# 5.4

# VAC068A Operation

## 5.4.1 Resetting the VAC068A

There are two reset methods on the VAC068A. A global reset clears all registers and a soft reset (interrupt reset) masks all interrupt requests.

### 5.4.1.1 Global Reset

A global reset is initiated by either asserting the RESET* signal for 1K processor clock cycles or asserting RESET* in conjunction with WORD*. Both global reset types reset all VAC068A registers to their default values.

To execute the first type of global reset, the RESET* signal must be held Low for more than 1K CPUCLK cycles. If RESET* is held for less than 1K CPUCLK cycles, a soft reset occurs.

#### *5.4.1.1.1 Power-On Reset*

The other option to execute a reset of VAC068A registers is to assert RESET* for at least 5 CPUCLK cycles and then assert WORD* in conjunction with RESET* for at least 10 CPUCLK cycles. This also resets all VAC068A internal registers.

Following a global reset, memory map and slave address decodes (other than EPROM and Local I/O address space) are disabled until a local processor write occurs to the VAC068A ID register.

### 5.4.1.2 Soft Reset

A soft reset is initiated by asserting RESET* and holding it for less than 1K CPUCLK cycles. This function masks all interrupt requests but does not affect the memory map configurations.

### 5.4.1.3 RESET* Termination

Upon the termination of the RESET* operation, EPROMCS* is asserted for all processor read cycles, independent of the processor address. EPROMCS* DSACKi* timings default

to the maximum time and 32-bit data size. To modify the EPROM data size, hold either ID bus ID[9] Low upon power-up during RESET* for 16-bit data size, or hold IDbus ID[8] Low for 8-bit data size. Upon completion of the FORCE EPROM mode, the user must configure DSACK*s in the DSACKi* Control register per the data size on the module. The Force EPROM mode is exited upon an access to the EPROM address space ($FF00 0000 to $FFEF FFFF). This dictates that a jump to EPROM should be one of the first instructions in the boot sequence. The default EPROM timing and data sizes are disabled by the first write to the EPROM DSACKi* Control register.

## 5.4.2 System Initialization

After reset and before VAC068A registers are initialized, only EPROM, the VIC068A, and the VAC068A may be accessed. As stated previously, processor reads are forced to EPROM and use the slowest DSACKi* timing. Once the EPROM address space is accessed on the local address bus, the force EPROM mode is exited. The user is expected to set DSACKi* timings for reliable module operation. The following sequence of events is anticipated after power-up reset:

1. The VAC068A samples ID[8,9] on the rising edge of RESET* to get the default EPROM data path width.
2. CPU reads the reset vector table entry (address $0000) and loads it into the interrupt stack pointer.
3. VAC068A asserts EPROMCS*, then DSACKi* is asserted after 7 CPUCLK cycles.
4. CPU captures $FF00 0008 (32-bit mode) from the data bus.
5. CPU reads address $0000 0004. The VAC068A asserts EPROMCS*, then DSACKi* cycle.
6. CPU captures data from the data bus.
7. CPU reads $FF00 0008.
8. The VAC068A asserts EPROMCS*, then DSACKi* after 7 clocks.
9. CPU reads, writes to a VAC068A register other than EPROM DSACKi* Control to verify proper operation.
10. Write remainder of VAC068A registers to fully define memory map.
11. Write VAC068A ID register $FFFD 29xx to enable select and decode outputs.
12. Read/write sufficient addresses to verify map decoding.

## 5.4.3 Configuring the Local Memory Map

Sections 5.4.3 and 5.4.4 describe a sample memory map definition. The specific registers and configuration values are given.

The example assumes the module contains 4 Mbytes of DRAM, 16 Mbytes of VSB space in region 1, and 256 Kbytes of SRAM in the shared resource area. All are 32 bits in width.

### 5.4.3.1 DRAM Size

The module DRAM memory area (region 0) starts at address $0000 0000 and ends at the address configured in the DRAM Upper-Limit Mask register ($FFFD 05xx). For the 4 Mbytes required by this example, this register is loaded with $003F. Unlike regions 1 and 2, region 0 is required to be the DRAM memory area.

To support a local mailbox area, the DRAM area must contain at least 256 bytes of address space.

### 5.4.3.2 VSB Space

VSB Space is configured in region 1. This requires setting bits 27 and 26 of the Region 1 Attribute register ($FFFD 09XX) to 10 to assert VSBSEL* when the local address falls into the address range for region 1. The lower address limit for region 1 is specified as the first address beyond the DRAM area configured in the DRAM Upper-Limit Mask register. The upper address limit for this region is configured in the Boundary 2 Address register ($FFFD 06xx) with $013F to allocate a 16-Mbyte space directly above the DRAM area.

### 5.4.3.3 VMEbus A32, D32 Access

The VMEbus accessible address space is configured in region 2. This requires setting bits 27 and 26 of the Region 2 Attribute register ($FFFD 0Axx) to 11 to assert MWB* when the local address falls into the address range for region 2. The lower address limit for region 2 is specified as the first address beyond the VSB space configured in the Boundary 2 Address register. The upper address limit for this region is configured in the Boundary 3 Address register ($FFFD 07xx). This register is loaded with $FEFB to allocate all other address space—except 256 Kbytes for a shared resource area and the top 16 Mbytes for local resources and EPROM—to the VMEbus.

#### 5.4.3.4 Shared Resource Area

The shared resource area is configured in region 3. This requires setting bits 27 and 26 of the Region 3 Attribute register ($FFFD 0Bxx) to 01 to assert SHRCS* when the local address falls into the address range for region 3. The lower address limit for region 3 is specified as the first address beyond the VMEbus area configured in Boundary Area 3 Address register. The upper address limit for region 3 is fixed at $FEFF FFFF.

#### 5.4.3.5 EPROM Space

The VAC068A supports a fixed EPROM space from $FF00 0000 to $FFEF FFFF. These address limits may not be modified. When any address in this range is accessed, the VAC068A asserts EPROMCS*.

### 5.4.4 Configuring the VMEbus Address Map

The module address map for slave accesses from other VMEbus masters is configurable in multiple areas. For this example, the 4 Mbytes of DRAM and the 256 Kbytes of SRAM in the shared resource area are both mapped to other addresses on the VMEbus through the two slave select areas.

#### 5.4.4.1 SLSEL0* Access

The SLSEL0* address range is used to define VMEbus access to the 4 Mbytes of module DRAM. SLSEL0* accesses can only be mapped to region 0 (DRAM). This memory is mapped into the $0880 0000 to $08BF FFFF address range using the SLSEL0* Base Address and Address Mask registers. The base address register is used to specify the required logic level of the upper address bits to be compared, while the mask register is used to specify which bits to compare. To map the specified space, the SLSEL0* Base Address register ($FFFD 03xx) is loaded with $0880 and the SLSEL0* Address Mask register ($FFFD 02xx) is loaded with $FFC0. This includes the A[31:22] address bus signals in the address comparison.

Because the SLSEL0* space is defined here to be A32 space, it is necessary to also configure the VIC068A Slave Select 0 Control register 0 (SS0CR0) for A32 address modifier codes and D32 address space.

By enabling redirection for the SLSEL0* region, it is possible for the local processor to access the 4 Mbytes of DRAM in both the region 0 and SLSEL0* areas of the address map.

Redirection is enabled for the SLSEL0* area by setting bit 19 in the Decode Control register ($FFFD 14xx). With this bit set, local processor accesses to either memory area will assert DRAMCS* and not assert SLSEL0*.

A VMEbus slave access to the SLSEL0* memory area (with proper AM codes) will assert both SLSEL0* (to the VIC068A) and DRAMCS*.

### 5.4.4.2 SLSEL1* Access

The SLSEL1* area can be configured to map VMEbus slave accesses into any of four areas on the module: EPROM, DRAM, VSB, or Shared Resource. The specific area is configured in the Decode Control register ($FFFD 14xx). For this example, the Shared Resource area is mapped into VMEbus space by setting bits 29 and 28 of the Decode Control register to 10.

The 256-Kbyte SRAM is mapped into the $C0 0000 to $CF FFFF address range using the SLSEL1* Base Address and Address Mask registers. The base address register is used to specify the required logic level of the upper address bits to be compared while the mask register is used to specify which bits to compare. To map a 256-Kbyte address space for SLSEL1 in the SHRCS region, the SLSEL1* Base Address register ($FFFD 01xx) is loaded with $xxC0 with the SLSEL1* Address Mask register ($FFFD 00xx) is loaded with $00FC. By clearing the upper 8 bits of the SLSEL1* Address Mask register, the A[31:24] address bus signals are excluded from the address comparison. This configures the shared resource to appear in A24/A32 address space by setting bit 27 of the Decode Control register.

Because the SLSEL1* space is defined here to be A24/A32 space, it is necessary to also configure the VIC068A Slave Select 1 Control register 0 (SS1CR0) for A24 or A32 address modifier codes and D32 address space.

When redirection for SLSEL1* is enabled by setting bit 20 of the Decode Control register, and an address in the SLSEL1* address mask range is present on the VMEbus, SHRCS* is asserted. Thus, a valid VMEbus slave access to SRAM occurs. If the SLSEL1* address is present on the local bus, no SLSEL1* assertion occurs.

If bit 20 in the Decode Control register is not set, no chip selects are asserted when the VMEbus address is within the SLSEL1* mask address range.

### 5.4.4.3 ICFSEL* Access

Decoding of VMEbus Interprocessor Communications Facility accesses are accomplished through the ICFSEL* Address register ($FFFD 04xx). The two bytes of this register are

both compared with A[15:8]. An exact match of these address bus signals with either byte causes the VAC068A to assert ICFSEL* to the VIC068A. This allows both global (or group) and module specific select addresses to be configured. The low address bits (A[7:0]) are used to select which specific module switches and registers are to be accessed.

Present convention is to use the upper byte of the ICFSEL* Address register to specify the global select address and the lower byte for the module specific address. Loading this register with $F00F would then specify that all A16 accesses to $F0xx are addressed globally, while a similar access to $0Fxx would only be responded to by a module.

These same ICF registers and switches are accessed by the local processor in the $FFFC xx5F through $FFFC xx7F range.

#### 5.4.4.4 VME A24 Master Cycle

The A24 space for VMEbus master accesses may be mapped to any 32-Mbyte section of the programmable regions. This is configured in the A24 Base Address register bits [31:25]. The value placed in these bits is compared to the local address bus LA[31:25]. Because any value may be placed in these register bits, it is possible for the A24 space to overlay other regions of the local address map. Values should be greater than $02 and less than $FE.

For this example, the A24 Base Address register ($FFFD 08xx) is loaded with $F000. This assigns the local address range of $F000 0000 to $F1FF FFFF to A24 space. This exists in the local address space presently assigned as VMEbus space in region 2. Any local processor access to this space causes the VAC068A to drive ASIZ0/1 with 11 to force the VIC068A to assert the proper AM codes for A24 addressing.

The data bus size for these A24 master accesses may also be configured in the A24 Base Address register for either D16 or D32 operation.

#### 5.4.4.5 VME A16 Master Cycle

VMEbus A16 master cycles are fixed at the top 128 Kbytes section of the address map ($FFFE 0000 to $FFFF FFFF). Other bits in the A24 Base Address register are used to configure the data bus width (D32 or D16) and cache inhibit control.

#### 5.4.4.6 Decode Control Register

To completely specify the VAC068A behavior as a VME slave device, the Decode Control register ($FFFD 14xx) must be configured. Bits [29:28] must be set to 10 to select SHRCS*

for assertion in response to a LAEN when SLSEL1* is asserted and FC2/1 specifies a VME slave access. Bit 27 must be set for A24 (or A32) operation. Bits [25:23] must be set to 1 in the general case. If any of these bits is a 0, the slave select output is independent of VAS*. Otherwise, the output is only asserted when VAS* is asserted. Since the qualification of the slave selects is done midway through the address decode process, timing verification must demonstrate that it is done early enough to avoid glitches.

Bits [20:19] are normally set to 1. When this is the case, local CPU accesses to addresses within the SLSEL0* or SLSEL1* range, respectively, result in the device chip select, DRAMCS* for SLSEL0* or SHRCS* for SLSEL1*, being asserted instead of MWB*. Otherwise, MWB* is asserted and a VME access occurs. When these redirection bits are set to 0, the VIC068A should be programmed to assert BERR* and LBERR* when it sees a valid slave select when it is bus master. This allows software to determine its own slave select address. If the redirection bits are not set to 1, attempted access to a module's slave address results in a VME bus timeout. Bit 22 provides a means of qualifying decodes of the local address map with PAS*.

## 5.4.5 VME Master Access

For VAC068A-controlled VMEbus accesses, the local address must be set up 10 ns before PAS* is asserted. After the VAC068A decodes an address that maps to VMEbus, the VAC068A asserts MWB* and sets ASIZ0/1 and WORD* High or Low according to the appropriate region attribute register. The VIC068A then obtains bus mastership and asserts ABEN*. The VAC068A uses ABEN* to enable its VME address drivers. The access completes when the VIC068A asserts DSACKi* or LBERR*. When ABEN* is deasserted, the VMEbus address drivers must three-state before the VIC068A deasserts both BBSY* and AS*.

When write-posting is enabled, the VIC068A asserts LADO to freeze the address outputs. Accordingly, the address path through the VAC068A is transparent when LADO is Low and latched when LADO is High.

## 5.4.6 VME Slave Operation

The VAC068A continuously monitors the VMEbus address bus for any of the SLSEL0*, SLSEL1*, and ICFSEL* addresses. When it detects such an address, qualified by AS* if so

enabled, it asserts the proper select output. The LADI input is used to insure that the address presented to the local bus remains stable throughout the slave access. This is accomplished by latching the local address outputs when LADO is High. The VIC068A may respond to an asserted slave select output with a local bus request and subsequent slave transfer. The VIC068A qualifies each slave select with the address modifier field and data strobe.

#### 5.4.6.1 Slave Transfer Sequence

The following sequence of events occurs on a slave transfer:

1. The VAC068A asserts SLSEL0* (or SLSEL1* or ICFSEL*).
2. The VIC068A qualifies SLSEL0* with address modifiers and data strobe(s) and asserts VICLBR*.
3. Module logic arbitrates for the local bus and asserts LBG* to the VIC068A. The VIC068A waits for local-cycle-end plus 3 CLK64M clocks, then asserts LAEN, sets FC2/1 to 10 to distinguish slave transfer from DMA or refresh, drives LA[7:0] appropriately, and enables the data path (DDIR, SWDEN*, DENIN*, DENIN1*, DENO*).
4. The VAC068A uses LAEN to enable its local address drivers and FC2/1 to select the VMEbus as the address source. Approximately 1 CLK64M clock from LAEN assertion, the VIC068A asserts PAS*.
5. Upon PAS* assertion, the VAC068A asserts DSACK0/1* and asserts the appropriate device select output (DRAMCS*, EPROMCS*, VSBSEL*, or SHRCS*).
6. The local memory reads/writes at the address on the local bus.
7. The VIC068A times out the DSACKi* to DTACK* delay, then asserts the LEDO output to capture the data in the bus interface latches, deasserts the local strobes, and asserts DTACK*. At some time previous to DTACK* assertion, the VIC068A asserts LADI to freeze the local address.
8. The VAC068A deasserts DSACK1/0* when PAS* is deasserted.
9. The VAC068A three-states its address drivers when LAEN is deasserted.

## 5.4.7 VME Master Block Transfer

The following sequence occurs on a master block transfer:

1. The local CPU initializes the VIC068A for a DMA block transfer, then asserts address and PAS* that maps to the VMEbus.
2. The VAC068A decodes the address and asserts MWB*, along with WORD* and ASIZ1/0 according to the appropriate Region Attribute registers.
3. The VIC068A detects MWB* asserted, requests the VMEbus, and asserts BLT*.
4. The VAC068A detects BLT* asserted and loads LD[31..8] into its DMA address counter for driving the local address and loads LA[31..8] into the BLT address counter for driving A[31..8].
5. The VIC068A receives VMEbus mastership and asserts LBR*.
6. Module logic arbitrates for the local bus and asserts LBG* to the VIC068A.
7. The VIC068A waits for local-cycle-end plus 3 CK64M clocks, then asserts LAEN, sets FC2/1 to 01 to distinguish DMA from a slave transfer or refresh, drives LA[7:0] appropriately, and enables the data path (DDIR, SWDEN*, DENIN*, DENIN1*, DENO*).
8. The VAC068A uses LAEN to enable its local address drivers with the DMA address and uses FC2/1 to select the DMA counters as the address source.
9. The VAC068A asserts LDMACK* when LAEN is asserted and the function codes reach the DMA (VME or local) state.
10. Approximately 1 CLK64M clock from LAEN assertion, the VIC068A asserts PAS* and DS*.
11. Upon PAS* assertion, the VAC068A asserts DSACKi*.
12. The local memory reads/writes at the addresses on the local bus as they are incremented and strobed by the VIC068A and VAC068A.
13. After burst counter expiration in the VIC068A, PAS* is deasserted. If a 256-byte boundary crossing occurs, BLT* or LADO is pulsed to increment the appropriate counter. Next, LBR* and LAEN are deasserted. If BLT* remains asserted, the address counters are preserved. If LADO toggles twice while LBR* is asserted, the local address counter is incremented. If BLT* toggles twice while LBR* is asserted, the local address counter is incremented. When BLT* or LAEN deassert, the local address drivers are three-stated. The DSACKi* drivers three-state soon thereafter as permitted by their rescinding circuit. When both BLT* and LBR* are deasserted, the DMA block transfer is over.

## 5.4.8 VIC068A/VAC068A Interconnect Diagram

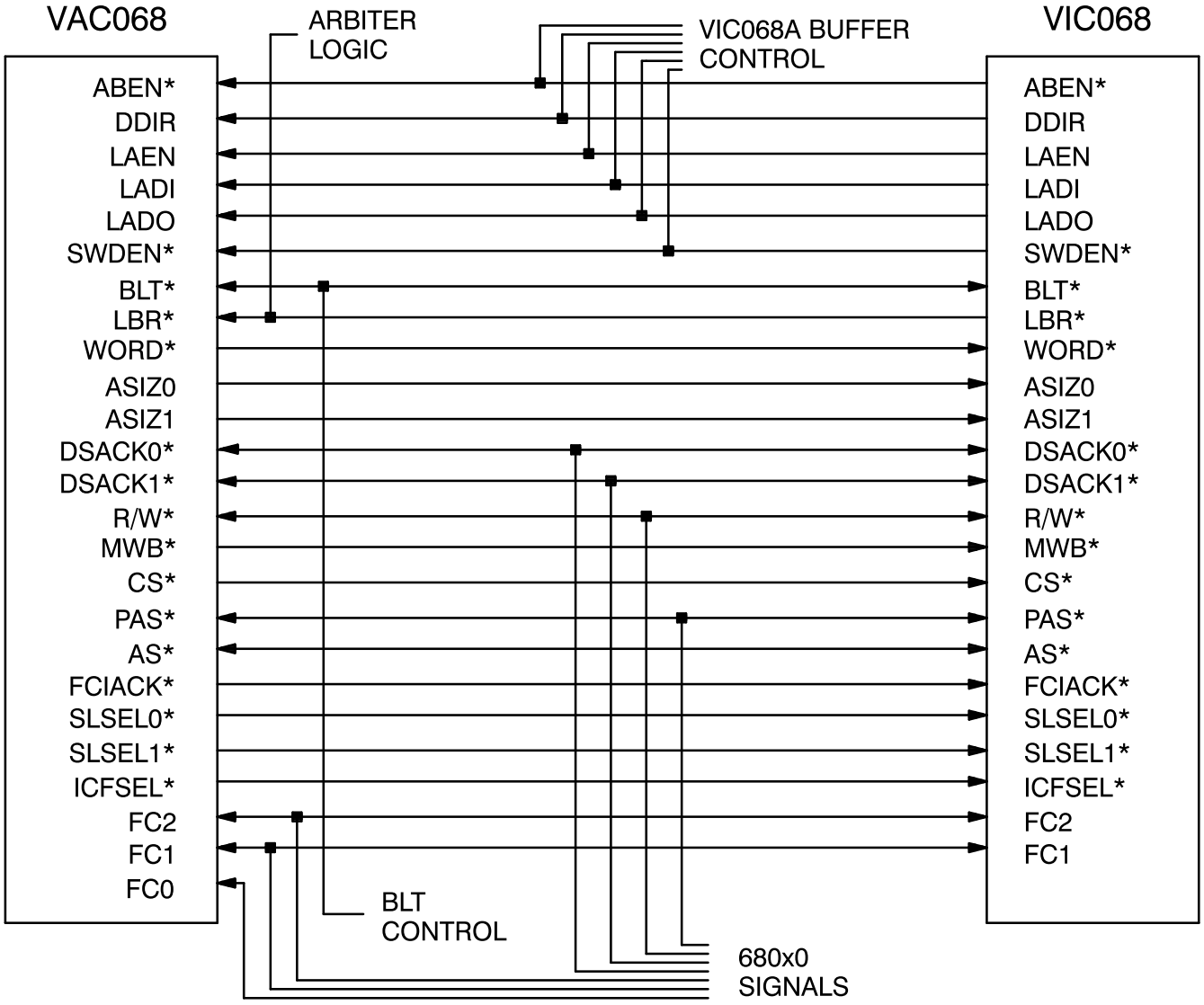


**Figure 5−3. VIC068A/VAC068A Interconnect Diagram**