C.A.T. PROCESSOR CARD
TECHNICAL MANUAL

Document Number: 495-701000

Issue: 1.0.2

NEW CAMBRIDGE HOUSE

LITLINGTON, NR. ROYSTON

HERTS. SG8 0SS

CONTENTS

1. Introduction

The C.A.T. Processor Card is a high performance VME compatible card based on the Texas Instruments TMS320C80 processor. The C.A.T. base card can be used alone as a powerful data processing unit, but it has been designed to work with closely coupled, high performance display boards and application specific daughterboards for applications such as:

- High Performance Graphics - Primalib, X-Windows
- Mixed Graphics and Radar displays
- Mixed Graphics and Video displays
- Video Capture and Compression
- Radar Tracking and Compression

This manual describes the C.A.T. base processor card and its operation. Other manuals describe the individual Display and Application Daughterboards.

1.1 Features

- TMS320C80 Processor operating at 50MHz
- 8 or 32MBYTES of Synchronous DRAM
- 1MBYTE of Flash EPROM
- Closely Coupled Video Display Board Site
- Highly flexible Application Daughterboard Site
- Shared memory VME I/F
- VME Master/Slave D64 BLT
- Dual RS-232 Serial I/O channels
- Dedicated Keyboard/Mouse connector
- Single Slot VME double Eurocard

1.2 Performance

1.2.1 Drawing Speeds

The following performance figures have been measured on a C.A.T. with a Lion Display Board.

Line Drawing
(10 Pixel Vectors, 8 bits per pixel)

- Vectors per second 1.2 Million

Image Copy
(500 pixel x 500 pixel, 8 bits per pixel)

- On screen to on screen 66 Million pixels/second
- Off screen to on screen 100 Million pixels/second

1.2.2 Image Processing

The following benchmarks are those provided by Texas Instruments as indicators of the TMS320C80 performance. The figures are for operations performed on data within the processor's internal memory and take no account of the overheads involved in transferring data into and out of the processor.

FFT
(16 bit data, 16 bit coefficient)

- 1024 point FFT (Parallel Processor) 0.16ms

JPEG
(800x600 pixels, Colour, Q = 90, Compression = 4:1)

- Encode time 42.0ms
- Decode time 58.8ms

1.2.3 Data Transfer

The TMS320C80 may access:

- Synchronous DRAM at 400MBytes/sec
- Display Board VRAM at 200MBytes/sec (write)
- Application Daughterboard at (up to) 200MBytes/sec

2. Installation

2.1 Getting Started

Power must be off before plugging in or removing the board from the backplane. All +5V, +12V, -12V and Gnd pins on both P1 and P2 must be connected.

The base C.A.T. does not use rows ‘a’ and ‘c’ of P2, however these signals are routed to the Application Daughterboard Site. Thus where an Application Daughterboard is fitted, the Daughterboard Technical Manual should be consulted for P2 compatibility.

When the C.A.T. is fitted to a VME compatible backplane with bus grant and interrupt acknowledge daisy chain links for each slot, the links must be removed from the slot in which the C.A.T. is fitted.

Forced air cooling may be required in systems where high ambient air temperatures are likely within the VME rack, see the Specification section - **3.4 Environmental**.

Before plugging the card into the rack check that the Link Settings are as you require. See the Installation section - **2.4 Link Settings**.

VME Interface parameters, such as Slave Address, Bus Request Level etc. are all set by software, however when using the standard C.A.T. firmware the values of these parameters are read from EERAM. Thus a user can edit the parameters in EERAM to set up the required Slave address, which will then be used by the C.A.T. each time it boots. See the Installation section - **2.5 EERAM settings**.

2.2 Front Panel

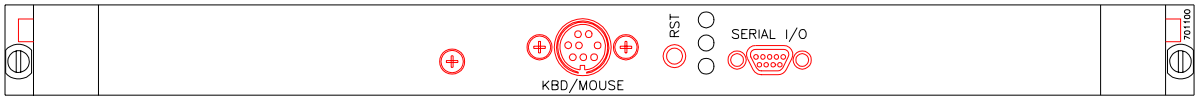

Figure 2.1: C.A.T. Front Panel

The front panel of the C.A.T. is shown in Figure 2.1. The illustrated version is of a C.A.T. without Display or Application Daughterboards.

The push button (marked RST) asserts reset to the C80 processor when pressed. If the C.A.T. is linked to be the VME Slot 1 controller, SYSRESET[L] will be asserted to the whole VME rack.

The three LED's have the following functions:

- Green: Power O.K. When this LED is ON it indicates that the +5V and +3V3 power rails are above their specified minimum voltage. When the supply rails are below their minima the C80 is held reset.
- Orange: Activity. This LED is ON when the C80 is accessing external memory, and OFF when the C80 external memory interface is idle. The brightness of this LED is an indication of how often the C80 is accessing memory.
- Red: S/W. This LED is turned ON at power-up and whenever the C80 is reset. The standard firmware turns the LED OFF on the completion of its self test and initialisation.

The micro D-Type connector labelled 'SERIAL I/O' allows connection of two RS-232 serial channels to the on board UART.

2.3 I/O Connectors

The CAT I/O connectors are shown in Figure 2.2.

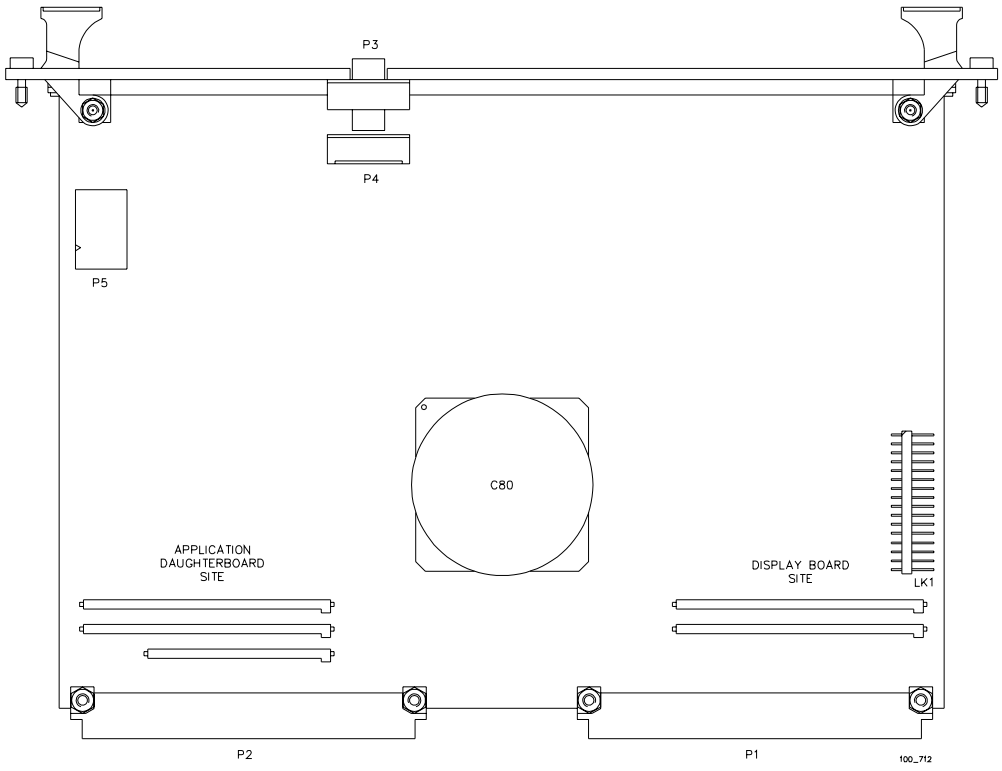

Figure 2.2: C.A.T. Connectors and Links

P1 and P2 are the standard VME backplane connectors. The VME signals used by the C.A.T. are listed in **Appendix C.** Rows 'a' and 'c' of P2 are routed to the adjacent Application Daughterboard Site.

P3 is the Micro D-Type on the front panel. It carries two RS-232 serial I/O channels. See **Appendix C** for the pinout. Channel A has TXD, RXD, RTS, CTS and DTR connections, whilst Channel B only has TXD and RXD. The RS-232 level translators are powered from the +12V and -12V supplies. All signals on this interface are protected by 18V varistors connected to 0V.

P4 is the dedicated Keyboard/Mouse connector, see **Appendix C** for the pin out. These are TTL level signals, protected by diodes to 0V and +5V. Also on the connector is a +5V supply to power external devices, this is protected by a 0.5A self resetting fuse.

P5 is programming header for the in-circuit-programmable FPGA's.

Also shown in Figure 2.2. are the Display Board connectors, and the user Links - LK1.

2.4 Link Settings

The C.A.T. has one block of eight links on the edge of the card, labelled LK1. These lie flat underneath the optional Display Board and can be changed without removing the Display Board.

Three of the links have H/W defined functions, the remaining links are used by software applications to select different modes of operation.

The individual links are D0 to D7, where D7 is nearest to P1, their functions are:

D0	Enable C.A.T. as VME Slot 1 System Controller. (Enable when link fitted)
D1	EERAM write protect 'A'. (Write is enabled when link fitted)
D2	EERAM write protect 'B'. (Write is enabled when link fitted)
D3	Reserved. Fit to set default VME Parameters (with Monitor F/W)
D4	Reserved. Fit to disable embedded application (with Monitor F/W)
D5,6,7	General Purpose. Function defined by S/W

2.5 EERAM Settings

When the C.A.T. is fitted with the standard Firmware, various parameters of the card, such as VME Slave Address, are defined by the contents of the EERAM. Thus to set up the card for a particular system, the EERAM contents will need to be edited. The standard C.A.T. firmware contains a menu driven program to edit these parameters. The human interface to this program is a Terminal (or a PC with terminal emulation S/W) connected to Serial I/O channel A.

When the C.A.T. is fitted with the standard firmware, the EERAM can be edited as follows:

i) Fit the EERAM Write Enable 'B' link to LK1 - D2
ii) Connect a Terminal (or PC with Terminal Emulation S/W) via an RS-232 cable to the front panel micro D-Type (channel A)
iii) Configure the Terminal/PC for 9600 Baud, 7 bits, no parity, 1 stop bit
iv) Power-up the C.A.T. and follow the on-screen menus to edit the various VME I/F parameters (typing **? rtn** will display the current menu)

v) If the C.A.T. has a Display Board fitted the video format may also be edited
vi) Power-down the C.A.T. and remove the EERAM Write enable link

For a full description of this process, and listings of all the menu options see the **C.A.T. Monitor and Support Utilities User Guide (494-538000).**

Note: The EERAM contents can also be viewed/modified via VME Short Address space, however as this is a direct manipulation of the EERAM contents it is far less convenient - see the C.A.T. Monitor and Support Utilities User Guide.

3. Specification

3.1 Functional

3.1.1 Processor

TMS320C80 operating at 50MHz

Master Processor (MP) - 32 Bit RISC, 100 MFLOPS, 4k I-Cache, 4k Data-Cache
Four Parallel Processors - 64 Bit op-code, 2k I-Cache, 8k Data-Cache
Transfer Controller - 400MBytes/sec transfers, supports SDRAM/VRAM/SRAM
Two Video Controllers - Programmable Video Timing, Capture/Display
3V3 Operation

3.1.2 Memory

Synchronous DRAM

8 or 32MBytes (see **Appendix D** Card Versions)
64 bit data
400MBytes/sec Read/Write Burst capability

Flash EPROM

Two 32 pin PLCC sites allowing 256k or 1MByte of Flash EPROM
+5V programming - can be programmed on board

3.1.3 Display Board Interface

Dedicated Display Board Interface with:

Up to three separate Framestores with (max) 16MBytes per store
200MBytes/sec Burst Write capability to Framestores
133MBytes/sec Burst Read capability from Framestores
Dedicated address decodes for DAC/LUTs/Control registers

3.1.4 Application Daughterboard Interface

Dedicated Application Daughterboard Site with:

Up to two stacked Daughterboards
Separate Data and Control decodes for each site
Data Widths of 8/16/32/64 bits
Up to 16MBytes of data memory per board
Shared Memory support
PDT (DMA) support

3.1.5 VME Interface

512k Shared SRAM

DTB Slave	A32 D32/16/8(EO) BLT D64/32

256 Byte shared EERAM/Registers

DTB Slave	A16 D8(O)
Master	A32/16 D32/D16/D8(EO) BLT D64/32
Interrupter	I (1-7) ROAK
Interrupt Handler	IH (1-7)
Slot 1 Arbiter	SGL/RRS/PRI

3.1.6 Serial I/O

Four serial I/O channels: Two RS-232, Two TTL (Keyboard/Mouse)

RTS/CTS/DTR on Channel A (RS-232)

Programmable baud rate from 50 to 38.4k baud

3.2 Mechanical

Board dimensions	:	233 x 160 mm (Double Eurocard, Single Slot)
VME Connectors	:	2 x DIN41612 96 Way
Front Panel	:	9 Way Micro D-Type (ITT Canon MDSM-9PE-Z10-VR1)
	:	Push Button reset switch
	:	Activity/Power LEDs
Weight	:	400 grammes

3.3 Power

SUPPLY	TOLERANCE	CURRENT TYP	CURRENT MAX
+5V	± 5%	1.5A	2.0A
+12V	± 10%	0.05A	0.1A
-12V	± 10%	0.05A	0.1A

The above figures are for the base C.A.T. only, any Display or Application Daughterboard current consumptions should be added to these.

The +3V3 supply for the C80 and Synchronous DRAM is generated by an on-board switching power supply module from the +5V supply. The power supply module is protected by a built-in over-current shutdown. The C.A.T. is protected from a fault in the module by a 4A SMT Time Delay Fuse (FS2) on the +5V input to the module.

3.4 Environmental

Storage only	:	Ambient temperature -40°C to +75°C
Storage and operating	:	Humidity 5% to 95%, non-condensing
Operating	:	Ambient temperature 0°C to +55°C
Maximum shock	:	30G Peak, 11ms duration.

Normally the C.A.T. would be expected to operate with a small amount of forced air cooling, however in benign environments this may not be necessary. The only component on the C.A.T. base card that requires special consideration is the C80 itself. The C80's cooling requirements are shown in the graph below.

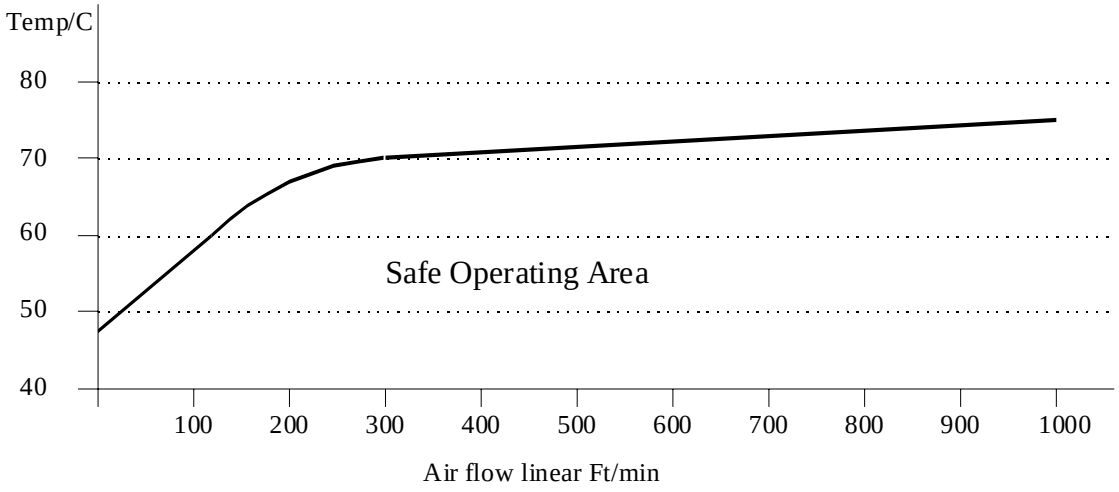

Figure 3.1: Maximum Ambient Temperature Versus Airflow

The C80 power dissipation is dependent upon processor activity, but it is expected that the C80 will operate below its maximum case temperature (85°C) in all situations where the above air flow requirements are satisfied.

4. Principles of Operation

This section describes the operation of the C.A.T. with reference to the block diagram Figure 4.1. The Lion Multilayer Display Board is included in this diagram, but only the interface to this card is described - for a full explanation see the Lion Technical Manual.

4.1 Overview

The block diagram in Figure 4.1 shows the various circuit elements of the C.A.T. and Lion Display Board. Each element has a fixed location in the C80 address space which is divided into 16 MByte blocks. Even circuit elements on the optional Display Board have their place in the memory map set by the main C.A.T. board. The C.A.T. memory map corresponding to the block diagram is given in **Appendix A**.

The circuit elements on the C.A.T. block diagram are:

- The TMS320C80 Processor
- SDRAM - 8 or 32MBytes of Synchronous DRAM for code and data
- FLASH EPROM - 1MByte of Firmware for C80 boot code
- QUART - Quad UART for serial communications
- SRAM/EERAM - Dual ported memory shared between VME and the C80
- VME I/F - Cypress VIC64 Chip set

4.2 The C80 Processor

The TMS320C80 is a single chip Multiple Instruction/Multiple Data parallel processor. It consists of a 32-bit RISC master processor (MP) with a 100-MFLOP IEEE floating point unit and four 32 bit parallel processing DSPs (PPs). Within the C80 is 50k Bytes of SRAM for instructions, parameters and data. The SRAM is divided into twenty-five 2k Byte segments, five per processor, to allow multiple simultaneous access. A large crossbar switch allows any processor to access any segment of SRAM.

The interface to external memory is via a Transfer Controller (TC) that supports direct connection to DRAMs, VRAMs and SDRAMs. The TC can automatically maintain Instruction and Data Caches for each of the five processors, and can be programmed to transfer blocks of data between on-chip and off-chip memory autonomously.

The C80 also contains a two channel Video Controller (VC). Each channel may be programmed to generate video timing signals (asynchronous to the C80 clock) and to generate VRAM transfer cycles for Video Display or Video Capture.

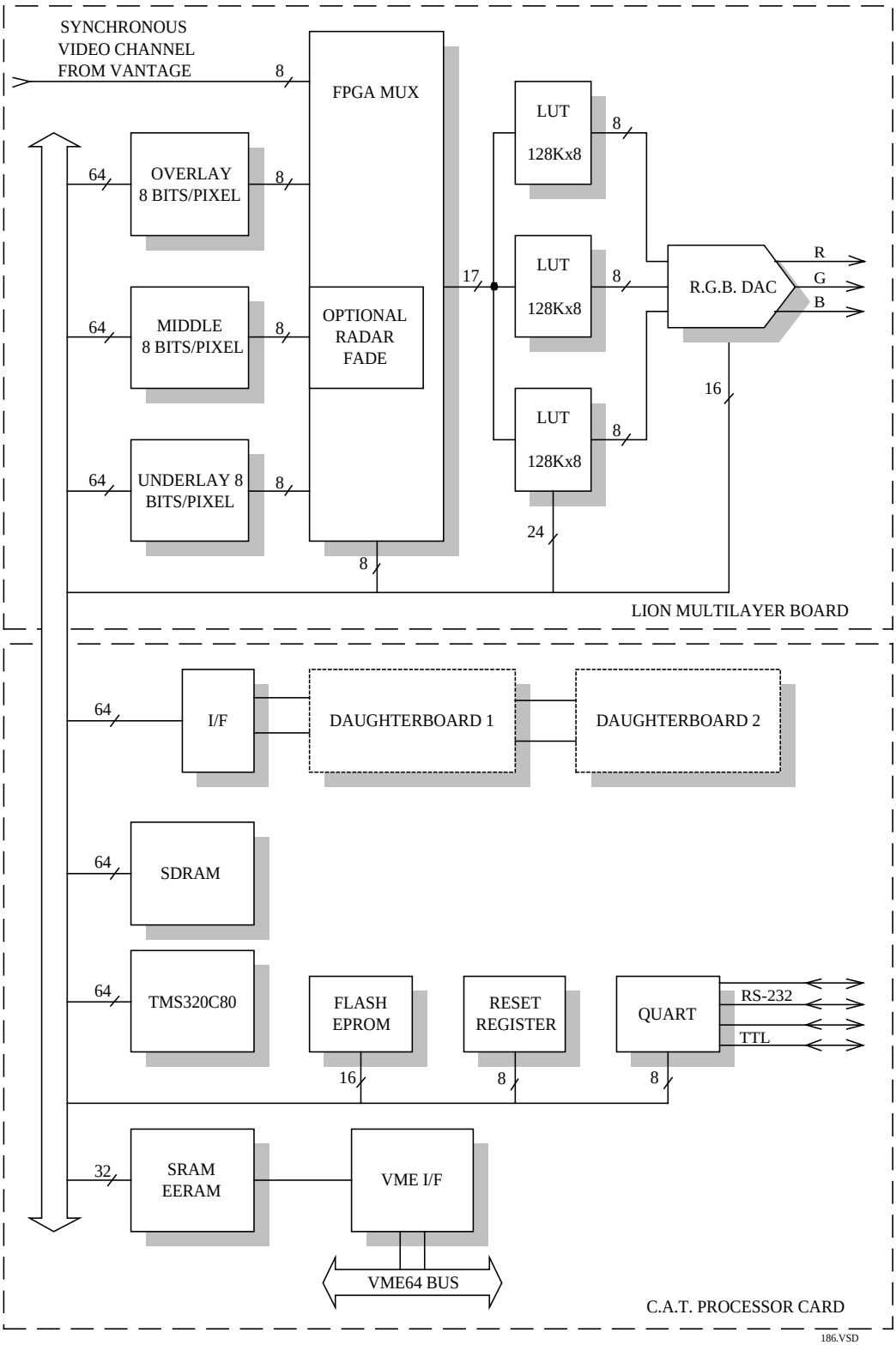

Figure 4.1: C.A.T. with Lion Multilayer Display Board

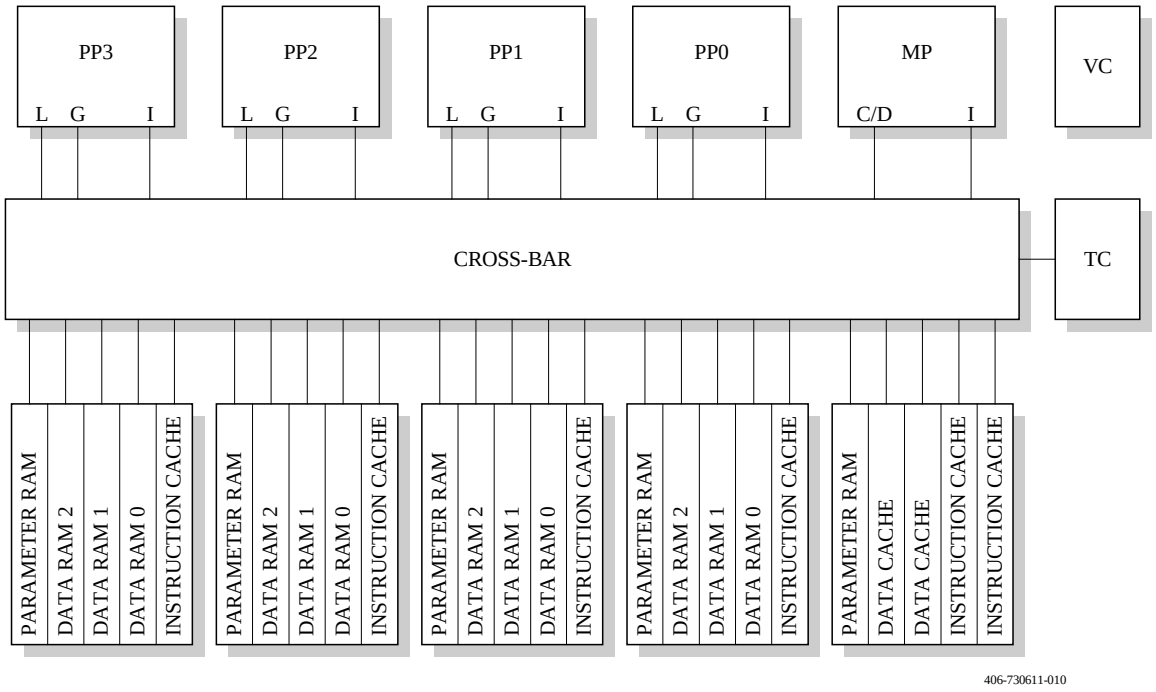

Figure 4.2: C80 Internal Architecture

4.3 The C80 External Memory Interface

The external memory interface of the C80 has 32 address lines, and 64 data lines. However the C80 supports memory devices of 8, 16, 32 or 64 bits; additionally the C80 supports multiplexed row/column address devices with a column address shift facility. This flexibility is achieved by having a three phase bus cycle. Every cycle starts with the C80 outputting the full 32 bit address it wishes to access, the C.A.T. hardware responds to the address by returning twelve configuration signals to the C80 that define:

- Bus Size: 8 bits, 16 bits, 32 bits, 64 bits
- Cycle Type: SDRAM/DRAM, pipelined/non-pipelined, speed
- Address Shift: Row to Column address shift for SDRAMs, VRAMs
- Page Size: Defines boundaries for Burst Transfers

The configuration signals define the capabilities of the memory at the address defined by the C80. At this stage the C.A.T. hardware has no idea whether the C80 wishes to transfer one byte or several hundred bytes.

On receipt of the configuration signals the C80 goes on to the second phase of the cycle and asserts RAS. The timing for the assertion of RAS depends on the Cycle Type selected. For the final phase of the bus cycle, the address output by the C80 is shifted as defined by the configuration signals and then CAS is asserted. If multiple data transfers are required, and the Page Size definitions allow, the C80 will continue to output new column addresses until either all the required data is transferred, or until a page boundary is hit.

Note that, once the configuration signals have been returned to the C80, the bus timing is entirely controlled by the C80. However, for slower devices even the C80's slowest bus cycle is too fast, in these instances the cycle is extended - under the control of the C.A.T. hardware by negating the C80 READY signal.

The C80 does not have any kind of acknowledge signal at the end of a bus cycle to confirm completion of the transfer. The C80 does however have a RETRY pin that can be used to defer an access to memory that is temporarily busy.

When reading data, the C80 always reads the full width of memory, i.e. if the C80 wishes to read one byte from a 64 bit memory, it will read all eight bytes and discard the unwanted data internally. When writing to external memory the C80 uses its eight individual CAS/DQM lines as byte write enables, so that it may write a single byte of data even to 64 bit memory.

In addition to the row address the C80 also outputs a STATUS code at the start of each cycle. This describes the forthcoming cycle and is used by the C.A.T. hardware to detect special cycles, such as Block Write and cycles with an invalid address such as Refresh.

4.3.1 C80 Memory Configuration Inputs

At the start of each Bus Cycle the C80 puts out the new row address and STATUS. In response the C.A.T. hardware must tell the C80 all about the type of memory it is trying to access - this is done by the C80 configuration inputs.

The Configuration inputs define:

BS1, BS0 - bus size:

BS1..0	Bus Size	Block Write
0 0	8 bits	Simulated
0 1	16 bits	*reserved*
1 0	32 bits	4x
1 1	64 bits	8x

These define the width of memory the C80 is accessing, NOT the amount of data the C80 is transferring. If the C80 Master Processor executes a 32 bit read from 8 bit memory, the C80 Transfer Controller will automatically read four sequential bytes and return a 32 bit quantity to the Master Processor.

During a Block Write cycle the C80 assumes a 64 bit bus size, and instead BS1, BS0 define the type of Block Write to be used with the selected memory. On the C.A.T. main board, only the SDRAM is 64 bits wide and for this BS1, BS0 = 0 (Simulated) during block writes. For Display and Application Daughterboard Block Write cycles BS1, 0 are defined by bits in the *Reset Register* (see below).

CT2..CT0 - Cycle Timing

CT2..0	Cycle Timing
0 0 0	Pipelined SDRAM, Burst length 1, CAS latency 2
0 0 1	Pipelined SDRAM, Burst length 1, CAS latency 3
0 1 0	Interleaved SDRAM, Burst length 2, CAS latency 2
0 1 1	Interleaved SDRAM, Burst length 2, CAS latency 3
1 0 0	Pipelined 1 Cycle/Column
1 0 1	Non-pipelined 1 Cycle/Column
1 1 0	2 Cycles/Column
1 1 1	3 Cycles/Column

For the C.A.T. main board and Display Board, only CT = 1, CT = 6 and CT = 7 are used. Application Daughterboards could potentially use any cycle type.

PS3..PS0 - Page Size

Page Size is relevant to block transfers of data between the C80 and its memory. When transferring a block of data, the C80 generates one row address cycle followed by as many column address cycles as are required to transfer the data. However if the transfer crosses a page boundary, a new row address cycle will be generated and the transfer will continue.

PS3..0	Page Size/Bytes
0 0 0 0	64
0 0 0 1	128
0 0 1 0	256
0 0 1 1	512
0 1 0 0	1k
0 1 0 1	256k
0 1 1 0	512k
0 1 1 1	1M
1 0 0 0	Disabled
1 0 0 1	2k
1 0 1 0	4k
1 0 1 1	8k
1 1 0 0	16k
1 1 0 1	32k

PS3..0	Page Size/Bytes
1 1 1 0	64k
1 1 1 1	128k

For most devices on the C.A.T. PS = 8, i.e. disabled. On the C.A.T. main board only the SDRAM and the VME shared SRAM have page mode enabled. On the Display Board only the VRAM has page mode enabled.

AS2..AS0 - Address Shift

AS2..0	Ram Type
0 0 0	Static
0 0 1	64k x n
0 1 0	256k x n
0 1 1	1M x n
1 0 0	4M x n
1 0 1	16M x n
1 1 0	64M x n
1 1 1	256M x n

AS2..0 set the amount of address shift between the row and column addresses for multiplexed address RAMs such as DRAM's. If AS = 0, then no shift is introduced and the row and column address are the same. On the C.A.T. main board only the SDRAM has a non-zero address shift.

4.3.2 C80 Status Codes

At the start of each Bus Cycle the C80 outputs a code on STATUS5... STATUS0 to describe the cycle. The codes define Read or Write, and special cycles such as Refresh or Block Write. They are used by the C.A.T. hardware in addition to the row address for memory decoding and for defining the memory configuration. The special cycles detected are:

STATUS 5..0	Cycle Type
0 0 0 0 1 0	Refresh
0 0 0 0 1 1	SDRAM DCAB
0 0 1 1 0 0	SDRAM MRS
0 0 1 0 0 1	Block Write
0 1 0 0 0 0	Frame 0 Read Transfer
0 1 0 0 1 0	Frame 0 split Read Transfer

STATUS 5..0	Cycle Type
1 0 0 1 1 0	XPT1 PDT Read
1 0 0 1 1 1	XPT1 PDT Write
1 0 1 0 1 0	XPT2 PDT Read
1 0 1 0 1 1	XPT2 PDT Write
1 0 1 1 1 0	XPT3 PDT Read
1 0 1 1 1 1	XPT3 PDT Write
1 1 0 0 1 0	XPT4/SAM1 PDT Read
1 1 0 0 1 1	XPT4/SAM1 PDT Write

4.3.3 C80 Ready

For slow devices, and for VME Master cycles, it is necessary to extend the basic C80 bus cycle. This is achieved by negating the READY line to the C80. For the following on-board devices, READY is negated for 100ns during the CAS phase of the bus cycle:

- Flash EPROM
- QUART
- VME EERAM and Registers
- Display Board FPGAs
- Display Board LUTs
- Display Board DAC

The timer that generates the READY signal is clocked from the C80 CLK-OUT, and is synchronised to the start of each bus cycle. For this reason wait states are limited to single data transfer bus cycles, and are not inserted into fast-page and burst transfers.

For VME Master cycles READY is negated until the end of the VME Bus cycle. READY may also be negated by the Application Daughterboards.

4.3.4 C80 Retry

The shared memory on the VME interface, and any memory on the Application Daughterboards may not always be immediately available to the C80. To support this the C.A.T. hardware will assert the C80 RETRY signal when the C80 tries to access memory that is not available. When RETRY is asserted, the C80 abandons its current bus cycle and starts a new cycle. The C80 will continue to attempt to access the shared memory until the memory becomes available, however during the time this is happening any higher priority bus accesses that occur will be allowed precedence.

4.3.5 C80 Bus Cycles

The C80 can produce a large variety of bus cycles. In the TMS320C80 data sheet there are 45 timing diagrams for different bus cycles. On a C.A.T. plus Display Board only 11 different cycles types are used and these split into two groups - Normal access (6 cycles), SDRAM access (5 Cycles). The cycle types are:

Normal: Read
Write
Load Colour Register
Block Write
Read Transfer (RAM -> SAM)
Refresh (CAS before RAS)

SDRAM: Read
Write
DCAB (a special pre-charge cycle performed once after reset)
MRS (Mode Register Set, performed once after reset)
Refresh

4.4 Bus Structure

The C.A.T. bus structure has been carefully designed with the following objectives:

- Maximum speed - by limiting the load on each data line
- Decoupling Daughterboards - daughterboards should not load on-board data lines
- Correct 3V3 to 5V translation - the C80 has 3V3 I/O

The C.A.T. bus structure is shown in Figure 4.3.

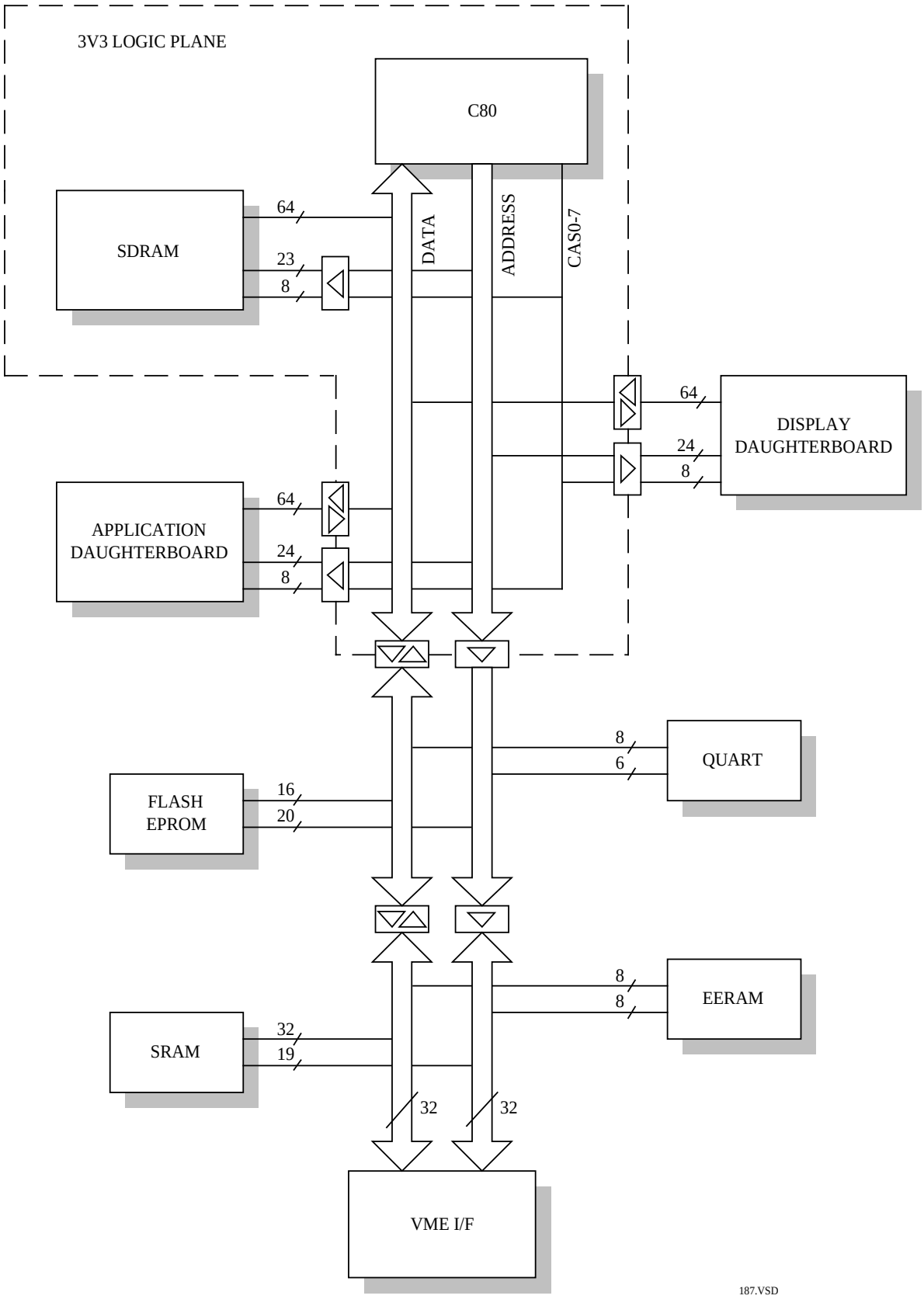

Figure 4.3: C.A.T. Internal Bus Structure

The C80 can be configured for Big-Endian or Little-Endian operation. The C.A.T. hardware is designed for Big-Endian operation to match VME, and the C80 is configured to boot in this mode.

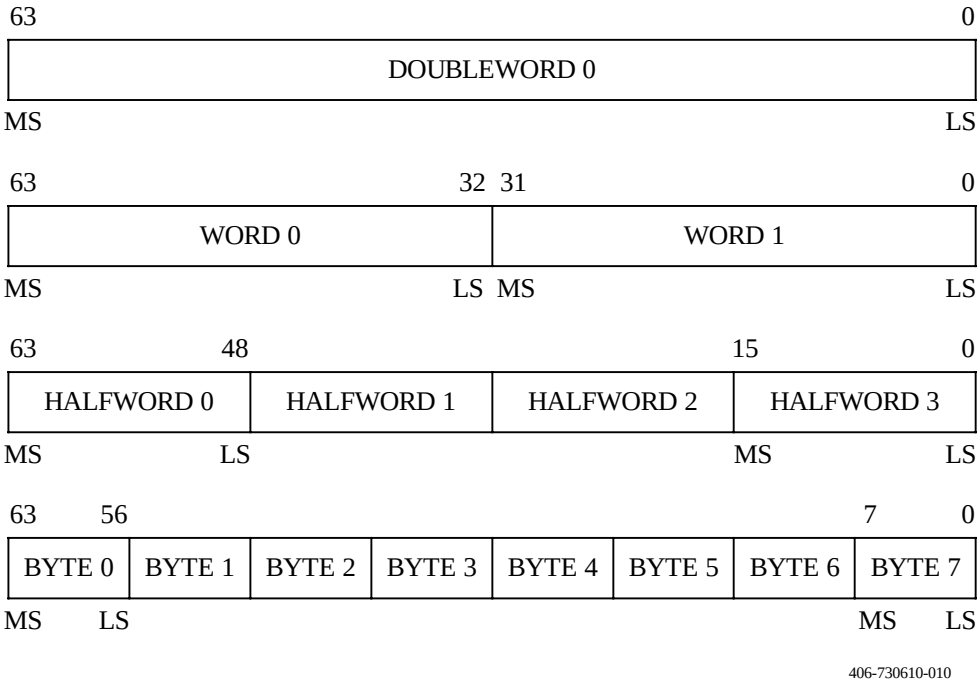

Figure 4.4: Big-Endian Bus Structure

4.5 Synchronous DRAM

The Synchronous DRAM is the C80's main instruction and data memory. The memory is 64 bits wide and is operated synchronously to the C80's 50MHz clock. Burst transfers between the C80 and the SDRAM occur at 400MBytes/sec.

Either four 16M bit devices may be fitted to locations E9, E10, H9, H10 on the top of the PCB to give 8MBytes of memory, or four 64M bit devices may be fitted to locations E12, E13, H12, H13 on the bottom of the PCB to give 32MBytes of memory.

The memory is implemented with all eight CAS/DQM lines so that individual bytes may be written, thus software may access the memory as **bytes, half words, words** or **double words.**

The SDRAMs have similar control signals and similar timing parameters to conventional DRAMs, however the control and address signals are treated as synchronous inputs and are only sampled when the SDRAM's chip select input is active. On the C.A.T. the SDRAM CE line is common to all four SDRAMs and is activated by decoding the C80 row address for normal read/writes, or by decoding the C80 STATUS for special SDRAM cycles.

The SDRAM is operated as sequential memory (not interleaved), and supports a burst length of one i.e. once a page of SDRAM has been activated the C80 may access a random sequence of addresses within the page on each clock edge. Only one bank is active at a time. To satisfy the C80's timing requirements the SDRAM is operated with a CAS read latency of 3 clocks.

Both types of SDRAMs used on the C.A.T. have 256 column locations per row. Thus the lower eight address lines to the SDRAMs are used for both Row and Column address. The next two address lines (A8, A9) are only used for Row addresses. The next highest address line (A10) is used by the SDRAM as a row address input, and later in the cycle as a control line. On the C.A.T. it is always set low after the row address phase. The higher address lines on the SDRAMs are bank selects, and on the C.A.T. these are latched during the row address phase and held for the rest of the cycle.

C80 Address Line	SDRAM Address pin	Use
C80_ADDR11	A0	Row/Column address
..	..	..
C80_ADDR18	A7	Row/Column address
C80_ADDR19	A8	Row address only
C80_ADDR20	A9	Row address only
C80_ADDR21	A10	Row address/control
C80_ADDR22	A11	Row address (64M), Bank select (16M)
C80_ADDR23	A12/BS0	Bank select (64M only)
C80_ADDR24	A13/BS1	Bank select (64M only)

Note: SDRAM address lines A0 to A9 are buffered versions of the C80 address outputs, the higher address/bank selects are generated by the FPGA P701_1.

4.6 Flash EPROM

The C.A.T. has two 32 pin socketed, PLCC sites for EPROMs. Both must be fitted to give a 16 bit wide interface. The EPROMs used on C.A.T. have 120ns access time, so READY is negated to extend the bus cycle. Whilst OTP EPROMs could be fitted to these sites most C.A.T.s will have Flash memory fitted. The sites are +5V only, so either 29F010s or 29F040s can be fitted to give 256K or 1MByte of memory.

The sites have individual write enables so the two sites can be independently erased or programmed. There is no hardware write protect on the Flash memory, but the sequence of writes required to erase or program a Flash EPROM is unlikely to occur accidentally.

After a C80 reset, the EPROMs are enabled to the top page of memory to allow the C80 to read its reset vector out of the top locations of the EPROM. Once code is executing, the C80 may disable the EPROM image at the top of memory (using the *Reset Register*) to free up the top 1G Byte of address space for VME access.

4.7 QUART

The Quad UART is a Phillips SCN26C94 device with four programmable UARTs and two Counter-Timers. The QUART operates from its own crystal to give programmable baud rates from 50 to 38.4k baud. Two of the four channels are brought out as RS-232 on the front panel Micro D-Type. The other two channels are routed to an on-board dedicated Keyboard/Mouse connector.

4.8 Reset Register

This is a simple 8 bit I/O register that is cleared each time the C80 is reset. It is used to control various hardware features on the board. Four bits are used to reset the VME I/F Chip set and the Application Daughterboards under S/W control. Two bits are used to set the type of Block Write to be used when accessing the Display and Application Daughter boards. Other bits are the REMAP bit that enables the EPROMs in the top memory page and the bit that controls the S/W LED on the front panel.

4.9 VME I/F and Shared Memory

The VME I/F is implemented using the Cypress VIC64 Chip set and shared memory. This allows:

- VME slave Extended Address accesses to shared SRAM (inc. BLT)
- VME slave Short Address accesses to shared EERAM
- the C80 to directly access external VME memory
- the Chip set to BLT data between shared SRAM and external VME memory
- VME interrupts to be generated
- VME interrupts to be handled
- the C.A.T. to be the Slot 1 System Controller

The Chip set and shared memory operate on a local bus that is separate from the main C80 bus. This allows VME slave accesses to the shared memory to be performed without affecting the C80's operation. When the C80 needs to access the shared memory, or if it wants to read or write directly to VME memory, it must have ownership of the local bus. If the bus is busy RETRY is asserted to the C80. This will ‘hold off’ the C80 until the local bus is free, but it will not lock the C80 bus and video screen refreshes will continue.

When the C80 directly accesses external VME memory, the READY line to the C80 is negated to extend the C80's memory cycle until the VME bus cycle has completed. As VME bus cycles are of indeterminate length, there is a serious risk that video screen refreshes will be lost during direct accesses of external VME memory - application code must be written with this in mind.

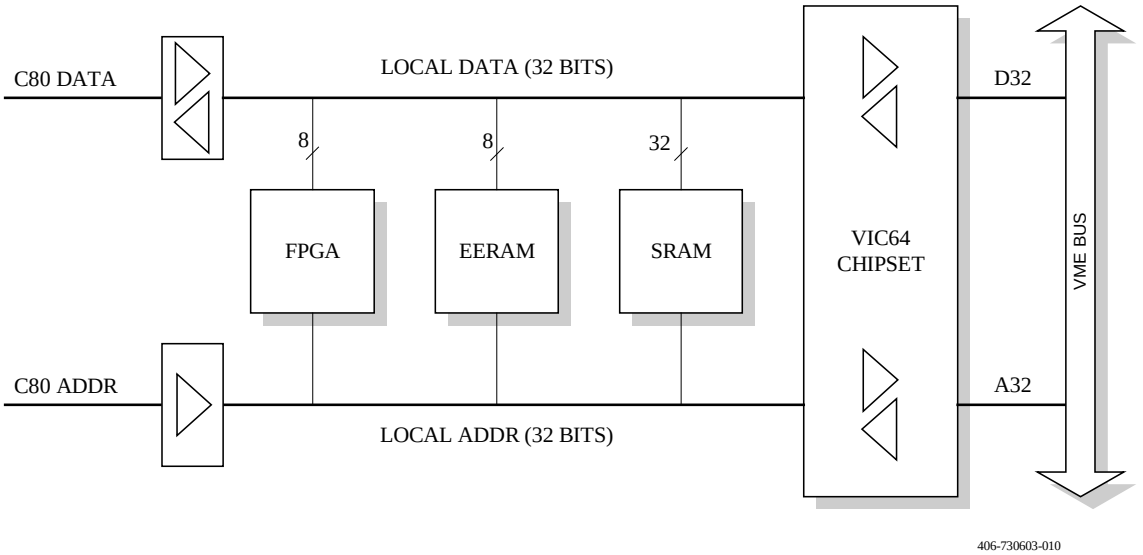

Figure 4.5: VME Interface and Shared Memory

When a block of data needs to be moved between the C.A.T. card and an external VME slave, this can be achieved by programming the VIC64 to move the data between the VME slave and the on-board shared SRAM. This is called a Local DMA Operation, and once initiated requires no C80 intervention.

4.9.1 Shared Memory and Registers

The 512k Bytes of shared SRAM on the VME interface Local Bus is 32 bits wide and supports block transfers to VME and the C80. The SRAM is mapped into VME Extended Address space. Its position in the VME memory map is set by software.

The EERAM and FPGA registers form a 256 Byte block that is mapped into VME Short Address space. Its location in Short Address space is set by Software. The EERAM stores the normal board identification parameters, and additional user defined parameters for initialising the VME interface and the Display Board.

The FPGA contains an assortment of registers connected with the VME interface. Two of the registers allow an external VME Master to interrupt or reset the C80. Other registers allow the C80 to check for bus related interrupts or to read back the User Links.

4.10 Display Board I/F

The Display Board Interface is quite specific as the Bus Size and Cycle Type for each area of memory on the Display Board is defined in the memory configuration FPGA on the C.A.T. main board. However as each memory area is defined for a 16 MByte block there is plenty of room for expansion.

The defined memory blocks are:

- Three separate VRAM Framestores, 64 Bits wide, 60ns Fast Page, 4k Page Size
- Video FPGA registers, 8 bits wide, 300ns single cycle
- Colour LUT, 32 bits wide, 300ns single cycle
- DAC, 16 bits wide, 300ns single cycle

Each of these has an associated RAS or CE signal on the interface. All 64 data bits and 24 address lines are routed to the Display Board via dedicated buffers.

4.11 Application Daughterboard I/F

The Application Daughterboard I/F is more flexible than the Display Board interface. Each Daughterboard has two decodes, a fixed, slow 8 bit decode and a flexible data decode. The C80 memory configuration parameters for the data decode are defined by pins on the interface. Thus the C80 will be automatically configured to access the Daughterboard data memory. The slow, 8 bit decode, is intended for board identification and control. The first two locations of this memory block should always return the Primagraphics part number of the Daughterboard when read by the C80.

Each of the two Daughterboards also has:

- a dedicated interrupt line to the C80
- a READY line to extend bus cycles
- REQUEST and GRANT lines to arbitrate for ownership shared memory
- PDT_REQ and PDT_ACK lines to control Peripheral Device Packet Transfers

The REQUEST and GRANT lines allow the Daughterboard memory to be shared between the C80 and the Daughterboard circuitry. When both wish to access the memory, the Daughterboard has the higher priority.

The PDT_REQ and PDT_ACK lines allow the C80 Transfer Controller to be used as a DMA controller to transfer data from the Daughterboard to other C.A.T. memory.

5. Software Specification

This section describes the programming, use and operation of the C.A.T. hardware. In hardware terms it covers:

- The C80 processor
- The Synchronous DRAM (SDRAM)
- The quad UART (QUART)
- The Reset Register
- The Flash EPROMs
- The VME Interface and Shared memory

In operational terms it covers:

- Booting from EPROM
- Resets and Interrupts
- Serial Communications
- VME Master, Slave and BLT accesses

5.1 The C80 Processor

5.1.1 Reset

The C80 reset is linked to the board reset, and thus when the C80 is reset both the QUART and the Reset Register (see below) are reset. The Reset Register holds the VME Interface and Application Daughterboards reset until written to by software.

A C80 reset will be generated by any of the following:

Power-up (i.e. either +5V or +3V3 being below normal levels)
Front Panel Reset Button
Assertion of VME Bus SYSRESET by another VME card
A write to the C.A.T. short address Mailbox Reset

The C.A.T. hardware is configured so that the C80 always runs after reset. During reset the C80 is set to Big Endian bus operation.

5.1.2 Interrupts

The four C80 interrupt lines are each dedicated to a single function:

External Interrupt 1 - Application Daughterboard 1
External Interrupt 2 - Application Daughterboard 2
External Interrupt 3 - VME Interface
External Interrupt 4 - QUART

Use of External Interrupts 1 and 2 are dependent on the Application Daughterboards fitted and will be described in the individual daughterboard documentation.

External Interrupt 3 has numerous sources within the VME Interface, its use is described in the Software Specification for the VME I/F, see section 5.6.11 **Interrupts to the C80**.

External Interrupt 4 is a direct connection from the QUART interrupt output. Thus this interrupt is enabled and cleared by writing to the appropriate registers within the QUART.

5.1.3 External Packet Requests

Only the Application Daughterboards can generate External Packet Requests (XPT's), so the use of these is not described here. However, the C.A.T. hardware defines that XPT0 and XPT1 are dedicated to Application Daughterboard 1, and that XPT2 and XPT3 are dedicated to application Daughterboard 2.

5.1.4 Transfer Controller Registers

The four Transfer Controller registers should be programmed as follows:

REFCNTL - This sets the dynamic memory refresh rate for the C.A.T. card SDRAM and Display Board VRAM. The C.A.T. card refreshes VRAM and SDRAM alternately, and thus a refresh is required every 7.5us. The refresh counter RPARLD is not used. Thus the register should be set to 0x0000 0186.

PTMIN - This sets the minimum duration for a Packet Transfer burst, and thus defines the *maximum* time that a equal priority bus access can be delayed. Increasing this number will boost overall sustained transfer rates at the expense of latency. Packet transfer bursts are interrupted for memory and display refresh cycles regardless of this setting.

PTMAX - This sets the maximum duration for a Packet Transfer burst, and thus defines the *maximum* time that a lower priority bus access can be delayed. This will normally depend on the application.

FLTSTS - Is read only.

5.1.5 Video Controllers

The C80 contains two Video Controllers each with numerous registers within the C80's internal address space. Video Controller 0 is dedicated to the Display Board, and thus details of programming these registers are covered in the appropriate Display Board Technical Manual. If no Display Board is fitted these registers need not be programmed.

Video Controller 1 signals are routed to the Application Daughterboard interface, again programming details for these registers will be in the appropriate Application Daughterboard Technical Manual.

5.1.6 Initialisation Sequence

After a hardware reset of the C80 the following sequence of events should occur:

Hardware Reset
|
C80 reads Reset Vectors from Flash EPROM @ FFFF FFF0
|
C80 begins Execution from Flash EPROM @ 01FX XXXX
|
EPROM Boot code sets REMAP bit in Reset Register.
|
EPROM Boot code loads REFCNTL (T.C. register) to set SDRAM refresh rate.
|
EPROM Boot code initialises C.A.T. hardware.
|
EPROM Boot code executes Embedded Application
OR
EPROM Boot code awaits Application download via VME

5.2 Synchronous DRAM

The SDRAM on the C.A.T. is the main code and data memory for the C80. It is contiguous 64 bits wide memory with individual byte write enables, and may be accessed as **bytes, half words, words** or **double words**.

The SDRAM supports block transfers of data to/from the C80 at 400 Mbytes/sec. The MP executes instructions directly from an on-chip instruction cache which is loaded in 64 byte bursts. Filling 64 bytes of instruction cache from EPROM will take 8us, from SDRAM it will take 300ns. It can be seen that there is a significant speed advantage in executing from SDRAM.

5.3 QUART

The QUART is a Philips SCN26C94. This is a quad version of the dual UART used on other Primagraphics cards. The device contains four UARTs and two 16 bit counter/timers. Each UART is programmable for - baud rate, number of data bits, parity, number of stop bits.

Channels A and B are for general purpose RS-232 I/O, and are routed to a 9 way micro-D-type connector on the C.A.T. card front panel. Channel A has CTS, DTR and RTS handshaking signals on the connector.

Channels C and D are connected to a dedicated Keyboard/Mouse connector on the C.A.T. card. This is intended to connect to a ‘SUN’ style keyboard and mouse. There are no hardware handshake lines on this interface.

The QUART is mapped into the C80's memory as an eight bit device, and must be accessed with **byte** reads and writes. The QUART has nearly fifty registers that are accessible to the C80 as a contiguous series of bytes above the QUART base address. The register definitions and the operation of the QUART are as described in the SCN26C94 Product Specification (data sheet) from Philips.

The QUART can be programmed to interrupt the C80 under various circumstances, and the QUART has a complex vectored interrupt scheme to speed up the interrupt service routine. Unfortunately, the C80 does not support vectored interrupts. The QUART is on a dedicated interrupt line to the C80 (Lint4), but the interrupt service routine will have to determine the exact source of the interrupt within the QUART.

None of the general purpose I/O pins on the QUART are connected, except those that are used as CTS, RTS, DTR on channel A.

5.4 Reset Register

The Reset Register is a simple 8 bit R/W register that is used to reset the VME interface and the Application Daughterboards. It should be accessed with **byte** reads and writes. All 8 bits of the register are set to 0 during a hardware reset of the C80.

D7	D6	D5	D4	D3	D2	D1	D0
Remap	BL WR1	BL WR0	S/W LED	App DB2 reset	App DB1 reset	VIC64 global rst	VIC64 reset

D0 - 1 These bits control the reset of the VIC64. Their use is described in the Software Specification for the VME I/F section **5.6.8 VIC64 and VME Resets**

D2 - 3 Control the resets to the Application Daughterboards. Lo = reset

D4 Controls the red LED on the C.A.T. Card front. Lo = ON

D5 - 6 These bits set the type of Block Write to be used with the Display Board and Application Daughterboard - see below

D7 Remap bit - see below. Lo = enable EPROMs at top of memory

The C80 can perform three different types of Block Write - x4, x8 and simulated. These two bits tell the C80 which kind of Block Write is to be used when accessing the Display Board frame stores and the data memory on the Application Daughterboards. The C.A.T. firmware must set these bits to match the capabilities of the memory devices being accessed.

D6	D5	Type of Block Write
0	0	Simulated
0	1	*Reserved*
1	0	x4
1	1	x8

The Remap bit is used to enable/disable the EPROMs at the top of memory. After a reset, the C80 Master Processor will go to the top of memory for its reset vectors, and with the REMAP bit set to 0 the EPROMs will be enabled to provide them. Once the Master Processor is executing code from EPROM at the normal address, the REMAP bit may be set to 1 to disable the EPROMs from the top of memory. This frees up the top 1 GByte of memory space for VME access.

5.5 Flash EPROMs

The C.A.T. card has two 32 pin PLCC sockets for Flash EPROMs. Either 1 MBit or 4 MBit devices may be fitted to give either 256k Bytes or 1M Bytes of storage. The two eight bit devices are configured as sixteen bits.

The EPROMs are used to store both the C80 boot code and the C80 reset vectors, and for this reason they are decoded to two addresses within the C80 memory map. It is intended that the boot code will be linked to run from the block of memory based at 0x1F00 0000.

The secondary address decode for the EPROMs is 0xFF00 0000 - this is used for the C80 reset vectors, and is only intended to be enabled for a short time after reset. The secondary address decode is enabled by a hardware reset of the C80; after the C80 has booted it should be disabled by software (by setting the REMAP bit in the Reset Register) to free up the top of memory for VME accesses.

The two Flash EPROMs have individual write enables so that they may be re-programmed individually. Flash EPROM programming algorithms vary from device to device and manufacturer to manufacturer. The initial devices fitted to the board will be AMD 29F010 and 29F040 devices. There is no hardware write protect on the Flash EPROMs, however the sequence of writes to the Flash to put it into the ERASE or WRITE mode is unlikely to occur accidentally. The Flash EPROMs cannot be accessed during the programming operation, so the loader program will have to be resident in main memory - most likely this will be an application program down loaded over VME.

The algorithms for ERASE and PROGRAM of the 29F010 and 29F040 are given in the AMD data sheets for these devices. (E.g. AMD Flash Memory Products Data Book 1994/5 pages 1-15 and 1-91).

5.6 VME Interface

The C80 accesses the VME Interface Local Bus at four different points in its memory map:

- Shared SRAM
- EERAM, FPGA registers and VIC64 registers
- Short Address Space VME Window
- Extended Address VME Window

When the C80 tries to access any of these addresses the C.A.T. hardware will arbitrate for ownership of the interface Local Bus. If the Local Bus is in use the C80 will be RETRY'ed. The C80 will continue to retry the access until the Local Bus is free, but during this time the C80 bus is NOT locked - memory refreshes, screen refreshes and packet transfers will continue.

However, once the C80 has ownership of the Local bus for a VME Master cycle, i.e. Short or Extended VME access, the C80 READY line will be negated and the processor will be held with the bus locked, until the VME bus cycle completes. During this time there will be no memory or screen refreshes.

5.6.1 Use of the Interface

In large disk based systems with a VME Host CPU the C.A.T. card will only be used as a VME Slave and Interrupter. The external Host will load commands and data into the C80 card's shared memory and will then interrupt the C80 via the mailbox interrupt. The C80 will respond by writing return messages into the shared memory and then interrupting the Host (via VME Interrupt request).

In small diskless systems, the C80 might be the only bus master, and it may need to initialise other boards and act as an Interrupt Handler. In this case the C80 card must perform read and write operations over VME to Slave cards - this can be done, but care is needed. In virtually all applications the C80 card will be driving a video display from its on-board video RAM. To maintain the display, the C80 needs to generate a constant stream of video RAM DRAM->SAM transfer cycles (with a high resolution display these may be needed every 7us). When the C80 performs a master read or write over VME there is the potential that the C80 will be halted until the cycle completes. If the external card responds too slowly then video RAM refreshes will be missed and the video display will flicker.

There are two situations where the C80 is halted by the VME I/F in this fashion, firstly ALL read cycles where the C80 is Bus Master (including IACK cycles), secondly when a write operation is attempted before the previous 'Post Write' cycle has completed. Most VME slave cards will respond in less than 7us, so this need not be a problem *provided* that Bus Ownership is obtained before the read/write operation is attempted. Bus Ownership can be obtained by setting the VIC64 RCR (*release control register*) to BCAP (Bus CAPture), and polling D7 of the VIC64 BESR (*bus error/status register*) to confirm ownership.

5.6.2 Programming the Interface

All of the operational parameters of the interface are software controlled rather than defined by Links or Jumpers. The parameters could be hard coded into the application software, but to improve portability it is suggested that application code uses the following parameters from the EERAM. These will be loaded into the EERAM to suit the users system configuration.

The parameters required to define the interface are:

Slave short address base	(0xnn)
Slave extended address base	(0xnnnn)
Interrupter Level	(Disabled, 1,2,3,4,5,6,7)
Interrupt Handler Level	(Disabled, 1,2,3,4,5,6,7)
BREQ level	(0,1,2,3)
Bus Release	(ROR/RWD/ROC)
Fairness	(On/Off)
Slot 1 Arbitration	(Priority/Round Robin)
Slot 1 Transfer Timeout	(Off, 4, 16, 32, 64, 128, 256, 512 us)

See **Appendix B** for the standard EERAM definition.

5.6.3 Shared Memory

The shared memory on the VME Interface is divided into two blocks, a large 512k Byte block of fast SRAM, and a smaller block of EERAM and registers.

The SRAM is used to pass packets of data and commands between the C80 and an external Host CPU. The SRAM is 32 bits wide and supports block transfers, both over VME and to the C80. The SRAM may be accessed as **bytes, half words** or **words**.

For VME Slave accesses the SRAM is positioned in VME Extended Address space by programming the VME Slave Page register (see below).

The EERAM and registers appear to the C80 as a 512 Byte block, but to VME Short Address space only the first 256 locations are visible. The C80 should always access this area as **bytes**. In addition to the usual board identification parameters, the EERAM will also store parameters for the Video Timing and VME interface. The FPGA registers include ‘Mail box’ registers for interrupting and resetting the C80 (see below).

C80 Address	VME Address	Usage
0x0D00 0000 to 0x0D00 001C	Base + 0x00 to Base + 0x1C	VME I/F FPGA registers
0x0D00 0020 to 0x0D00 00FC	Base + 0x20 to Base + 0xFC	EERAM
0x0D00 0100 to 0x0D00 01FF	N/A	VIC64 Registers

The standard EERAM usage is defined in Appendix B.

5.6.4 Slave Operation

The VIC64 has two slave address decode inputs - Slave Select 0 and 1. The C.A.T. card hardware uses Slave Select 0 for Extended Address space and Slave Select 1 for Short Address space. Short Address operations will access the EERAM and the registers in the VME I/F FPGA. Extended Address operations will access the shared SRAM.

The following parameters are required during the set up of the interface:

Slave Short Address - positions the EERAM in VME Short Address Space.
Slave Extended Address - positions the SRAM in VME Extended Address Space.

Before a Slave access to the C80 card can be performed the following VIC64 registers need to be initialised:

SS0CR0 *(Slave Select 0 Control Reg 0)*
SS0CR1 *(Slave Select 0 Control Reg 1)*
SS1CR0 *(Slave Select 1 Control Reg 0)*
SS1CR1 *(Slave Select 1 Control Reg 1)*
LBTR *(Local Bus Timing Register)*
AMSR *(Address Modifier Source Register)* - Optional AM code

Additionally the *Slave Page Register* also needs to be initialised.

Once these registers are programmed the interface will allow external VME Masters to access the shared memory. Slave accesses can be disabled using the Slave Select Control registers. A hardware reset will clear these registers and thus disable Slave Operations.

The slave interface performance can be boosted by enabling the Slave Post Write facility, Bit 7 of *SS1CR0.*

5.6.5 Master Operation

To minimise the time that the C80 is kept waiting by VME read/write operations it is advised that the Master Post Write facility is enabled and that Bus ownership is obtained using the Bus Capture and Hold (BCAP) facility before read or write operations are performed.

Unlike Motorola processors the C80 does not generate external signals for Byte/Word/LWord transfers. Thus to define the required transfer width the VME Bus Windows in the C80 memory map are duplicated three times - once for byte accesses, once for Word accesses and once for LWord accesses.

For example, the VME Short Address Windows are:

C80 Address	VME Short Address	Transfer Width
0x0E00 0000 to 0x0E00 FFFF	0x0000 to 0xFFFF	Byte (8 bits)
0x0E01 0000 to 0x0E01 FFFE	0x0000 to 0xFFFE	Word (16 bits)
0x0E02 0000 to 0x0E02 FFFC	0x0000 to 0xFFFC	LWord (32 bits)

Thus,

A C80 write to 0E00 0010 will initiate a Byte write to VME address 0010
A C80 write to 0E01 0010 will initiate a Word write to VME address 0010
A C80 write to 0E02 0010 will initiate a LWord write to VME address 0010

The Extended Address Windows are:

C80 Address	VME Extended Address	Transfer Width
0x4000 0000 to 0x7FFF FFFF	0x0000 0000 to 0x3FFF FFFF	Byte (8 bits)
0x8000 0000 to 0xBFFF FFFE	0x0000 0000 to 0x3FFF FFFE	Word (16 bits)
0xC000 0000 to 0xFFFF FFFC	0x0000 0000 to 0x3FFF FFFC	LWord (32 bits)

In this table the lowest 1GByte of VME Extended Address Space is being addressed, however the second, third or fourth GigaByte of VME memory could alternatively be addressed by setting the *Master Page* register in the VME I/F FPGA.

The following parameters are required during the setup of the interface:

BREQ Level (0/1/2/3)
Bus release (ROR/RWD/ROC)
Fairness (On/Off)

The following VIC64 registers must be initialised before a VME Master read or write can be performed:

ARCR *(Arbiter/Requester Configuration Register)* - BREQ level, Fairness
RCR *(Release Control Register)* - ROR/RWD/ROC
SS1CR0 *(Slave Select 1 Control Register 0)* - Bit 6 enables Master Post Write
AMSR *(Address Modifier Source Register)* - Optional AM codes

In addition to initialising the required VIC64 registers the *Master Function* register and the *Master Page* register in the VME I/F FPGA must be set as required (see below).

5.6.6 Master Block Transfers

The VIC64 may be programmed to DMA blocks of data between the local shared SRAM and external VME memory. Once programmed, the VIC64 automatically acquires the bus and controls the data transfer - the C80 is not involved. On completion of the transfer the VIC64 can interrupt the C80 if this is desired. In the VIC64 documentation this is called a 'master block transfer with local DMA'.

The C.A.T. card supports either D32 or D64 block transfers. The VIC64 is normally programmed to perform the transfer as a series of bursts, with pauses in between to allow other VME cards to use the bus. During the pauses (interleave periods), the C.A.T. card can handle single read/write slave accesses without corrupting the block transfer - in the VIC64 documentation this feature is referred to as 'Dual Path Enabled'.

Whilst a BLT with Local DMA is happening the C.A.T. card cannot handle Slave BLT's.

Whilst a BLT with Local DMA is happening the C80 must NOT attempt any Master Read/Writes over VME (including IACK cycles) - or the BLT will be corrupted.

The completion of a BLT can be confirmed by reading the VIC64 DMASR (*DMA Status Register*), or alternatively, the VIC64 DMASICR (*DMA Status Interrupt Control Register*) may be programmed to interrupt the C80 on BLT completion.

The parameters required for a Master BLT are:

VME source/destination address
Local destination/source address
Length of Transfer (Bytes)
Burst Length (Cycles)
Interleave period (multiples of 250ns)

The VIC64 registers that need to be programmed before a transfer are:

BTCR *(Block Transfer Control Register)* - Interleave period/Transfer direction
BTDR *(Block Transfer Definition Register)* - D32/D64
BTLRs *(Block Transfer Length Registers)* - Transfer Length
RCR *(Release Control Register)* - Burst length

The BLT is started by the C80 performing a pseudo-write to VME immediately after the C80 has set Bit 6 of the BTCR. The pseudo-write provides the interface with the external VME address and the local address. To the C80 the pseudo-write is like any other D32 VME Master Write cycle, except that:

The VME address written to is the External VME source/destination address.
The data value written is the Local destination/source address.

The BLT is always initiated with a pseudo-write; the direction of data transfer is defined by bit 4 of the BTCR.

NOTE: On completion of the BLT, Bit 6 of the BTCR should be cleared to 0, or any subsequent Master cycle (including IACK cycles) will initiate another BLT.

During a BLT the VIC64 will grab the local bus, and this will prevent the C80 from accessing the shared memory and VIC64 registers. However if the BLT is programmed to occur in bursts, the C80 can access local resources during the interleave periods. A BLT may be terminated by clearing Bit 6 of the BTCR.

5.6.7 System Controller Function

The System Controller function is enabled during Board Reset if a Jumper is fitted to User Link location D0. This Jumper cannot be read back directly, but a read of the VIC64 ICR *(Interface Configuration Register)* will show if the VIC64 is operating as System Controller.

When operating as System Controller the VIC64 may be programmed to define:

Arbitration Type:	Priority/Round Robin	(Default is Round Robin)
Transfer Timeout:	4/16/32/64/128/256/512us or Off	(Default is 32us)

Arbitration Type is set by the VIC64 ARCR *(Arbiter/Requester Config Register)* bit 7.

The VIC64 includes a fixed 8us Arbitration Timeout that is always active when the VIC64 is acting as the bus arbiter. After issuing a Bus Grant to an external Bus Request, the VIC64 allows 8us for the external card to assert Bus Busy - if the card has not responded in this time the VIC64 asserts its own BBSY signal to clear the bus. The VIC64 may be programmed to interrupt the C80 when such a timeout occurs - see bit 5 of the EGICR *(Error Group Interrupt Control Register).*

When operating as the System Controller, the VIC64 has an optional Transfer Timeout that may be enabled. This starts timing on the assertion of DS1/0[L], and if neither DTACK[L] nor BERR[L] is seen before the end of the timeout period - it will assert BERR[L]. The Timeout period is set by programming the VIC64 TTR *(Transfer Timeout Register).* The period may be set to various times between 4us and 512us.

When the VIC64 is configured as the System Controller, any reset of the C.A.T. card will also cause the VME SYSRESET[L] signal to be asserted. Thus the front panel reset button and the Mailbox Reset will both reset the entire VME rack.

5.6.8 VIC64 and VME Resets

Cypress recommends that the VIC64 is reset by the on-board processor under software control. Thus the reset line to the VIC64 comes from Bit 0 of the Reset Register. It is further recommended, that after power-up the VIC64 is given a Global reset - this is achieved on the C.A.T. card by setting Bit 1 (Global Reset) of the Reset Register before Bit 0 is set.

Note that every time the C80 is reset, the Reset Register is cleared, and the VIC64 is also reset. If the VIC64 is configured as the Slot 1 system controller, then the Global reset following a C80 reset will cause the VIC64 to assert VME SYSRESET[L], and the entire system will be reset.

The C.A.T. card will respond to an externally generated VME SYSRESET[L], but only after the reset to the VIC64 has been removed (i.e. Bit 0 of the Reset Register has been set high).

5.6.9 Interrupter Operation

The VIC64 can be programmed to generate an interrupt on any of the seven VME BUS IRQ lines. The VIC64 has a separate interrupt vector register for each of the interrupt lines. When an external interrupt handler responds to the interrupt by generating an IACK cycle, the value in the vector register is returned to the handler and the interrupt request is cleared. The VIC64 may also be programmed to generate an interrupt to the C80 when it receives the IACK cycle.

Interrupts are generated by writing to the VIC64 VIRSR *(VMEbus Interrupt Request/Status Register).* The vectors are stored in VIVBR1-7 *(VMEbus Interrupt Vector Base Registers).* If it is required that the VIC64 interrupts the C80 when the external VME host acknowledges the interrupt generated by the C.A.T. card - then program the VIC64 VIICR *(VMEbus Interrupter Interrupt Control Register).*

5.6.10 Interrupt Handler Operation

The VIC64 monitors the seven VME BUS IRQ lines and can be programmed to interrupt the C80 when any particular IRQ line is activated. Normally when this occurs the C80 will respond by executing a READ to the appropriate IACK address in the C80 memory map, this will generate an IACK cycle on the bus and acknowledge the interrupt.

There is a separate VIC64 register associated with each VME Interrupt line handler:

VICR1-7 *(VMEbus Interrupt Control Registers)*

The IACK address that is used to acknowledge a particular VME bus interrupt request depends on the ipl value programmed into the VICR register - the C80 must acknowledge the ipl value, not the VME Interrupt Level. (See Interrupts to the C80).

5.6.11 Interrupts to the C80

The VIC64 is designed to work with CPUs that support vectored interrupts, unfortunately the C80 does not. Instead the three bit vector generated by the VIC64 (ipl0, ipl1, ipl2) can be read from the Interrupt Pending register within the VME I/F FPGA. On receipt of an interrupt from the VME I/F the C80 should read this register first to determine the source of the interrupt. Possible sources are:

Mailbox Interrupt - An external VME Master has written to the C80 Mailbox address.
VME Bus Error - VME BERR was asserted whilst C.A.T. was bus Master.
VIC64 - Numerous sources. Use the ipl code to determine which of...

Error Group Interrupt.
Local DMA transfer complete.
VMEbus Interruptor - IACK cycle received.
VMEbus Interrupt Handler - IRQ received.

All of the possible interrupt sources within the VIC64 have an associated register that is programmed with an 'ipl' value. When one of the sources is activated, and an interrupt to the C80 is generated, the source of the interrupt can be quickly determined by reading the ipl value output by the VIC64. When it is desired to acknowledge a VIC64 generated interrupt, the ipl value is used to determine which address the C80 should read:

To acknowledge ipl value...	...read C80 Address
1	0x0C00 0003
2	0x0C00 0005
3	0x0C00 0007
4	0x0C00 0009
5	0x0C00 000B
6	0x0C00 000D
7	0x0C00 000F

5.6.12 VME I/F FPGA Registers

The VME I/F FPGA links the VIC64 to the Local Bus and to the C80 bus. It also contains seven registers that are accessible to both the C80 and to VME slave accesses via Short Address Space. The FPGA registers should be accessed as **Bytes**.

C80 Address	VME Short Address	Register
0x0D00 0000	Base + 0x00	Mailbox Reset
0x0D00 0004	Base + 0x04	Mailbox Interrupt
0x0D00 0008	Base + 0x08	Interrupts Pending
0x0D00 000C	Base + 0x0C	VME Master Function
0x0D00 0010	Base + 0x10	VME Master Page
0x0D00 0014	Base + 0x14	User Links
0x0D00 0018	Base + 0x18	Configuration links
0x0D00 001C	Base + 0x1C	*Reserved*

5.6.12.1 Register Descriptions

Mailbox Reset

A write of 0x01 to this register will initiate a timed, delayed hardware reset of the board. After the write has occurred there is a 1us pause before reset is asserted, after which reset is asserted for 1us.

Mailbox Interrupt

D7	D6	D5	D4	D3	D2	D1	D0
-	-	-	-	-	-	-	Mailbox interrupt

An external VME CPU may set this bit to interrupt the C80. Once set it can only be cleared by the C80 writing 0 to this location. The C80 cannot set this bit.

Interrupts Pending

D7	D6	D5	D4	D3	D2	D1	D0
-	-	Deadlock	Mailbox interrupt	Bus Error interrupt	ipl2	ipl1	ipl0

The VIC64 has numerous internal interrupt sources. Each active source is assigned a 3 bit 'ipl' code by software, and when a source asserts its interrupt this code is output by the VIC64. This register allows the C80 to read the 'ipl' code, and thus determine the source of the interrupt.

Additionally the Mailbox Interrupt status and the Bus Error Interrupt bit can be checked in this register. The Bus Error bit is set whenever a bus cycle generated by the C.A.T. card is terminated in error; if the cause of the error was a deadlock situation, the Deadlock bit will also be set. Bus Error and Deadlock are both cleared by writing 0 to this register.

VME Master Function

D7	D6	D5	D4	D3	D2	D1	D0
-	-	-	-	-	-	FC2	FC1

This register defines the additional parameters needed by the VME I/F when it is performing a Master read or write.

D1- 0 (0, 0) Access User Data
(0, 1) Access User Program
(1, 0) Access Supervisor Data
(1, 1) Access Supervisor Program

VME Master Page Register

D7	D6	D5	D4	D3	D2	D1	D0
-	-	-	-	-	-	Page 1	Page 0

This register defines the two highest address bits when the VME I/F is performing a Master read or write to the VME Extended Address Space. This applies to both single cycles and BLTs.

D1 - 0 (0, 0) Access VME Page 0000 0000 to 3FFF FFFF
(0, 1) Access VME Page 4000 0000 to 7FFF FFFF
(1, 0) Access VME Page 8000 0000 to BFFF FFFF
(1, 1) Access VME Page C000 0000 to FFFF FFFF

User Links

D7	D6	D5	D4	D3	D2	D1	D0
D7	D6	D5	D4	D3	EERAM write protect B	EERAM write protect A	-

There are 8 user links on the board, all but one of them can be read back here. Link D0 sets the VME chipset into SLOT 1 System Controller mode - unfortunately it cannot be read back.

Write protect ‘A’ guards the factory set region of the EERAM. Write protect ‘B’ guards the User defined parameters in the EERAM. The remaining Jumpers have no fixed functions, and can be used by the software to select default set-ups or change operational modes.

D0	Not connected	
D1	EERAM write protect A	Lo = Jumper fitted = write enabled
D2	EERAM write protect B	Lo = Jumper fitted = write enabled
D3	Reserved	Lo = Jumper fitted, Hi = no jumper fitted
D4	Reserved	Lo = Jumper fitted, Hi = no jumper fitted
D5-7	General purpose	Lo = Jumper fitted, Hi = no jumper fitted

With the standard firmware, fitting the Jumper to D3 will cause the initialisation code to use default parameters for the VME interface rather than use those stored in EERAM, fitting the Jumper to D4 will prevent an embedded application from booting.

Config Links

The configuration links are used to signal build standard changes in the hardware to the software. They are set by adding or removing surface mount resistors on the back of the board. The default value read back value for these links is 0x00.

5.6.13 Slave Page Register

The two VME Slave Addresses for the C80 card (Short address and Extended address) are programmed into one 32 bit register on the VME I/F. Additionally there is a mask register associated with the slave address register that defines which address lines are to be included in the address comparison.

Page Reg	Function	A31.....A24	A23.....A16	A15.....A8	A7.....A0
0x0B00 0004	Address	Extended (hi)	Extended (lo)	Short	not used
0x0B00 0000	Mask	Extended (hi)	Extended (lo)	Short	not used

The Short Address Space decoding for the C80 card is set by data bits 8 to 15 of the address and mask registers. Normally bits 8 to 15 of the Mask register will be set to 0, so that the C.A.T. Card occupies 256 locations of Short Address space.

The Extended Address Space decode for the C80 card is set by data bits 16 to 31 of the address and mask registers. The normal value for mask bits 16 to 31 is 0x0007 as this maps all of the 512k of shared SRAM onto VME. However changing the mask could reduce this to 256k, 128k or 64k.

The Short and Extended Address decode and mask values must be combined and written as 32 bit quantities. If it is wished to disable one of the slave address spaces, then this should be done using the VIC64 *Slave Select Control Registers*. The Page registers cannot be read back, but can be reloaded at any time, however slave accesses should be disabled by writing to the appropriate register in the VIC64 before this is done.

The registers **must** be loaded in the correct sequence: Address register first, followed by Mask register, as the act of loading the Address register clears the Mask register.

5.6.14 VIC64 Registers

The VIC64 registers are only accessible to the C80, in a 256 byte block from 0x0D00 0100 to 0x0D00 01FF. The VIC64 has eight data lines connected to D0 - D7, thus the VIC64 registers appear on every fourth byte address starting at address 0x0D00 0103...

The following table lists all of the VIC64 registers their addresses and their mnemonics. The final four columns indicate if the register is used in **M**aster transfers, **S**lave transfers, **B**lock transfers or **I**nterrupts. Registers in bold type are described below.

Address		Description	M	S	B	I
0x0D00 0103	**VIICR**	VME bus Interrupter Int Con Reg				*
0x0D00 0107-1F	**VICR1-7**	VME bus interrupt (Handler) Con Regs				*
0x0D00 0123	**DMASICR**	DMA Status Int Con Register			*	
0x0D00 0127-3F	LICR1-7	Local Interrupt Con Regs (not used)				
0x0D00 0143	ICGSICR	ICGS Interrupt Con Reg (not used)				

Address		Description	M	S	E	I
0x0D00 0147	ICMSICR	ICMS Interrupt Con Reg (not used)				
0x0D00 014B	**EGICR**	Error Group Int Con Reg				
0x0D00 014F	ICGSVBR	ICGS Vector Base Reg (not used)				
0x0D00 0153	ICMSVBR	ICMS Vector Base Reg (not used)				
0x0D00 0157	LIVBR	Local Interrupt Vector Reg (not used)				
0x0D00 015B	EGIVBR	Error Group Interrupt Vector (not used)				
0x0D00 015F	ICSR	Interprocessor Comms Switch (not used)				
0x0D00 0163-73	ICR0-4	Interprocessor Comms Regs (not used)				
0x0D00 0177	ICR5	Interproc Comm, Chip Version				
0x0D00 017B-7F	ICR6,7	Interprocessor Comms Regs (not used)				
0x0D00 0183	**VIRSR**	VMEbus interrupt Request Status Reg				*
0x0D00 0187-9F	**VIVBR1-7**	VMEbus Interrupt Vector Reg (IRQn)				*
0x0D00 01A3	**TTR**	Transfer Timeout Reg	*		*	*
0x0D00 01A7	**LBTR**	Local Bus Timing Reg		*	*	
0x0D00 01AB	**BTDR**	Block Transfer Definition Reg			*	
0x0D00 01AF	**ICR**	Interface Configuration Reg				
0x0D00 01B3	**ARCR**	Arbitor/Requester Config Reg	*			
0x0D00 01B7	**AMSR**	Address Modifier Source Reg				
0x0D00 01BB	**BESR**	Bus Error Status Reg	*	*	*	
0x0D00 01BF	**DMASR**	DMA Status Reg			*	
0x0D00 01C3	**SS0CR0**	Slave Select 0 Control Reg 0		*		
0x0D00 01C7	**SS0CR1**	Slave Select 0 Control Reg 1		*		
0x0D00 01CB	**SS1CR0**	Slave Select 1 Control Reg 0		*		
0x0D00 01CF	**SS1CR1**	Slave Select 1 Control Reg 1		*		
0x0D00 01D3	**RCR**	Release Control Register	*			
0x0D00 01D7	**BTCR**	Block Transfer Control Register			*	
0x0D00 01DB	**BTLR1**	Block Transfer Length Register 0			*	
0x0D00 01DF	**BTLR0**	Block Transfer Length Register 1			*	
0x0D00 01E3	**SRR**	System Reset Register				
0x0D00 01E7	**BTLR2**	Block Transfer Length Register 2			*	
0x0D00 01EB-FF		*Reserved Locations*				

Detailed bit descriptions of each of the registers can be found in the VIC068A Users Guide. Additional VIC64 specific data can be found in the VIC64 Design Notes.

5.6.14.1 Register Descriptions

VIICR VME bus Interrupter Interrupt Control Register (Addr 0x0D00 0103)

D7	D6	D5	D4	D3	D2	D1	D0
Mask Bit	-	-	-	-	ipl2	ipl1	ipl0

When the C80 Card is interrupting an external VME Host CPU, the VIC64 can be programmed to interrupt the C80 when the external Host CPU generates an IACK cycle - this is done by clearing the Mask bit in this register to 0. The ipl code set on D2-0 will be read by the C80 from the VME I/F FPGA Interrupt Pending register when the interrupt is asserted.

VICR1-7 VME bus Interrupt Control Register (Addr 0x0D00 0107-1F)

D7	D6	D5	D4	D3	D2	D1	D0
Mask Bit	-	-	-	-	ipl2	ipl1	ipl0

These registers are used to enable the C.A.T. card as an Interrupt Handler. The seven VICR registers are each associated with one of the VME IRQ lines. When the Mask Bit in a particular register is cleared to 0, the VIC64 will interrupt the C80 whenever the associated IRQ line is activated. The ipl code set on D2-0 will be read by the C80 from the VME I/F FPGA Interrupt Pending register when the interrupt is asserted. The C.A.T. Card can be programmed to respond to any, or all seven, of the IRQ lines; however only seven unique ipl codes are available.

DMASICR DMA Status Interrupt Control Register (Addr 0x0D00 0123)

D7	D6	D5	D4	D3	D2	D1	D0
Mask Bit	-	-	-	-	ipl2	ipl1	ipl0

The VIC64 can be programmed to interrupt the C80 when it has completed a local DMA operation - this is done by clearing the Mask bit in this register to 0. The ipl code set on D2-0 will be read by the C80 from the VME I/F FPGA Interrupt Pending register when the interrupt is asserted.

EGICR Error Group Interrupt Control Register (Addr 0x0D00 014B)

D7	D6	D5	D4	D3	D2	D1	D0
AC Fail Mask	Post Write fail mask	Arbiter Timeout mask	SysFail interrupt mask	SysFail	ipl2	ipl1	ipl0

The VIC64 can be programmed to interrupt the C80 when it detects any of the above four error conditions by clearing the appropriate mask bit to zero. D3 - the SysFail Bit may be read at any time to determine the state of the VME SysFail line. The ipl code set on D2-0 will be the interrupt identifier code read by the C80 from the VME I/F FPGA.

VIRSR VME bus Interrupt Request/Status Register (Addr 0x0D00 0183)

D7	D6	D5	D4	D3	D2	D1	D0
IRQ7	IRQ6	IRQ5	IRQ4	IRQ3	IRQ2	IRQ1	enable

With this register the C80 Card can generate interrupts to external processors over VME by asserting the appropriate IRQ line. An interrupt can be generated by writing a one to both the required IRQ bit (D1 to D7) and to the enable bit (D0). The status of the interrupt can be monitored by reading back this register to see if the IRQ bit has been cleared. Normally the IRQ bit is cleared by the (remote) interrupt handler generating an IACK cycle; however a set IRQ bit may also be cleared locally by writing one to the set IRQ bit (D1 to D7) and zero to the enable bit (D0).

VIVBR VME bus Interrupt Base Registers 1-7 (Addr 0x0D00 0187-9F)

When the card is operating as an Interrupt Requester, these seven registers store the interrupt vectors returned by the C.A.T. Card during an IACK cycle. Each register corresponds to a particular IRQ line, only those in use need be programmed.

TTR Transfer Timeout Register (Addr 0x0D00 01A3)

D7	D6	D5	D4	D3	D2	D1	D0
VMEbus timeout2	VMEbus timeout1	VMEbus timeout0	Local timeout2	Local timeout1	Local timeout0	Arbiter timeout	Inc Bus acquire

This register controls two ‘Timeout’ timers.

The VME bus transfer timer is active only when the VIC64 is the System Controller, whereupon it will terminate any bus cycle that has not completed in the defined time period (see below).

The Local Bus Timeout timer can be used to terminate a Master VME cycle if VME bus ownership cannot be obtained within a specified time period. Bit 0 of this register must be set to do this.

D1 is a status bit that is set when the VIC64 detects a bus arbitration timeout. This timeout is always set to 8us, but only operates when the VIC64 is the System Controller.

The timer settings are:

D7/D4	D6/D3	D5/D2	Timeout Period us
0	0	0	4
0	0	1	16
0	1	0	32
0	1	1	64
1	0	0	128
1	0	1	256
1	1	0	512
1	1	1	Disabled

LBTR Local Bus Timing Register (Addr 0x0D00 01A7)

The values in this register set various timing parameters on the C.A.T. Card VME I/F local bus, that are used when the VIC64 is in control of the local bus. This register should always be set to 0x94.

BTDR Block Transfer Definition Register (Addr 0x0D00 01AB)

D7	D6	D5	D4	D3	D2	D1	D0
2K boundary crossing	Enable D64 Slave	Accelera te BLT's	Enable D64 Master	VME 256 boundary crossing	Local 256 boundary crossing	Enable optional AM codes	Enable Dual Path

This register selects between D64 or normal Block Transfers and enables optional hardware features that accelerate block transfers.

D0 Dual path enable - should be set to 1

D1 Enable user defined AM codes for block transfer - normally set to 0

D2 Enable local bus 256 byte boundary crossing - should be set to 1
D3 Enables 256 byte boundary crossing on VME - should be set to 1

D4 Enables D64 Master Operations (during BLT with local DMA) - set as required

D5 Enables accelerated BLT's - normally set to 0

D6 Enables D64 slave operations - should be set to 1

D7 Enables 2k byte boundary crossing for D64 Master Cycles - normally set to 0

ICR Interface Configuration Register (Addr 0x0D00 01AF)

This register controls features on the interface between the VIC64 and the Local Bus, it is normally set to 0x10. D0 of this register is read only, and can be read to determine whether the VIC64 is the System Controller (0 = System Controller function enabled).

ARCR Arbiter/Requester Configuration Register (Addr 0x0D00 01B3)

D7	D6	D5	D4	D3	D2	D1	D0
Arbiter Mode	BREQ Level1	BREQ Level0	DRAM refresh	Fairness timer3	Fairness timer2	Fairness timer1	Fairness timer0

This register sets the Master Bus Request level, turns Fairness On and Off and sets the type of arbitration to be used when the card is System Controller.

D3-0 Set to 0 to disable Fairness, set to 0xF to disable Fairness timeout, other values set Fairness timeout in increments of 2us

D4 DRAM refresh is not required. Always set to 0

D6-5 Set Bus Request Level (0,0) use BREQ0
(0,1) use BREQ1
(1,0) use BREQ2
(1,1) use BREQ3

D7 Set to 0 for Round Robin, Set to 1 for Priority. (Only applies when System Controller)

AMSR Address Modifier Source Register (Addr 0x0D00 01B7)

D7	D6	D5	D4	D3	D2	D1	D0
AM2-0 generate	AM5-3 qualify	AM5	AM4	AM3	AM2	AM1	AM0

This register is programmed to allow User defined Address Modifier Codes to be used in place of the standard VME codes. The C.A.T. Card hardware does not support this feature for Master Operations, but it can be used for Slave Operations. The User defined Address Modifiers for Slave Operations are enabled in conjunction with SS0CR0 and SS1CR0.

D5-0 User Defined Address Modifier Code

D6 0 - Use standard AM Codes for VME Slave accesses
1 - Qualify AM5-3 with User defined AM code for VME slave accesses

D7 Not implemented on C.A.T. card. Set to 0

If User defined AM codes are not being used, this register can be ignored and it will default to 0.

BESR Bus Error Status (Addr 0x0D00 01BB)

D7	D6	D5	D4	D3	D2	D1	D0
VME BUS Master	Local BUS Error (LBERR*)	VME BUS Error (BERR*)	VME BUS Timeout	Local Bus Timeout	SL0 self access	SL1 self access	Local Bus Timeout (acquire)

The various bits of this register are set if an error occurs during a VME operation. Once set, the bit must be cleared by writing 0 to the register.

D0 - Timeout trying to acquire VME Bus, would be generated by a Master Cycle or Local DMA

D1 - A C.A.T. Card Master or Local DMA cycle has tried to access its own Slave Short Address, this is not allowed

D2 - A C.A.T. Card Master or Local DMA cycle has tried to access its own Slave Extended Address, this is not allowed

D3 - Local Bus Timeout

D4 - A VME Bus Timeout has occurred. (This bit is only valid if the C.A.T. Card is the System Controller)

D5 - VME Bus error. VME BERR* has been asserted

D6 - Local Bus error. LBERR* has been asserted

D7 - VME BUS Master. This bit can be read at any time, and does not need to be cleared. It is set when the C.A.T. card is Bus Owner

DMASR DMA Status Register (Addr 0x0D00 01BF)

D7	D6	D5	D4	D3	D2	D1	D0
Post Write data in store	1	1	VME BUS Error (BERR*)	Local Bus Error (LBERR*)	BERR* during DMA	LBERR * during DMA	BLT in progress

This register is used to check the success of Local DMA transfers.

D0 - BLT in progress, this bit is set when a BLT starts, and cleared by the VIC64 on completion of the DMA transfer

D1 - LBERR* during DMA, this bit is set if a Local bus error is generated during a DMA. It must be cleared by writing 0 to this location

D2 - BERR* during DMA, this bit is set if a VME Bus error occurs during a DMA. It must be cleared by writing 0 to this location

D3 - Local Bus Error, read only, is a copy of D6 in the BESR

D4 - Bus Error, read only, is a copy of D5 in the BESR

D5,6 - Not Used

D7 - Post Write data in store, this bit is set when write data has been 'posted' to the VME I/F by the C80, but the VME Write has yet to be completed

SS0CR0 Slave Select 0 Control Register 0 (Addr 0x0D00 01C3)

This register controls certain aspects of VME Slave accesses in Extended Address Space to the C.A.T. card. It also defines how block transfers are handled on the Local Bus. On the C.A.T. card only two different values should be programmed into this register:

Normal Operation: 0x12
Supervisory Access Only: 0x32

SS0CR1 Slave Select 0 Control Register 1 (Addr 0x0D00 01C7)

This register sets the Extended Address Slave access and local Bus Block Transfer timings for the interface, it should be set to 0x11.

SS1CR0 Slave Select 1 Control Register 0 (Addr 0x0D00 01CB)

D7	D6	D5	D4	D3	D2	D1	D0
Enable Slave Post Write	Enable Master Post Write	Supervisor access only	32 Bit Local Bus	Address space 1	Address space 0	BLT Mode 1	BLT Mode 0

This register controls certain aspects of VME Slave accesses in Short Address Space to the C.A.T. card, and also enables the Post Write facilities.

D0 - D4 These bits should always be set to 0x18

D5 When set this bit limits Slave Extended Address accesses to Supervisory only

D6 When set this bit enables the Master Post Write

D7 When set this bit enables the Slave Post Write facility

For optimal performance both D6 and D7 should be set hi.

SS1CR1 Slave Select 1 Control Register 1 (Addr 0x0D00 01CF)

This register sets the Short Address Slave access timings for the interface, it should be set to 0x0F.

RCR Release Control Register (Addr 0x0D00 01D3)

D7	D6	D5	D4	D3	D2	D1	D0
Release Mode 1	Release Mode 0	Burst Length 5	Burst Length 4	Burst Length 3	Burst Length 2	Burst Length 1	Burst Length 0

This register configures the VME bus release mode of the VIC64, and sets the burst count for block transfers with local DMA. By setting the release mode to BCAP, the VME bus can be grabbed and held thus eliminating bus arbitration for the duration of a master cycle.

D0 - 5 Block transfer Burst length. This value sets the number of **transfer cycles** per burst (note: a value of 0 defines a burst of 64 cycles). For D64 block transfers the value defined here will be multiplied by 4

D6,7 Bus Master Release Mode.

0 0	ROR - Release on Request
0 1	RWD- Release when done
1 0	ROC - Release on BCLR*
1 1	BCAP - Capture and Hold

BTCR Block Transfer Control Register (Addr 0x0D00 01D7)

D7	D6	D5	D4	D3	D2	D1	D0
Module-based DMA	BLT with Local DMA	MOVEM DMA	Data Direction	Interleave period 3	Interleave period 2	Interleave period 1	Interleave period 0

This register defines the type, direction and burst interleave period of block transfers generated by the VIC64. On the C.A.T. card only the BLT with local DMA mode is supported.

D0 - 3 Block transfer Burst Interleave period. This sets the time interval between bursts, in increments of 250ns. It should be set to 1 or greater

D4 Transfer direction. When set, data is read from external VME memory, when clear data is written to VME external memory

D5 MOVEM BLT - not allowed on C.A.T. card - always set to 0

D6 BLT with Local DMA - Set high immediately before initiating a BLT

D7 Module-Based DMA - not allowed on C.A.T. card - always set to 0

BTLR2/BTLR1/BTLR0 Block Transfer Length Register (Addr 0x0D00 01E7/DB/DF)

These three 8 bit registers combine to define the total number of bytes to be transferred in a BLT with Local DMA transfer.

SRR System Reset Register (Addr 0x0D00 01E3)

Writing 0xF0 to this register will cause SYSRESET[L] to be asserted for 200ms. This will reset the system. The C.A.T. card will also be reset, unless the VIC64 is being held reset by D0 of the *Board Reset register*.

5.6.15 VIC54 Non-Implemented Features

The VIC64 has two features which have NOT been implemented on the C.A.T. Firstly, the interprocessor communication facilities are not supported; secondly none of the Local Interrupts are used.

6. Logic Description

This section describes the logic and circuit operation of the C.A.T. with reference to the circuit diagrams. Grid references to circuit elements on the schematics are given thus:

[3C6] sheet 3, grid reference C6

Individual components are referred to by their PCB reference e.g. K4.

Signal names are followed by [H] or [L] to indicate active high, or active low.

6.1 Circuit Overview

The C.A.T. circuitry is reasonably simple. Aside from the VIC64 Chipset, all control functions are implemented in three FPGA's, the remaining circuitry is either Data Buffers, Address Buffers or Memory chips.

The diagram in Figure 6.1 shows an overview of the three FPGA's that control the C.A.T. circuitry. P701_1 is the main address decoder for the C.A.T. generating individual RAS and chip select signals. It also generates the READY and RETRY signals for the C80, and contains the arbiters that control access to the shared memory on the VME I/F and Applications Daughterboards.

The smaller FPGA P701_2 generates the memory configuration for the C80. For memory on the C.A.T. and the Display Daughterboard the configurations are defined in the source code for P701_2. For the Application Daughterboards the memory configuration is defined by signals from the daughterboard.

The third FPGA, P701_3 interfaces the C80 bus control signals to the VIC64 Chipset, and additionally contains a number of 8 bit registers that may be accessed by either the C80 or an external VME CPU.

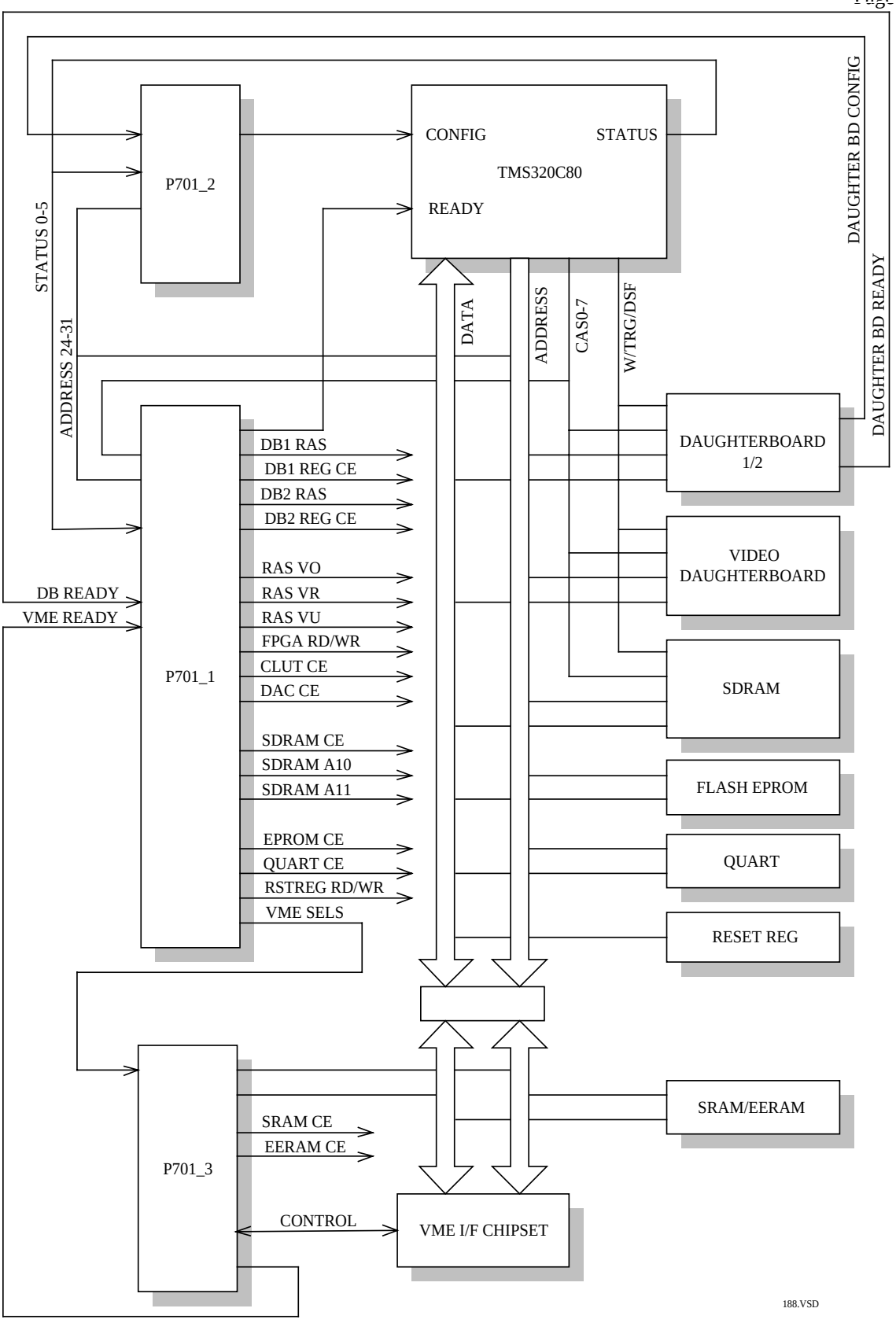

Figure 6.1: C.A.T. FPGA's and Control Signals

6.2 The C80 Processor

The C80 is a 3V3 device and unfortunately its I/O lines are not 5V tolerant, thus all of the C80's address, data and control signals need to be translated. The address and data buffers connected to C80 DATAn[H] and C80 ADDRn[H] are 74LPT series logic, these devices operate from the 3V3 supply, but have 5V tolerant I/O. Other signals in and out of the C80 are translated by Quickswitch buffers - E4, E5, E6, E7, E8, F8 and G8 - all on page 3 of the circuit diagram. All 3V3 signals are connected to the C80, and are prefixed by 'C80' on the schematics e.g. 'C80 TRG[L]' [3E3] is a 3V3 signal, whilst the translated 5V tolerant signal is 'TRG[L]' [3E4].

6.2.1 C80 I/O Signal Definitions

A31-0	O	C80 address output. This is always driven, even on cycles where the Address is undefined
D63-0	I/O	C80 data
RAS	O	Row Address Strobe. Although this signal is always high at the start of a bus cycle, its subsequent transitions depend on the type of memory being accessed and the cycle type
CAS/DQM7 - CAS/DQM0	O	These eight signals operate as individual byte write enables on the C80's data bus. For VRAMs and DRAMs they operate as CAS, for SDRAMs they operate as Data Qualifiers (DQM)
TRG/CAS	O	For VRAMs and DRAMs this signal operates as a Transfer/OE signal, for SDRAMs it is the CAS
W	O	Write signal
DSF	O	VRAM special function. This is activated for VRAM block write, Load Colour register and Split Transfer cycles
STATUS0 - STATUS5	O	The STATUS lines define the type of bus cycle the C80 performing, they are used by P701_1 and P701_2 to detect special bus cycles such as Refresh or Block Write
AS2 - AS0 BS1 - BS0 CT2 - CT0 PS3 - PS0	I I I I	These four groups of signals form the memory configuration signals that tell the C80 the capabilities of the memory it is trying to access. See section **4.3.1 C80 Memory Configuration Inputs**
RL	O	This signal is output from the C80 as a row address latch, however a high to low transition of this signal is the only reliable indication of the start of a new bus cycle. It is used by P701_1 to synchronise the READY timer, and to latch row address decodes

READY	I	This signal is driven low by P701_1 to extend the bus cycle for slow devices
RETRY	I	This signal is driven low by P701_1 when the C80 attempts to access shared memory that is busy (VME I/F or App Daughterboard)
CLK-OUT	O	This is the C80's internal 50MHz clock. It is buffered and fed to the SDRAMs, P701_1 and the App Daughterboard. P701_1 uses this clock to generate READY[H], RETRY[L] and the SDRAM select and upper address lines
FAULT	I	Memory Fault line - not used on C.A.T
UTIME	I	User defined Timing - not used on C.A.T. This signal is driven low during reset to configure the C80 as Big-Endian
HREQ HACK	I O	The Host Interface is not used on C.A.T. HREQ is driven low during reset to activate the Master Processor after reset
REQ1-REQ0	O	
EINT1	I	Rising edge triggered interrupt, driven by App Daughterboard 1
EINT2	I	Rising edge triggered interrupt, driven by App Daughterboard 2
EINT3	I	Rising edge triggered interrupt, driven by P701_3 (VME I/F)
LINT4	I	Level triggered interrupt, driven by QUART
RESET	I	The C80 reset line is driven by P701_2. See section **6.8 Resets**
XPT2-XPT0	I	External Packet Transfer. On the C.A.T. only the App Daughterboards generate XPT's
CLK-IN	I	The C80's x2 clock input. This is driven by a 100MHz TTL Oscillator
EMU1-EMU0 TCK, TMS, TRST TDI, TDO		The C80's JTAG emulation port is tracked to P11 to allow a Texas Instruments emulator to be attached. This connector is not fitted as standard on C.A.T. boards
CBLNK, CSYNC HSYNC, VSYNC FCLK, SCLK CAREA		Video controller 0 signals are routed to the Display Board, Video controller 1 signals are routed to the App Daughterboard site

6.2.2 C80 Data Bus

The 3V3 64 bit data bus of the C80, (C80 DATAn[H]), is connected directly to the data pins of the SDRAM's as these are also 3V3 devices. The full 64 bits are buffered two ways by 74LPT16245 data buffers to the Display Board (M6, M8, Q6, Q8 - Sheet 6), and to the Application Daughterboard site (F4, F6, H4, H6 - Sheet 7).

Thirty-two bits of the data bus are split off and buffered (M9, Q9 Sheet 5) onto a third bus - SLOW DATAn[H]. This bus connects to the QUART [5E3], EPROM's [5B3], Reset Register [8B3] and VME I/F [2A5].

6.2.3 C80 Address Bus

The top eight address lines of the C80 are translated to +5V by Quickswitch E5 [3E2] and used by P701_1 and P701_2 for address and configuration decoding. The bottom 24 bits of the address are buffered two ways by 74LPT16244 buffers to the Display Board (H2, F2 Sheet 6) and to the Application Daughterboard site (H3, F3 Sheet 7).

A third set of buffers (M10, Q10 Sheet 5) buffers the full 32 bits to the QUART [5E3], EPROM's [5B3] and VME I/F [2C5]. This is SLOW ADDRn[H].

A fourth address buffer (H8 4A3], buffers the address to the SDRAM's to minimise loading on the C80 address lines and to improve the drive to the SDRAM array.

6.2.4 C80 CAS/DQM Lines

The C80 CAS/DQM lines (C80 CASn[L], [3D4]) are as time critical as the address and data lines when performing memory accesses. They are treated similarly with dedicated buffers to the Display Board (L8 [6G4]), the Application Daughterboard Site (E2 [7G4]) and the SDRAMs ([J8 [4A4]). The CAS/DQMs are also translated by Quickswitch G8 [3E3], for P701_1 and the VME I/F [2A1].

6.2.5 C80 RL/RAS/W/TRG/DSF

The C80 row latch signal (C80 RL[L]) is the only reliable start of cycle indicator, it is buffered to the Application Daughterboard site [7F4], and translated by Quickswitch E7 for P701_1. A delayed version of RL[L] is also fed to P701_1, so that it may be registered on the alternate clock edge.

C80 RAS[L] is only used by P701_1 and P701_2. All other RAS signals are generated by P701_1 and are qualified by address and STATUS.

C80 W[L], C80 TRG[L] and C80 DSF[H] are buffered and fed to both the Display Board [6F4] and the Application Daughterboard [7F4].

6.2.6 C80 CLKOUT

The 50MHz clock output from the C80 has very little drive capability, and is buffered three ways to improve propagation. Two outputs are driven by buffer J8 [4A5], one output is used to drive the SDRAMs, the other to drive P701_1. The clock is also buffered to the Application Daughterboard Site by E2 [7G4].

6.2.7 C80 STATUS and Configuration

The STATUSn[H] lines are translated by Quickswitch E7 [3E5] and fed to P701_1 and P701_2. The twelve configuration signals are generated by P701_2 and fed back to the C80 via translators E6 and E8 [3A5]. The C80 only allows 20ns from issuing the new row address to sampling the configuration signals, thus a fast ispLSI 2064-125 is used for P701_2.

The configuration for a particular address is hard coded into P701_2 as listed in **Appendix A**. For the Application Daughterboards, the configuration is defined by signals from the daughterboards that are fed into P701_2 e.g. DB1 AS0[H] [3G3]. These signals are intended to be static due to the short time interval from row address to configuration input.

In addition to the address, P701_2 also uses the STATUSn[H] signals to detect special cycles. When P701_2 detects the STATUS for a SDRAM DCAB or MRS cycle then the SDRAM configuration is output irrespective of the row address.

When P701_2 detects the STATUS for a Block Write or Load Colour Register cycle the normal BS1, 0 configuration for the address selected is overridden. If the corresponding row address indicates a Block Write to SDRAM BS1, 0 are set to 0 to give a simulated block write. If the row address is for the Display board VRAM or Application Daughterboard data memory then BS1, 0 are determined by bits 5 and 6 of the *Reset register*.

When P701_2 detects the STATUS for a Refresh cycle, either VRAM or SDRAM configuration is output depending on the state of the TFRESH[H] signal from P701_1.

6.2.8 C80 Video Controller Signals

The C80 Video Controller signals are translated by Quickswitches (E4, E8) and fed to the Display and Daughterboards. As the Quickswitches are inherently bi-directional the sync signals may be programmed as inputs or outputs as required by the daughterboards.

6.3 Memory Cycles

The normal bus cycles on the C.A.T. are 3 Cycles per Column (CT2..0 = 7), and these are used for all on-board accesses (except SDRAM).

Figure 6.2 shows a single 3 Cycle/Column write cycle. At the start of the cycle RAS is taken high and the Row Address and STATUS for the new cycle are output. From these the memory configuration is determined and returned to the C80. On receipt of the configuration signals the C80 knows how to continue the cycle - in this case it waits two clocks and asserts RAS, and then waits two more clocks before outputting the column address. For non-multiplexed memories (AS2..0 = 0) the column address is the same as the row address. CAS is asserted one clock after the column address, for two clocks.

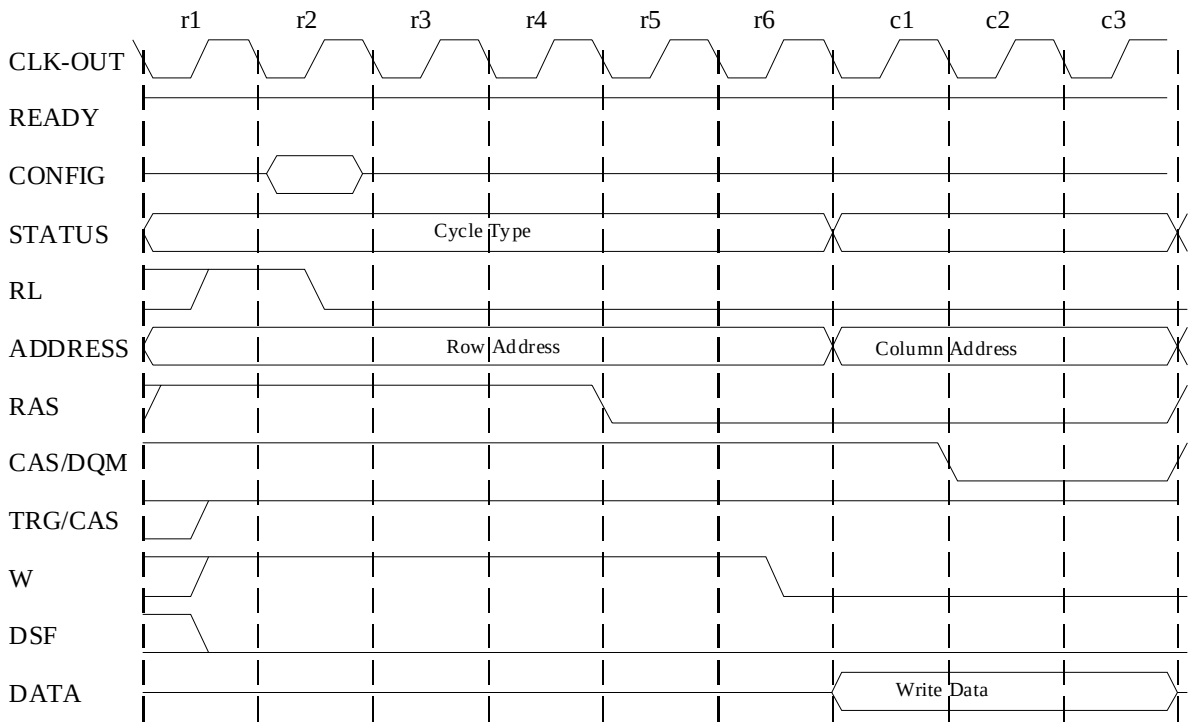

Figure 6.2: Single Three Cycle/Column Write Cycle

At the end of the bus cycle shown in Figure 6.2, RAS goes high ready for the next cycle, however for memories that support Fast Page access RAS may be held low and further column accesses may be made. In a memory Write Cycle all the column accesses will be Writes, the C80 does not perform read/modify/writes. Additional Column accesses will not necessarily be back to back, as in practice the C80 tends to hold RAS active until a page boundary is crossed, or until a different area of memory is to be accessed. Since the processors within the C80 execute from internal instruction caches, RAS can be asserted until the next Refresh Cycle.

After a memory refresh cycle the C80 always takes RAS high. The Bus then sits in the 'r1' state with RAS and RL high, but with no row address or STATUS. This may continue for some microseconds, and thus the only valid indication of the start of a new cycle is the High to Low transition of RL in state 'r2'.

For slow devices P701_1 negates READY to extend the Cycle. This is done when CAS is low, the C80 sampling the READY line on the c2 to c3 transition. See Figure 6.3.

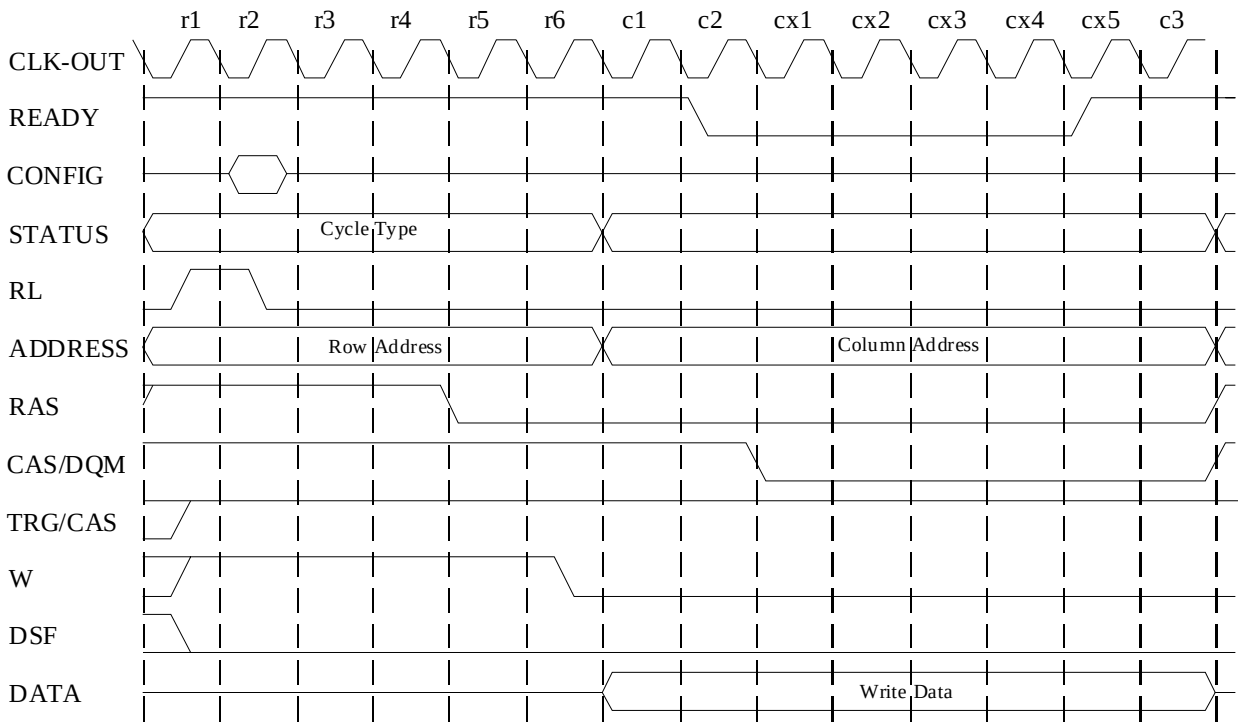

Figure 6.3: Single Three Cycle/Column Write Cycle, extended by READY

Apart from the Refresh Cycle (Figure 6.4), the other non-SDRAM bus cycles differ from the Write Cycle in Figure 6.2 only in the assertion of W, TRG/CAS and DSF - see the TMS320C80 data sheet for these bus cycles.

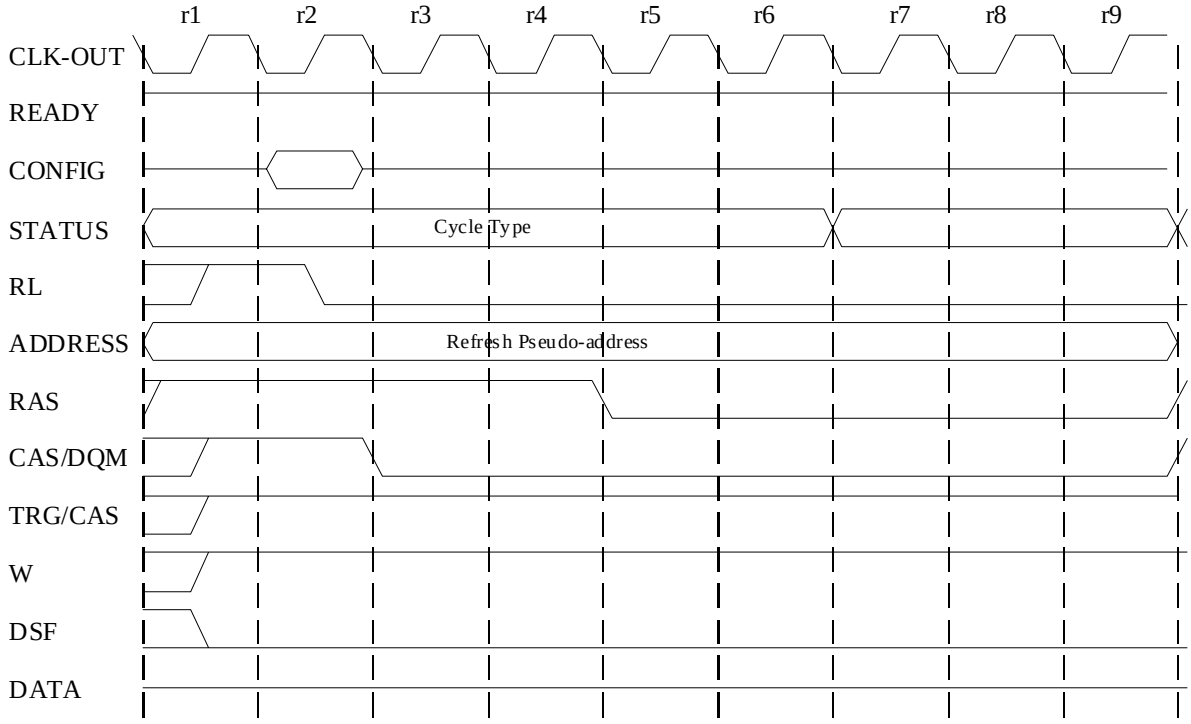

Figure 6.4: Three Cycle/Column Refresh Cycle

6.4 SDRAMs

A C.A.T. with 8MBytes uses four 16M Bit devices, each arranged as 2 banks x 512k x 16 bits, these are sited on the top of the board - E9, E10, H9, H10.

A C.A.T. with 32MBytes uses four 64M Bit devices, each arranged as 4 banks x 1024k x 16bits, these are sited on the bottom of the board - E12, E13, H12, H13.

Only one set of RAMS may be fitted to the board at a time.

The address and control signals to the SDRAMs are common to all four RAMs, except for the eight CAS/DQM lines which are used as individual byte write enables.

The clock, control and CAS/DQM signals are all driven from the same buffer (J8 [4A5]) to minimise skew, and series resistors are included to reduce ringing. The address lines are driven by a similar buffer (H8 [4A3]).

The chip select, SD CS[L] is generated by P701_1 synchronously to the rising edge of the clock. The select line is activated at the start of the bus cycle and held active for the whole cycle, a delayed version of RL[L] is used to synchronously set or clear the chip select.

The bank select signals SD A11[H], SD BS0[H], SD BS1[H] are generated by P701_1. These are simply synchronously latched address lines, set during the row address phase when the delayed RL[L] is high.

The SD A10[H] address line to SDRAMs is used as a row address input when RAS is active, but later in the cycle it controls the optional Auto-Precharge of the SDRAM. With the C80 Auto-Precharge must not be used, so A10 is forced low (by P701_1) after the row address phase.

6.4.1 SDRAM Bus Cycles

In addition to Read, Write and Refresh cycles the C80 has two additional cycle types to support SDRAM operation. The DCAB (Precharge) and MRS cycle are both used by the C80 in its post-reset sequence to initialise the SDRAMs. The DCAB cycle deactivates the SDRAMs, it is the equivalent of the Low to High transition of RAS in a normal DRAM. The MRS cycle programs the Mode Register inside the SDRAMs for Read Latency (3) and Burst Length (1).

SDRAM commands are:

CS[L]	RAS[L]	CAS[L]	WE[L]	A13..11	A10	A9..0	Command
H	X	X	X	X	X	X	Device Deselected
L	L	H	H	V	V	V	Activate Bank and Row
L	H	L	L	V	L	V	Write to specified column
L	H	L	H	V	L	V	Read from specified column

CS[L]	RAS[L]	CAS[L]	WE[L]	A13..11	A10	A9..0	Command
L	L	H	L	X	L	X	Deactivate (Precharge) both banks
L	L	L	L	V	V	V	Mode Register Set
L	L	L	H	X	X	X	Auto Refresh

A SDRAM read cycle, with four reads from one page is shown in Figure 6.5.

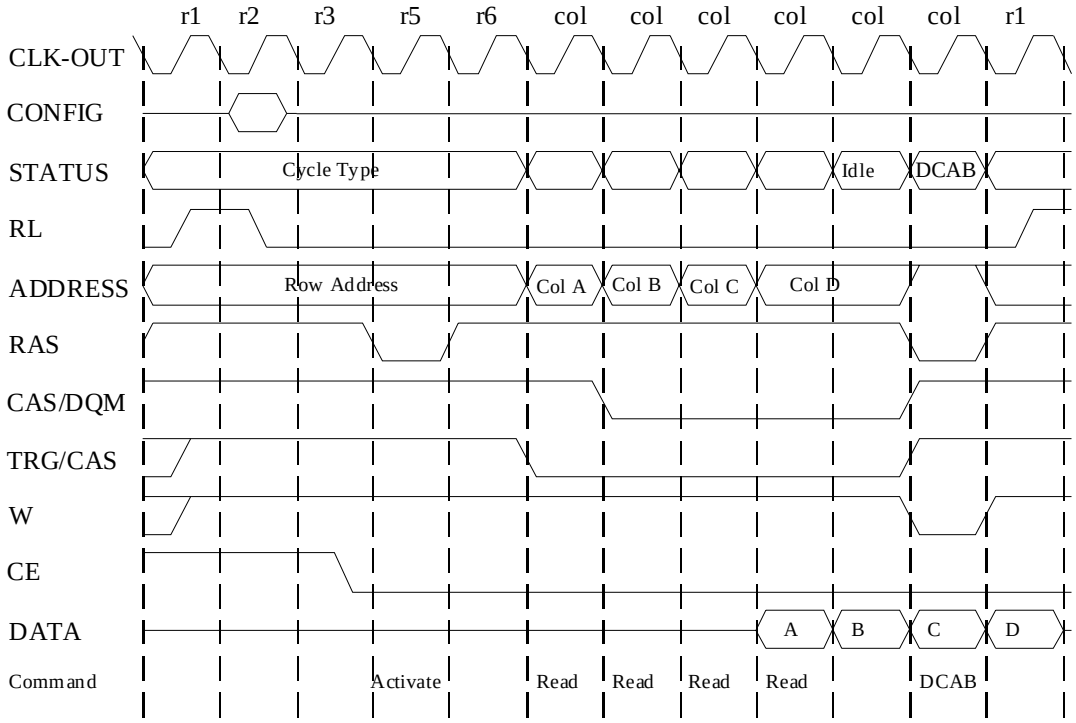

Figure 6.5: SDRAM Read Cycle

6.5 P701_1

The FPGA P701_1 [3F3] performs the following tasks:

- Address decoding
- READY[H] generation
- Shared memory arbitration and RETRY[L] generation
- Data buffer control

6.5.1 Address Decoding

Address decoding in P701_1 is based on the top eight address lines of the C80, thus all decodes occupy 16M Bytes of the address map (see **Appendix A**). For each of the decoded outputs e.g. EPROM CE[L], there is a corresponding latch within the FPGA that is set at the start of the bus cycle when RL[L] is high. The decoded output is generated by qualifying the latched decode with an appropriate signal. Where the decode signal is being used directly as a write pulse e.g. RST REG WE[L], the qualifier is CASn[L]. Where RAS[L] is used as a qualifier, some further signal must be used as a write qualifier at the target device.

On certain bus cycles the C80 outputs an invalid or 'Don't Care' address. P701_1 detects these cycles using the STATUS signals, and disables the unwanted decode outputs. These cycle types are:

REFRESH cycles
SDRAM MRS cycles
SDRAM DCAB cycles

P701_1 also detects Video Controller 0 RAM->SAM transfer cycles, and on these cycles all three Framestore RAS signals to the Display Board are activated.

6.5.2 Refresh Cycles

The C80 Transfer Controller has a built in refresh timer that regularly generates refresh cycles, but the C.A.T. hardware must determine the type of refresh cycle performed using the configuration inputs to the C80.

Two types of refresh cycle are required for the C.A.T. hardware - SDRAM refresh and VRAM refresh. Support for both is simply arranged on the C.A.T. by alternating Refresh cycle types. This is achieved by having the refresh cycle type defined by a flip-flop output that is toggled after each refresh cycle. Both the address decode FPGA (P701_1) and the configuration FPGA (P701_2) need to know which type of refresh is coming next, and this is defined by the signal TFRESH[H] [3G5] - high = VRAM, low = SDRAM.

6.5.3 READY[H]

To meet the C80 setup and hold requirements, READY[H] is generated synchronously on the falling edge of CLK[H].

READY[H] can be negated to extend the cycle by VME READY[H], DB1 READY[H] or by DB2 READY[H]. These lines are only sampled when the appropriate address decode is active, i.e. DB1 READY[H] is disabled until DB1 RAS[L] is asserted.

For slow devices READY[H] is asserted for 100ns to extend the cycle. In these cases the timing for READY[H] is determined by a counter, incrementing on the falling edge of the clock, and synchronised to the bus cycle by RL[L]. See timing diagram in Figure 4.4.

6.5.4 Shared Memory Control

The VME I/F and the two Application Daughterboards can request ownership of their memory and thus 'lock out' the C80. This feature is controlled by P701_1 using request and grant lines to each of these circuits and by asserting RETRY[L] to the C80.

The C80 is normally assumed to 'own' all of the shared memory. When, for instance, a VME Slave Cycle causes the VIC64 to request ownership of the Local Bus it must assert VME REQ[H] [3F2] to P701_1 and wait for VME GRANT[H] [3G2]. If the C80 is not currently accessing the Local Bus then VME GRANT[H] will be returned immediately, otherwise the grant will not be issued until the start of the next C80 bus cycle. Once VME REQ[H] has been asserted any new C80 cycles that attempt to access the local bus will cause RETRY[L] to be asserted.

The request/grant mechanism is the same for the two Application Daughterboards.

The RETRY[L] signal is generated synchronously to the falling edge of CLK[H]. The C80 will abandon and retry a bus cycle *only* when RETRY[L] is asserted early in the row-address phase of the cycle (on the r2 to r3 transition). Because of this the arbitration logic is synchronous, to the falling edge of the clock.

6.5.5 Other Features

P701_1 controls the External Packet request lines to the C80 (XPTn[L]), these are asserted in response to a request from one of the Application Daughterboards (DBn PDTn REQ[L]). Once a request has been asserted, the XPTn[L] lines are held asserted to the C80 until the C80 generates an XPTn bus cycle. This is detected using the STATUS lines.

P701_1 also generates the enable and direction signals for the three sets of data buffers on the C80 data bus. The enables are controlled by the latched address decodes, the direction is set by the DDIN[L] signal from the C80. When a XPT cycle is detected the Application Daughterboard data buffers are enabled in the direction defined by STATUS 5..0.

The three WEn[L] lines are generated from CASn[L] gated with a write enable determined from STATUS 5..0. They are used as write enables by the EPROMs the QUART and the Colour LUTs on the Display Board.

P701_1 also drives the Orange LED on the C.A.T. front panel. This is intended to monitor C80 bus activity - the brighter the LED the more the C80 is doing. The LED is driven by a gate within the FPGA that turns the LED off when the STATUS lines are in their idle condition, and turns the LED on when any other STATUS is output.

6.6 VME I/F

6.6.1 VIC64 Chip Set

All connections to the VME Bus are made through the chip set, there are no connections between the VME bus and other C.A.T. circuitry. The chip set consists of one VIC64 (M2) and three CY7C964s (H1, M1, Q1 Sheet 1). The VIC64 is the master chip and controls the other three. It connects to all the VME bus arbitration signals, the address modifiers and the interrupt request and acknowledge lines. It contains the registers that control the interface's operation. The VIC64 is a synchronous device and operates from its own 64MHz TTL oscillator, XT3 [1F2].

The CY7C964s are mainly used as address and data buffers and operate under the direct control of the VIC64. In addition to latching addresses, the CY7C964s can increment addresses during Slave Block Transfers, and generate the Local Bus addresses during Block Transfers with Local DMA. The CY7C964's other function is to decode VME addresses during slave cycles, each CY7C964 contains an 8 Bit address register and comparator.

On the C.A.T. side of the chip set, the address and data lines connect directly to the Local Bus, LB LDATn[H] and LB LADDn[H]. All the VIC64 controls lines connect to the FPGA P701_3 [1E5]. The CY7C964s are controlled by the VIC64, the only C.A.T. signals that connect to them are the slave address decode outputs which go to P701_3, and the Slave Page Register load signal - VME CE P[L] which is generated by P701_1.

Attendant to the chip set is the upper/lower word swap buffer T11 [2E5]. This is used by the VIC64 on Slave Cycles to the Local bus, and by the C80 on Master cycles to the VME. It is not used by the C80 when accessing the Local Bus.

6.6.2 VIC64 Local Control Signals

The following VIC64 signals control the flow of data across the interface. The direction and function of the signals depends on who owns the Local Bus. On Master and IACK Cycles the Bus is owned by the C80, on Slave and Local DMA cycles it is owned by the VIC64.

VIC64 Pin	I/O	Active	Master Cycle	Slave Cycle
PAS	I/O	[L]	Address strobe input	Address strobe output
DS	I/O	[L]	Data strobe input	Data strobe output
DSACK0,1	I/O	[L]	Data Acknowledge output	Data Acknowledge input
R/W	I/O	[H]	Read/Write input	Read/Write output
WORD	I	[L]	Defines Local Bus Width	Defines Local Bus Width
BLT	I/O	[L]	Block Transfer input	Block Transfer output

VIC64 Pin	I/O	Active	Master Cycle	Slave Cycle
FC1,2	I/O	[H]	Inputs define cycle type: (0, 0) - User Data (0, 1) - User Program (1, 0) - Supervisor Data (1, 1) - Supervisor Program	Outputs define cycle type: (0, 0) - Slave BLT (0, 1) - Local DMA (1, 0) - Slave Access (1, 1) - DRAM refresh
SIZ0,1	I/O	[H]	Inputs define transfer width: (0, 0) - Longword (0, 1) - Byte (1, 0) - Word (1, 1) - 3 Byte	Outputs define transfer width: (0, 0) - Longword (0, 1) - Byte (1, 0) - Word (1, 1) - 3 Byte
ASIZ0,1	I	[H]	Inputs define address space: (0, 0) - User defined (0, 1) - A32 (1, 0) - A16 (1, 1) - A24	Not used

Other important control signals between the VIC64 and the P701_3 FPGA are listed below, these control bus ownership, slave cycle address decodes and resets.

VIC64 Pin	I/O	Active	Function
LBR	O	[L]	Local Bus Request, asserted when VIC64 wants to own Local Bus
LBG	I	[L]	Local Bus Grant, gives VIC64 ownership of Local Bus
CS	I	[L]	Asserted when C80 accesses the VIC64 internal registers
MWB	I	[L]	Asserted when the C80 is accessing external VME memory
FCIACK	I	[L]	Asserted when the C80 reads the Interrupt Acknowledge address
SLSEL0	I	[L]	Asserted when the CY7C964s detect a valid VME extended address
SLSEL1	I	[L]	Asserted when the CY7C964s detect a valid VME short address
LBERR	I/O	[L]	Local Bus Error - asserted by the VIC64 if a Master cycle fails, not used on slave cycles
DEDLK	O	[L]	Asserted by the VIC64 if the C.A.T. tries a Master cycle to one of its own Slave addresses
IRESET	I	[L]	Resets the VIC64 when asserted
RESET	O	[L]	Asserted by the VIC64 if VME SYSRESET[L] is asserted

The Local Bus request and grant signals are treated as asynchronous signals, even though the arbiter in P701_1 is synchronous. The request line is double clocked by P701_1 to remove metastability.

6.6.3 Local Bus Memory

The shared SRAM on the Local Bus is accessed by the C80 when the address decode FPGA P701_1 asserts VME CE S[L]. The timing of these cycles is controlled by the C80 and is set for 3 Cycle/Column access, multiple reads/writes per cycle are allowed. On read cycles all four SRAMs (S6, S8, T6, T8 Sheet 2) are enabled, on write cycles individual CAS/DQMs are used as write enables to the SRAMs. To ensure that data and address hold times are met on multiple write cycles to the SRAM's latched buffers are used for the C80 address and data.

When the VIC64 has control of the Local Bus, single accesses to the SRAMs are timed by P701_3 asserting the DSACKn[L] lines to the VIC64. On multiple accesses (slave BLTs and Local DMA) the timing is controlled by registers within the VIC64.

The EERAM (Q11 [2E3]), the VIC64 internal registers and the FPGA registers are all on the same C80 address decode - VME CE E[L]. The FPGA P701_3 generates the chip selects for the EERAM and the VIC64 based on the C80 address. Cycle timing is controlled by the C80 with P701_1 inserting the 100ns extension. Only single access cycles are performed. VME slave accesses to the C.A.T.'s short address access the EERAM and FPGA registers, but cannot access the VIC64 registers. On these cycles timing is controlled by P701_3.

Writes to the EERAM are timed the same as reads, however it must be remembered that after a write cycle the EERAM will be inaccessible for 1ms - the system software must take account of this. The Write Protect links for the EERAM are connected to P701_3 and operate by disabling the EERAM chip select - LB EERAM CE[L].

The specified EERAM has a built-in power supply monitor and a 5ms timer that disable all writes to the EERAM after power-up to prevent corruption of the EERAM contents.

6.6.4 P701_3

The main function of this FPGA is to interface the C80 control signals to the VIC64 control signals. Additional functions include:

- Controlling the SLOW Bus to Local Bus buffers
- Controlling the Local SRAM and EERAM
- Timing Slave access cycles

The FPGA also contains registers for:

- Generating Reset and Interrupt
- Setting Master Cycle parameters
- Reading back User Links

The address decodes that P701_3 receives and its responses are:

Decode Signal	Function
VME CE S[L]	Access shared SRAM
VME CE E[L]	Access EERAM, FPGA registers or VIC64 registers
VME CE P[L]	Write to VME Slave Page Registers (in CY7C964's).
VME CE M[L]	Perform VME master cycle, short/extended address space defined by address
VME CE I[L]	Acknowledge VIC64 or external VME interrupt

Additionally the signal VME SEL[L] is active whenever any of the above are active. Control signals generated by P701_3 to the other C.A.T. hardware are:

Signal	Function
VME REQ[H]	A buffered version of the VIC64 signal LBR[L] - this is the request for ownership of the Local Bus, P701_1 responds with VME GRANT[H]
VME READY[H]	Negated by P701_3 to extend C80 Master and IACK cycles, it is held low until the VIC64 asserts DSACK0,1
VME EINT[L]	Interrupt line to C80. Can be generated by Mailbox Interrupt, LBERR[L] or VIC64
SW RST[L]	Reset line to P701_2. Can be generated by Mailbox Reset, or High to low transition of SYSRESET[L]

P701_3 is clocked by the same oscillator as the VIC64, though none of the VIC64 signals are treated as synchronous. The clock is used for timing slave cycles, delaying signals, and for generating the timed reset pulse that is output on SW RST[L].

The FPGA determines the kind of Master cycle required by the C80 from the hi-order address lines LB LADD15,16[H] and SLOW ADDR30,31[H] and from registers within the FPGA.

6.7 Switching 3V3 PSU

The +3V3 supply for the C80 and the SDRAM's is generated by a DC to DC switching power supply module - PSU1 [8C5]. The supply is non-isolated and operates from the +5V rail. The supply has built in protection for over current on the +3V3 output, and should the module go faulty, the board is protected by a 4A SMT fuse on the module input. Switching noise is minimised by the external smoothing capacitors C3, C4 and C5 - these are low ESR types.

To prevent over current situations the module is turned off by the MAX810 reset monitor (VR1 [8B5]) whenever the +5V supply rail is below 4.6V. As the MAX810 also generates a 200ms reset on power-up, there is a pause before the module is enabled thus preventing large surge currents during power-up.

The +3V3 rail also has its own reset monitor, a MAX809 (VR2 [3A3]). This generates a reset whenever the +3V3 rail is below 3.0V. The two reset monitor outputs are combined in P701_2 to drive the green LED on the C.A.T. front panel, when this is lit both +5V and +3V3 are within their operational range.

6.8 Resets

The various reset sources are brought together in P701_2 to generate RST BD[H] and RST BD[L]. These signals reset the C80, the QUART and the Reset Register. Sources for resets are:

- +5V Power Monitor
- +3V3 Power Monitor
- Front Panel Reset Button
- VME I/F

Both of the Power Monitor I/C's (VR1, VR2) have built in timers that assert reset for 200ms after power-up.

After it has been initialised, the VME I/F will generate a reset when:

- SYSRESET[L] does a high to low transition
- An external VME Bus Master writes to the C.A.T.'s Reset Mailbox

When the VME I/F is held in reset by Bit 0 of the Reset Register, the C.A.T. will NOT respond to SYSRESET[L] and the Reset Mailbox cannot be activated as slave accesses are disabled.

6.9 Test Points

The following signals can be probed via test points to facilitate manufacturing test and fault finding.

On the top side of the PCB:

Test Point	Signal
TP1	Gnd
TP3	Gnd
TP4	Gnd
TP5	+5V
TP6	+3V3
TP7	-12V
TP8	+12V
TP9	Gnd
TP10	Gnd

On the back of the PCB:

Test Point	Signal	Description
TP11	RAS[L]	C80 row address strobe output
TP12	REFRESH[H]	VRAM/DRAM refresh cycle indicator
TP13	READY[H]	C80 ready input
TP14	EPROM CE[L]	Flash EPROM select line
TP15	RST BD[L]	C80 reset input
TP16	QUART CE[L]	Quad UART select line
TP17	RL[L]	C80 row address latch output
TP18	RETRY[L]	C80 retry input
TP19	SDRAM CE[L]	SDRAM select line
TP20	LBG[L]	VIC64 local bus grant input
TP21	VIC RST OUT[L]	VIC64 reset output
TP22	SLSEL0[L]	C.A.T. extended address slave decode
TP23	DS[L]	Local bus data strobe
TP24	DSACK[L]	Local bus acknowledge signal
TP25	SLSEL1[L]	C.A.T. short address slave decode

6.10 C.A.T. Power Up Sequence

As the +5V power comes up the card is held in reset by the power supply monitor I.C. VR1. This I.C. monitors the +5V rail and has a built in timer such that reset is asserted until the +5V has been above 4.6V for 200ms. When VR1 removes it's reset the 3V3 power supply is enabled. At this time the card is still held in reset by the second power supply monitor I.C. VR2. This I.C. monitors the +3V3 power supply output and also has a built in timer. When the +3V3 rail has been above 3.0V for 200ms, VR2 removes its reset. When both the +5V and +3V3 power supply monitors have removed their resets, the green LED on the front panel is illuminated, and the reset is removed from the C80.

From this point onwards, the initialisation sequence is the same as for a reset.

During reset the C80 tristates all its data, address and control lines apart from the clock output and the STATUS lines. On completion of reset all the tristated signals are driven to their inactive level. The C80 will then execute at least 16 refresh cycles to initialise external memory.

On either the first or second refresh cycle (depending on the state of TFRESH[H]) the configuration FPGA (P701_2) will give the C80 the cycle type for Synchronous DRAM (CT3..0 = 0 0 1). The C80 will respond by generating two special cycles to initialise the SDRAMs. Firstly a DCAB cycle is generated to deactivate the SDRAMs, then a MRS cycle is generated to program the Mode Register within the SDRAMs. The C80 will then continue it's 16 refresh cycles, alternating between DRAM refresh cycles and SDRAM cycles as controlled by P701_2 and the TFRESH[H] signal.

On completion of the refresh cycles, the C80 Master Processor (MP) will attempt to read it's reset instruction form the top of memory. This is stored in the top 8 bytes of the Flash EPROMs at 0xFFFF FFF8 to 0xFFFF FFFF, however the MP instruction cache is always filled with a 64 byte burst, so the C80 reads in the top 64 bytes of memory i.e. 0xFFFF FFC0 to 0xFFFF FFFF. As the EPROMs are arranged as 16 bits wide this takes 32 read cycles. The EPROM locations from 0xFFFF FFC0 to 0xFFFF FFF7 are filled with test data so that the EPROMs output can be checked with an oscilloscope.

The reset instruction will cause the C80 to start executing code from the EPROMs at the defined start address for the Monitor Firmware, this will be somewhere in the range 0x1F00 0000 to 0x1F0F FFFF. Again the code will be read into the MP instruction cache in 64 byte bursts. The Monitor Firmware will start by trying to initialise its stack and static variables in SDRAM. It will then go on to initialise the QUART so that it can report the results of the Built In Test.

On completion of the Built In Test, the Monitor Firmware will initialise the C.A.T. and any Display Board that is present using parameters from the EERAM. The Firmware may then go on to execute an embedded application or it may await the download of an application via VME.

APPENDIX A - MEMORY MAP

C.A.T. Memory Map

Base Address (hex)	Width/bits	Size/Bytes	Function
0000 0000	32/64	32M	C80 internal memory and registers
0200 0000	64	8 or 32M	Synchronous DRAM (code/data)
0400 0000	64	0 - 16M	Underlay Framestore
0500 0000	64	0 - 16M	Middle Framestore
0600 0000	64	0 - 16M	Overlay Framestore
0800 0000	8	0 - 16M	Daughterboard 1 registers/prom
0900 0000	8	0 - 16M	Daughterboard 2 registers/prom
0A00 0000	32	512k	VME I/F shared SRAM.
0B00 0000	32	8	VME Slave Page registers
0C00 0000	32	4	VME IACK address
0D00 0000	8	8k	VME I/F shared EERAM and registers
0E00 0000	8	64k	VME Short Address Window (Byte)
0E01 0000	16	64k	VME Short Address Window (Word)
0E02 0000	32	64k	VME Short Address Window (LWord)
1000 0000	32	512k	Video CLUTs
1100 0000	8	0 - 512	Video FPGAs
1200 0000	16	8	Video Palette DAC and Cursor
1D00 0000	8	1	Reset Register
1E00 0000	8	64	Quad UART (QUART)
1F00 0000	16	1M	Flash EPROM
2000 0000	64/32/16/8	0 - 64M	Daughterboard 1 data
3000 0000	64/32/16/8	0 - 64M	Daughterboard 2 data
4000 0000	8	1G	VME Extnd Address Window (Byte)
8000 0000	16	1G	VME Extnd Address Window (Word)
C000 0000	32	1G	VME Extnd Address Window (LWord)
FF00 0000	16	1M	Flash EPROM (REMAP = 0)

C.A.T. Memory Configuration:

Address (Hex)	C.A.T.	AS2...0	BS1..0	CT2..0	PS3...0
0000 0000 0100 0000	C80 Internal memory	x x x	x x	x x x	x x x x
0200 0000 0300 0000	Synchronous DRAM	0 0 1	1 1	0 0 1	1 0 0 1
0400 0000	Underlay VRAM	0 1 0	1 1	1 1 1	1 0 1 0
0500 0000	Middle VRAM	0 1 0	1 1	1 1 1	1 0 1 0
0600 0000	Overlay VRAM	0 1 0	1 1	1 1 1	1 0 1 0
0800 0000	App Daughterboard 1 - Reg	0 0 0	0 0	1 1 1	1 0 0 0
0900 0000	App Daughterboard 2 -Reg	0 0 0	0 0	1 1 1	1 0 0 0
0A00 0000	VME shared SRAM	0 0 0	1 0	1 1 1	0 1 1 0
0B00 0000	VME Slave Page Registers	0 0 0	1 0	1 1 1	1 0 0 0
0C00 0000	VME IACK address	0 0 0	1 0	1 1 1	1 0 0 0
0D00 0000	VME shared EERAM	0 0 0	1 0	1 1 1	1 0 0 0
0E00 0000	VME Short Address Window	0 0 0	1 0	1 1 1	1 0 0 0
1000 0000	Video Colour LUT	0 1 0	1 0	1 1 1	1 0 0 0
1100 0000	Video FPGAs	0 1 0	0 0	1 1 1	1 0 0 0
1200 0000	Video DAC	0 0 0	0 1	1 1 1	1 0 0 0
1D00 0000	Reset Register	0 0 0	0 0	1 1 1	1 0 0 0
1E00 0000	QUART	0 0 0	0 0	1 1 1	1 0 0 0
1F00 0000	Flash EPROM	0 0 0	0 1	1 1 1	1 0 0 0
2000 0000	App Daughterboard 1 - Data		Defined	by App	DB 1
3000 0000	App Daughterboard 2 - Data		Defined	by App	DB 2
4000 0000 FFFF FFFF	VME Extended Address Window (REMAP = 1)	0 0 0	1 0	1 1 1	1 0 0 0
FF00 0000 FFFF FFFF	Flash EPROM (REMAP = 0)	0 0 0	0 1	1 1 1	1 0 0 0

APPENDIX B - EERAM AND VME I/F FPGA REGISTERS

The following 64 locations are visible to both VME Short address space and to the C80, only every fourth location is accessed.

The C.A.T. card has two EERAM write protect jumpers. Jumper A protects those parameters that are set at the time of manufacture. Jumper B protects the user defined parameters.

Addr	Use	Write protect	TYP value	Default value	Comment
0x00	C.A.T. Card Mailbox Reset	None			FPGA register, write only
0x04	C.A.T. Card Mailbox Interrupt	None			FPGA register, read/write
0x08	Interrupt pending register	None			FPGA register, read/write
0x0C	VME Master functions	None			FPGA register, read/write
0x10	VME Master page register	None			FPGA register, read/write
0x14	Read User Links	None			FPGA register, read only
0x18	Read H/W Configuration Links	None			FPGA register, read only
0x1C	*Reserved*	None			
0x20 to 0x38	Display Board hardware parameters. Factory Set	A			Contents depends on Display Board fitted
0x3C	Checksum for locations 20 - 3C	A			
0x40 to 0x78	Display Board Video Timing Parameters. User Set	B			Contents depends on Display Board fitted
0x7C	Checksum for locations 40 - 7C	B			
0x80	Board Type (MSB)	A	0x07		Hex value 0x0701 for
0x84	Board Type (LSB)	A	0x01		board part number
0x88	PCB artwork issue	A	0x01		e.g. 1,2...
0x8C	Board Version	A	0x00		e.g. 0,1,2...
0x90	SDRAM Memory size (MBytes)	A	0x08		8 = 8 MBytes
0x94	Shared SRAM size (1/4 MByte)	A	0x02		2 = ½ MByte
0x98	*Reserved*	A		0x00	
0x9C	EPROM Firmware version	A	0x01		Hex value for version
0xA0	EPROM Firmware revision	A	0x00		and revision.e.g. V:1.0
0xA4	C.A.T. card FPGA Code Version	A	0x01		e.g. 1,2,3....
0xA8	*Reserved*	A		0x00	
0xAC	Display Board (MSB)	A	0x07		Hex value 0x0705 for Display
0xB0	Display Board (LSB)	A	0x05		Board part number
0xB4	Video FPGA code version	A	0x02		E.g. 02.03

Addr	Use	Write protect	TYP value	Default value	Comment
0xB8	Video FPGA code revision	A	0x03		
0xBC	Checksum for locations 80 - BC	A			
0xC0	App Daughterboard 1 (MSB)	B			Hex value for application
0xC4	App Daughterboard 1 (LSB)	B			Daughterboard 1 part number
0xC8	App Daughterboard 2 (MSB)	B			Hex value for application
0xCC	App Daughterboard 2 (LSB)	B			Daughterboard 2 part number
0xD0	VME Slave Short Address	B		0xBE	VME base addr = 0xBE00
0xD4	VME Slave Extnd Address (MSB)	B		0x1C	VME base addr = 0x1C800000
0xD8	VME Slave Extnd Address (LSB)	B		0x80	
0xDC	Interrupt Vector	B		0x00	
0xE0	Interrupt Handler Levels	B		0x00	Enable/disable levels 1-7
0xE4	BREQ Level, Bus release, Interrupter Level	B		0x00	Level 0-3, ROR/RWD/ROC, Level 1-7
0xE8	Slot 1 Arb, Timeout, Fairness	B		0x00	Priority/Round Robin etc.
0xEC	Checksum for locations C0 - EC	B			
0xF0	*Reserved for Monitor*	B			
0xF4	*Reserved for Primalib*	B			
0xF8	*Reserved for Primalib*	B			
0xFC	*Reserved for Primalib*	B			

The FPGA registers are more fully defined in section 5.6.12 of this manual.

APPENDIX C - CONNECTOR PINOUTS

Serial RS-232 Connector (P3)

Pin	Signal	Direction
1	Ch A - Clear To Send	input to C.A.T.
2	Ch A - Receive Data	input to C.A.T.
3	Ch B - Transmit Data	output from C.A.T.
4	Ch B - Receive Data	input to C.A.T.
6	GND	
7	Ch A - Ready To Send	output from C.A.T.
8	Ch A - Transmit Data	output from C.A.T.
9	Ch A - Data Terminal Ready	output from C.A.T.

Note (i) Mating connector should be a 9 Way Micro D-Type
E.g. ITT Cannon part No. MDSM-9SC-Z11-VS1

Note (ii) All I/O signals on this connector are protected by 18V Varistors to 0V

Keyboard/Mouse Connector (P4)

Pin	Signal	Direction
1	+5V	Fused power outlet
2	GND	
3	Ch C - Receive Data (Kbd)	Input to C.A.T.
4	Ch C - Transmit Data (Kbd)	Output from C.A.T.
5	Ch D - Receive Data (Mouse)	Input to C.A.T.
6	+5V	Fused power outlet
7	Ch D - Transmit Data	Output from C.A.T.
8	GND	

Note (i) Mating connector is 8 Way 2.5mm pitch socket
E.g. Toby XH250-08H crimp housing and XH250T-010 crimp contacts

Note (ii) All I/O signals on this connector are protected by diodes to +5V and 0V

VME P1

Only signals used by C.A.T. are shown. The signals in this table are given their standard VME bus names, on the schematics these signal names are prefixed by 'E' e.g. EBBSY[L].

Pin	Row A	Row B	Row C
1	D0[H]	BBSY[L]	D8[H]
2	D1[H]	BCLR[L]	D9[H]
3	D2[H]	ACFAIL[L]	D10[H]
4	D3[H]	BG0IN[L]	D11[H]
5	D4[H]	BG0OUT[L]	D12[H]
6	D5[H]	BG1IN[L]	D13[H]
7	D6[H]	BG1OUT[L]	D14[H]
8	D7[H]	BG2IN[L]	D15[H]
9	GND	BG2OUT[L]	GND
10	SYSCLK[H]	BG3IN[L]	SYSFAIL[L]
11	GND	BG3OUT[L]	BERR[L]
12	DS1[L]	BR0[L]	SYSRESET[L]
13	DS0[L]	BR1[L]	LWORD[L]
14	WRITE[L]	BR2[L]	AM5[H]
15	GND	BR3[L]	A23[H]
16	DTACK[L]	AM0[H]	A22[H]
17	GND	AM1[H]	A21[H]
18	AS[L]	AM2[H]	A20[H]
19	GND	AM3[H]	A19[H]
20	IACK[L]	GND	A18[H]
21	IACKIN[L]		A17[H]
22	IACKOUT[L]		A16[H]
23	AM4[H]	GND	A15[H]
24	A7[H]	IRQ7[L]	A14[H]
25	A6[H]	IRQ6[L]	A13[H]
26	A5[H]	IRQ5[L]	A12[H]
27	A4[H]	IRQ4[L]	A11[H]
28	A3[H]	IRQ3[L]	A10[H]

Pin	Row A	Row B	Row C
29	A2[H]	IRQ2[L]	A9[H]
30	A1[H]	IRQ1[L]	A8[H]
31	-12V		+12V
32	+5V	+5V	+5V

VME P2

Row B signals used by C.A.T. are shown here. Row A and C signals are connected to the Application Daughterboard Site and may be used by Daughterboards - consult the relevant Technical Manual.

Pin	Row A	Row B	Row C
1		+5V	
2		GND	
3			
4		A24[H]	
5		A25[H]	
6		A26[H]	
7		A27[H]	
8		A28[H]	
9		A29[H]	
10		A30[H]	
11		A31[H]	
12		GND	
13		+5V	
14		D16[H]	
15		D17[H]	
16		D18[H]	
17		D19[H]	
18		D20[H]	
19		D21[H]	
20		D22[H]	
21		D23[H]	
22		GND	
23		D24[H]	
24		D25[H]	
25		D26[H]	
26		D27[H]	
27		D28[H]	
28		D29[H]	

Pin	Row A	Row B	Row C
29		D30[H]	
30		D31[H]	
31		GND	
32		+5V	

Display Board I/F Connectors

P9 **P10**

Pin	Signal	Pin	Signal		Pin	Signal	Pin	Signal
1	+5V	2	+5V		1	+5V	2	+5V
3	VDB_DATA0	4	VDB_DATA1		3	VDB_ADDR0	4	VDB_ADDR1
5	GND	6	VDB_DATA2		5	GND	6	VDB_ADDR2
7	VDB_DATA3	8	VDB_DATA4		7	VDB_ADDR3	8	VDB_ADDR4
9	VDB_DATA5	10	VDB_DATA6		9	VDB_ADDR5	10	VDB_ADDR6
11	VDB_DATA7	12	GND		11	VDB_ADDR7	12	GND
13	VDB_DATA8	14	VDB_DATA9		13	VDB_ADDR8	14	VDB_ADDR9
15	VDB_DATA10	16	VDB_DATA11		15	VDB_ADDR10	16	VDB_ADDR11
17	GND	18	VDB_DATA12		17	GND	18	VDB_ADDR12
19	VDB_DATA13	20	VDB_DATA14		19	VDB_ADDR13	20	VDB_ADDR14
21	VDB_DATA15	22	VDB_DATA16		21	VDB_ADDR15	22	VDB_ADDR16
23	VDB_DATA17	24	GND		23	VDB_ADDR17	24	GND
25	VDB_DATA18	26	VDB_DATA19		25	VDB_ADDR18	26	VDB_ADDR19
27	VDB_DATA20	28	VDB_DATA21		27	VDB_ADDR20	28	VDB_ADDR21
29	GND	30	VDB_DATA22		29	GND	30	VDB_ADDR22
31	VDB_DATA23	32	VDB_DATA24		31	VDB_ADDR23	32	RAS_VU[L]
33	VDB_DATA25	34	VDB_DATA26		33	RAS_VO[L]	34	RAS_VR[L]
35	VDB_DATA27	36	GND		35	VDB_RL[L]	36	GND
37	VDB_DATA28	38	VDB_DATA29		37	VDB_WE[L]	38	VDB_TRG[L]
39	VDB_DATA30	40	VDB_DATA31		39	VDB_DSF[H]	40	VDB_CAS0[L]
41	GND	42	VDB_DATA32		41	GND	42	VDB_CAS1[L]
43	VDB_DATA33	44	VDB_DATA34		43	VDB_CAS2[L]	44	VDB_CAS3[L]
45	VDB_DATA35	46	VDB_DATA36		45	VDB_CAS4[L]	46	VDB_CAS5[L]
47	VDB_DATA37	48	GND		47	VDB_CAS6[L]	48	GND
49	VDB_DATA38	50	VDB_DATA39		49	VDB_CAS7[L]	50	FPGA_RD[H]
51	VDB_DATA40	52	VDB_DATA41		51	FPGA_WR[L]	52	CLUT_CE[L]
53	GND	54	VDB_DATA42		53	GND	54	WE7[L]
55	VDB_DATA43	56	VDB_DATA44		55	WE6[L]	56	WE5[L]
57	VDB_DATA45	58	VDB_DATA46		57	WE4[L]	58	DAC_CE[L]
59	VDB_DATA47	60	GND		59	R/W[H]	60	GND
61	VDB_DATA48	62	VDB_DATA49		61	VDB_DIR[H]	62	VDB_DEN[L]
63	VDB_DATA50	64	VDB_DATA51		63	VDB_VSYNC[L]	64	VDB_HSYNC[L]
65	GND	66	VDB_DATA52		65	GND	66	VDB_HBLANK[L]
67	VDB_DATA53	68	VDB_DATA54		67	VDB_FCLK[H]	68	VDB_CAREA[H]

Pin	Signal	Pin	Signal		Pin	Signal	Pin	Signal
69	VDB_DATA55	70	VDB_DATA56		69	VDB_SCLK[H]	70	VDB_VBLANK[L]
71	VDB_DATA57	72	GND		71	REFRESH[H]	72	GND
73	VDB_DATA58	74	VDB_DATA59		73	-12V	74	+12V
75	VDB_DATA60	76	VDB_DATA61		75		76	
77	GND	78	VDB_DATA62		77	+3V3	78	+3V3
79	VDB_DATA63	80	+5V		79	+5V	80	+5V

Application Daughterboard Connectors

P6 **P7**

Pin	Signal	Pin	Signal		Pin	Signal	Pin	Signal
1	DB_DATA0	2	DB_DATA1		1	+5V	2	+5V
3	GND	4	DB_DATA2		3	+5V	4	+5V
5	DB_DATA3	6	DB_DATA4		5	DB_ADDR0	6	DB_ADDR1
7	DB_DATA5	8	DB_DATA6		7	GND	8	DB_ADDR2
9	DB_DATA7	10	GND		9	DB_ADDR3	10	DB_ADDR4
11	DB_DATA8	12	DB_DATA9		11	DB_ADDR5	12	DB_ADDR6
13	DB_DATA10	14	DB_DATA11		13	DB_ADDR7	14	GND
15	GND	16	DB_DATA12		15	DB_ADDR8	16	DB_ADDR9
17	DB_DATA13	18	DB_DATA14		17	DB_ADDR10	18	DB_ADDR11
19	DB_DATA15	20	DB_DATA16		19	GND	20	DB_ADDR12
21	DB_DATA17	22	GND		21	DB_ADDR13	22	DB_ADDR14
23	DB_DATA18	24	DB_DATA19		23	DB_ADDR15	24	DB_ADDR16
25	DB_DATA20	26	DB_DATA21		25	DB_ADDR17	26	GND
27	GND	28	DB_DATA22		27	DB_ADDR18	28	DB_ADDR19
29	DB_DATA23	30	DB_DATA24		29	DB_ADDR20	30	DB_ADDR21
31	DB_DATA25	32	DB_DATA26		31	GND	32	DB_ADDR22
33	DB_DATA27	34	GND		33	DB_ADDR23	34	DB_CAS0[L]
35	DB_DATA28	36	DB_DATA29		35	DB_CAS1[L]	36	DB_CAS2[L]
37	DB_DATA30	38	DB_DATA31		37	DB_CAS3[L]	38	GND
39	GND	40	DB_DATA32		39	DB_CAS4[L]	40	DB_CAS5[L]
41	DB_DATA33	42	DB_DATA34		41	DB_CAS6[L]	42	DB_CAS7[L]
43	DB_DATA35	44	DB_DATA36		43	GND	44	DB_DSF[H]
45	DB_DATA37	46	GND		45	DB_RL[L]	46	DB_DIR[L]
47	DB_DATA38	48	DB_DATA39		47	DB_W[L]	48	DB_DEN[L]
49	DB_DATA40	50	DB_DATA41		49	DB_TRG[L]	50	GND
51	GND	52	DB_DATA42		51	DB1_RAS[L]	52	DB_CLK[H]
53	DB_DATA43	54	DB_DATA44		53	DB1_SEL[L]	54	DB2_RAS[L]
55	DB_DATA45	56	DB_DATA46		55	GND	56	DB2_SEL[L]
57	DB_DATA47	58	GND		57	DB1_CE_REG[L]	58	DB2_CE_REG[L]
59	DB_DATA48	60	DB_DATA49		59	DB1_GRANT[H]	60	DB2_GRANT[H]
61	DB_DATA50	62	DB_DATA51		61	DB1_PDT1_REQ[L]	62	GND
63	GND	64	DB_DATA52		63	DB1_PDT1_ACK[L]	64	DB2_PDT1_REQ[L]
65	DB_DATA53	66	DB_DATA54		65	DB1_REQ[L]	66	DB2_PDT1_ACK[L]
67	DB_DATA55	68	DB_DATA56		67	GND	68	DB2_REQ[L]

Pin	Signal	Pin	Signal		Pin	Signal	Pin	Signal
69	DB_DATA57	70	GND		69	DB1_READY[H]	70	DB2_READY[H]
71	DB_DATA58	72	DB_DATA59		71	DB1_EINT[L]	72	DB2_EINT[L]
73	DB_DATA60	74	DB_DATA61		73	DB1_RST[L]	74	GND
75	DB_DATA62	76	DB_DATA63		75	DB1_PDT2_REQ[L]	76	RST_DB2[L]
77	DB1_AS0[H]	78	DB2_AS0[H]		77	REFRESH[H]	78	DB2_PDT2_REQ[L]
79	DB1_AS1[H]	80	DB2_AS1[H]		79	GND	80	DB2_PDT2_ACK[L]
81	DB1_AS2[H]	82	DB2_AS2[H]		81	DB1_PDT2_ACK[L]	82	
83	DB1_BS0[H]	84	DB2_BS0[H]		83		84	DB_CAREA[H]
85	DB1_BS1[H]	86	DB2_BS1[H]		85	+3V3	86	+3V3
87	DB1_CT0[H]	88	DB2_CT0[H]		87	DB_FCLK[H]	88	DB_SCLK[H]
89	DB1_CT1[H]	90	DB2_CT1[H]		89	+12V	90	+12V
91	DB1_CT2[H]	92	DB2_CT2[H]		91	DB_VSYNC[L]	92	DB_HBLANK[L]
93	DB1_PS0[H]	94	DB2_PS0[H]		93	-12V	94	-12V
95	DB1_PS1[H]	96	DB2_PS1[H]		95	DB_VBLANK[L]	96	DB_HSYNC[L]
97	DB1_PS2[H]	98	DB2_PS2[H]		97	+5V	98	+5V
99	DB1_PS3[H]	100	DB2_PS3[H]		99	+5V	100	+5V

P8

Pin	Signal	Pin	Signal
1	P2 row a pin 1	2	P2 row c pin 1
3	P2 row a pin 2	4	P2 row c pin 2
5	P2 row a pin 3	6	P2 row c pin 3
7	P2 row a pin 4	8	P2 row c pin 4
9	P2 row a pin 5	10	P2 row c pin 5
11	P2 row a pin 6	12	P2 row c pin 6
13	P2 row a pin 7	14	P2 row c pin 7
15	P2 row a pin 8	16	P2 row c pin 8
17	P2 row a pin 9	18	P2 row c pin 9
19	P2 row a pin 10	20	P2 row c pin 10
21	P2 row a pin 11	22	P2 row c pin 11
23	P2 row a pin 12	24	P2 row c pin 12
25	P2 row a pin 13	26	P2 row c pin 13
27	P2 row a pin 14	28	P2 row c pin 14
29	P2 row a pin 15	30	P2 row c pin 15
31	P2 row a pin 16	32	P2 row c pin 16
33	P2 row a pin 17	34	P2 row c pin 17
35	P2 row a pin 18	36	P2 row c pin 18
37	P2 row a pin 19	38	P2 row c pin 19
39	P2 row a pin 20	40	P2 row c pin 20
41	P2 row a pin 21	42	P2 row c pin 21
43	P2 row a pin 22	44	P2 row c pin 22
45	P2 row a pin 23	46	P2 row c pin 23
47	P2 row a pin 24	48	P2 row c pin 24
49	P2 row a pin 25	50	P2 row c pin 25
51	P2 row a pin 26	52	P2 row c pin 26
53	P2 row a pin 27	54	P2 row c pin 27
55	P2 row a pin 28	56	P2 row c pin 28
57	P2 row a pin 29	58	P2 row c pin 29
59	P2 row a pin 30	60	P2 row c pin 30
61	P2 row a pin 31	62	P2 row c pin 31

Pin	Signal	Pin	Signal
63	P2 row a pin 32	64	P2 row c pin 32
65 to 79	N/C	66 to 80	N/C

APPENDIX D - C.A.T. PROCESSOR CARD VERSIONS

701000 8MBytes of SDRAM, Stand alone front panel

701001 8MBytes of SDRAM, no front panel

701002 32MBytes of SDRAM, no front panel

701003 32MBytes of SDRAM, Stand alone front panel

701004 32MBytes of SDRAM, Semi-rugged, no front panel

APPENDIX E - CIRCUIT DIAGRAMS AND ASSEMBLY DRAWINGS

Circuit Diagram: 302-701000-02X

Assembly Details: 300-701000-02X