5.3
VAC068A Overview

5.3.1 Applications

The VAC068A is a complementary chip to Cypress's VIC068A VMEbus Interface Controller. As the VAC068A is intended to work exclusively with the VIC068A, the user should be familiar with VIC068A operation. Section 1 of this book must be used in conjunction with the VAC068A section to fully understand the operation of the chip set.

The VAC068A includes drivers and receivers to interface directly to the local address and data bus. For connection to the local address bus, the VIC068A drives the lower local address bus signals LA[7:0] and the VAC068A drives the upper local address bus signals LA[31:8]. For the local data bus, the VIC068A drives LD[7:0] and the VAC068A drives LD[31:8].

For the VMEbus address signals, the VIC068A drives A[7:1] and the VAC068A drives A[31:8]. VMEbus data signals D[7:0] are driven directly by the VIC068A. The VAC068A uses an alternate IDbus to drive the next eight VMEbus data bits D[15:8] with an external '543. An external '245 is needed to swap between LD[15:8] and ID[15:8]. The upper VMEbus data signals D[31:16] are driven through external '543s and enabled by the VIC068A buffer control signals. Thus the only additional logic required with the VIC068A and the VAC068A for a complete VMEbus and local interface are two '543s and three '245s for direction and isolation of data signals from ID[15:8]. The VAC068A internally includes one of the two swap buffers required for unaligned transfers (see VAC068A block diagram). The internal swap buffer moves the ID bus data from ID[15:8] to LD[31:24].

The VAC068A IDbus also allows slower I/O devices to be utilized with system processor clock rates of 25 MHz or higher. This is accomplished by I/O read, I/O write, and DSACKi* programmable time delays. These parameters are set in the DSACKi* Control register along with assertion and recovery times for local I/O chip selects (IOSELi*).

When the IDbus is used to interface to slower 8-bit peripherals, certain registers must be configured. These are the DSACKi* Control register, PIO Function register, and the Device Location register. Region 5 of the VAC068A address map defines each of six different address areas for these 8-bit peripherals with separate I/O select signals for each address area. Two of these select signals have dedicated pins and the other selects are shared with the programmable I/O signals. To enable these devices on the ID bus, the corresponding bit in the Device Location register must be set.

The VAC068A provides VMEbus address decoding when functioning as a slave. This is accomplished by two separate slave select base address registers. These registers are compared to the respective VMEbus address signals and, when a match occurs, the correspond-

ing SLSELi* signal is asserted to the VIC068A. The address resolution of slave selects is 64 Kbytes; i.e., only the upper 16 bits are compared. For A16 slave selects, bits 26 and 27 of slave select 1 must be used. This is enabled in the Decode Control register.

In addition to VMEbus slave selects, the VAC068A can assert a variety of chip selects to the local module. These are activated by internal comparators that compare the local or the VMEbus address to the VAC068A address map. Two means exist for asserting the chip selects. One is when the local address matches the corresponding region address. The other is when redirection is enabled in the Decode Control register. When the VAC068A's own VMEbus slave select address is detected and redirection is enabled in the Decode Control register, the corresponding chip select is asserted. For example, the VAC068A can assert DRAMCS* (after VIC068A asserts LBR* and receives a LBG*) in response to the detection of its own slave select 0 address on the local bus. The slave select address is located in the SLSEL0 Base Address register. SLSEL1* may be redirected to EPROMCS*, DRAMCS*, SHRCS*, or VSBSEL* (if it is enabled in the Decode Control register) when the slave select 1 address is detected on the VMEbus address bus.

The VAC068A also provides address decode for local I/O devices (IOSEL0–5*), the VIC068A (CS*), and local resources. The upper address range is reserved for VMEbus short address space (A16), and can be programmed for either automatic D32/D16 decode (based on A16), D16, or D32 data path.

The VAC068A includes local and VME address counters required to support VMEbus block transfers. These counters automatically increment when a 256-byte boundary is crossed. The VAC068A also contains a dual-address path that allows VIC068A master accesses to the VMEbus during the interleave period of a block transfer.

The VAC068A may be used in conjunction with the VIC068A for local DMA transfers. This includes transfers from memory to local I/O or across the VSB (VME Subsystem Bus). That is, the VIC068A/VAC068A architecture allows design of VSB and I/O LSI devices that utilize DMA capability.

The VAC068A decodes processor and VIC068A signals to identify the current state of the local module. For example, the function codes are decoded from the VIC068A to interpret whether the current local cycle is a slave block transfer, normal slave transfer, local DMA cycle, or DRAM refresh cycle. The result of this decode is used to determine the slave cycle type and to assert the signals LDMACK* (local DMA activity) or REFGT* (refresh grant).

FC2	*FC1*	*CYCLE*
0	0	Slave Block Transfer
0	1	Local DMA
1	0	Slave Access
1	1	DRAM Refresh

The VAC068A includes circuitry to decode the processor's function codes. This is used to supply floating-point-unit chip select (FPUCS*) to the modules coprocessor.

The VAC068A also identifies if the current cycle is an interrupt acknowledge cycle. This is done through the assertion of FCIACK* by the VAC068A when any one of these three conditions occur:

- function code bits 0–2 are all set to 1s
- bit 30 of the PIO Direction register is set and an access is made to $FFFF FFxx and function code inputs are NOT all set to 1s
- bit 31 of the PIO Function register is set and there is an access to local I/O 5 address space

The VAC068A local interrupts can occur on any of three interrupt-request output signals: PIO7, PIO10, or PIO11. These signals are enabled as output interrupt signals through the PIO Function register.

Enabling/disabling specific interrupts is accomplished in the Interrupt Control register. This register also specifies the mapping of interrupts on PIO7, 10, or 11. Normally, the PIO7, 10, or 11 signals are connected to the VIC068A LIRQi* signals. The Interrupt Status register may be read to determine which interrupt condition is causing the output to be asserted. The conditions causing these interrupts are:

- programmable timer
- mailbox access
- UART channel A or B service
- programmable I/O signals PIO4, PIO7, PIO8, or PIO9

The timer, mailbox, and PIO4, 7, 8, and 9 interrupts are active Low and edge triggered. The interrupt request goes inactive when the local processor clears its respective bit in the Interrupt Control register.

To disable the UART channel A and B interrupts, the user must first determine what caused the UART interrupt. The cause of the interrupt is located in the respected UART Serial I/O Channel Interrupt Status register. This register is read-only. Once the cause is determined, the respective bit in the UART Serial I/O Channel Interrupt Mask register must be disabled. Upon completion of this activity, the Interrupt Control register bit may be cleared.

Other signals the VAC068A decodes include:

- BLT* (local block transfer address counter)
- LBG* (VIC068A local bus activity)

The VAC068A also decodes the VIC068A buffer control signals. These include:

- SWDEN* (swap data enable)
- DDIR (data direction)
- LAEN (local address enable)
- LADO (latch address out)
- LADI (latch address in)
- ABEN* (VME address bus enable)

The VAC068A also furnishes status and size signals to the VIC068A and the local processor. These include:

- WORD* for data size
- ASIZ1/0 for address size
- LDMACK* for local DMA activity
- CACHINH* cache inhibit signal
- REFGT* for refresh grant

The VAC068A also includes shared-function programmable I/O signals. These dual-function pins are defined in the PIO Function register. When not serving as general-purpose I/O signals they may be used for:

- dual UART channels (A and B)
- I/O read enable for local I/O
- I/O write enable for local I/O
- local I/O chip selects (2–5)
- shared-resource chip select

The VAC068A also includes a timer with prescaler, programmable DSACKi* control, and a chip select (ICFSEL*) for access to the VIC068A global and module switches from the VME-bus.

5.3.2 VMEbus Address Decoding

The VAC068A's VMEbus address map consists of an address region for VMEbus A16 master cycles. The A24 Space Base Address register specifies where the A24 address overlay is for A24 master cycles. This register also contains the bits to program the data size of the A16 and A24 master cycles. An automatic decode of the data size is accomplished through the VAC068A by monitoring LA[16] for A16 space and LA[24] for A24 space. If this address signal is High, a D16 data path is selected (WORD* asserted). If this address bit is Low, a D32 data path is selected (WORD* deasserted). A fixed data size option also exists.

There are three programmable regions that can be programmed to assert MWB*, VSBSEL*, or SHRCS*. MWB* signals the VIC068A to acquire the VMEbus for a master access. Any or all regions are capable of this operation. VSBSEL* and SHRCS* are chip selects to module resources.

The VAC068A also has two slave select base address registers, two slave select mask registers, and an interprocessor communications select address register. The slave select base address registers and mask registers are used to decode valid slave selects and generate SLSELi* to the VIC068A. Any of A32, A24, or A16 slave access can occur. The ICFSEL address register is used to access the VIC068A global and module switches from the VMEbus.

5.3.2.1 Master Access

There are five types of master accesses that may be accomplished with the VAC068A. They are:

- A32 address with programmable data size
- A24 address with D16
- A24 address with D32
- A16 address with D32
- A16 address with D16

5.3.2.2 Programmable VMEbus Space

Any or all of the three programmable regions can be used to set up VMEbus accesses by defining an address range and programming the attribute register to assert MWB*. The address range is configured by programming the Boundary 2 or 3 Address registers. When the Region Attribute register is set for MWB* assertion, the VIC068A arbitrates for the VMEbus. The corresponding ASIZ1/0, WORD*, and CACHINH* settings are driven upon the assertion of the processor address strobe (PAS*). See Section 1 of this book for details on VMEbus master transfers.

5.3.2.3 A24 VMEbus Space

The A24 Base Address register specifies an A24 address space overlay. This overlay is 32 Mbytes in size. The data path may be D16, D32, or automatically decoded. Bit 24 of the A24 Base Address register determines if the entire region is D16 or D32. Automatic decode is enabled through bit 20 of the A24 Base Address register. The A24 address space data path is determined by LA[24]. If LA[24] is High, the data path is D16. If LA[24] is Low, the data path is D32. WORD* is driven accordingly on these master cycles.

This address space can be divided into two 16-megabyte blocks if automatic decode of LA[24] is enabled. The first 16 Mbytes are D32 and the second 16 Mbytes are D16. If automatic datapath decoding is not used, bit 24 of the register determines the data path of the entire 32-Mbyte overlay. The A24 address space may be overlaid on the top of any of the three programmable regions (1, 2, or 3) regardless of what it is set for (i.e., VSBSEL*, MWB*, SHRCS*, or inactive).

The VAC068A compares the upper 7 bits [31:25] of the A24 Base Address register to local address bits LA[31:25] and, upon a match, overlays 32 Mbytes of A24 address space on the programmed region. When this overlay occurs, MWB* is asserted and an A24 VMEbus master cycle takes place. The VAC068A drives WORD*, ASIZ1/0, and CACHINH* as specified by the settings of the A24 Base Address register.

The requirements for forcing A24 address accesses are that it must fall between the address range of $02 and $FE00 0000 and that the A24 overlay address is above the DRAM region (region 0). This address space decode cannot be disabled.

As in a programmable master access, MWB* is asserted and the VIC068A requests the VMEbus and initiates the transfer per VMEbus protocol.

5.3.2.4 A16 VMEbus Space

The upper address space, starting at $FFFE 0000 and continuing to $FFFF FFF0, is reserved for A16 VMEbus accesses. This address space is fixed and should not be used for any other purpose. There are two ways to set up the data size for A16 master accesses. The first is to allow the VAC068A to automatically decode LA[16]. This is similar to the automatic decode of the A24 VMEbus space. If LA[16] is High, the data path size is D16 (WORD* asserted). If LA[16] is Low, the data path size is D32 (WORD* deasserted). This decode reflects region 6 of the VAC068A address map as A16/D32. The other option is to set bit 22 in the A24 Space Base Address register. This enables bit 21 of the A24 Space Base Address register to control the data path size. If bit 21 is set, the data path is D32 (WORD* deasserted). If bit 21 is clear, the data path is D16 (WORD* asserted). CACHINH* is always asserted during VMEbus A16 access and is not programmable.

Region 6 and the programmable regions provide the only way for an A16 access to take place when using the VAC068A. As before, MWB* assertion triggers the VIC068A to acquire the VMEbus and initiate the VMEbus transfer.

5.3.3 VMEbus Slave Access

The VAC068A contains four registers to define VMEbus slave address decode. These are the Slave Select 1 and Slave Select 0 Base Address registers, and Slave Select 1 and Slave Select 0 Address Mask registers. The VAC068A also contains an Interprocessor Communications Select register. This is used to access the VIC068A global switches, module switches, and communication registers.

The base address registers are loaded with the address that determines a valid slave select to assert SLSEL0* or SLSEL1*. The mask registers are used to qualify which bits in the base address register are to be compared to its respective VMEbus address signal. When a bit is set in the mask register, it allows the base address register contents to be compared to the corresponding VMEbus address signal. If the compare matches, the corresponding SLSELi* is asserted. If clear, no compare is made and the VAC068A does not care what value is on that particular VMEbus address signal. It is important to use these mask bits with care. It is possible to get multiple slave selects if a small number of mask bits are set (i.e., 0s in the high-order mask bits).

The lower VMEbus address bits may also be compared for an A16 slave access. This is accomplished by setting bit 26 of the Decode Control register. When this bit is set, VMEbus A[15:8] is compared to the Slave Select 1 Base Address register bits [31:24]. This allows the user to decode VMEbus short address space and assert any of DRAM, EPROM, VSB, or shared resources chip selects. The chip select is determined using bits 28–29 in the Decode Control register.

The VAC068A address decoder constantly compares the register values to the VMEbus address, and when a match occurs it asserts the proper slave select signal. When SLSELi* is asserted to the VIC068A, the VIC068A asserts LBR* (qualified by VMEbus AS*), which is also connected to the VAC068A's LBR* signal. When LBG* is asserted by the modules local bus arbiter, the following sequence occurs:

1. The VIC068A asserts LAEN with valid FC2/1, SIZ1/0, and LA[7:0].
2. The VAC068A asserts LA[31:8].
3. The VIC068A asserts PAS* and DS*.
4. The VAC068A drives ASIZ1/0 and WORD*, then asserts the proper chip select.
5. The VAC068A asserts DSACK0/1* (if enabled).
6. The VIC068A times out SAT delay.
7. The VIC068A asserts DTACK* and VAC068A deasserts DSACK0/1* with PAS* deassertion.
8. the cycle completes, VAC068A three-states its address signals

Slave select 0 is always associated with DRAMCS*. This assumes that a local bus grant on assertion of SLSEL0* results in an access of local DRAM. Similarly, slave select 1 can be enabled to assert any of the following chip selects:

- SHRCS*
- EPROMCS*
- DRAMCS*
- VSBSEL*

This chip select is asserted according to the Decode Control register bits 28–29 (FFFD 14xx).

Slave select 1 is only asserted when slave select 0 is not asserted per an internal interlock. Slave select 0 is asserted for slave transfers matching its base address register (and when the proper mask bits are set) or redirection of the local address to DRAM and DRAMCS* assertion.

If both slave selects are in use and each is enabled for a different address space, it is possible for both SLSEL0* and SLSEL1* to be asserted simultaneously as in the following case:

> When bit 26 is set in the Decode Control register, the VAC068A compares the VMEbus address signals A[15:8] to SLSEL1* Base Address register bits [31:24]. If a match occurs with SLSEL0* address bits [31:24] and VME address signals A[31:24], qualified by the address mask register, then both selects are asserted. The slave select address mask register would also have to be enabled for a compare of these address bits. When this occurs, and the two slave selects are mapped to different regions, the local module must decide which slave select has precedence.

If both SLSEL0* and SLSEL1* are asserted and point to the same device (i.e., DRAMCS* via SLSEL1* redirection in the Decode Control register) or they are decoded at non-overlapping address ranges in the same address space, no conflict should arise. In this case, to access the full address space, slave select 0 should correspond to the larger address space for access to the Mailbox region.

For recognition of the slave select address on the local bus, additional comparators monitor the local bus for the SLSEL0* address range. If this function is enabled in the Decode Control register bit 19, the VAC068A asserts DRAMCS* when it recognizes a valid slave address on the local address bus. This allows the local processor to access data in DRAM at the same address as other modules on the VMEbus.

Additionally, the slave address may be determined by using the VIC068A's ability to assert LBERR* when it sees a qualified slave select. This is called self-access. The resulting operation is:

- VMEbus BERR* is driven by the VIC068A
- LBERR* is driven by the VIC068A
- the VIC068A Bus Error Status register indicates a self-access has occurred

The Interprocessor Communications Facility Select register is another type of slave select. It enables an A16 access to VIC068A internal global switches, module switches, and communication registers. This register compares VMEbus A[15:8] to each of the two bytes in the register. When a match occurs, the VAC068A asserts ICFSEL* to the VIC068A. The upper byte is normally used for global accesses to the four Interprocessor Communications Global switches. If used for this function, all VAC068A (and all other non-VAC068A modules) must be set to the same value. The lower byte is used to access the Interprocessor Communications Module switches and Interprocessor Communications registers. When used for this function, the values must be different for each VMEbus module. Section 1 of this user's guide should be referenced when accessing specific switches and registers.

5.3.4 Local Memory Map Decoding

The VAC068A segments the local processor's address space into a number of fixed- and variable-sized segments with a resolution of 64 Kbytes (i.e., bits [31:16]). Separate segments are available for:

- DRAM
- VMEbus subsystem bus select (VSBSEL*)
- shared resource chip select (SHRCS*)
- EPROM
- local I/O (I/O selects 5–0, VIC068A and VAC068A register access)

The DRAM, VSBSEL*, and Shared Resource segments are programmable. These regions occupy address space from $0000 to $FFFD FFFF. DRAM is hard-coded to start at $0000

0000, thus making it region 0. The means for sectioning VSBSEL* or SHRCS* in regions 1, 2, or 3 are in the respective Region Attribute register. Boundary Address registers 2 and 3 divide the respective contiguous regions. Each of these regions has the following attribute settings associated with it:

- cache inhibit (CACHINH*)
- address size (ASIZ1/0)
- data word size (WORD*)

In addition to the above programmable segments that may be assigned boundaries and attributes, fixed-size attribute segments are mapped to EPROM and local I/O, which include IOSEL5–0* and the VIC068A and VAC068A register maps. Areas also exist for assertion of MWB* in any of the three programmable regions, VME A16 address space, and an A24 address overlay space (see *Figure 5-2* for graphical representation).

5.3.4.1 DRAM Decode

DRAM is hard-coded to start at $0. This region continues to the value programmed into the DRAM Upper-Limit Mask register. The minimum value that can be programmed in this register is 0000 0001. A 256-byte region is always available for the DRAM mailbox area. This register value is also the starting address for region 1.

An interrupt can be enabled in the Interrupt Control register to indicate an access to the mailbox address space. DRAMCS* is asserted for $0000 00xx after the Force EPROM mode is exited.

The DRAM Upper-Limit Address register is NANDed with the local address bus LA[31:16] to determine the output of the compares. If any are Low, the access is not to DRAM. If all compares are High, then DRAM is being accessed and DRAMCS* is asserted.

There is another way to assert DRAMCS* and access DRAM address space using bits in the Decode Control register to redirect the Slave Select 0 address when it is present on the local address bus. Qualification of DRAMCS* assertion may be with or without PAS* assertion. This is programmable in the Decode Control register. DSACKi* control for DRAM access is also available in the Decode Control register. DRAM accesses may be set for a 32-bit data path, or VAC068A DSACKi* may be three-stated and external DSACKi* generation used. The three-state option is useful for processors that need synchronous termination of DRAM access as with the Motorola 68040.

For redirection of the SLSEL0* address, bit 19 in the Decode Control register must be set. The slave select base address register must be present on the local address bus. If this bit is clear, DRAMCS* is not asserted when the SLSEL0* address is present on the local bus. The correct mask bits must be set to enable a compare of the address bits.

The user may also elect to enable qualification of DRAMCS* with PAS* asserted by setting bit 30 in the Decode Control register. This bit is normally set to condition the VAC068A to look only at the address during the proper time period.

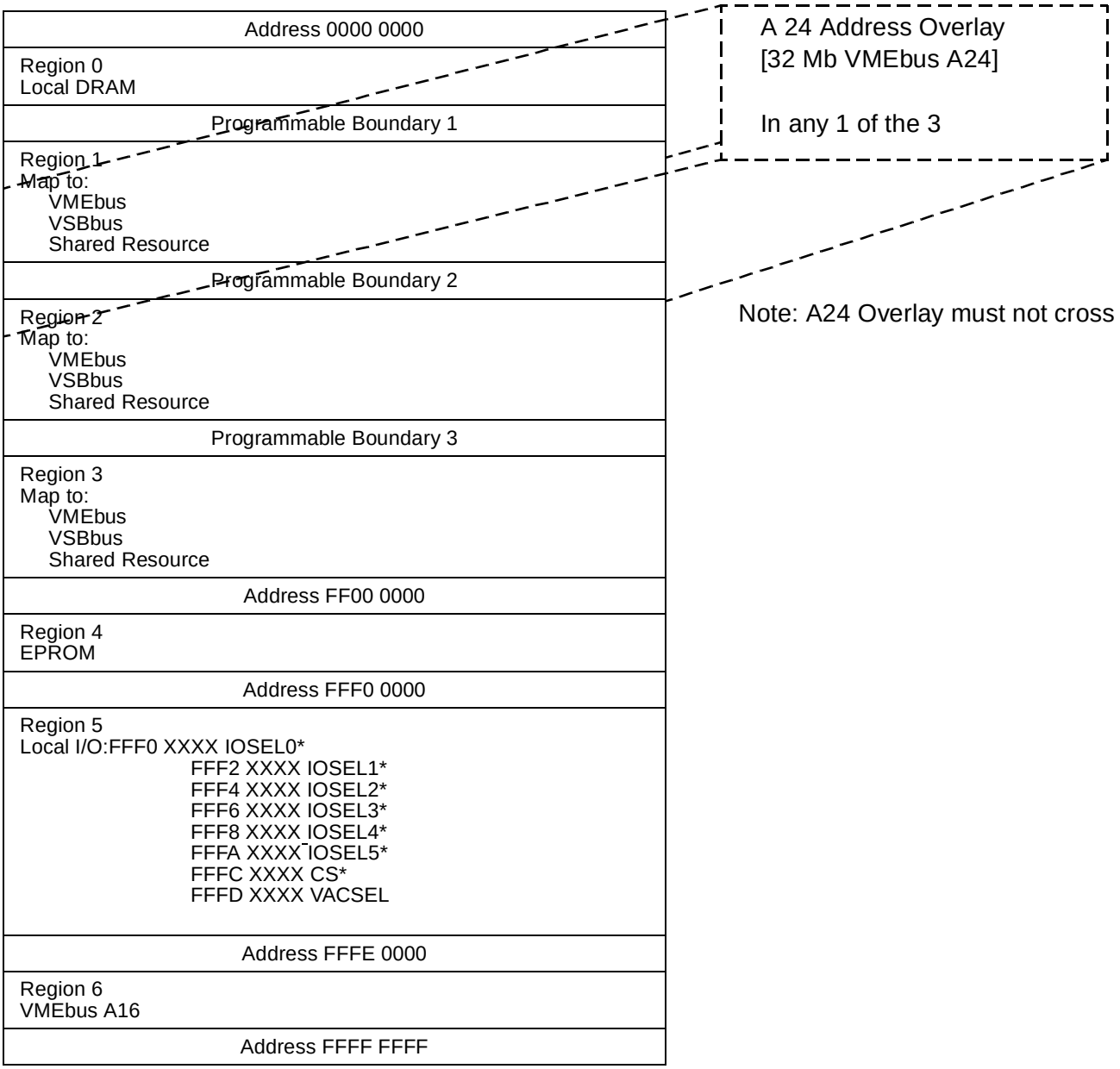

Figure 5-2. VAC068A Memory Map

It should be noted that DRAMCS* is always asserted during a VMEbus SLSEL0* cycle unless redirection is disabled. DSACKi* may also be three-stated on LAEN assertion for VMEbus slave cycles. This is accomplished using bit 31 of the Decode Control register. The user may chose to set the DSACKi* delay for DRAM accesses in the Decode Control register using bits 17–18. The time intervals are in CPUCLK cycles, from 0 to 3.

The upper 256 bytes of DRAM serve as a mailbox area. If the mailbox interrupt is enabled via the Interrupt Control register, a write to the upper 256 bytes of DRAM via SLSEL0* interrupts the local processor with the specified interrupt signal.

5.3.4.2 Programmable Decode

Following the DRAM Upper-Limit Mask register are two Boundary Address registers. These registers define the next three regions of address space that may be assigned to any of the following chip selects or VMEbus access:

- SHRCS* (Shared Resource Chip Select)
- VSBSEL* (VMEbus Subsystem Bus Select)
- inactive (not programmed)

Normally SHRCS* is used as an SRAM chip select, although it can be used for any shared resource on the module including DRAM. When used in this fashion, it has no effect on the VIC068A block transfers with local DMA.

When configured for “inactive,” the region attributes are still driven as programmed when the region is accessed. These attributes can be qualified with PAS* when enabled in the Decode Control register bit 20.

Since the DRAM Upper-Limit Mask register defines the upper address space of DRAM, it is also the starting address for region 1. This means that the Boundary 2 Address register specifies the upper limit of region 1 and the starting address of region 2. The Boundary 3 Address register specifies the upper address limit of region 2 and the starting address of region 3. Care should be exercised in programming region 3 so that it does not overlap the EPROM address space. If this occurs, EPROMCS* is asserted with the chip select programmed in the Region 3 Attribute register. If this situation arises, external logic must decode chip selects.

5.3.4.3 EPROM Decode

The EPROMCS* address starts at $FF00 0000 and continues to $FFEF FFFF. This gives the user 15 Mbytes of dedicated EPROM address space. When the user accesses this address space, the VAC068A asserts EPROMCS*. The default EPROM data size is 32 bits. Different EPROM data sizing is selected by setting signals ID8 or ID9 upon power-up reset. If ID9 is Low during RESET*, a 16-bit data path is selected. If ID8 is Low during RESET*, an 8-bit data path is selected. After reset, the user must configure the data size acknowledge in the EPROM DSACKi* Control register bits 27 and 28. It is assumed that the IORD*, IOWR*, and IOSEL* assertion/recovery times are not to be used for EPROM accesses.

The DSACKi* Control register also allows for different-speed EPROM to be used on the module. Bits 29–31 are used to set proper DSACKi* assertion timing or disable DSACKi* assertion by VAC068A altogether.

The EPROMCS* timing should be determined upon power-up by loading the DSACKi* time in the DSACKi* Control register. There is also a “Force EPROM” mode that the VAC068A initially enters upon power-up. This mode is exited by any memory access in the EPROM address space ($FF00 0000 to $FFEF FFFF).

5.3.4.3.1 Forced EPROM Mode

The map decode function is overridden at power-up to force read accesses to EPROM independent of its mapping until the EPROM address appears on the local address bus. This implies a jump to EPROM address space ($FF00 0000 to $FFEF FFFF) should be one of the first instructions in the boot-up sequence. An initialization routine is given in Chapter 20. This routine or a similar one should be initiated after exiting from system reset. When the forced EPROM mode is exited, DRAMCS* is asserted for address $0000 00xx.

5.3.4.4 Local I/O Select Decode

The local I/O address space begins at $FFF0 0000 and ends at $FFFD FFFF. IOSEL5–0* accesses, VIC068A registers, and VAC068A registers reside in this address space. This address space is available to the local module only. The IOSELi*s each occupy 128 Kbytes of address space starting at $FFF0 0000 to $FFFA 0000. The VIC068A register accesses start at $FFFC 0000 and the VAC068A register accesses start at $FFFD 0000. Thus each register map consumes 64 Kbytes of address space. The upper limit for this region is $FFFD FFFF. Each IOSEL5–0* has an individual DSACKi* Control register associated with it. IOSEL5–0* also have a Device Location Register attribute that enables the particular IOSEL5–0* device to be located on the ID bus. IOSEL5/2* do not have their own dedicated signal on the VAC068A, instead they share a PIO signal function and must be enabled in the PIO Function register. IOSEL1/0* signals have their own pin and do not share functionality.

In the DSACKi* Control register, it is possible to set IORD* and IOWR* assertion and deassertion parameters. These are commonly called “cycle end control.” Cycle end control is defined as a means to provide increased hold time to peripheral devices located on the local module. Qualification of IORD* or IOWR* deassertion is either with PAS* or when the DSACKi* assertion delay has elapsed. For assertion of IORD* or IOWR*, the delay is programmed in half CPUCLK cycles after PAS* assertion. The IORD* and IOWR* assertion and recovery delay is not used for EPROM or shared resources selects.

IOSEL5–0*s are available to the user in one of two modes of operation. The first mode is when they are located on the local address and data bus. This is the typical operating condition. The second mode is when they are used on the ID bus. This mode is used when slow peripherals reside on the local module.

When the ID bus is used, the Device Location register should be properly programmed. This register indicates to the VAC068A which device is located on the ID bus. Latching is provided internal to the VAC068A. When using the IOSEL5–0*, no I/O device access is allowed to start until the select signal for the previously accessed I/O device has been deasserted for the number of clock cycles configured in the DSACKi* Control register. Programmable attributes for IOSEL5–0*s in the DSACKi* Control register are:

- assertion delay for the IOSEL5–0* from PAS* in half CPUCLK cycles (otherwise assertion is with PAS*)
- deassertion delay (recovery time) for IOSEL5–0* inactive in CPUCLK cycles

EPROMCS* and SHRCS* devices do not use these timing parameters.

The VAC068A registers are also located in region 5 address space. They are accessed at local address $FFFD 0000 through $FFFD 0029. Accesses to these registers occur on the local address and data bus. Acknowledge occurs with the assertion of DSACK1*. These registers are not byte-accessible and must be set upon power-up to configure the VAC068A operational mode. The undefined bits are read as 1s. When all registers are configured, the VAC068A ID register must be written to enable decode and control functions.

5.3.5 Local Decode Control/Status

The VAC068A decodes function codes FC2–0 to provide status and control signals to the local module. These signals include:

- FCIACK* for local interrupt acknowledge cycle
- FPUCS* for floating-point coprocessor chip select
- REFGT* for refresh grant
- LDMACK* for local DMA activity
- CACHINH* for cache inhibit status

5.3.5.1 Function Code Decode

The VAC068A decodes FC2–0 from the processor and FC2–1 from the VIC068A. To determine when function codes from the VIC068A are to be decoded, the signal LAEN must be asserted. For local processor function decodes, LAEN is not asserted. The functions decoded on the local bus from the VIC068A are one of the following:

- slave transfer (normal)
- slave block transfer
- DRAM refresh cycle
- local DMA

When these activities are detected by the VAC068A, the proper output signals are asserted. When the VIC068A drives the function codes (defined in Section 1 of this book) the REFGT* and LDMACK* signals are decoded and asserted.

When decoding local processor cycles (LAEN deasserted), the VAC068A is capable of asserting FCIACK* and FPUCS*. These signals can only be generated when the local processor function codes specify CPU space (FC2–0 = 111). The FCIACK* signal, used for interrupt acknowledge to the VIC068A, is asserted when FC2–0 = 111 and LA[17:15] = 111. The FPUCS* signal, used to select a floating-point coprocessor, is asserted when FC2–0 = 111 and LA[17:13] = 10001.

Care should be taken when these signals are used in a robust system that makes use of coprocessors or other parts of CPU space as only those local address bus bits listed are decoded.

FPUCS* may also be asserted through an access to IOSEL* region 4. This is enabled by setting bit 31 of the PIO Function register.

Assertion of FPUCS* is further qualified by bit 16 in the Decode Control register. This allows the assertion of FPUCS* to occur either when the CPUCLK goes Low or when PAS* is asserted.

Both FCIACK* and FPUCS* are inhibited when the VIC068A owns the local bus (LAEN is asserted).

5.3.6 Programmable Input/Output

The programmable I/O (PIO) signals of the VAC068A provide a general-purpose I/O facility that can alternatively be programmed to provide IORD* (I/O read), IOWR* (I/O write), IOSEL5/2* (I/O selects), interrupt, SHRCS*, or Serial I/O (asynchronous serial I/O) functions. Multiple registers are used to configure the operation of these pins and read status of a particular function. These include PIO Function, Data Out, Pin, and Direction registers.

When the PIO signals are configured in the PIO Function register as general-purpose I/O signals, the user may choose to use these signals to support the serial I/O function on VAC068A (i.e., RTS, CTS, DCD).

Bits 30 and 31 of the PIO Function register have special functions. Bit 30 enables a debounce delay circuit associated with PIO9 (see section 5.3.8.1). When bit 31 is set, it enables the user to assert FPUCS* on accesses to IOSEL4* address space and assert FCIACK* on accesses to IOSEL5* address space. This is useful for decoding an interrupt acknowledge cycle to obtain the status/ID of a service routine.

The PIO Data Output register contains data to be put out on the PIO pins that are defined as outputs. Reading this address causes the data bus to be driven with the contents of the register. Writing to this register causes the value in the PIO Data Output register to be driven on the PIO pins that are defined as outputs in the PIO Function register.

The PIO Pin register can be read to examine the data being presented on the PIO inputs. This register is a read-only register. Reading the PIO Pin register does not affect the contents of the PIO Data Output register. Writing to the Pin register causes a DSACK1* assertion, but has no effect on the data present on the PIO pins.

The PIO Direction register specifies the direction of the PIO signals. When set the signal is an output, when cleared it is an input. This register has no effect if the corresponding PIO Function Register bit is set to disable the general-purpose I/O capability of that pin. Bit 30 of the PIO Direction register has a special function (HIACKIN). It allows FCIACK* to be asserted on accesses to $FFFF FFxx. This is used to allow non-680x0 processors to assert FCIACK*.

5.3.6.1 Serial I/O

The VAC068A contains a dual, full-duplex UART. Each channel is double-buffered on transmit and quad-buffered on receive. The UARTs always transmit two stop bits and require a single stop bit on receive.

The UART Serial I/O Channel A and B Mode registers allow configuration for:

- looping of transmitter or receiver
- break transmission or no break
- enable the transmitter/receiver pair
- transmitter/receiver reset and run
- baud rate selection from 300 to 9600 baud as well as intermediate frequencies (via the CPU Clock Divisor register)
- 7 or 8 data bits per character
- odd, even, or no parity generation and check

The CPU Clock Divisor register must be loaded with a value to produce the correct baud rate generation. Examples are given in the register description section for different CPUCLK rates.

Each UART contains a four-byte-deep FIFO (UART Channel A and B Receiver FIFO register) for receiving serial information. These FIFOs are accessed through the receiver FIFO registers. Included in these registers are indications of errors in parity, frame, and break detection. Once a character is read, the next character becomes available to read.

The Channel A and B Transmit Data registers are accessed from the local data bus. They are loaded with the data to be transmitted. When enabled to run, the next character may be written into the register.

The Interrupt Control register is configured to indicate which interrupt signal (PIO7, 10, or 11) is driven as a serial I/O interrupt. Serial I/O interrupts are not cleared by disabling the UART interrupt alone. The following sequence should be followed when handling a serial I/O interrupt:

1. Determine which UART channel (A or B) interrupt is pending in the Interrupt Status register.
2. Determine the cause of the interrupt in the UART Channel (A or B) Interrupt Status register.
3. Mask the interrupt in the UART Channel (A or B) Interrupt Mask register.
4. Service the interrupt based on the status.
5. Clear the Interrupt Control register for the specific UART interrupt.

The UART Channel A or B Interrupt Mask registers can be configured to interrupt the local module for other conditions. These conditions include:

- transmitter empty; transmitter ready
- single character received
- FIFO full
- break change
- parity error
- frame error
- overrun

5.3.6.2 I/O Select

The VAC068A incorporates individual local I/O select signals. IOSEL1* and 0 are individual signals and are not multiplexed with other function. IOSEL2* through IOSEL5* share functions with PIO13, PIO6, PIO8, and PIO9 signals, respectively. As stated previously, these signals have the ability to communicate on the local data bus LD[31:16] or the ID bus ID[15:8].

The Device Location register specifies mapping of the IOSEL5–0* signals. When a bit is set, it indicates that the respective IOSEL5–0* is located on the ID bus instead of the local data bus.

The PIO Function register controls the multiplexed signal selection. When a bit is cleared, the signals are in the general-purpose I/O mode, otherwise they have the shared function.

IOSEL4* has a special function associated with it. When PIO Function register bit 31 is set, FPUCS* is asserted on accesses to IOSEL4* address space. Additionally FCIACK* is asserted on access to IOSEL5* address space. Another way to assert FCIACK* is to set bit 30 in the PIO Direction register (HIACKEN). When this bit is set, FCIACK* is asserted on access to $FFFF FFxx.

The other signals of concern when using the IOSEL5–0* function are IORD* (I/O Read) and IOWR* (I/O Write). These signals are multiplexed with PIO3 and PIO4 respectively. Control for IORD* and IOWR* is located in the DSACKi* Control register. If these signals are to be used as IORD* and IOWR*, the PIO Function register must be set accordingly.

CACHINH* is enabled by bit 23 in the A24 Space Base Address register for accesses to region 5. This includes IOSEL5–0*, VIC068A, and VAC068A register accesses.

5.3.7 Interrupt Support

The VAC068A may be configured to generate up to three interrupt requests that are designed to connect to local interrupt request signals on the VIC068A. The interrupting functions are serial I/O, timer, mailbox, and PIO interrupt. The various interrupt requests can be multiplexed on up to three of the PIO signals, as configured in the Interrupt Control register. These interrupt requests are typically connected to the VIC068A LIRQi* signals.

5.3.7.1 Interrupt Status Register

The Interrupt Status register allows the processor to determine which of several possible events caused an interrupt. Except for the UARTs, a serviced interrupt request is withdrawn when the processor clears its mapping bit in the Interrupt Control register ($FFFD 16xx). A separate Interrupt Status register is implemented for each of the serial I/O channels (channel A at $FFFD 25xx and channel B at $FFFD 26xx). These are cleared by determining the cause in the Interrupt Status register, then masking the interrupt in the Interrupt Mask register, then clearing the interrupt in the Interrupt Control register.

5.3.7.2 PIO Interrupt

The interrupt output signals share functions with PIO7, PIO10, and PIO11. These output signals are mappable from any of the following sources:

- timer (register-controlled)
- SIO Channel A and B (register-controlled)
- mailbox (access to upper 256 bytes of DRAM address space)
- PIO4 (user defined)
- PIO7 (user defined)
- PIO8 (user defined)
- PIO9 (user defined)

An interrupt request is generated if a falling edge is present on PIO4, 7, 8, or 9 input signals and output on PIO7, 10, or 11. The PIO interrupts are enabled in the Interrupt Control register. The interrupt sources are active Low and edge-triggered (except UART channel A and B). PIO4, PIO7, and PIO8 can be used for VSB write post failure, parity error, or any other user-defined interrupt. PIO9 has a special debounce delay when enabled in the PIO Function register. A change of state on this input is not recognized until it has been stable for 255 CPUCLK cycles. This provides a debounce delay of 26.7 ms when a 9600 baud rate is generated from the CPU Clock Divisor register.

5.3.7.2.1 Timer Interrupt

The timer interrupt is enabled by loading a value into the Timer Data register and configuring the Timer Control register per the user's requirement. The timer consists of a 16-bit programmable timer with a 6-bit prescaler. Interaction with the timer is by means of a Timer Control register and a Timer Data register. The control register contains a 6-bit prescaler, which is loaded with a count value. The carry of this counter clocks the value loaded in the Timer Data register. A run/load bit and a once/continuous bit are also included. A prescale value of 0 results in a DC prescale output. When the timer control register is read, LD[15:8] are driven with the instantaneous state of the prescaler counter and the value loaded in the prescale counter. Whenever the timer overflows, or when it is disabled, the timer is loaded with the contents of the Timer Data register. When the Timer Data register is read, the data bus is

driven with the instantaneous state of the timer (e.g., the 16-bit counter). The prescaler counter is clocked by CPUCLK.

5.3.7.2.2 Serial I/O Interrupt

The SIO channel A and B interrupts are enabled in the UART Serial I/O Channel Interrupt Mask registers. UART A and B interrupts are the only interrupts that are level-sensitive and not edge-triggered. All others are edge-triggered. Any of several interrupting conditions on the UART ports may cause an interrupt. These include transmit ready, transmit shift register empty, receiver FIFO full, received character ready, break change received, and error conditions such as overrun, frame, or parity. They can be masked in the UART Interrupt Mask registers (A at $FFFD 23xx and B at $FFFD 24xx) and monitored in the UART Interrupt Status register (A at $FFFD 25xx and B at $FFFD 26xx). The SIO interrupts are cleared in the UART Interrupt Mask registers.

5.3.7.2.3 Mailbox Interrupt

The mailbox interrupt is enabled in the Interrupt Control register. It is generated by writing to the top 256 bytes of local DRAM as defined by the DRAM Upper-Limit Address register. This mailbox region is always present in the VAC068A DRAM address space. The mailbox interrupt is activated by a slave access using SLSEL0* or when the local processor's access to the SLSEL0* address is redirected to local DRAM. It may also be generated by redirecting SLSEL1* to DRAM in the Decode Control register. The CACHINH* signal is asserted on accesses to the mailbox region.

5.3.8 Miscellaneous Features

Individual features of the VAC068A including the PIO9 Debounce, ID bus, DSACKi* control, local DMA support, and IORD* and IOWR* are discussed here.

5.3.8.1 PIO9 Debounce

PIO9 has a special debounce circuit associated with it. This makes it suitable for mechanical switch interrupt input. Switch debouncing is accomplished using a counter. The counter is clocked at the maximum rate of the baud rate timing chain. Because of the debounce circuit, PIO9 must be held Low for 255 clock cycles of this counter in order to generate an interrupt. This provides a debounce circuit delay of 26.7 ms if a 9600 baud rate is generated from the CPU Clock Divisor register. The debounce delay is enabled by bit 30 of the PIO Function register.

5.3.8.2 Isolated Data Bus

The VAC068A pinout includes ID[15:8], the isolated data bus. On the module, a '245 is connected between LD[15:8] and ID[15:8] and a '543 is connected between ID[15:8] and D[15:8]. The VAC068A provides the data swap connection between D[15:8] and LD[31:24]. Analysis

has shown that connecting peripheral controller chips directly to the 68030's data bus is difficult at higher processor-clock frequencies. Accordingly, systems using the VIC068A/VAC068A should be designed to connect low-speed peripheral devices to ID[15:8]. The VAC068A enables this data path on accesses to such I/O devices, as well as when SWDEN* is asserted by the VIC068A, and provides data latching and output enable control to ease interface timing. The DDIR signal provides direction flow.

5.3.8.3 Programmable DSACKi* Timing

The VAC068A generates DSACKi*s with programmable timing for each of its device select outputs (i.e., IOSEL5–0*, SHRCS*, and EPROMCS*) except the VSBSEL*, the VIC068A, and the VAC068A register accesses.

Upon power-up, DSACKi* sizing for EPROM space is as follows. For an EPROM data path of 16 bits, force ID[9] to a 0 at power-up. If an 8-bit data path is needed, force ID[8] to 0 at power-up. The default DSACKi* assertion for EPROM (prior to the DSACKi* EPROMCS* Control register being written) is for a 32-bit data path (ID8 and ID9 High).

The VAC068A asserts DSACKi*s on the processor access to DRAM with programmable wait state timing (set in the DSACKi* Control register). When used with the Motorola 68020, the VAC068A three-states its DSACKi* drivers on DRAM access to allow either the VAC068A or external logic to control their timing. When used with a synchronous local bus, the VAC068A allows for termination of DRAM accesses externally. On VME slave and VIC068A DMA cycles, the VAC068A asserts DSACKi* with minimum delay, following the assertion of local processor address strobe PAS*. This allows the high-resolution DSACKi*-to-DTACK* delay timer in the VIC068A to delay until data is valid. (See VIC068A registers $C7 and $A7.)

5.3.8.4 VIC068A/VAC068A DMA Support

The VAC068A provides upper-address counters and control logic for both the VMEbus address bus A[31:8] and the local address bus LA[31:8] to extend the address range from 8 bits to 32 bits. This allows for crossing of 256-byte boundaries on either the local or VMEbus during block transfers. VIC068A/VAC068A DMA always has local memory as the source or destination of data. The data is transferred to either the VME interface or local I/O port in a pass-through mode. VMEbus block transfer DMA is a dual-address operation in which the source and destination are required to be on opposite sides of the VME interface. Data is transferred to/from some address in the local memory from/to some address accessed via the VMEbus. Memory-to-memory transfers, where both the source and sink address are local memory, are not supported.

In the VIC068A-supported module-based local DMA operation, DMA data transfers to/from the specified local memory address across an interface boundary to the local destination. This destination cannot be local memory. Thus, it allows the implementation of a VSB and/or daughterboard interface with DMA capability or fixed-address I/O operation as would be appropriate with a SCSI or Ethernet implementation.

There is also a dual-address path between the local bus and VMEbus that permits VMEbus accesses during periods of processor activity or during the interleave time between block transfers. The VAC068A also supports block transfers as a slave by latching incoming address information on the falling edge of AS*.

5.3.8.5 IORD* and IOWR*

The VAC068A generates IORD* and IOWR* as alternate I/O read and write signals. These outputs are synchronous to CPUCLK with a 0–2 clock delay from assertion IOSEL5–0* to the assertion of IOWR* or IORD* as well as the usual DSACKi* delay assertion. The IOSELi* outputs may be combinatorial with a minimum delay, or synchronous with a clock period delay to insure address set-up time. The IOSELi*s also have a programmable minimum deassertion time, since most peripheral controller chips cannot handle back-to-back accesses. These features significantly reduce the amount of support logic otherwise required to interface peripherals to the 68K.

5.3.8.6 I/O Recovery Timer

The VAC068A provides a programmable recovery time between individual I/O device accesses (IOSEL5–0* High). This timer function is provided when a value is written to the Recovery Time bits in the DSACKi* Control registers. No IOSEL5–0* device access is allowed to start until the select signal for the previously accessed IOSEL5–0* device has been deasserted for the programmed number of clock cycles.

5.3.8.7 IACK Cycle Emulation for Non-680X0 Processors

The VAC068A provides two alternatives to 680X0 function code decoding for asserting the FCIACK* signal. This allows non-680X0-type processors to emulate the 680X0's interrupt acknowledge hardware protocol. If bit 31 of the PIO Function Register is set, accesses to IOSEL5* address space results in FCIACK* being asserted, while accesses to the IOSEL4* address space results in FPUCS* being asserted. If bit 30 of the PIO Direction register is set, then accesses to $FFFF FFxx results in FCIACK* being asserted, independent of the function codes.

5.3.8.8 Cache Inhibit Output

The VAC068A provides a CACHINH* (cache inhibit) signal typically connected to the corresponding input of the local processor. CACHINH* is asserted on any A16 access and is programmable in regions 1, 2, 3, 5, and in the A24 overlay region. For regions 1, 2, and 3, this is configured in the corresponding region attribute register. It is enabled for local I/O device access with bit 19 in the A24 Base Address register. For the A24 overlay, CACHINH* is enabled by setting bit 23 in the A24 Base Address register. CACHINH* is always asserted on redirected access of DRAM, and access to the top 256 bytes of DRAM if the mailbox interrupt is enabled.

CACHINH* can be asserted for all region 5 local I/O device selects (IOSEL5-0*), including VIC068A and VAC068A register accesses and region 6 (VMEbus A16) master accesses, and it may be asserted in the individual region attribute registers ($FFFD 09xx region 1, $FFFD 0Axx region 2, and $FFFD 0Bxx region 3), including the A24 address space overlay.

CACHINH* is deasserted in the remainder of DRAM and in region 4, the EPROM address space.