1.3

Overview of the VIC068A

The VIC068A provides an economical and convenient means to interface between a local CPU bus and the VMEbus. The local bus interface of the VIC068A emulates Motorola’s family of 32-bit CISC processor interfaces (68K). Other processors can easily be adapted to interface to the VIC068A with appropriate logic. All of the following items are discussed in further detail in later sections of this manual.

1.3.1 Resetting the VIC068A

The VIC068A can be reset by any of three distinct reset conditions.

- Internal Reset. This reset is the most common means of resetting the VIC068A. It resets most register values and all mechanisms within the device. This reset is usually issued as a push-button reset.
- System Reset. This reset provides a means of resetting the VIC068A through the VMEbus backplane. The VIC068A may also signal a SYSRESET* by writing a configuration register.
- Global Reset. This is the most complete reset of the VIC068A. This resets all of the VIC068A’s configuration registers. This reset should be used with caution since SYSCLK is not driven and the BG*/IACK* daisy-chains are disabled while a global reset is in progress (while it is system controller). This is usually issued as a power-up reset.

All three reset options are implemented in a different manner and have different effects on the VIC068A configuration registers. See section 1.11.1.

1.3.2 The VIC068A VMEbus System Controller

The VIC068A is capable of operating as the VMEbus system controller. It provides VMEbus arbitration functions including:

- priority (PRI), round-robin (RRS), and single-level (SGL) arbitration schemes
- driving IACK* daisy-chain
- driving BGiOUT* daisy-chain (all four levels)
- driving SYSCLK output
- VMEbus arbitration timeout timer
- VMEbus transfer timeout timer

The system controller functions are enabled by the SCON* pin of the VIC068A. When strapped Low, the VIC068A functions as the VMEbus system controller. See Chapter 1.4.

1.3.3 VIC068A VMEbus Master Cycles

The VIC068A is capable of becoming the VMEbus master in response to a request from local resources. In this situation, the local resource requests that a VMEbus transfer is desired. The VIC068A then makes a request for the VMEbus. When the VMEbus is granted to the VIC068A, it then performs the transfer, acknowledges the local resource, and the cycle is complete. The VIC068A is capable of all four VMEbus request levels (see section 1.5.1). The following release modes are supported (see section 1.5.2):

- Release On Request (ROR)
- Release When Done (RWD)
- Release On Clear (ROC)
- Release under RMC* control
- Bus Capture And Hold (BCAP).

The VIC068A supports A32, A24, and A16 as well as user-defined address spaces.

1.3.3.1 Master Write-Posting

The VIC068A is capable of performing master write-posting (bus-decoupling) during both block and single-cycle transactions. In this situation, the VIC068A acknowledges the local resource immediately after the request to the VIC068A is made, thus freeing the local bus. The VIC068A latches the local data to be written and performs the VMEbus transfer without the local resource having to wait for the VMEbus. See section 1.5.5.

1.3.3.2 Indivisible Cycles

Read-modify-write cycles and Indivisible Multiple-Address Cycles (IMACs) are easily performed using the VIC068A. Significant control is allowed to:

- request the VMEbus on the assertion of RMC* independent of MWB* (this prevents any slave access from interrupting local indivisible cycles)
- stretch the VMEbus AS*
- make the above behaviors dependent on the local SIZi signals

See section 1.5.6.

1.3.3.3 Deadlock

If a master operation is attempted when a slave operation to the same module is in progress, a deadlock has occurred. The VIC068A signals a deadlock condition by asserting the DEDLK* signal. This should be used by the local resource requesting the VMEbus to try the transfer after the slave access has completed. See section 1.5.7.

1.3.3.4 Self-Access

If the VIC068A is selected as the slave while it is VMEbus master, a self-access has occurred. The VIC068A asserts both BERR* and LBERR* in this situation.

BESR[2,1] also indicates when a self-access has occurred.

1.3.4 VIC068A VMEbus Slave Cycles

The VIC068A is capable of receiving slave accesses (see Chapter 1.6). The VIC068A contains a highly programmable environment to allow for a wide variety of slave configurations. The VIC068A allows for:

- D32 or D16 configuration
- A32, A24, A16, or user-defined address spaces
- programmable block transfer support including:
  - accelerated block transfer (PAS* held asserted)
  - non-accelerated-type block transfer (toggle PAS*)
  - no support for block transfer
- programmable data acquisition delays
- programmable PAS* and DS* timing
- restricted slave accesses (supervisory accesses only)

When a slave access is required, the VIC068A requests the local bus. When local bus mastership is obtained, the VIC068A reads or writes the data to/from the local resource and asserts the DTACK* signal to complete the transfer.

1.3.4.1 Slave Write-Posting

The VIC068A is capable of performing a slave write-post operation (bus-decoupling) during single cycle transactions. When enabled, the VIC068A latches the data to be written and acknowledges the VMEbus (by asserting DTACK*) immediately thereafter. This prevents the VMEbus from having to wait for local bus access. See section 1.6.7.

1.3.5 Address Modifier (AM) Codes

The VIC068A encodes and decodes the VMEbus address modifier codes. For VMEbus master accesses, the VIC068A encodes the appropriate AM codes through FCi status, ASIZi status, and the block transfer status. For slave accesses, the VIC068A decodes the AM Codes and checks the Slave Select Control registers to determine if the slave request is to be supported with regard to address spaces, supervisory accesses, and block transfers. The VIC068A also supports user-defined AM codes. That is, the VIC068A can be configured to assert and respond to user-defined AM codes. See section 1.6.1.

1.3.6 VIC068A VMEbus Block Transfers

The VIC068A is capable of both performing (as master) and receiving (as slave) block transfers. The master VIC068A performs a block transfer in one of two modes:

- MOVEM-type block transfer
- master block transfer with local DMA

The VMEbus specification restricts block transfers from crossing 256-byte boundaries. The VIC068A works around this problem by simply toggling the AS* at VMEbus page boundaries. The VIC068A is also able to break the total transfer length into smaller bursts. The VIC068A allows for easy implementation of large block transfers by releasing the VMEbus and local bus between these bursts and, at the appropriate time, re-requesting the buses at a programmed time later. This in-between time is referred to as the interleave period. All of this is performed without processor/software intervention until the transfer is complete. See section 1.10.1.1.

The VIC068A contains two separate address counters for the VMEbus and the local address buses. In addition, a separate address counter is provided for slave block transfers. The VIC068A address counters are 8-bit up-counters that provide for transfers up to 256 bytes. For transfers that exceed the 256-byte limit, Cypress CY7C964s, Cypress VAC068A or external counters and latches are required.

The VIC068A allows slave accesses to occur during the interleave period. Master accesses are also allowed during interleave with programming and external logic. This is referred to as the dual-path option. See section 1.10.1.1.6.

The Cypress Semiconductor CY7C964s or VAC068A may be used in conjunction with the VIC068A to provide much of the external logic required for extended block transfer modes such as the 256-byte boundary crossing and dual path. Three CY7C964s extend the 8-bit counters in the VIC068A to support full 32-bit incrementing addresses on both the local bus and VMEbus. The CY7C964s also contain the latches required for extended address block transfers as well as those required for supporting the dual-path option. The CY7C964 enhances boards that support block transfers by greatly reducing the necessary support logic.

The Cypress Semiconductor VAC068A may also be used to provide the latching and counting of upper data and addresses also reducing necessary support logic.

1.3.6.1 MOVEM Master Block Transfers

This mode of block transfer provides the simplest implementation of VMEbus block transfers. In this mode, the local resource configures the VIC068A for a MOVEM block transfer and proceeds with the consecutive-address cycles (such as a 68K MOVEM instruction). The local processor continues as the local bus master in this mode. See section 1.10.1.2.

1.3.6.2 Master Block Transfers with Local DMA

In this mode, the VIC068A becomes the local bus master and reads or writes the local data in a DMA-like fashion. This provides a much faster interface than the MOVEM block transfer, but with less control and error detection. See section 1.10.1.1.

1.3.6.3 Slave Block Transfers

The process of receiving a block transfer is referred to as a slave block transfer. The VIC068A is capable of decoding the address modifier codes to determine if a slave block transfer is desired. In this mode, the VIC068A captures the VMEbus address, and latches it into internal counters. For subsequent cycles, the VIC068A increments this counter for each transfer. The local protocol for slave block transfers can be configured in a full handshake mode by toggling both PAS* and DS* and expecting DSACKi* to toggle, or in an accelerated mode in which only DS* toggles and PAS* is asserted throughout the cycle.

The VIC068A is capable of acting as a DMA controller between two local resources. This mode is similar to that of master block transfers with local DMA except that a local I/O acts as the second source or destination.

1.3.7 VIC068A Interrupt Generation and Handling Facilities

The VIC068A is capable of generating and handling a seven-level prioritized interrupt scheme similar to that used by the Motorola 68K processors. These interrupts may be the result of the seven VMEbus interrupts, seven local interrupts, five VIC068A error/status interrupts, and eight interprocessor communication interrupts.

The VIC068A can be configured as an interrupt handler for any of the seven VMEbus interrupts. The VIC068A can generate the seven VMEbus interrupts as well as supplying a user-defined status/ID vector. The local priority level (IPL) for VMEbus interrupts is programmable. When configured as the system controller, the VIC068A drives the VMEbus IACK daisy-chain.

The following characteristics of local interrupts may be configured in VIC068A registers:

- user-defined local Interrupt Priority Level (IPL)
- option for VIC068A to provide the status/ID vector
- edge or level sensitivity
- polarity (rising/falling edge, active High/Low)

The VIC068A is also capable of generating local interrupts on certain error or status conditions. These include:

- ACFAIL* asserted
- SYSFAIL* asserted

- failed master write-post (BERR* asserted)
- local DMA completion for block transfers
- arbitration timeout
- VMEbus interrupter interrupt

The VIC068A can also issue interrupts by setting a module or global switch in the interprocessor communication facilities (mailbox interrupts).

1.3.8 Interprocessor Communication Facilities

The VIC068A includes interprocessor registers and switches that can be written and read through VMEbus accesses. These are the only registers that are directly accessible from the VMEbus. Included in the interprocessor communication facilities are:

- four general-purpose 8-bit registers
- four module switches
- four global switches
- VIC068A version/revision register (read-only)
- VIC068A Reset/Halt condition (read-only)
- VIC068A interprocessor communication register semaphores

When set through a VMEbus access, the switches can interrupt a local resource. The VIC068A includes module switches that are intended for a single module, and global switches that are intended to be used as a broadcast.