bit CO.

Chapter 6 SCSI

6.1 Introduction

This chapter describes the SCS1 interface used in the Acorn Technical Publishing System. This interface has a large cache buffer memory which is used by the host computer to store and retrieve data to be passed to the peripheral devices. Access to this buffer is shared with an LSI SCSI controller. The controller used in the Acorn Technical Publishing System design is the Western Digital WD33C93A SB1C.

Most of the logic within the interface is contained in a number of programmable array logic devices (PALs) and this logic is NOT described in detail.

At the end of this chapter are three sections detailing the following:

- the memory map of the SCSI interface
- an alphabetical list of all the control signals within the interface and between the interface and ARM, together with a brief summary of their function
- a summary of states within each of the major state machines.

Additional documents will be required. These are listed in Appendices E and H, but principally the data on the SCSI controller chip (Western Digital WD33C93A - SBIC) and the ANSI SCS1 standard will be required, together with data on the specific peripherals to be interfaced.

6.2 SCSI circuit elements

All of the SCS1 electronics are contained in the schematic diagrams in Appendix A. To ease understanding it may be split into the following functional blocks:

- High speed, 32 bit bi-directional latched interface. This comprises IC37, 1C45, 1C56 and 1C65, and RN11-14.
- 256 kbyte (64 k ARM word) data buffer/cache. Implemented with 64 k x 4 dynamic RAM. May be accessed by either ARM or the SB1C in its data transfer phase.
- 32 bit to 8 bit bi-directional interface. This is the data funnel that provides the correct organisation for ARM data to/from the SBIC.
- Western Digital SCS1 interface chip WD33C93A (SB1C) together with SCSI bus connectors, pullups and filters. This is the port to the SCSI bus. All SCS1 signals apart from RST* are generated by this device. RST* comes from the control port.
- Control port. Provides software control functions of reset.
- Address generation and multiplexing. When the SB1C is accessing the DRAM buffer it cannot generate the data addresses directly, so a 16 bit preloadable counter is provided. This must be set under software control before the transfer may start. When ARM accesses the buffer the data address is taken from the latched address bus. There are two multiplexers which are used to select between latched address bus and the preloadable counter. These multiplexers also select row and column addresses as required by the DRAM buffer.
- Control logic and refresh generation. A number of programmable devices are used to contain the complex logic needed to control accesses and arbitrate between ARM and SB1C.

6:3 Summary of access modes

The following different types of access are supported. Each access type is summarised here, but a more detailed description follows in section 6.12, *Detailed description of circuit operation.*

ARM <==> DRAM buffer

ARM may read and write data in ARM words or in bytes to and from the buffer. If when ARM starts an access, the buffer is currently owned by the SB1C, then a sequence of logical events is initiated to force the SB1C to relinquish control of the buffer.

ARM <==> SBIC

It is necessary for the ARM to access the SB1C in order to issue commands and monitor status. There are limitations as to when the SBIC status may be monitored. These are:

- Read SB1C status
- Read SCS1 bus status
- Program SBIC control registers
- Initiate SCSI bus transactions

Additionally, it is possible to read and write data in polled mode directly to and from the SB1C.

SBIC <==> DRAM buffer

When the SB1C enters a data transfer phase and wishes to send/receive data to/from the peripheral device(s) it normally does so in direct buffer access mode, in which it takes control of the DRAM buffer and generates the appropriate read or write strobes.

SBIC => Control Port

The SB1C may write control and reset values to the control port. This will pre-empt any current DBA operation, but normally will not happen during a DBA transfer.

SBIC => Address Latch/Counter

The SBIC may write the start address within the DRAM buffer of the next DBA data transfer.

Block diagram

A block diagram is shown below:

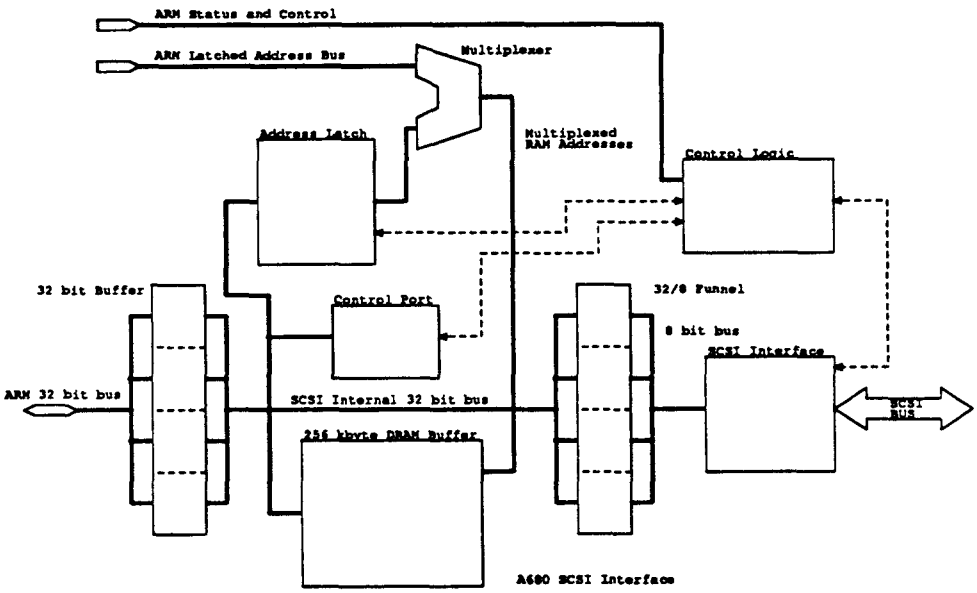

Figure 6.1 : SCSI block diagram

The 256 kbyte buffer may be accessed by either ARM or by the SBIC when it is in the direct buffer access mode (DBA). It is organised as 64 k words where each word is 32 bits. Data to and from the SBIC is only eight bits, and this transformation is performed by the four buffers, collectively known as the funnel. These four buffers actually implement a four to one bi-directional multiplexer/demultiplexer.

When the SB1C is performing a DBA access, the address of the RAM accessed is found from the 16 bit address counter/latch. The action of this is described in more detail later in this chapter.

The control of the multiplexer is automatic and transparent to the programmer. When ARM begins an access cycle to the DRAM, the current state of the SBIC may or may not allow an immediate access. If the SBIC is accessing the buffer, then its action will be suspended and ARM will be granted access.

When ARM access to the buffer is allowed, the multiplexer will change to allow the ARM latched address bus to select the required ward in the buffer. When DBA cycles are started the multiplexer selects the address latch/counter as the source of the SCS1 DRAM buffer addresses.

The initial value in the address latch/counter is set as the lower 16 bits of an ARM access to a specified location. Each time the address latch/counter is written it also resets the 2 bit counter in the funnel which performs the byte to ARM word translation. This means that any DBA accesses must start on an ARM word boundary. (Note: this may be changed in future hardware.)

The address latch/counter is 16 bits for the available 64 k words of the buffer. To arrange that accesses from ARM correspond to this, the latched address lines which are used by the address selector are LA2 to LA17. Byte accesses are further decoded by LA0 and LA1.

6.4 Accessing the SBIC

The SB1C is addressed at two different address values (actually it appears in the memory map as a block of address pairs, but it is strongly advised that the lowest two addresses are the only pair used). The INDIRECT addressing mode is used.

The organisation is as follows:

LA2	Register Accessed by WRITE	Register Accessed by READ
0	Register File Address	Auxiliary Status
1	Selected Register File	Selected Register File

Most of the registers in the SB1C are organised as a register file. The particular register in the register file is selected by setting the register file address register. This is done by writing the offset (0 - 0x18) of the required element of the register file into the register file address register. The address of the register file address register is at the appropriate memory block to select the SB1C with LA2 = 0.

If the same location is read it will return the auxiliary status register.

There is also a method of auto-increment addressing. For further details see the WD33C93 A data sheet, listed in Appendix E.

6:5 Interrupts

The Acorn Technical Publishing System SCSI interface generates one interrupt. This is derived directly from the 1NT pin of the SB1C, inverted, and passed to the 1L3* pin of IOC. This means that the SB1C interrupt is level sensitive, inverted, and of IRQ type. The interrupt must be cleared by reading the SBIC status register (17 in the register file) before any more commands may be issued. The interrupt status is also indicated in bit 7 of the auxiliary status register. If this bit is set and a command is issued, the last command ignored flag is set.

6:6 Reset control port

The SCSI reset control port is used to reset either the SBIC, the SCSI bus, or both. There is a facility to program the control port such that the SBIC may be automatically reset by a SCSI bus reset, or independently as required. The port is organised as a 7 bit latch which is addressable as detailed in Figure 6.3, the memory map.

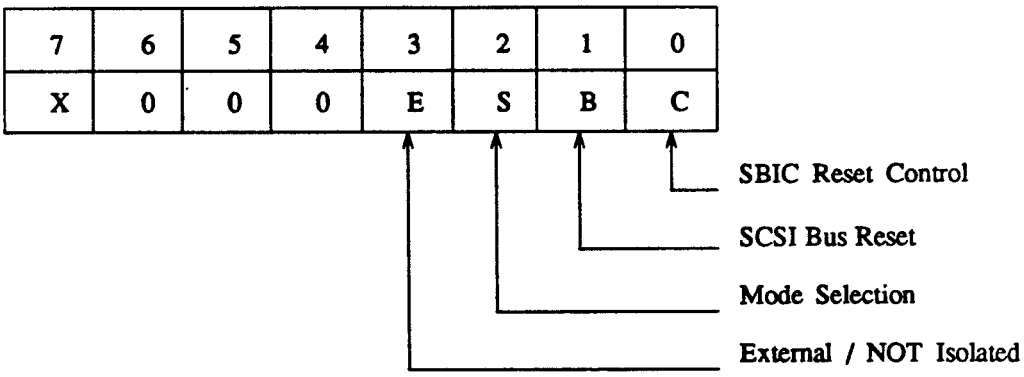

Figure 6.2 : SCSI control port bits

Mode selection ('S' bit)
If this bit is set to 0 then the SBIC reset ('C' bit) and SCSI bus ('B' bit) reset inputs may be used to reset the SBIC or SCSI bus respectively.

If this bit is set to 1 the 'B' and 'C' bits are ignored. The external reset mode signal may be selected.

External/NOT isolated ('E' bit)
If this bit is written as 1 when the 'S' bit is written as a 1, then if the SCSI bus reset line (RST*) becomes active the SBIC chip will be reset at the same time. If this bit is written as a 0 when the 'S' bit is written as a 1, a reset on the SCSI bus will not reset the SBIC chip.

SBIC reset control ('C' bit)
If the 'S' bit is 0, then a 1 to this bit will generate an active reset signal to the SBIC. It is latched, and so must be written as 1 then 0 to provide the appropriate reset pulse. It should be noted that the SCSI specification has a minimum requirement for the width of the reset pulse.

SCSI bus reset ('B' bit)
If the 'S' bit is 0, then a 1 to this bit will generate an active reset signal to the SBIC. It is latched, and so must be written as 1 then 0 to provide the appropriate reset pulse.

Examples of use of the control port

To reset SBIC but not reset SCSI bus:

Write: 00000001 to control port
Write: 00000000 to control port

To reset SCSI and not SBIC:

Write: 00000010 to control port
Write: 00000000 to control port

To reset both SCSI and SBIC simultaneously:

Write: 00000011 to control port
Write: 00000000 to control port

To enable SRESIN to reset SBIC:

Write: 00001100 to control port

To disable SRESIN from resetting SBIC:

Write: 00000100 to control port

Note: Whilst the 'E' bit is active neither SRESIN or the 'C' bit being active will generate a reset for the SBIC. Also, when the 'E' bit is active the 'B' bit will not reset the SCSI bus. The 'E' bit is NOT latched, and so is active only for the duration of the control signal.

The SCSI specification requires a reset width of 25 µs.

6:7 The address latch/counter

This is a 16 bit synchronous presettable counter. Its purpose is to generate the address to (from) which data is written (read) in DBA mode. This only applies to the data transfer phase. The value written to this register corresponds to the ARM word offset from the base of the SCSI RAM buffer at which the transfer should start. A point to note is that it is the WORD offset and so the byte offset must be divided by 4 when this register is loaded.

As previously noted, accessing this register resets the data funnel to point to the first byte of the word.

The address latch/counter is incremented after every four bytes have been transferred under DBA. It is practically impossible for a transfer to involve a non-word multiple of bytes, but if it is required to make non-word transfers in the data transfer phase, use must be made of the single byte transfer mode of the SBIC. Refer to the SBIC manual for more details.

6:8 Data alignment

ARM data is aligned such that for a given ARM word the data byte corresponding to data lines D0 to D7 will be addressed at the lowest numeric byte address for that word. This convention is observed in transfers in and out of the funnel. The implication behind this is that as words are converted to bytes by the funnel and transferred to the SBIC this is in sequentially ascending address order.

This must also be the alignment observed when loading or unloading the SCSI buffer by ARM.

6:9 Arbitration and preferred transfer mechanisms

The state of the buffer memory falls into one of three categories:

- Idle - SBIC is not transferring data and no refresh is occurring.
- Refresh of dynamic RAM is occurring. This lasts for one clock tick (S cycle) and any ARM access will be delayed by this amount.
- SBIC is transferring data by DBA and thus access to the buffer is delayed until the SBIC has been throttled off.

If the SCSI interface is in the first category, then an ARM access will occur in 1N + 1S cycle. If the SCSI interface is in the second category then an additional 1S cycle will be inserted. If the SCSI interface is in the third category then a number of additional S cycles will be inserted until the logic of the SBIC has released the buffer access to ARM. The exact number of S cycles varies, but is typically 5 to 6. It should be noted that adjacent accesses by ARM will override any attempt to move to the second or third categories:

As a consequence it is advised that data transfers to/from ARM to SCSI buffer are performed with LDM and STM instructions, as this will continuously assert the selection signal and mean that only one SBIC arbitration will occur for a given number of words of data.

6.10 Device IDs

The following represents the preferred allocation of ID's to devices. All internal hard discs will have an ID of 0, and all SCSI host interfaces will be programmed to have an ID of 7.

ID	Type of device
0	Acorn Technical Publishing System Internal Hard Disc Drive
1	External Hard Disc Drive #1
2	External Hard Disc Drive #2
3	External Hard Disc Drive #3
4	External Hard Disc Drive #4 / Streamer #2
5	Streamer #1
6	Processor / Printer Device
7	Host (Acorn Technical Publishing System)

Table 6.1 : Device IDs

6.11 Supporting documentation

1. Western Digital WD33C93 SBIC Data Sheet
2. Western Digital SBIC Applications Notes
3. ANSI Small Computer System Interface X3:131 - 1986.

6:12 Detailed description of circuit operation

Main state machine (IC 20)

The interface is controlled by a synchronous state machine called the main state machine. It is implemented as a PAL (20R6) clocked by the positive going edge of the ARM REF8M clock. It is the principle mechanism for arbitration.

It has seven states:

- Idle
- Access
- SBIC strobe
- DBA active
- Throttle
- DBA refresh pending
- DBA refresh occurring.

Its inputs are:

IORQ*	ARM I/O request
BPHI2	Buffered PHI2 clock from the master MEMC
LR*/W	ARM latched write strobe (active high for write)
SCBSEL*	Early decode of buffer select by ARM
SCCSEL*	Early decode of control select by ARM (ie SBIC, control port, address latch)
LA9,10	Latched addresses (not used in this PAL)
DBAGT	DBA mode is in progress if TRUE
DBARQ	DBA mode is being requested if TRUE
RST*	System reset signal
REFREQ	Refresh request
REF8M	8 MHz system timing clock

Its outputs are:

DRQ	DBA data request signal to SBIC. When TRUE the main state machine indicates that the SBIC may perform a DBA transfer.
ABRL	ARM buffer read latch. This strobe signal is used to latch data from ARM into the ARM interface latches.
FSM2-0	Finite state machine outputs.
DOREF*	Refresh control strobe. When TRUE indicates that a refresh may occur. Active low.
IOGT	ARM I/O grant signal. Active high from the PAL because it is inverted by the open collector buffer (part of IC97). See the MEMC data sheet for description of timings.

Here is a description of each state.

Idle state
This is the condition after system reset, when no access by ARM is current, and no DBA transfer is in progress. A DRAM refresh will not induce a change of state. The need for a refresh is indicated by the REFREQ input becoming active. A refresh will be allowed unless an access is about to start, which is indicated by SCBSEL or SCCSEL being active.

The main state machine will leave its idle state under the following conditions:

- A valid access by ARM to the DRAM buffer
- A valid access by ARM to the control port, SBIC or address latch
- A DBA request from the SBIC. (Assertion of DBARQ).

For the first two conditions, the definition of valid is that IOGT should NOT be active (ie a cycle has not already started) and that IORQ should be active. It is not sufficient to simply use SCBSEL or SCCSEL as these may be active during a MEMC controlled DMA cycle for VIDC. For a further understanding of this, consult the ARM chip set documentation (Bibliography, references 1, 2, 3 and 4).

In the case of the first two conditions above the main state machine will move to the access state. If DBARQ is active, then the main state machine will move to the DBA active state, and indicate this to the SBIC by asserting DRQ.

Access state
This state is entered for either a DRAM buffer access or a control access. DRAM buffer accesses are only one REF8M clock cycle long, so if SCCSEL is NOT active, then the state following the access state will be idle if DBARQ is not true or the DBA active state if it is. If SCCSEL is active, then the next state will be the SBIC strobe state.

SBIC strobe state
This state is used to ensure that when reading or writing to the SBIC pulse widths are within specification for the SBIC. It should be noted that the binary values used in the state machine to represent the access state and the SBIC strobe state are separated by a unit distance (grey) code, in order to avoid glitches on strobe signals which are common to the two states.

SBIC strobe always returns to the idle state. This is to allow 125 ns between a SBIC access and the possible resumption of DBA. This ensures that precharge times are satisfied on the DRAMs.

DBA active state
This is the state during which DBA transfers are allowed. It *is* maintained until one of two sets of conditions arise.

Firstly, if an SCBSEL, SCCSEL or REFREQ signal become active it is necessary for the DBA transfer to be temporarily suspended. This is known as throttling the DBA. The main state machine will then move to the throttle state:

The second exit condition occurs when **DBARQ** becomes inactive: This indicates that the DBA transfer is terminated *because the* SCSI bus state has moved from the data transfer phase. There are a number of reasons for this and it is discussed in more detail in the description of DBA. For the main state machine it will terminate the DBA active state and move the state machine to the idle state.

Throttle state
This state is entered from the DBA active state. It is maintained until DBAGT goes inactive, which shows that the SBIC has relinquished control of the DRAM buffer.

The state assumed after the throttle state is dependent upon REFREQ. If REFREQ is active, the next state will be the DBA refresh pending state. Otherwise it will be the access state, to allow a SBIC, buffer or similar access to occur.

DBA refresh pending state
This state is included to ensure that there is a 125 ns clock cycle between the preceding DBA state and the refresh state. It is decoded as part of the logic for the DOREF* signal. The next state is always DBA refresh.

DBA refresh state
This is the state during which refresh occurs if the REFREQ occurred in the DBA mode. After this state the state machine returns to the idle state. As DBARQ is probably pending, this will cause the restarting of the DBA mode. See the idle state for more details.

The select PAL and the MSM decode PAL (IC25 and IC22)

These are two combinatorial PALs fed with the finite state machine outputs from the main state machine PAL. Using the current values of the main state machine and other associated control signals they generate a number of miscellaneous strobes and signals.

Select PAL signals (IC25)

CSSBIC* - active low SBIC chip select
This is a combinatorial decode of the SCCSEL* signal and the appropriate values of LA9 and LA10. It is used to select the SBIC for control and status accesses.

ACSTB* - address counter strobe (pin 19)
A write to the address counter will make this signal active. It is conditioned by the main state machine being in the SBIC strobe state. This signal is used to generate a clock signal for the address latches.

CSCON* - control port chip select
This is decoded from the LA9, LA10, SCCSEL and LR*/W signals. It is also conditioned by the main state machine being in the access state. This means that data will be latched into the control port PAL on the rising edge of this signal (half way through the access cycle) and thus the data will be valid at this point in the cycle.

ACLD* - address counter load signal
When ACLD* is active, clocking the address counters will cause them to take the data at their inputs and transfer it to the output registers as opposed to counting. This allows the transfer address to be set. The ACLD* signal is active throughout the SBIC strobe and the access state. It is also conditioned by SCCSEL*, LR*/W, LA9 and LA10.

ABEN* - ARM buffer enable (active low)
This signal is used to control the enable of the ARM interface buffer. It is driven from either SCCSEL* or SCBSEL* being active: DBAGT must be inactive. For reads it is also conditioned by IORQ being active.

ABWL* - ARM buffer write latch
This signal is used to select either latched data or *direct data* from the ARM interface buffers. These are 74ACT646 type devices, and ABWL* drives both the mode select signal and also the clock signal. It is an asynchronously timed signal and is driven off the transitions of IORQ, PHI2, SCBSEL and SCCSEL.

The first negative going transition of IORQ after the beginning of the cycle makes ABWL* latch the data. The beginning of the cycle is determined from the state of PHI2, which goes active at the end of the preceding cycle. ABWL is cleared by PHI2 going inactive.

MSM decode PAL signals (IC22)

ARMCYC* - buffer access strobe (active low)
This strobe is generated in the access state when SCBSEL is active. It is used to indicate to the DRAM control PAL (IC17) that an *access* to the DRAM should now take place.

SDIR - funnel direction control
When SDIR = 1 data is being read from the DRAM buffer to the SBIC. When SDIR = 0, data is being written into the DRAM buffer by the SBIC. When in the idle state, the access state or the SBIC strobe state, SDIR is controlled by LR*/W. In any other state it is controlled by WESBIC* which is the SBIC write data strobe. WESBIC* is active when the SBIC is writing to the DRAM buffer and so then SDIR will be low.

RAMWE* - RAM write enable
When in the access state, RAMWE* is the inverse of LR*/W. Otherwise RAMWE* is controlled by WESBIC*.

CONCYC* - control strobe (active low)
This is true when SCCSEL is active and the main state machine is in the access or SBIC strobe states. It has two purposes:

- In the funnel PAL it is used to condition ACLD to generate the address latch clock ACCK.
- In the DBA state machine (IC15) it is used to condition the WESBIC* and RESBIC* strobes.

The funnel PAL (IC45) and 32 bit to 8 bit conversion

This PAL has a number of functions:

- It is a two bit counter used to select which byte is read/written by the SBIC in DBA mode.
- It generates a clock for the address counters in DBA mode. It also clocks the address counters when they are written by ARM to load the transfer address.
- It controls the four buffers used to implement the funnel. Only one buffer may be enabled at a time:

The two bit counter
This has to be an asynchronous counter. This is because the width of the data strobes generated in DBA mode is variable and not necessarily synchronised to any particular clock. As such it is actually an eight state asynchronous state machine. The counter is incremented on every high to low transition of DLY50 and is cleared every time ACLD is active when in the DBA mode.

When not in the DBA mode (ie DBAGT is inactive) the current count state is held, unless the address counter *is* accessed.

The DLY50 signal is used to increment the counter because its last transition represents the end of the cycle, and as such means that the funnel will clock on to the next byte at the very end of each transfer.

Two of the three state bits are used to indicate to the DRAM PAL (IC17) which byte is currently the selected byte for a DBA transfer. These are signals CO1 and CO2 on the schematic.

Generation of the ACCK signal
In DBA mode as the funnel counter state machine passes through its last (8th) state it generates the ACCK signal. This clocks the address counter.

When not in the DBA mode, the ACCK signal is generated when CONCYC, ACLD and ACSTB are all active. This corresponds to an access to the address counter by ARM.

Output buffer control for the funnel
There are four control signals to select the appropriate ACT245. They are known as SE*(0:3). The count signals CO2 and CO1 are decoded to select which byte is being transferred. The truth table for the logic is as follows (in DBA mode):

CO2	CO1		SE*0	SE*1	SE*2	SE*3
0	0		1	0	0	0
0	1		0	1	0	0
1	1		0	0	1	0
1	0		0	0	0	1

(1 = ACTIVE)

Table 6.2 : Output buffer control for the funnel - truth table

Additionally, all the select signals are conditioned by DBACYC being active.

It should also be noted then when not in DBA mode (DBAGT inactive) then if CSSBIC* is active SE*0 is made active.

RAM control PAL (IC17)

This is a combinatorial PAL (20L8) which *is* fed with a number of signals which are used to generate the RAS* and CAS* signals for the DRAM.

Accesses in non-DBA mode
In the non-DBA mode, transfers are byte or word-wide. A transfer occurs when the ARMCYC* signal becomes *active.* This is a 125 ns pulse generated by the main state machine going into the access state. This signal is used to generate the RASO* and RAS1* signals. It is also passed directly to the RAMCYC output and the delay line IC21 then delays this by steps of 10 ns. Two of these steps - the 40 ns tap and the 50 ns tap - are fed back to IC17 as DLY40 and DLY50 respectively. When ARM is writing into the buffer, the DLY50 tap is used to generate the CASxx* signals, otherwise for ARM reads the DLY40 tap is used.

Both RAS signals (RAS0* and RAS1*) are identical and all four CAS signals (CAS0L*, CAS0H*, CASIL* and CAS1H*) are identical when accesses in non-DBA mode are taking place.

To allow ARM to write the selected byte within the buffer, the PAL IC23 decodes the selected byte. This PAL has the following inputs and outputs:

Inputs
LA0 and LA1 from IC57 (the address latch)
LB*/W - latched Not Byte/Word
DBAGT- DBA GranT
LR*/W - latched Not Read/Write
RAMWE* - RAM Write Enable Strobe

Outputs
WE(0..3)* - Four byte write enables
RAMOE* - RAM output enable

In DBA mode or non-DBA mode with LB*/W high (ie, a word access) RAMOE* is always active and each WE(0..3)* signal is driven by RAMWE*:

Otherwise (ie, a byte access is initiated by ARM) if LR*/W is high indicating that there will be a write cycle RAMOE* is driven inactive and one of WE(0..3)* is selected by LA0 and LA1. RAMOE* has to be driven inactive because with the appropriate RAS and CAS active but WE(n)* inactive, the unselected DRAMs will execute read cycles and contend with the bus.

If LR*/W is low (ie, a read) RAMOE* is active and all of WE(0..3)* are inactive in non-DBA mode.

The row/column address selection is always driven from a 20 ns tap on IC21. This is passed to the two address multiplexers, IC66 and IC58.

Accesses in DBA mode
When the DBAGT signal is active it is the DBACYC signal which is used to time reads and writes by the SBIC from/to the DRAM buffer. The DBACYC signal is derived from the WESBIC* and RESBIC* signals. These are combined with the ACT00 NAND gate IC7:

RESBIC*	WESBIC*	DBACYC	
0	0	1	should never occur
0	1	1	a SBIC read of data
1	0	1	a SBIC write of data
1	1	0	SBIC inactive

Note: '0' and '1' refer to TTL Logic low and TTL logic high respectively. RESBIC* and WESBIC* are ACTIVE when at logic low.

The counter inputs CO1 and CO2 are used to select the appropriate RAS/CAS pair to read/write the appropriate byte of data. The truth table for this selection is as follows:

CO2	CO1	RAS0	RAS I	CAS OL	CAS0H	CAS 1L	CAS1H
0	0	DBACYC	0	DLY50	0	0	0
0	1	DBACYC	0	0	DLY50	0	0
1	1	0	DBACYC	0	0	DLY50	0
1	0	0	DBACYC	0	0	0	DLY50

Table 6.3 : Accesses in DBA mode - truth table

The RAS-CAS delay is constant at 50 ns for DBA reads and writes.

Refresh in non-DBA mode
In both modes refreshes are by the automatic (CAS before RAS) refresh method. In the non-DBA mode the refresh cycle is generated by either REFQ0 or REFQ1 being active but the actual strobe of data is the DOREF* signal from the main state machine.

Both banks of DRAM are refreshed simultaneously. To comply with the CHR parameter of the dynamic RAM it is necessary to stretch the CAS pulse by making it the combination of both the DOREF* signal and the DLY40 signal. Once again, all four CAS signals are the same and both RAS signals are the same.

The REFCYC signal is generated from the same combination of DOREF* and DLY40 as is used for the CASs. This signal is passed to the refresh control state machine IC27 to indicate that the refresh has occurred.

Refreshes in DBA mode
Whilst in DBA mode, the dynamic RAM is owned by the SBIC. Two methods to control refresh are provided:

- Transparent refresh which occurs during SCSI bus data transfers.
- Non-DBA mode word refresh which occurs if the SBIC is in DBA mode but no data transfers are occurring.

This approach provides very high performance in DBA mode as refresh requests do not interfere with SCSI bus transfers.

DRAM buffer

The DRAM buffer is implemented as eight 64 k x 4 dynamic RAM devices. The early write mode of operation is employed, which means that the DRAM output enable pins (pin 1) may be permanently tied low. The organisation of the DRAM is as 64 k 32 bit words.

The RAS, CAS and WE* signals are all fed via series terminator resistors to avoid overshoot. The values of these *series* terminators vary to allow for the different loading of each line.

Refresh clock (IC19) and refresh control (IC27)

The refresh clock is produced with a standard 16R8 type PAL. This is programmed as a seven bit synchronous counter which gives a squarewave of 16 µs period when fed with an 8 MHz clock. The 8 MHz clock used is REF8M.

The refresh control is more complex. This has to be an asynchronous state machine as there is no coherent clock to generate the timing for this PAL:

The refresh state machine has two possible modes of operation; transparent mode and non-DBA mode.

When DBAGT is not asserted, all refreshes are in non-DBA mode. The refresh cycle may occur when the main state machine is in the idle state and generating the DOREF* strobe (ie, no pending access by ARM).

When DBAGT is asserted, the refresh control attempts to synchronise refresh cycles with access cycles. This is possible because only a byte is accessed at a time and so the DRAM buffer has been split into two banks (Bank0 is the even half ward and Bank1 is the odd half ward).

If, however, access cycles are not occurring even though DBA mode is selected (for example during a seek) then the refresh state machine detects this condition, throttles the SBIC and performs a non-DBA mode refresh.

There are six possible states in the refresh state machine:

DONE1
PEND0
DORF0
DONE0
PEND1
DORF1

These states are controlled by the four inputs:

REFCLK	16 Its refresh clock squarewave
REFCYC	refresh cycle from DRAM control (IC17)
DBAGT	DBA mode in progress
DLY50	DRAM access timing

The principle of operation is as follows:

Non-DBA mode
All four bytes of RAM are refreshed simultaneously. The cycle is initiated by REFCLK going active. The refresh state machine would be in the DONE0 state and moves to the PEND1 state which generates a refresh request to the main state machine via the REFREQ signal.

The first occurrence of the DOREF signal after REFREQ is asserted generates the RFQ0 and RFQ1 strobes which actually cause the DRAM control logic to generate the refresh cycle: This is indicated by REFCYC going active: This moves the refresh state machine to the DORF1 state. This is terminated by DLY50 going inactive which indicates that the cycle is complete.

The refresh state machine will now move to the DONE1 state. This state is maintained until REFCYC goes low (ie, 8 s later). At this point if DBAGT is inactive, the refresh state machine moves directly to the DONE0 state in preparation for the next refresh cycle.

DBA mode

Initially, the refresh state machine will attempt to synchronise a refresh cycle on one bank to an access cycle on the other bank. This is achieved by modifying the logic such that RFQ0 and RFQ1 are conditioned by REFCLK. The state machine now goes through two additional states, PEND0 and DORF0.

This allows the two banks to be refreshed separately, with the refresh cycle being requested to the DRAM control PAL (IC17) byte one of REFQO or REFQ1 going active.

The positive going edge of REFCLK initiates the Bank0 refresh and the negative going edge of REFCLK initiates the Bank1 refresh.

As before, the refresh state machine moves from the PENDx state to the DONEx state when the refresh cycle actually occurs (REFCYC or DLY50 active). The refresh cycle now occurs when the OTHER bank is being accessed.

It should be noted that each bank is still refreshed every 16 µs but that the refreshes are interleaved at 8 s intervals.

A further complication exists in that at any time, the data transfer on the SCSI bus may be suspended. This would cause the DRAM refresh to fail unless appropriate steps were taken.

If, whilst DBA is active, the refresh state machine has stayed in its PENDx state (either 0 or 1 depending upon REFCLK) until the next transition of REFCLK because there has been no access, then REFREQ is asserted. This will cause the main state machine to drive the DBA state machine out of DBA. Then a normal non-DBA type refresh may occur.

It should be noted that for transparent refresh to be effective there should be an access

cycle on the SCSI bus less than every 2 µs. This is to ensure that refresh cycles are constantly maintained without requiring the system to move to non-DBA refresh.

Control port

The control port is a PAL. It has seven inputs from the lowest seven bits of the data bus. The eighth input of the PAL comes from the reset line of the SCSI bus. Initially, this PAL controls the system resets. It is possible to reset the SBIC and also to reset the SCSI bus independently. It is also possible to program the control port such that if there is an external reset on the SCSI bus, the SBIC is reset. This only would be used where there are two Acorn Technical Publishing Systems communicating on the SCSI bus.

For further details of how to program the control port, see section 6.6, *Reset control port.*

Two other signals may be controlled by this port. These are BS0 and BS1. These two signals are intended to select a starting byte for the funnel (IC31). This will allow transfers to start with byte alignment. It is not currently implemented.

Address latch/counter

This is a pre-loadable 16 bit synchronous up counter. It is implemented with two PALs. The programming for each PAL is identical. There are eight data load bits, and eight count outputs. The PALs also have an active low COUNT_ENABLE* input and an active low CARRY* output

When COUNT_ENABLE* is active the counter will count with a binary sequence every time the ACCK signal goes positive. This is assuming that the ACLD signal is high. When all the count registers are '1' then CARRY* is asserted to show that a carry out is required.

COUNT_ENABLE* of the least significant byte of the counter is always tied low. This means that this counter will always count. The CARRY* of this device is connected to the COUNT_ENABLE* of the second byte of the counter. Thus this will only increment on every 256th transition of the common clock signal (ACCK).

The output of these PALs is permanently enabled.

When ACLD* goes low, this indicates that the counter is to be reloaded. Under these circumstances the data at the data inputs is transferred to the count registers on the positive going edge of ACCK. This is synchronous pre-load.

Address multiplexers (IC66 and IC58)

The selection between latched ARM address and DBA mode address counter output is controlled with DBAGT. When DBAGT is asserted the multiplexers use the address counter values, and when DBAGT is not asserted the multiplexers use the latched ARM address values.

The address multiplexers also select between row and column addresses. This selection is performed with the SELRNC signal which is an output from the delay line IC21.

Each address multiplexer has the following inputs:

8 ARM latched addresses	LA(2:9) or LA(10:17)
row/column selection	SELRNC
ARM/counter selection	DBAGT

and outputs:

MUXA(0:3) or MUXA(4:7)

DBA state machine and SBIC

The SBIC is an advanced LSI device which provides all the electrical interfaces and timings for the SCSI bus. In the Acorn Technical Publishing System design the WD33C93 device is used which has single-ended open collector SCSI bus drivers.

SBIC interface modes
The SBIC may function in three modes:

- processor slave interface
- DMA
- DBA (direct buffer access).

In the Acorn Technical Publishing System only two of these modes are used. These are:

Processor slave interface
This is the standard read/write I/O access. The SBIC has two addressing modes, direct and indirect. In the Acorn Technical Publishing System the indirect mode is used. The host processor can access a register, the register must first be selected by writing its address to the special address register in the SBIC. For some registers in the SBIC there is an auto-increment addressing mode.

The processor slave interface mode is used for all command and status operations with the SBIC: It is a very powerful device and is capable of combining several SCSI bus operations into one command.

Direct Buffer Access (DBA)
In the DBA mode the SBIC takes over the host RAM buffer. To do this a number of control signals which are normally inputs in the processor interface mode become outputs. These signals are:

RE*	SBIC Read Data strobe
WE*	SBIC Write Data strobe
RCS*	RAM Chip Select

Also the DRQ signal becomes an input

DRQ - Data Request
The transition from inputs to outputs is controlled by the SBIC itself. The exact mechanism is rather complex and is further complicated by the requirement to suspend the DBA mode in the middle of a data transfer. This allows ARM to access the DRAM buffer memory and accommodate a refresh cycle if required. To control the DBA mode requires a separate state machine. In the Acorn Technical Publishing System design this is the DBA state machine, IC15.

In the DBA mode, RCS* is asserted to indicate to the host processor that the SBIC expects control of the buffer memory. Whilst both DRQ and RCS* are active the SBIC will generate the RESBIC* and WESBIC* when it wishes to transfer data in or out of the buffer memory.

Description of DBA mode and DBA state machine
The DBA state machine has the following states:

No DBA
DBA pending
DBA active
Throttle
Kickstart,

and the following inputs:

RCS*	SBIC Ram Chip Select signal
DRQS	System Data Request (synchronised)
LR*/W	Read/Write control CONCYC* Control Cycle Strobe
MRSBIC	SBIC Reset Strobe,

and the following outputs:

WESBIC*	SBIC Write strobe
RESBIC*	SBIC Read strobe
DBAGT	DBA Grant signal
DBARQ	DBA Request signal
DBASM(2:0)	DBA Finite State Machine registers (these are not connected but are enabled).

The DBA state machine is a synchronous state machine using the same clock as the SBIC.

States are as follows:

No DBA
This is the idle state of the DBA state machine. Whilst in this state there is no current DBA transfer and a standard processor slave interface may be used to read/write data to the SBIC. The CONCYC* signal and the LR*/W signal are used to generate the RESBIC* and WESBIC* signals. DBAGT and DBARQ are FALSE.

If a DBA transfer is to start, the SBIC will assert RCS*: It should be noted that RCS* *is* actually high-impedance in the processor slave interface and thus is pulled high with R132. The assertion of RCS* moves the DBA state machine into the DBA pending state. For the WD33C93A, DRQ must be previously asserted. The main *state machine* asserts DRQ in any idle state.

DBA pending
Once RCS* has been asserted the DBA transfer cannot actually start until DRQ is asserted by the main state machine to signal to the SBIC that the host buffer memory is free. When DRQ *is* asserted the DBA state machine moves to the DBA active state. DBARQ is asserted but DBAGT *is* not asserted.

DBA active
Whilst this state is maintained (RCS* and DRQ active) DBA transfers may occur. The transfers are controlled by either RE* or WE* from the SBIC driving RESBIC* or WESBIC* respectively. These signals are passed to the RAM Control PAL, IC17 to generate all the appropriate memory signals.

DBAGT and DBARQ are both asserted. Note carefully that DRQ is now the handshake leader for suspending DBA transfers - throttle mode.

If RCS* goes inactive but DRQ stays active it indicates that the SBIC has transferred all the bytes specified by the transfer count registers or a bus error has arisen. In either case, the SBIC will have moved out of the DBA transfer state. This moves the DBA state machine into the No DBA state.

If DRQ goes inactive but RCS* is still active then a throttle of the current DBA transfer *is* required. This will be because ARM wishes to access the DRAM buffer memory for example. However, in the throttled state there is no way of telling that the SBIC wishes to resume the transfer in that both DRQ and RCS* are inactive and this is exactly as they are when there is no DBA transfer in progress. Thus the DBA state machine will move to the throttle state.

Throttle state
In the Acorn Technical Publishing System design, DBARQ is still asserted in the throttle state to indicate to the main state machine that when SCBSEL* or SCCSEL* have been removed a DBA transfer should resume. However, whilst throttled an abort might have been issued to the SBIC, in which case there is no DBA transfer to continue. To allow for this state the DBA state machine moves to the kickstart state when DRQ is reasserted.

Kickstart state
This state is only entered from the throttle state: DRQ will have been re-asserted to both the DBA state machine and the SBIC. If there is still data to transfer under DBA the SBIC will re-assert RCS* (within about 40 ns!). This will move the DBA state machine to the DBA active state. Otherwise, if RCS* is not asserted then the DBA state machine will move to the No DBA mode.

DBARQ is still asserted in the kickstart state. DBAGT is an inversion of RCS* as this must change over the control of the memory as soon as RCS* is reasserted because of the first byte may start within 40 ns of the assertion of CS*. This will occur whilst the DBA state machine *is* still in the kickstart state.

Enables for RESBIC* and WESBIC*
These two signals are tri-stated when the SBIC has control of the bus. This is indicated by RCS* being active. Thus RCS* is used as the output enable for these two pins.

Register file addressing
As the SBIC is used in the indirect addressing mode the ALE pin (pin 24) is tied low, and the A0 pin (pin 19) is connected to ARM latched address LA2.

SBIC interrupt
The SBIC interrupt pin is inverted by IC7. A series terminator is fitted as IC7 is an ACT gate and thus may produce significant overshoot. A test point is provided on the interrupt pin - this is used during production testing.

Miscellaneous synchronisation and clock circuits

It is possible for the SBIC to be operated at both 8 MHz or 16 MHz in the Acorn Technical Publishing System design. When it is operated at 8 MHz there is no real requirement for synchronisation as all the control signals are timed with respect to REF8M. However, if the 16 *MHz* operation is selected then the DRQ signal must be synchronised into the SBIC. This is done with the 74F74 flip-flop IC11. This is specifically an F series device because of the exceptionally short recovery time exhibited by such devices. It is clocked from the negative going edge of the SBIC clock (inverted by IC7). This provides approximately 90 ns of DRQ to clock setup.

The 16 MHz clock signal is derived from the 96 MHz clock by means of a synchronous divide by 6 circuit.

SCSI external bus

The SBIC drives the SCSI bus directly. In compliance with the SCSI specification, the output drivers are open drain drivers with a sink capability of 40 mA per pin. Provision is made to terminate the bus close to the SBIC by means of 220R/330R resistive terminator packs.

There are two 50 way SCSI connectors. One of them, J9, is a 50 way IDC which is used for the internal hard disc drive connection. The other connector, J3, is a 50 way Delta type connector mounted on the back panel of the PCB. This is the external bus connector for additional external SCSI devices. Between the two connectors is a set of RFI filters that prevent radiation on the external SCSI bus.

The SCSI RST signal is driven from Q13 which is an open collector transistor. This is in turn driven from the control port PAL IC30. The status of the reset line is monitored by means of the clamping circuit D11, D12 and R2.

There *is* a separate fused terminator power circuit. D1 provides reverse isolation on the terminator power circuit, and Fl is a 1 A fuse to prevent damage if the terminator circuit is inadvertently grounded.

The requirement for shield isolation *is* met by LK 4 on the external bus connector.

6:13 SCSI memory map

The following memory map represents a sub-set of the available ARM map in that all the SCSI addresses fall within a limited range.

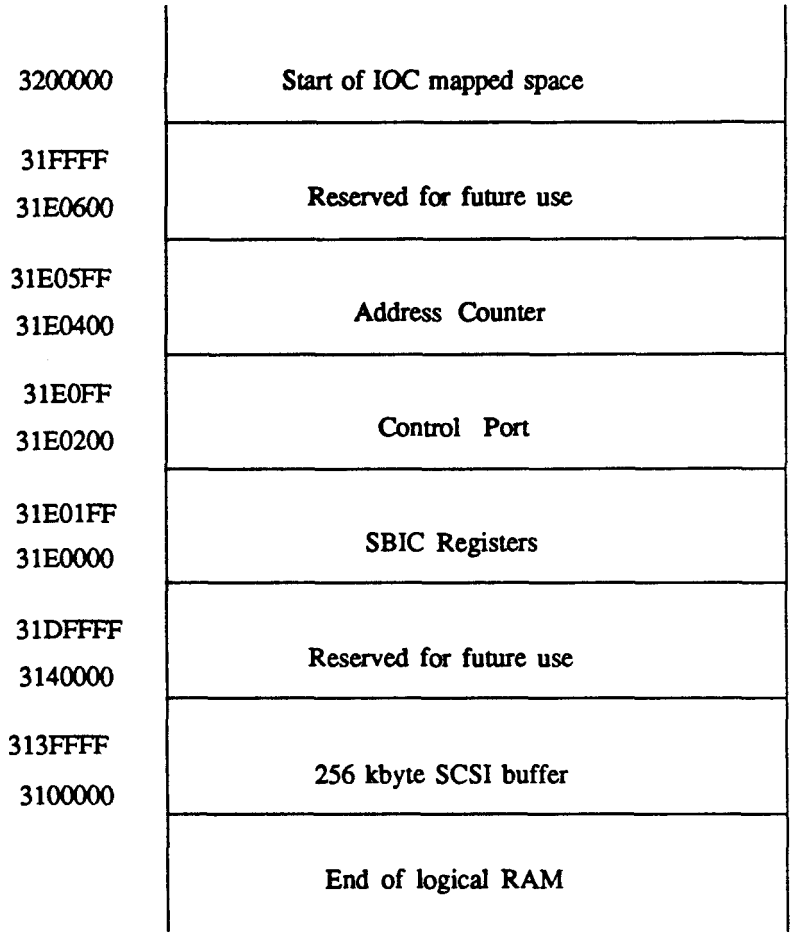

Figure 6.3 : SCSI memory map

Note:
Normal addressing of the SBIC would be:

Aux status/Address Register = 31E0000
Register File (R/W) = 31E0004 (selected by LA2).

6:14 List of signals

ABEN*	ARM Buffer Enable
ABRL*	ARM Buffer Read Latch Strobe
ABWL*	ARM Buffer Write Latch Strobe
ACCK	Address Counter Clock
ACLD*	Address Counter Load Strobe
ARMCYC*	ARM Initiated Access Cycle Strobe
BPHI2	Buffered PHI2 Clock
BS0,1	Byte Select 0 and 1
CA(0:15)	Address Counter Bus
CAS0L,0H*	Bank 0 CAS Lines for DRAM
CAS1L,1H*	Bank 1 CAS Lines for DRAM
CLK9M6	9.6 MHz Clock for SBIC
CLKDBA	SBIC and DBA State Machine Clock
CO2,1	Funnel Byte Select Counter Outputs
CONCYC*	Control Cycle Strobe
CSCON*	Control Port Chip Select
CSSBIC*	Chip Select for SBIC
D(0:31)	32 bit ARM un-buffered Data Bus
DBACYC	DBA Bus Cycle Signal
DBAGT	DBA Grant Signal
DBARQ	DBA Request Signal
DBASM(2:0)	DBA Finite State Machine Registers
DLY40,50	40 ns and 50 ns Taps from Delay Line
DOREF*	Refresh Control Strobe
DRQ	DBA Data Request Signal to SBIC
DRQS	System Data Request (synchronised)
FSM(0:2)	Finite State Machine Outputs
IOGT	ARM I/O Grant Signal (uninverted)
IOGT*	ARM I/O Grant Signal (open-collector)
IORQ*	ARM I/O Request
LA(2:17)	ARM Latched Address Bus
LA2,9,10	Latched Addresses
LR*/W	ARM Latched Write Strobe
MRSBIC*	Master Reset for SBIC
MRSCSI	Reset for SCSI Bus
MUXA(0:7)	DRAM Multiplexed Address
RAMCYC	RAM Cycle Strobe
RAMWE*	RAM Write Enable Strobe
RAS0,1*	RAS Lines for DRAM
RCS*	SBIC Ram Chip Select Signal
REF8M	8 MHz System Timing Clock
REFCLK	16 µs Squarewave
REFCYC	Refresh Cycle from DRAM Control (IC17)
REFQ0,1	Appropriate Bank Refresh Select
REFREQ	Refresh Request
RESBIC*	SBIC Read Strobe
RST*	System Reset Signal
SCBSEL*	Early Decode of Buffer Select by ARM
SCCSEL*	Early Decode of Control Select by ARM
SD(0:31)	32 bit Internal DRAM Buffer Data Bus
SDIR	Funnel Direction Select
SE*(0:3)	Funnel Buffer Enables
SELRNC	Select Row/Column Control
WESBIC*	SBIC Write Strobe

6:15 State machine states

A summary of states within each of the major state machines:

Main state machine - IC20

Idle
Access
SBIC strobe
DBA active
Throttle
DBA refresh pending
DBA refresh occurring

Refresh state machine - IC27

DONE0
PEND1
DORF1
DONE1
PEND0
DORF0

DBA state machine - IC15

No DBA
DBA pending
DBA active
Throttle
Kickstart

6:16 Links and configuration options

LK 4	External Shield Isolation
	Disconnect to isolate shield from equipment 0 V.
LK 60	Clock Selection
	Selects between 16 MHz clock for SBIC and 8 MHz clock.

Chapter 7 The video system

This chapter describes the video system of the Acorn Technical Publishing System.

7.1 Introduction

The video circuit is based around the Acorn video controller chip (VIDC). Refer to the VIDC data sheet (Bibliography, reference 3) for full details of this chip.

The video controller (VIDC) accepts video data from memory under DMA control and serialises it to drive the CRT: The chip controls all the display timing parameters and controls the position of the cursor. The VIDC is a programmable device offering a very wide choice of display formats.

The Acorn Technical Publishing System design does not support all the facilities that a full VIDC implementation could. In particular the low resolution colour display modes and supremacy are not supported.

The 4 bits of digital data which normally drive the red DAC in a full VIDC implementation are externally serialised to a single bit-stream of four times the VIDC pixel rate. With the VIDC operating at 24 MHz, four bits/pixel mode, 96 MHz bit-rates are generated giving a very high resolution monochrome display.

When used in this mode, VIDC must be programmed to a particular set of values described in the section *Programming* below.

7.2 Screen modes

Four screen modes will normally be supported:

- 1152 * 900 pixel graphics mode

  This mode provides a display area of 1152 * 900 pixels plus a two pixel wide border.

- 144 * 45 character mode

  In this mode the display area is treated as 144 character columns by 45 character rows. Each character is eight pixels wide by 20 pixels high. The border is eight pixels wide at the sides and two pixels deep at the top and bottom.

- 96 * 32 character mode

  In this mode the display area is treated as 96 character columns by 32 character rows. Each character is 12 pixels wide by 28 pixels high: The border is eight pixels wide at the sides and four pixels deep at the top and bottom.

- 80 * 20 character mode

  In this mode the display area is treated as 80 character columns by 20 character rows. Each character is 8 pixels wide by 20 pixels high. The border is eight pixels wide at the sides and eight pixels deep at the top and bottom.

7.3 Data display

Pixels are displayed starting at the top left hand corner of the screen. In the modes described above, bit 0 of word 0 in the display buffer is the first displayed pixel, bit 1 is the next pixel etc.

7:4 Programming

The VIDC contains three 32 bit wide FIFOs (video, cursor and sound) and 46 write-only registers of up to 13 bits each. In all cases the address of the register is contained in the top six bits (26-31) of the data field. Bits 25 and 24 are not used (see Figure 7.1 below). The actual data bits are distributed among the remaining 24 bits of the data field according to the register in question. Refer to the VIDC data (see Appendix H *Bibliography)* for details of the bit positions.

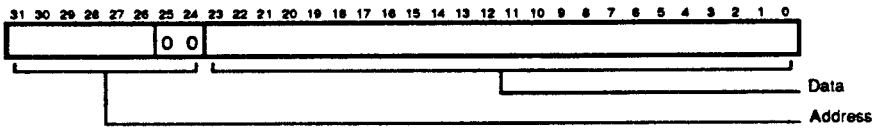

Figure 7.1 : Register fields

In order to define the display format correctly, the registers need to be programmed as shown in Figure 7.2 and Tables 7.1 and 7.2.

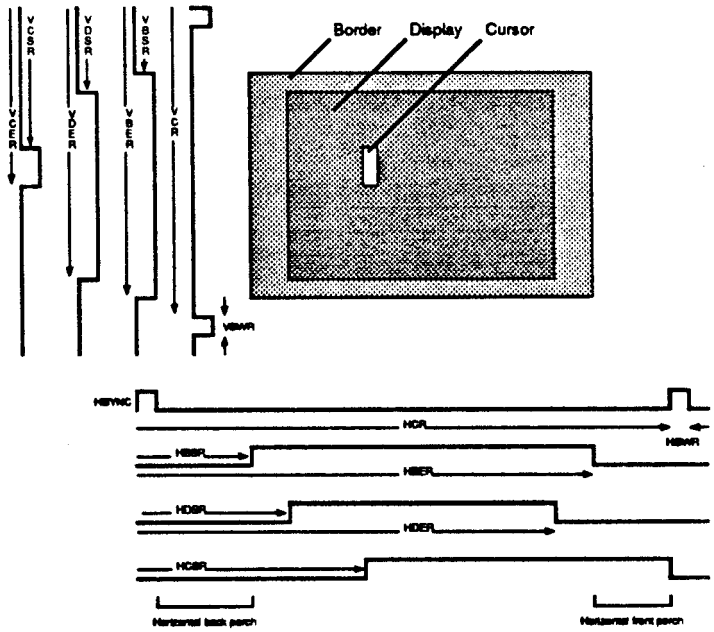

Figure 7:2 : Display format

Address (Hex)	Register	Value
Colour/Sound Registers		
0x00-0x3C	Video Palette logical colours	See text
0x40	Border Colour Register	See text
0x44-0x4C	Cursor Palette logical colours	See text
0x60-0x7C	Stereo Image Register	See next chapter
Horizontal Timing Registers (Graphics Mode, 144 * 45 Character Mode and 96 * 32 Character Mode)		
0x84	Horizontal Sync Width	(208/4-2)/2
0x88	Horizontal Border Start	((208+188)/4-1)/2
0x8C	Horizontal Display Start	((208+188+8)/4-7)/2
0x90	Horizontal Display End	((208+188+8+1152)/4-7)/2
0x94	Horizontal Border End	((208+188+8+1152+8)/4-7)/2
0x80	Horizontal Cycle	((208+188+8+1152+8+4)/4-2)/2
0x9C	Horizontal Interlace	Ignored, see Control Register
Horizontal Timing Registers (80 * 20 Character Mode)		
0x84	Horizontal Sync Width	(208/4-2)/2
0x88	Horizontal Border Start	((208+444)/4-1)/2
0x8C	Horizontal Display Start	((208+444+8)/4-7)/2
0x90	Horizontal Display End	((208+444+8+1152)/4-7)/2
0x94	Horizontal Border End	((208+444+8+1152+8)/4-7)/2
0x80	Horizontal Cycle	((208+444+8+1152+8+260)/4-2)/2
0x9C	Horizontal Interlace	Ignored
Vertical Timing Registers (Graphics and 144 * 45 Character Mode)		
0xA4	Vertical Sync Width	3-1
0xA8	Vertical Border Start	3+43-1
0xAC	Vertical Display Start	3+43+2-1
0xB0	Vertical Display End	3+43+2+900-1
0xB4	Vertical Border End	3+43+2+900+2-1
0xA0	Vertical Cycle	3+43+2+900+2-1
Vertical Timing Registers (96 * 32 Character Mode)		
0xA4	Vertical Sync Width	3-1
0xA8	Vertical Border Start	3+43-1
0xAC	Vertical Display Start	3+43+4-1
0xB0	Vertical Display End	3+43+4+896-1
0xB4	Vertical Border End	3+43+4+896+4-1
0xA0	Vertical Cycle	3+43+4+896+4-1

Table 7:1 : Register values

Vertical Timing Registers (80 * 20 Character Mode)

0xA4	Vertical Sync Width	3-1
0xA8	Vertical Border Start	3+287-1
0xAC	Vertical Display Start	3+287+8-1
0xB0	Vertical Display End	3+287+8+400-1
0xB4	Vertical Border End	3+287+8+400+8-1
0xA0	Vertical Cycle	3+287+8+400+8+244-1

Control Register

0xE0	Control	Composite Sync. Interlace Off DMA Level - End of words 2,6 4 Bits per pixel 24 MHz VIDC pixel rate

Cursor Position Registers

0x98	Horizontal Cursor Start Register	See text
0xB8	Vertical Cursor Start Register	See text
0xBC	Vertical Cursor end Register	See text

Table 7.2 : Register values (continued)

For an in-depth discussion on how to calculate these parameters, refer to the VIDC data sheet. These values have *been* selected to correctly drive the monitor *as described in* Chapter 17, *Monochrome monitor.*

The external serialiser complicates the calculation of the timing parameters slightly. This is because the programmed VIDC pixel rate is one quarter of the external pixel rate. The vertical timing parameters are unaffected by this as they are defined in units of a raster, but the horizontal timing parameters which are defined in units of two (24 MHz) pixels can only be programmed in units of eight (96 MHz) pixels. There are now four times as many pixels on a line as are actually programmed.

Control register (CR) : address 0xE0

The control register should configure VIDC to produce:

- composite sync
- no interlace
- DMA at the end of words two and six
- four bits per pixel
- 24 MHz pixel rate.

Video palette logical colours Ox0-0xF : addresses 0x00-0x3C

In this design, where only the high resolution monochrome display is supported, all 16 locations of the video palette should be programmed to be a 1:1 logical to physical mapping, as shown in Table 7.3 below. White on black and black on white displays *are* illustrated. Supremacy is not supported so bit D[12] is zero.

Video Palette Entry	Value	Effect on display data
0x0	0x0000	
0x1	0x0111	
0x2	0x0222	0 = Black
0x3	0x0333	1= White
etc		
0xE	0x0EEE	
0xF	0x0FFF	
Alternately,		
0x0	0x0FFF	
0x1	0x0EEE	
0x2	0x0DDD	0 = White
0x3	0x0CCC	1= Black
etc		
0xE	0x0111	
0xF	0x0000	

Table 7.3 : Video palette, high resolution mode

Border colour register : address 0x40

In this design the border can be black or white. For a black border 0x0000 should be written to the border colour register, and for a white border 0x000F should be written. It is also possible to produce vertical stripe patterns using values 0x001 - 0x00E. No other values should be used.

7.5 The cursor

The cursor is the same format in all video modes, and is automatically defined to be 32 VIDC pixels wide (128 display pixels), though it may be any number of rasters high. Any pixel may be defined as being transparent, enabling smaller cursors to be constructed. The format is always two bits per pixel, with bits 0, 1 in the first word in the cursor FIFO representing the first pixel, etc. In the high resolution monochrome mode the cursor can only be defined horizontally in units of four (96 MHz) pixels, though it can be positioned anywhere on the screen to within one (96 MHz) pixel.

The cursor should not be programmed to be outside the display area vertically, but it may be programmed to start or end outside the display area horizontally. The cursor must not be programmed to 'run off the right hand side of the screen, though it may be programmed to start before the left hand side. If a cursor of, say only eight pixels wide is required, then the image should be programmed to be at the right hand end of the 32-pixel block, and the first 24 pixels should be programmed to be transparent. In this way the displayed cursor may be positioned anywhere on the screen. Note that the cursor will not be displayed outside the border area either vertically or horizontally.

Horizontal cursor start register (HCSR) : address 0x98

This register defines the time, in units of single pixel periods, from the start of the HSYNC pulse to the start of the cursor display. If M display pixels are required in this time, then value (M-6) should be programmed into the HCSR.

This is a 13 bit register (bits 11 to 23), with bit 11 the least significant.

Note that only the cursor start position needs to be defined, as the cursor is automatically disabled after 32 pixels.

Vertical cursor start register (VCSR) : address OxB8

This register defines the time, in units of a raster, from the start of the VSYNC pulse to the start of the cursor display. If N rasters are required in this time, then value (N-1) should be programmed into the VCSR. This is a 10 bit register, with bit 14 the least significant.

Vertical cursor end register (VCER) : address OxBC

This register defines the time, in units of a raster, from the start of the VSYNC pulse to the end of the cursor display. (ie the first raster on which the cursor is *not* present). If N rasters are required in this time, then value (N-1) should be programmed into the VCER. This is a 10 bit register, with bit 14 the least significant.

Cursor palette logical colours 1-3 : addresses 0x44-0x4C

In this design the cursor palette should be programmed to the values in the table below.

cursor colour 1 : 0x10
cursor colour 2 : 0x20
cursor colour 3 : 0x30

Table 7.4 : Cursor palette, high resolution mode

The colour of each pixel in the cursor is then defined by two bits according to the table below.

MSB	LSB	
0	0	transparent
0	1	cursor black
1	0	do not use
1	1	cursor white

Table 7.5 : Cursor logical colours, high resolution mode

7:6 Monitor

A specification for the monitor *is* in Chapter 17, *Monochrome monitor.*

7.7 Circuit description

This section describes the video circuit.

Register writes

Write data is strobed into VIDC internal registers by the active low VIDW strobe.

The address of the register to be written is supplied on the upper data bits while the lower data bits carry the data to be written.

Video data

VIDC (IC28) requests data from the memory by signalling with video request - VIDRQ. This signal is asynchronous to the system clock and is synchronised by an F174 (IC1). The capacitor/resistor/diode (C38/R114/D3) network may be required to prevent excessive system bus bandwidth being requested by VIDC on system power-up. In practice this occurs rarely and is limited to particular VIDC devices. For *this* reason these three components are not normally fitted.

The addressing of the data in memory is controlled by the memory controller, MEMC, which signals valid data is on the bus by making video acknowledge - VIDAK active.

The data is buffered in one of two FIFOs (video and cursor) within VIDC before being passed through the pallet and DACs.

The inverse of the four bits of data which are fed to the red DAC are output on the VED(0:3) pins on VIDC. These are serialised to a single bit stream at 96 MHz by an F166 shift register IC2 (see Figure 7.3 below).

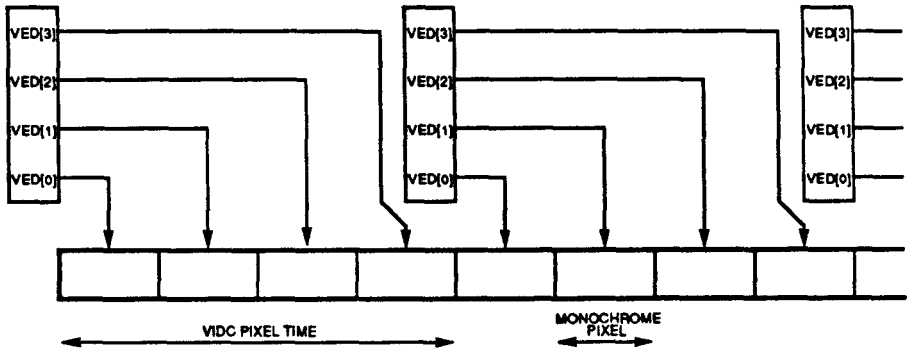

Figure 7:3 : External serialiser

The 96 MHz oscillator is divided down to 24 MHz by a quadrature divide by 4 circuit constructed from an AS74 (IC14). The logical NAND of the two phases of the 24 MHz clock provides the parallel load signal for the shift register.

The output of the serialiser is buffered to provide a 75 CI output at approximately 0.7 V.

The serialiser and output buffer operate from a regulated 5 V supply to reduce noise on the display.

On early VIDC samples the worst case timing parameter for CLK24SYS to VED(0:3) does not allow one particular edge to be used to latch VED(0:3). LK 55 allows this edge to be selected according to the speed of VIDC. All examples to date have been fast enough for LK4 to be set to link pins 1 and 2.

Cursor data

The cursor data FIFO is loaded during the horizontal sync period preceding the line on which cursor data is displayed. Cursor data is addressed, sent and strobed into VIDC in a similar manner to video data. MEMC uses HS to select the cursor data buffer which is separate from the video data buffer. It is not necessary to synchronise HS into MEMC as the timing is still controlled by VIDRQ.

Composite sync: output

With the VIDC control register set to produce composite synchronisation pulses, these emerge on the V/CSYNC pin. The composite sync signal is the exclusive NOR of VSync and HSync (see the figure below).

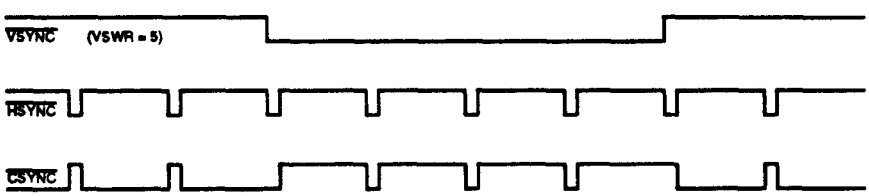

Figure 7.4 : VSYNC output

The V/CSYNC output is inverted and buffered by Q4 and output to the monitor via a BNC on the rear panel. Note that only the falling edge of SYNC is critical to the monitor.

Chapter 8 The sound system

8.1 Introduction

The sound circuit is based around the Acorn video controller chip (VIDC). Refer to the VIDC data sheet for full details of this chip.

It should be noted that this design does not support all the facilities that a full VIDC implementation could. In particular the sound filtering circuitry has been optimised to produce a monaural sound channel.

The video controller (VIDC) accepts sound data from memory under DMA control, serialises it, and performs digital to analogue conversion to drive a speaker via an external power amplifier and filter.

The bandwidth of the sound channel is approximately 100 Hz to 1.5 kHz.

The sound system within VIDC consists of a four word FIFO and bytewide latch which drive a 7 bit exponential DAC: The eighth (sign) bit steers the DAC output to one of two output pins, one designated `+' and the other designated `-'. The sound signal is generated externally by integrating and then subtracting these two pairs of signals.

In a stereo VIDC implementation the stereo image is synthesised by time-division multiplexing the output between the 'left' and 'right' pairs of output signals. In this monaural design this *is* not supported and the left and right channels are summed externally to produce a mono signal.

8.2 Programming

The addressing of the data in the sound buffer is controlled by the memory controller, MEMC. See Appendix H *Bibliography*.

The VIDC can operate in 1, 2, 4 or 8 channel modes. In this design it should be operated in single channel mode. In single channel mode all eight stereo image registers must be programmed to the same value. Stereo sound data can be played through the sound system but will be reproduced in mono.

In single channel mode sequential bytes in the sound buffer are sequential samples and are applied to the DAC in the order they appear in the buffer. The frequency with which samples are applied to the DAC *is* programmable and set in the sound frequency register.

In all cases the address of the VIDC register to be written to contained in the top six bits (26-31) of the data field. Bits 25 and 24 are not used. The actual data bits are distributed among the remaining 24 bits of the data field according to the register in question. Refer to the VIDC data sheet for details of the bit positions:

Stereo Image registers, channels 0-7 : addresses 0x60-0x7C

In a full VIDC implementation these eight registers define the stereo image position for each of the eight possible channels. In this design these have no function and should be programmed to a central position.

Sound frequency register (SFR) : address OxC0

This register defines the byte sample rate of the sound data. It is defined in units of 1

If a sample period of N µs is required, then (N-1) should be programmed into the SFR. N may take any value between 3 and 256.

The sample rate is selectable by the SFR in units of 1 µs, with a minimum value of 3 µs.

8:3 Circuit description

VIDC (IC28) requests data from the memory by signalling with sound request - SNDRQ . This signal is asynchronous to the system clock and is synchronised by an F174 (IC 1). The addressing of the data in memory is controlled by the memory controller, MEMC which signals valid data is on the bus by making sound acknowledge - SNDAK active.

The data is buffered in the sound FIFO in VIDC before being serialised and digital to analogue converted.

The sound outputs CH- and CH+ are current sinks. The magnitude of the output current is a function of the sound reference input current generated by R71 from the filtered reference voltage SNDVDD. (The reference current is equal to the step size in the highest chord.) The output currents are integrated and converted to a voltage by R135/R134 and C77/C75. The result is further filtered by R128/R131 and C78/C70.

The filtered sound outputs are summed and buffered by a low voltage audio amplifier, an LM386 (IC92), which has a gain of approximately 20 in this circuit. The inputs to the LM386 have 50 kOhm pull-down resistors and together with C73/C76 provide 0V DC restoration. The output capacitor C72 removes the 2.5 V output offset from the speaker.

Chapter 9 Keyboard and mouse

The keyboard is a separately housed unit which communicates with the main processor box via a duplex serial link. It is the keyboard that is responsible for detecting keys states and tracking mouse movement.

The specification of the keyboard is given in this chapter.

One hundred and three keys are provided with a general layout similar to the IBM 101 standard.

The rolling ball type mouse plugs into the keyboard and provides an additional three keys. The specification of the mouse is in this chapter.

The keyboard may reset the main processor (if this function is enabled) but is designed so that plugging and unplugging it from the main processor does not cause a system reset or software failure.

The keyboard identifies each key by its row and column address in the keyboard matrix. Tables 9.2, 9.3 and 9.4 show the correspondence between the physical key position and the row and column codes.

9.1 Keyboard interface

At the keyboard end the serial link is controlled by a microcontroller, and at the processor (host) end by the IOC keyboard interface.

The serial link uses NRZ encoding and is a two wire full duplex system which operates at 31.25 K baud. The keyboard is powered via the serial link.

The keyboard protocol is described in the following sections.

9.2 Mouse interface

The mouse interface has three switch sense inputs and two quadrature encoded movement signals for each of the X axis and Y axis directions. Mouse key operations are debounced and then reported to the host via the keyboard and keyboard protocol. The mouse keys are allocated row and column codes within the main key matrix: See Table 9.4:

Circuitry in the keyboard decodes the quadrature signals from the mouse and maintains a signed 7 bit count for each axis of mouse movement.

When count overflow or underflow occurs on either axis both X and Y axis counts lock and ignore further mouse movement until the current data has been sent to the host computer.

Overflow occurs when a counter holds its maximum positive count (0111111 binary). Underflow occurs when a counter holds its maximum negative count (1000000 binary).

Each axis can count up to 5,000 state changes per second minimum, this corresponds to approximately 16 inches per second on a 300 points per inch mouse.

Serial data transmissions from the keyboard are either one or two bytes in length. Each byte sent is individually acknowledged by the host computer. The keyboard will only transmit a byte when the previous byte has been acknowledged. The exception to this *is* the reset code indicating either that a power on or user reset occurred, or that a protocol error occurred.

Each data byte (eight data bits) is preceded by a single start bit (logic 1). The least significant data bit (DO) is sent first. The last data bit (D7) is followed by two stop bits (logic 0). Note that data is sent in inverted form, that is a logic 1 data bit will appear on the serial line as a logic 0 and vice versa.

Reset protocol

The keyboard restarts when it receives a one byte HardReSet (HRST) code from the host.

To initiate a restart the keyboard sends a HRST code to the host, which will then send back HRST to command a restart.

The keyboard sends HRST to the host if :

- a power on reset occurs
- a user reset occurs
- a protocol error is detected.

After sending HRST, the keyboard waits for a HRST code. Any non HRST code received causes the keyboard to resend HRST to the host.

The pseudo program below illustrates the reset sequence or protocol.

```
START reset

ON error send HRST code to host then wait for code from host
IF code = HRST THEN restart ELSE error

ON restart clear mouse position counters
        set mouse mode to data only in response to an RMPS request
        stop key matrix scanning and set key flags to up
        send HRST code to host

Wait for next code
IF code = RAK1 THEN send RAK1 to host ELSE error

Wait for next code
1F code = RAK2 THEN send RAK2 to host ELSE error

Wait for next code
IF code = SMAK THEN mouse mode to send if not zero and enable key scan
ELSE IF code = SACK THEN enable key scanning
     ELSE IF code = MACK THEN set mouse mode to send when not zero
       ELSE IF code = NACK THEN do nothing ELSE error

END reset
```

Note: the on/off state of the <Shift> and <Caps lock> key LEDs does not change on reset. The LED state is not defined at power on. The host computer is always responsible for selecting the LED status. After the reset sequence, key scanning is only enabled if a scan enable acknowledge (SACK or SMAK) is received from the host.

Data transmission

When enabled for scanning, the keyboard informs the host of any new key down or new key up by sending a two byte code incorporating the key row and column addresses.

The first byte gives the row and is acknowledged by a byte acknowledge (BACK) code from the host. If BACK is not received, then the error process (ONerror) is entered. See *Reset protocol* above.

If the BACK code is correctly received the keyboard will send the column information and wait for an acknowledge. If either a NACK, SACK, MACK or SMAK acknowledge code is received then the keyboard will continue by processing the acknowledge code. See the section *Command and acknowledge code summary* below.

If the character received as the second byte acknowledge was not one of NACK/MACK/SACK/SMAK then the error process is entered. See *Reset protocol* above.

While key scanning is suspended (after NACK or MACK) any new key depression is ignored and does not result in a key down transmission unless the key remains down after scanning resumes following a SACK or SMAK. Similarly a key release is ignored while scanning is off.

Mouse data

Mouse data is sent by the keyboard if requested by a RQMP request from the host or if a SMAK or MACK have enabled transmission of non-zero values.

Two bytes are used for mouse position data. Byte one encodes the accumulated movement along the X axis while byte two gives Y axis movement. Both X and Y counts are transferred to temporary registers when data transmission is triggered, so that accumulation of further mouse movement can occur. The X and Y counters are cleared upon each transfer to the transmit holding registers. Therefore the count values are relative to the last values sent.

The host acknowledges the first byte (Xcount) with a BACK code and the second byte (Ycount) with any of NACK/MACK/SACK/SMAK. A protocol failure causes the keyboard to enter the error process. See *Reset protocol* above.

When transmission of non-zero mouse data *is* enabled, the keyboard gives key data transmission priority over mouse data except when the mouse counter over/underflows.

Description of acknowledge codes

There are seven acknowledge codes which may be sent by the host

Mnemonic	Function
RAK1, RAK2	These are used during the reset sequence described in *Reset protocol* above.
BACK	This is the acknowledge to the first byte of a two byte keyboard data set.
NACK	Acknowledge and disable key scanning. The transmission of key up/key down data *is* disabled and mouse data *is* sent only on RQMP request.
SACK	Acknowledge and enable key scanning. The transmission of key up/key down data *is* enabled but mouse data is sent only on RQMP request.
MACK	Acknowledge and disable key scanning. The transmission of key up/key down data is disabled, but the transmission of non-zero mouse data is enabled.
SMAK	Acknowledge and enable key scanning. The transmission of key up/key down and non-zero mouse data is enabled.

9.3 IOC keyboard asynchronous receiver transmitter (KART)

At the processor end of the serial link is the IOC KART. This provides an asynchronous serial link to the keyboard. It is of fixed format with eight bits to a character which is framed with one start bit and two stop bits. The least significant bit KD[0] is transmitted/received first. The KART divides into two, the receiver and the transmitter.

The ARM accesses the receiver via the serial Rx data register. A clock of 16 times the data rate is used by the KART to clock in the serial data from the KIN pin. When a data byte has been received, the SRx bit is asserted in the IRQ status B register to indicate that the byte is available for reading. See *IOC interrupt allocations in* chapter 5 for details of this register.

False start bits of less than a half bit duration are ignored.

The ARM accesses the transmitter via the serial Tx data register. The byte written to the serial Tx data register is transmitted serially from the KOUT pin, and the STx bit is asserted in the IRQ status B register to indicate that the transmission is finished and the serial Tx data register may be reloaded. **See** *10C interrupt allocations in* chapter 5 for details on this register.

The receive and transmit speeds are the same and may programmed using counter 3 in IOC. The current keyboard operates at 31.25 kbaud.

Serial Tx data register

Writing to this register loads the serial output shift register, clears any outstanding TRx interrupt and starts the transmission. An interrupt is raised when the register is ready to be reloaded.

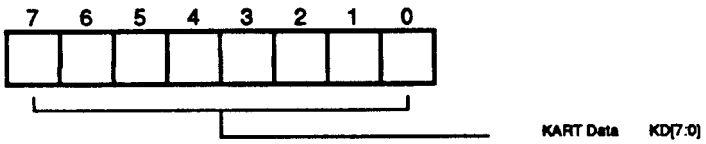

Figure 9.1 : Serial Tx data register 0x04 write

Serial Rx data register

Reading from this register clears any outstanding SRx interrupt and returns the current received byte. Data is only valid while the SRx bit is set in the IRQ status B register.

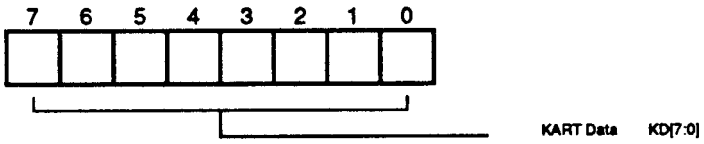

Figure 9.2 : Serial Rx data register 0x04 read

Initialisation

After power-on, the KART is in an undefined state. The KART is initialised by programming the serial line speed using counter 3 and performing a read from the serial Rx data register, discarding the data byte. This clears any outstanding receive interrupt and enables the KART for the next reception. Finally the Tx data register is written to. This will abort any transmission in progress, cause a new one to be started, and clear any STx interrupt.

Receive Interrupt

The SRx interrupt is set halfway though the reception of the last data bit.

Note: Care should be taken to ensure that the last bit has been received before the serial Rx data register is read, to prevent this bit being interpreted as the start bit of the next packet.

Keyboard protocol

The keyboard protocol is described in the previous section. This section describes actions that the processor takes to conform to this protocol.

When the keyboard has sent a byte, in normal operation it will not send again until it has received an acknowledge from the ARM. The only exception to this is during the reset protocol used to synchronise the handshaking, where each side is expecting specific responses from the other, and will not respond further until these are received. Unexpected responses cause the reset sequence to be restarted.

Key changes are transmitted as key up and key down codes, appended to row and column data. This is only transmitted when enabled by an acknowledge scan code. An acknowledge scan code should always be issued at the end of a keyboard packet (two bytes), as there is no command code allowing scanning to be enabled later.

The treatment of mouse data is a modification of this simple protocol The keyboard will not send mouse data unless specifically allowed to. This is indicated to the keyboard by an acknowledge mouse code. This code instructs the keyboard to transmit the cumulative change of mouse coordinates since the last instruction, or if there has been no change the next move made by the mouse. While it is not allowed to send mouse coordinates the keyboard will update a counter with any changes. If the mouse is moved too far between requests for mouse data, the count will overflow or underflow. See *Mouse data* above for more details.

Description of commands

In addition to acknowledging codes transmitted by the keyboard the host can also initiate actions in the keyboard by sending commands. Commands may be sent at any time and may be interleaved with acknowledge codes.

The LEDS and PRST commands may be acted on immediately. Commands which require a response are held pending until the current data protocol is complete.

Repeated commands only require a single response from the keyboard.

Mnemonic	Function
HRST	Reset keyboard. If the HRST command is received the keyboard immediately enters the restart sequence. See *Reset protocol* above.
LEDS	Turns key cap LEDs on/off. A three bit field indicates which state the LEDs should be in. Logic 1 is on, logic 0 (zero) off. D0 controls Caps Lock D1 controls Num Lock D2 controls Scroll lock
RQMP	ReQuest Mouse Position (X, Y counts). See *Mouse data* above.
RQID	ReQuest keyboard Identification code. The keyboard is manufactured with a six bit code to identify the keyboard type to the host. Upon receipt of RQID the keyboard transmits KBID to the host.
PRST	Reserved for future use. The keyboard ignores this command
RQPD	For future use. Current keyboards encode the four data bits into the PDAT code data field and then send PDAT to the host.

9.4 Command and acknowledge code summary

Mnemonic	msb	lsb	Comments
HRST	1111	1111	One byte command, keyboard reset
RAK1	1111	1110	One byte response in reset protocol
RAK2	1111	1101	One byte response in reset protocol
RQPD	0100	xxxx	One byte from host, encodes four bits of data
PDAT	1110	xxxx	One byte from keyboard, echoes four data bits of RQPD
RQ1D	0010	0000	One byte host request for keyboard ID
KBID	10xx	xxxx	One byte from keyboard encoding keyboard 1D
KDDA	1100	xxxx	New key down data. Encoded row (1st byte) and column (2nd byte) numbers
KUDA	1101	xxxx	Encoded row (1st byte) and column (2nd byte) numbers for a new key up
RQMP	0010	0010	One byte host request for mouse data
MDAT	0xxx	xxxx	Encoded mouse count, X (byte1) then Y (byte2)
BACK	0011	1111	Ack for first keyboard data byte pair
NACK	0011	0000	Ack and disable keys, disable mouse
SACK	0011	0001	Ack and enable keys, disable mouse
MACK	0011	0010	Ack and disable keys, enable mouse
SMAK	0011	0011	Ack and enable keys, enable mouse
LEDS	0000	0xxx bit flag to turn LED(s) on/off	
PRST	0010	0001 From host, one byte command, does nothing	

x is a data bit in the code e.g. xxxx is a four bit data field

Table 9.1 : Command and acknowledge code summary

9:5 Key codes

The key codes transmitted by the keyboard are defined in Tables 9.2, 9.3 and 9.4 below.

Key Posn	Key Size	Key Name	Row Code	Col. Code	COMPTEC Legend	Notes	Key cap front Legend
F1	1	Esc	0	0	E843	1	
F2	1	F1	0	1	F851	2	
F3	1	F2	0	2	F852	2	
F4	1	F3	0	3	F853	2	
F5	1	F4	0	4	F854	2	
F6	1	F5	0	5	F855	2	
F7	1	F6	0	6	F856	2	
F8	1	F7	0	7	F857	2	
F9	1	F8	0	8	F858	2	
F10	1	F9	0	9	F859	2	
F11	1	F10	0	A	F860	2	
F12	1	F11	0	B	F861	2	
F13	1	F12	0	C	F862	2	
F14	1	Print	0	D	P761	1,3	
F15	1	Scroll	0	E	S100	1	
F16	1	Break	0	F	B461	1	TBA (P959) Pause
E1	1	`~	1	0	KK6D		
E2	1	1	1	1	1215		
E3	1	2	1	2	2215		
E4	1	3	1	3	3215		
E5	1	4	1	4	4225		
E6	1	5	1	5	5215		
E7	1	6	1	6	6225		
E8	1	7	1	7	7225		
E9	1	8	1	8	8205		
E10	1	9	1	9	9215		
E11	1	0	1	A	0216		
E12	1	- _	1	B	Mil S		
E13	1	=+	1	C	DD1S		
E14	1	£ ¤	1	D	ZZY0		
E15	1	Backspc	1	E	AA0L	1	
E16	1	Insert	1	F	1835	1	
E17	1	Home	2	0	H229	1,3	
E18	1	Pg up	2	1	P955	1	
E19	1	Numlock	2	2	N100	1,4	
E20	1	/	2	3	KK9J	1	
E21	1	*	2	4	FF5L	1	
E22	1	#	2	5	ZZAB	1	

Table 9.2 : Key codes

Key Posn	Key Size	Key Name	Row Code	Col. Code	COMPTEC Legend	Notes	Key cap front Legend
D1	1.5	Tab	2	6	T077	1	
D2	1	Q	2	7	Q838		
D3	1	W	2	8	W838		
D4	1	E	2	9	E838		
D5	1	R	2	A	R838		
D6	1	T	2	B	T838		
D7	1	Y	2	C	Y838		
D8	1	U	2	D	U838		
D9	1	I	2	E	1838		
D10	1	0	2	F	0838		
D11	1	P	3	0	P838		
D12	1	[{	3	1	JJ1L		
D13	1	] }	3	2	JJ1M		
D14	1.5	\ |	3	3	KK22		
D15	1	Delete	3	4	D841	1	
D16	1	Copy	3	5	C01P	1	TBA (E1100 End)
D17	1	Pg dwn	3	6	P956	1	
D18	1	7	3	7	7837		
D19	1	8	3	8	8837		
D20	1	9	3	9	9837		
D21	1	-	3	A	DD9L	1	
C1	1.75	Ctrl	3	B	C036	1,3	
C2	1	A	3	C	A838		
C3	1	S	3	D	S838		
C4	1	D	3	E	D838		
C5	1	F	3	F	F838		
C6	1	G	4	0	G838		
C7	1	H	4	1	H838		
C8	1	J	4	2	J838		
C9	1	K	4	3	K838		
C10	1	L	4	4	L838		
C11	1	9 •	4	5	MMD0		
C12	1	' "	4	6	PP74		
C13	2.25	Return	4	7	R024	1	
C14	1	4	4	8	4837		
C15	1	5	4	9	5837		
C16	1	6	**4**	**A**	6837		
C17	1	+	4	B	DD2W	1	
B1	2.25	Shift	4	C	S014	1,3	
B2		"KEY104"		4	D		
B3	1	Z	4	E	Z838		
B4	1	**X**	**4**	F	X838		
B5	1	C	5	0	C838		
B6	1	V	5	1	V838		

Table 9:3 : Key codes (continued)

Key Posn	Key Size	Key Name	Row Code	Col. Code	COMPTEC Legend	Notes	Key cap front Legend
B7	1	B	5	2	B838		
B8	1	N	5	3	N838		
B9	1	M	5	4	M838		
B10	1	, <	5	5	BB4C		
B11	1	. >	5	6	BB4B		
B12	1	/?	5	7	GG54		
B13	2.75	Shift	5	8	S044	1,3	
B14	1	CrsrUp	5	9	AA01	1	
B15	1	1	5	A	1837		
B16	1	2	5	B	2837		
B17	1	3	5	C	3837		
A1	1.5	Caps	5	D	C206	1,4	
A2	1.5	Alt	5	E	A046	1,3	
A3	7.0	Space	5	F			
A4	1.5	Alt	6	0	A046	1,3	
A5	1.5	Ctrl	6	1	C150	1,3	TBA(A1009 Action)
A6	1	CrsrLt	6	2	AA0L	1	
A7	1	CrsrDn	6	3	AA0M	1	
A8	1	CrsrRt	6	4	AA00	1	
A9	2.0	0	6	5	0140		
A10	1	-	6	6	MM0P		
A11	2.0	Enter	6	7	E095	1	
LEFT		-	7	0	5		
MIDDLE		-	7	1	5		
RIGHT		-	7	2	5		

Notes : Unless shown otherwise, keytop colour is Pantone warm grey 3
Legend marking is black
Row and column codes are in hexadecimal
Key positions are as in Acorn drg. 0276,100/L

1. Dark keytop
2. Red keytop
3. Key position with N key rollover
4. Green light emitting diode under key cap
5. Located on mouse

Table 9.4 : Key codes (continued)