1.6

VIC068A VMEbus Slave Operations

The act of writing or retrieving data for a VMEbus master is referred to as a slave operation. The VIC068A is able to perform slave operations with extensive configuration options.

The following VIC068A registers are used in performing and configuring slave operations:

- Slave Select 0 Control Register 0 (SS0CR0), bits 0−5
- Slave Select 0 Control Register 1 (SS0CR1)
- Slave Select 1 Control Register 0 (SS0CR0), bits 0−5
- Slave Select 1 Control Register 1 (SS0CR1)
- Local Bus Timing Register (LBTR)
- Address Modifier Source Register (AMSR)

1.6.1 The Valid Slave Select

The VIC068A contains two separate signals that are used to indicate that the VIC068A has been selected for a slave access. These signals are SLSEL0* and SLSEL1*. These signals are usually the result of external VMEbus address decoding. When the VIC068A detects a SLSELi* asserted, the VIC068A waits for:

- AS* asserted
- DSi* asserted for the current cycle (not left over from previous cycle)
- DTACK* or BERR* deasserted from previous cycle

The VIC068A then checks the AM codes for:

- address sizing (A32/A24/A16)
- transfer type (supervisory/user)

If the VIC068A is configured, in SSiCR0, to accept the slave access as indicated by the AM codes, the VIC068A proceeds with the slave request by asserting the LBR* signal to obtain the local bus. If the VIC068A is not configured for the particular type of slave access, the VIC068A ignores the request and does not assert LBR*.

If the VIC068A has been selected for valid D32 slave access and the VIC068A is configured to accept only D16 operations, the VIC068A asserts BERR*. If the WORD* signal is asserted, the VIC068A will only accept D16 slave cycles, independent of how the VIC068A may be configured in the SSiCR0. If the VIC068A is not configured to accept block transfers, any block transfer request will be BERRed.

The VIC068A is also capable of bypassing the standard VMEbus AM codes and qualifying the SLSELi* with user-defined AM codes. If enabled for this operation (in SSiCR0[3:2]), the VIC068A compares the AM codes with the value contained in the AMSR[5:0]. If the values match, a valid slave select has occurred. The VIC068A may also be configured to only compare AM[5:3] to AMSR[5:3]. When enabled for this operation by setting AMSR[6], the address size is also qualified by the address size information in the SSiCR0. *Table 1−7* summarizes the AM code operations for VIC068A slave accesses.

In some situations, it is possible for both SLSEL1* and SLSEL0* to be asserted simultaneously. In this case, the VIC068A checks each SLSELi*'s register configuration with the AM codes to determine which, if any, valid slave select has occurred.

Also included in the AM codes is information specifying if the transfer is a block transfer. The VIC068A checks the SSiCR0 register to determine if the VIC068A is enabled to receive slave block transfers. Slave block transfers are discussed in detail in section 1.10.2.

Table 1−7. Slave Transfer AM Code Control Map

VIC068A AM Code Inputs		**VIC068A Slave Access Outputs**		
Operation Type	AM[5:0]	Address Size	Block Transfer	FC2/1
User Data User Program Supervisory Data Supervisory Program	\$09 \$0A \$0D \$0E	A32 Addressing	No	1 0
User Block Supervisory Block	\$0B \$0F		Yes	0 0
User Data User Program Supervisory Data Supervisory Program	\$39 \$3A \$3D \$3E	A24 Addressing	No	1 0
User Block Supervisory Block	\$3B \$3F		Yes	0 0
User Access Supervisory Access	\$29 \$2D	A16 Addressing	No	1 0

Table 1−7. Slave Transfer AM Code Control Map (continued)

VIC068A AM Code Inputs		VIC068A Slave Access Outputs		
Operation Type	AM[5:0]	Address Size	Block Transfer	FC2/1
User Defined	User Defined	User Defined	No	1 0
User Defined	User Defined	User Defined	Yes	0 0

1.6.2 The Local Bus Request

After a valid slave select has been signaled, the VIC068A bids for the local bus by asserting the LBR* signal. This signal should be input to a local bus arbiter, which in turn issues a bus grant (assert LBG*) to the VIC068A. The VIC068A does not perform local bus arbitration.

1.6.3 The Local Bus Grant

Once the VIC068A issues the LBR*, the VIC068A waits for the assertion of the local bus grant. If the local processor expects BGACK-type signaling in response to the assertion of BG*, this must be generated by external logic. This BGACK must continue to be asserted until LBR* is deasserted.

After the assertion of LBG*, the VIC068A waits for $3T+t_{PD}$ before enabling the local address drivers (and data drivers if writing). At this point, the VIC068A waits $1T+t_{PD}$ before driving PAS*. If local resources cannot be guaranteed to be off the local bus by these times after the assertion of LBG*, additional delay may need to be introduced between the assertion of a bus grant from the local arbiter and the assertion of LBG* to the VIC068A. See *Figure 1−31* in Chapter 1.13 for more details on local bus timing.

When the VIC068A begins driving the local bus, it also drives the FC2/1 signals with information on the type of local cycle it is performing. These function codes should not be confused with the function codes that are driven to the VIC068A when it is performing VMEbus master cycles (see section 1.5.12). The function code outputs for the VIC068A are as follows:

FC2	*FC1*	*Description*
0	0	Slave Block Transfer
0	1	Block Transfer with Local DMA
1	0	Standard Slave Access
1	1	DRAM Refresh

1.6.4 Local Bus Timing

The VIC068A contains a Local Bus Timing Register (LBTR) for configuring the minimum PAS* assertion and deassertion and DS* deassertion times. For single cycle slave accesses, the minimum deassertion time is not considered in that PAS* and DS* are asserted according to the timing discussed in section 1.6.3. The minimum deassertion times for PAS* and DS* are usually used when performing multiple back-to-back local cycles such as DRAM refresh cycles, slave block transfers, and master block transfers with local DMA. The minimum assertion time for PAS* should be set so that illegal memory access cycles can never occur for the particular memory devices being used. Minimum PAS* asserted time is usually dictated by the assertion of the DSACKi* signals, which start an additional delay circuit called the SAT delay (see section 1.6.8). *Figures 1−6* and *1−7* show an example of local reads and writes.

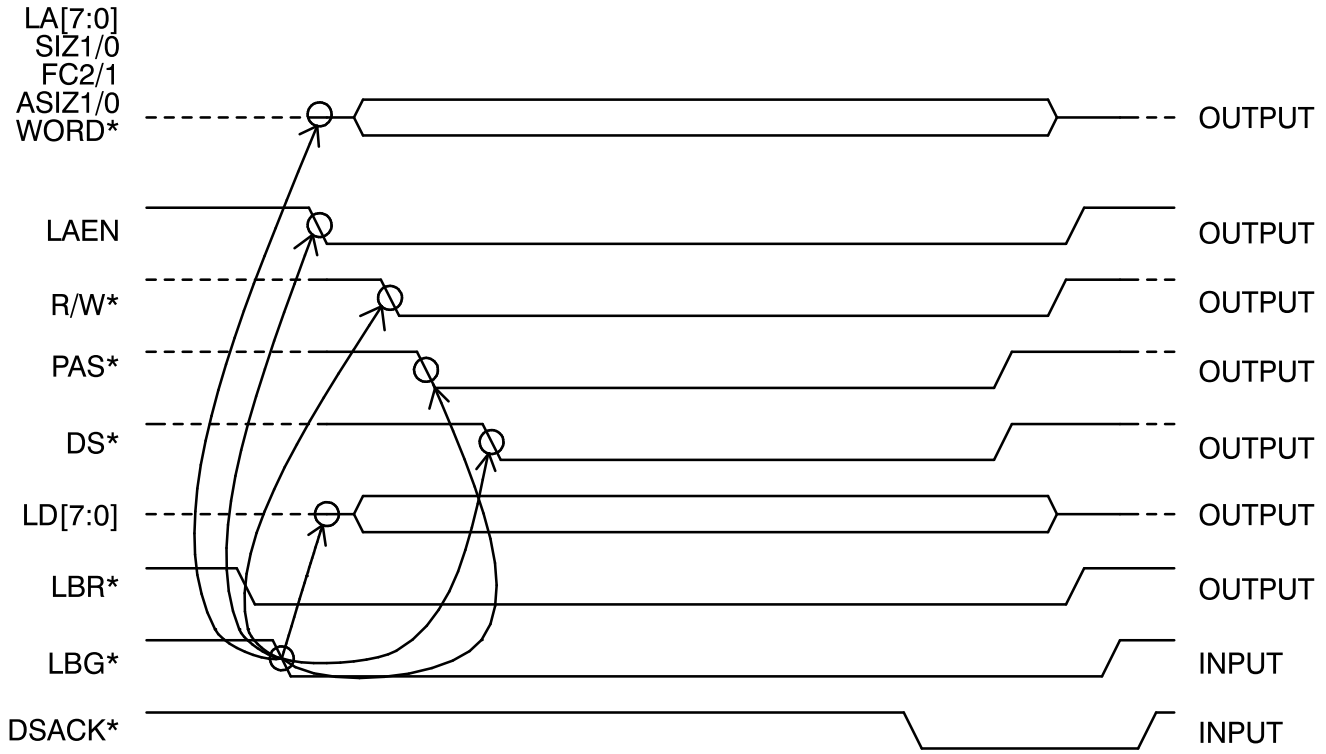

Figure 1−6. Local Bus Cycle—Write

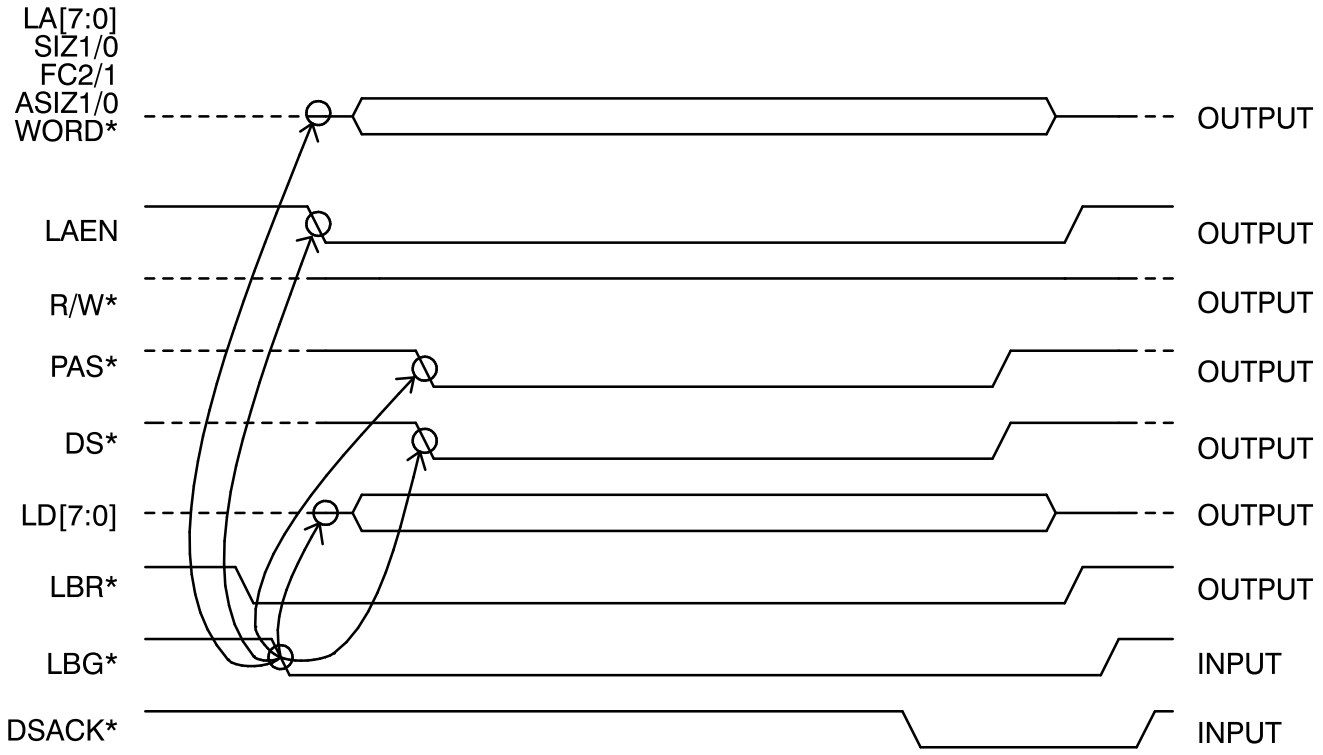

Figure 1−7. Local Bus Cycle—Read

1.6.5 VMEbus/Local Bus Data and Port Size

After receiving a valid slave select, the VIC068A examines the DS1/0*, A01, and LWORD* signals to determine the size and the alignment of the transfer. The VIC068A then drives the SIZ1/0 and LA[1:0] signals with the appropriate values. The SIZ1/0 codes are given below:

SIZ1	*SIZ0*	*Transfer Size*
0	0	Longword
0	1	Byte
1	0	Word
1	1	3-byte

The ability to handle D32 transfers is configured in the SSiCR0[4]. If configured as a D32 port, the VIC068A accepts any size VMEbus transfer. If programmed as a D16 port only,

the VIC068A will assert BERR* for all D32 requests. The VIC068A determines D32 and D16 from the LWORD* signal.

1.6.6 The Latched Bus Interface

When a slave read is performed, data is read from local memory and latched into the VMEbus data transceivers. LD[7:0] is latched into the VIC068A and LD[31:8] is latched into external devices such as the CY7C964s. After DSACKi* is asserted, the VIC068A begins the SAT delay(SSiCR1[3:0]). After this delay times out, the VIC068A asserts both DTACK* and LEDO. These signals are held until both DS1/0* are deasserted.

During a slave write access, the VIC068A asserts the LEDI signal immediately after receiving a valid slave select latching the data into the data transceivers.

1.6.7 Slave Write Posting

The VIC068A is able to perform slave write posting, when SS1CR0[7] is set. In this mode, the VIC068A, after receiving a valid slave select, latches the incoming data (asserts LEDI) and immediately asserts DTACK*. The VIC068A requests the local bus and then performs the local write cycle independent of VMEbus activity.

If a slave write post is requested while a previous slave write post is currently being serviced, DTACK* is not asserted until the local write is complete. At this point, the VIC068A posts the write normally. Slave write posts should be used with caution. If there was a problem with the local portion of the cycle (i.e., LBERR* was asserted), the initiator of the cycle may never know the the cycle did not complete since DTACK* was already asserted.

1.6.8 Slave Acknowledge Timing (SAT)

Once DSACKi* is asserted, the delay defined by SSiCR1[3:0] begins. After this delay expires, DTACK* is asserted, and PAS* and DS* deassert. LEDO is also asserted if the slave cycle is a read. An assertion of any or both of the DSACKi* signals is considered an acknowledge and begins this timeout. The default value for these delays is 0. Usually delays need

only be used for memory designs that use an advance acknowledge or late bus-error algorithms. Once DTACK* is asserted, the VIC068A maintains DTACK* until the VMEbus DSi* signals are deasserted.

If the LBERR* signal is asserted, the VIC068A asserts BERR* until the DSi* signals are deasserted. If LBERR* is asserted after either DSACKi* has been asserted, and the DSACK*- to-DTACK* delay has not yet expired, DTACK* is inhibited and the BERR* signal will be asserted on the VMEbus without delay. If the user employs some delayed LBERR* algorithm (such as late parity check), the SAT delay must be programmed sufficiently long to allow for this delay of LBERR*.

Table 1−8. Buffer Control Signals: D32 VMEbus Slave Write Operation

Data Path Size	VMEbus Stimulus			Local Bus Response			Address Control			Data Control			Swap Control				
	DS1/0*	A01	LWORD*	SIZ1/0	LA[1:0]	DSACKi*	ABEN*	LADI	LADO	DENO*	LEDI	LEDO	DDIR	SWDEN*	ISOBE*	DENIN*	DENIN1*
D32	0 0	0	0	LL	LL	0	H	↑	L	H	↑	L	L	H	L	L	L
D32, UAT (0−2)	0 1	0	0	HH	LL	0	H	↑	L	H	↑	L	L	H	L	L	L
D32, UAT (1−3)	1 0	0	0	HH	LH	0	H	↑	L	H	↑	L	L	H	L	L	L
D32, UAT (1−2)	0 0	1	0	HL	LH	0	H	↑	L	H	↑	L	L	H	L	L	L
D16 (0−1)	0 0	0	1	HL	LL	0	H	↑	L	H	↑	L	L	L	L	L	H
D16 (2−3)	0 0	1	1	HL	HL	0	H	↑	L	H	↑	L	L	L	L	L	H
D8 (0)	0 1	0	1	LH	LL	0	H	↑	L	H	↑	L	L	L	L	L	H
D8 (1)	1 0	0	1	LH	LH	0	H	↑	L	H	↑	L	L	L	L	L	H
D8 (2)	0 1	1	1	LH	HL	0	H	↑	L	H	↑	L	L	L	L	L	H
D8 (3)	1 0	1	1	LH	HH	0	H	↑	L	H	↑	L	L	L	L	L	H

Table 1−9. Buffer Control Signals: D32 VMEbus Slave Read Operation

Data Path Size	VMEbus Stimulus			Local Bus Response			Address Control			Data Control			Swap Control				
	DS1/0*	A01	LWORD*	SIZ1/0	LA[1:0]	DSACKi*	ABEN*	LADI	LADO	DENO*	LEDI	LEDO	DDIR	SWDEN*	ISOBE*	DENIN*	DENIN1*
D32	0 0	0	0	LL	LL	0	H	↑	L	L	L	↑	H	H	L	H	H
D32, UAT (0−2)	0 1	0	0	HH	LL	0	H	↑	L	L	L	↑	H	H	L	H	H
D32, UAT (1−3)	1 0	0	0	HH	LH	0	H	↑	L	L	L	↑	H	H	L	H	H
D32, UAT (1−2)	0 0	1	0	HL	LH	0	H	↑	L	L	L	↑	H	H	L	H	H

Table 1−9. Buffer Control Signals: D32 VMEbus Slave Read Operation (continued)

Data Path Size	VMEbus Stimulus			Local Bus Response			Address Control			Data Control			Swap Control				
	DS1/0*	A01	LWORD*	SIZ1/0	LA[1:0]	DSACKi*	ABEN*	LADI	LADO	DENO*	LEDI	LEDO	DDIR	SWDEN*	ISOBE*	DENIN*	DENIN1*
D16 (0−1)	0 0	0	1	HL	LL	0	H	↑	L	L	L	↑	H	L	H	H	H
D16 (2−3)	0 0	1	1	HL	HL	0	H	↑	L	L	L	↑	H	H	L	H	H
D8 (0)	0 1	0	1	LH	LL	0	H	↑	L	L	L	↑	H	L	H	H	H
D8 (1)	1 0	0	1	LH	LH	0	H	↑	L	L	L	↑	H	L	H	H	H
D8 (2)	0 1	1	1	LH	HL	0	H	↑	L	L	L	↑	H	H	L	H	H
D8 (3)	1 0	1	1	LH	HH	0	H	↑	L	L	L	↑	H	H	L	H	H

Table 1−10. Buffer Control Signals: D16 VMEbus Slave Read Operation

Data Path Size	VMEbus Stimulus			Local Bus Response			Address Control			Data Control			Swap Control				
	DS1/0*	A01	LWORD*	SIZ1/0	LA[1:0]	DSACKi*	ABEN*	LADI	LADO	DENO*	LEDI	LEDO	DDIR	SWDEN*	ISOBE*	DENIN*	DENIN1*
D16 (0−1)	0 0	0	1	HL	LL	0	H	↑	L	L	L	↑	H	L	H	H	H
D16 (2−3)	0 0	1	1	HL	HL	0	H	↑	L	L	L	↑	H	H	L	H	H
D8 (0)	0 1	0	1	LH	LL	0	H	↑	L	L	L	↑	H	L	H	H	H
D8 (1)	1 0	0	1	LH	LH	0	H	↑	L	L	L	↑	H	L	H	H	H
D8 (2)	0 1	1	1	LH	HL	0	H	↑	L	L	L	↑	H	H	L	H	H
D8 (3)	1 0	1	1	LH	HH	0	H	↑	L	L	L	↑	H	H	L	H	H

Table 1−11. Buffer Control Signals: D16 VMEbus Slave Write Operation

Data Path Size	VMEbus Stimulus			Local Bus Response			Address Control			Data Control			Swap Control				
	DS1/0*	A01	LWORD*	SIZ1/0	LA[1:0]	DSACKi*	ABEN*	LADI	LADO	DENO*	LEDI	LEDO	DDIR	SWDEN*	ISOBE*	DENIN*	DENIN1*
D16 (0−1)	0 0	0	1	HL	LL	0	H	↑	L	H	↑	L	L	L	L	L	H
D16 (2−3)	0 0	1	1	HL	HL	0	H	↑	L	H	↑	L	L	L	L	L	H
D8 (0)	0 1	0	1	LH	LL	0	H	↑	L	H	↑	L	L	L	L	L	H
D8 (1)	1 0	0	1	LH	LH	0	H	↑	L	H	↑	L	L	L	L	L	H
D8 (2)	0 1	1	1	LH	HL	0	H	↑	L	H	↑	L	L	L	L	L	H
D8 (3)	1 0	1	1	LH	HH	0	H	↑	L	H	↑	L	L	L	L	L	H