To the Graduate Council:

I am submitting herewith a thesis written by Thomas C. Perry entitled "Design of an Application Specific Integrated Circuit for a VMEbus Industrial Control System." I have examined the final copy of this thesis for form and content and recommend that it be accepted in partial fulfillment of the requirements for the degree of Master of Science, with a major in Electrical Engineering.

Donald W. Bouldin

Donald W. Bouldin, Major Professor

We have read this thesis
and recommend its acceptance:

Robert B. Balchunier

Marshall Pace

Accepted for the Council:

Lee Minkel

Associate Vice Chancellor
and Dean of the Graduate School

STATEMENT OF PERMISSION TO USE

In presenting this thesis in partial fulfillment of the requirements for a Master's degree at The University of Tennessee, Knoxville, I agree that the library shall make it available to borrowers under rules of the Library. Brief quotations from this are allowable without special permission, provided that accurate acknowledgment of the source is made.

Permission for extensive quotation from or reproduction of this thesis may be granted by my major professor, or in his absence, by the Head of Interlibrary Services when, in the opinion of either, the proposed use of the material is for scholarly purposes. Any copying or use of the material in this thesis for financial gain shall not be allowed without my written permission.

Signature Thomas C. Perry

Date 11/30/92

DESIGN OF AN APPLICATION SPECIFIC INTEGRATED CIRCUIT FOR A VMEBUS INDUSTRIAL CONTROL SYSTEM

A Thesis

Presented for the

Master of Science

Degree

The University of Tennessee, Knoxville

Thomas C. Perry

December 1992

ABSTRACT

The introduction of ASIC technology in recent years has provided the digital designer with a unique method for developing electronic systems offering higher levels of performance and functionality coupled with the requirement for less physical space. This project describes the application of ASIC technology into a series of I/O modules. These modules were important components in the development of a VMEbus Industrial Control System at Texas Instruments®. Following the initial documentation of the ASIC design specifications, the project included a sequence of tasks, developed from established ASIC development guidelines, containing distributed responsibilities between the designer and the ASIC device manufacturer. A key requirement in completing the project involved the use of high level software design capture and simulation tools, available from the ASIC manufacturer, operating in an electronic workstation environment.

Verification of the ASIC prototype devices, fabricated from the development process data, provided assurance of a successful design effort. The ASIC device performed all specified functionality during target system operation without failure. Only one minor modification, external to the device, was required to ensure reliable operation under all specified conditions. The ASIC device was considered a significant achievement in the development of the Texas Instruments VMEbus Industrial Control System.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

Analog-to-Digital	A/D
Application Specific Integrated Circuit	ASIC
Computer-Aided Design	CAD
Data Transfer Bus	DTB
Digital-to-Analog	D/A
Input / Output	I/O
Test Description Language	TDL
Printed Circuit Board	PCB
Programmable Logic Controller	PLC

CHAPTER 1

INTRODUCTION

In recent years, significant technological advances in industrial automation systems have given factories the tools necessary to increase productivity, improve quality, reduce the manufacturing cycle time, and decrease material waste.[8] Modern industrial control systems utilize PLC devices coupled with intelligent I/O control point interfaces to provide machine and process control automation. The incorporation of ASIC technology into industrial control systems has proven to be a significant factor in the development of compact automation equipment providing extensive control capabilities. The advantages associated with ASIC technology application in digital design provided the initiative at Texas Instruments for development of an industrial control system utilizing the VMEbus backplane bus. A subset of this development effort included the design of a single, ASIC device to operate in a series of I/O modules which provide an interface to various control point types in the VMEbus Industrial Control System. The specification, design, and verification of this ASIC device is the subject of the sections and chapters that follow.

The Modern Industrial Control System

Industrial control systems provide automated control of applications ranging from a single machine or process to the coordination of multiple machines and processes throughout an entire factory. The foundation of a modern industrial control system consists of a PLC device coupled with

multiple I/O control point interfaces defined by the application. This system provides automated control of machines and processes located within a constrained area of the factory. Multiple PLC devices are often interconnected by a management and control system allowing multiple automated areas distributed throughout the factory to be monitored and controlled from a single network control center.

In the industrial control system, the PLC serves as a central processing unit responsible for processing data received from input control points and providing updated data to output control points. Using the I/O control point interface, the PLC executes a program, stored in internal memory, which is developed for the specific control application. This program enables the PLC to scan the logic state of input control points, make control decisions based on these inputs, and update the logic state of output control points at a periodic rate. The time required by the PLC to perform a scan of all assigned input and output control points is labeled as the system scan time. It is extremely important in the development of industrial control systems that scan times do not exceed the specified system reaction times for output control point updates from the transitions of input control points which ensure reliable operation of the processes and machines under control.

The I/O control point interfaces present in the industrial control system provide the physical connection between the PLC and the field equipment being controlled such as switches, valves, alarms, motors, and relays. Devices containing multiple I/O control point interfaces are referred to as I/O modules. In the industrial control system, I/O modules are modular in design and are mounted into racks or bases. The I/O module

base contains a fixed number of slots, usually eight to sixteen, providing supply power and data communication lines between the PLC and I/O modules present in the base. I/O module bases are located either in close proximity to the PLC or at remote locations by using distributed base controllers. The number and type of I/O control points needed for an industrial control application determines the required number of I/O modules for the application.

In general, I/O modules contain multiple control points with possibly a mixture of input and output point types. I/O modules are classified into one of three categories; discrete, analog, or special function. Discrete I/O modules provide an interface to field devices whose control point physical state is described as one of two possible conditions. In the PLC digital logic, a single logic bit is required to describe the discrete control point physical state. As an example, the LOW and HIGH logic states of a binary logic signal represent the control point physical state as either ON/OFF or OPEN/CLOSED.

Analog I/O modules interface to field devices whose control point physical state can be any one value from a defined range of possibilities. These modules interface to control points containing analog voltages and/or currents as their physical states. Multiple logic bits, organized as a data byte or data word, are required to represent the control point physical state in the PLC digital logic. Analog I/O modules generally contain D/A converters for output control points and A/D converters for input control points which perform the conversion of analog signals into the required digital format. I/O control point data is stored in random access memory located on the

analog I/O module. A microcontroller, also part of the analog I/O module configuration, continuously updates control point data by providing control signals to the analog signal converters and performing read/write cycles to the control point data memory.

To access control point data, the PLC performs read and write data transfer cycles to the memory present on the analog I/O module. Since the microcontroller updates the data on a periodic basis asynchronous to the PLC, the PLC must arbitrate for access to the memory. When the PLC requests memory, the microcontroller completes any current read/write cycle to the memory and grants access to the PLC. This handshaking operation between the microcontroller and the PLC prevents bus contention on the I/O module data bus and thus the possible corruption of control point data.

A final category of I/O modules available in an industrial control system is the special function I/O module. Special function I/O modules are similar to analog I/O modules in that they generally contain a single microcontroller or microprocessor device coupled with on-board memory. These I/O modules provide special interfaces, which are often programmable, for field device control point signal conditioning. By performing complex control point signal conditioning independently of PLC operations, the special function I/O module utilizes parallel processing to ensure the most efficient use of the PLC scan time. This technique allows the PLC to perform control functions reliably and accurately without the burden of controlling complex signal conditioning tasks.

VMEbus Industrial Control System

The VMEbus is a 16/32 bit backplane bus providing an interconnection medium for data processing, memory, and peripheral equipment in a compact mechanical configuration. Suppling a backbone for an industrial control system, the VMEbus offers high performance, versatility, and utilization of the Eurocard format.[3] The VMEbus is comprised of four signal groups or sub-buses providing power and data communication control signals to all devices located on the bus. In addition, the VMEbus contains a communication protocol that specifies the sequence of events present when two or more devices exchange information on the bus. This protocol is designed to provide reliable data communication between multiple devices on the bus without interference to the internal activities of non-participating devices located on the bus. Adherence to the protocol specifications ensures that a device will operate reliably with any other device designed for the VMEbus system. Devices in the VMEbus system are generally classified as either a master or a slave. Master devices arbitrate for control of the VMEbus and perform data transfer cycles either to another master or a slave device. Using a broadcast addressing scheme, the master can communicate with multiple master and/or slave devices simultaneously. Slave devices do not control bus cycles but simply monitor the address, control, and data information present during the cycles. Slave devices are distinguished from one another in the VMEbus system by a unique, binary address assigned by the master device during system initialization. By assigning unique addresses, masters can issue commands and transfer data to an individual slave device without disturbing the internal activities of other slaves.

The VMEbus backplane is contained in a card cage enclosure providing a maximum of 21 slots for plug-in devices or cards. Plug-in cards must conform to a specified mechanical form factor and can occupy a single or multiple backplane slots. The power supply for the backplane is either a plug-in device usually occupying multiple slots or a separate enclosure mounted above or below the card cage. The backplane bus provides power and sub-bus signal connections to all slots in the card cage. Hardware jumpers on the backplane allow daisy-chained signals on the bus to pass through unused slots.

Development of a VMEbus Industrial Control System involves the design of a PLC as the bus master and accompanying series of I/O modules as slave devices in conformance with VMEbus mechanical and electrical specifications. The I/O module series includes modules of the three common types in various I/O control point configurations. A sample representation of a VMEbus Industrial Control System configuration is illustrated in Figure 1.1 with the power supply occupying the first five slots of the backplane. The PLC is designed as a double wide, plug-in VMEbus card occupying slots six and seven. The remaining slots are populated with an assortment of I/O modules required for the particular control application. Any slots not occupied by I/O modules are labeled as empty slots with the backplane jumpers set appropriately. I/O modules are designed as either single or double wide, plug-in VMEbus cards based on the required PCB area for the specific implementation. Wiring connections to field equipment control points are provided through connectors located on the I/O module front panel. Additional I/O modules are placed in any unused card cage slot

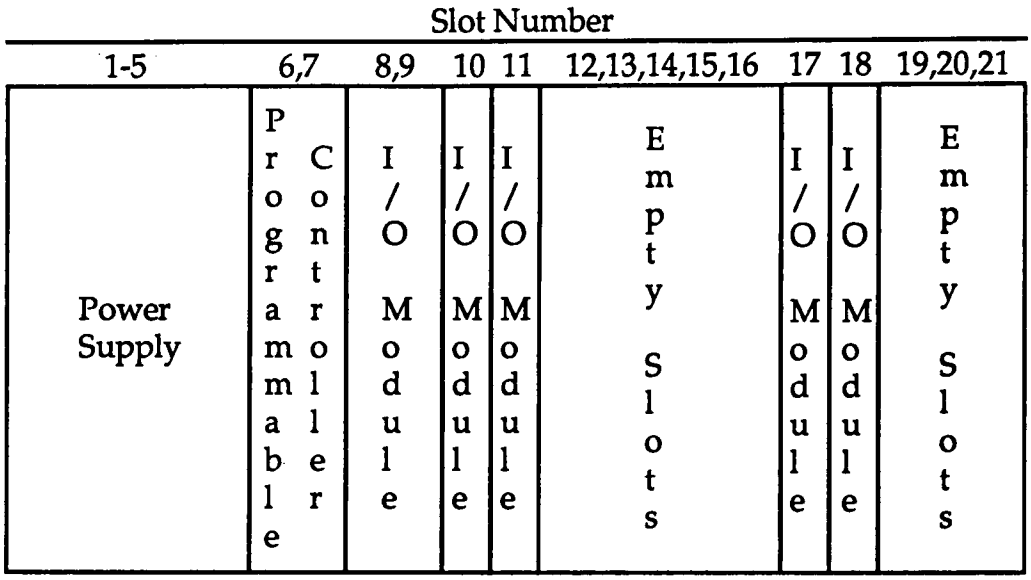

Figure 1.1. VMEbus Industrial Control System

to accommodate future system expansion as the control application is modified by the user.

ASIC Technology in Digital Design

An ASIC is a semiconductor device developed by the design engineer to perform a specific set of logic functions in a unique circuit application. ASICs are generally used to integrate the logic functions of multiple, small- to medium-scale integrated circuits into a single, large-scale integration logic device. The advantages associated with ASICs include increased system reliability and performance coupled with decreased component costs and required PCB area. By reducing the number of signal interconnections present on a PCB, the reliability of the circuit is increased and in general, signal propagation delays are decreased thus enhancing circuit performance. Prototyping costs, integration testing complexity, and costs associated with

design revisions are the primary disadvantages of ASICs. The inability to probe internal ASIC signal interconnections proves difficult for design debug during the initial phases of system testing. In addition, any future ASIC design revision requires another pass through the manufacturer's prototyping process involving significant costs and schedule delays. Coupled with the decision of whether or not to incorporate ASIC technology into a circuit design is the selection of which available ASIC technology type is best for the application.

The two most common types of ASIC technology available to the designer of complex digital systems are gate arrays and standard cells. Gate arrays are the most common ASIC device offering lower cost and quicker turnaround of prototypes. The basic structure of a gate array consists of multiple columns of unwired transistor pairs separated by wiring channels. The perimeter of the gate array contains interface macros for each device I/O pin which are programmable as either an input, output, bidirectional, or power signal pad. Capacity of the gate array is measured in terms of equivalent, 2-input NAND logic gates built from transistor pairs. The transistors contained inside the gate array are wired together to form gates, flip-flops, and other logic functions and then interconnected using the dedicated wiring channels.[6]

Since the gate array contains a fixed array of transistors configured as equivalent gates, the final metal interconnect layers provide the interconnection between gates as specified by the design application. In general, gate arrays are produced by the manufacturer in large quantities up to the final interconnect layers. Only these interconnect layers are required

to customize the device to the design application thus resulting in faster prototype fabrication and lower prototype cost. To simplify the design process, the manufacturer provides a complete library of logic function macros, built from equivalent logic gates, available for use in the gate array. These macros provide logic functions ranging in complexity from basic NAND and NOR gates to multiple-bit adders and comparators. Each macro consumes a specified number of equivalent logic gates from the total gate array device capacity. Using these logic macros, the designer assembles the circuit into a schematic diagram. The manufacturer extracts information from the schematic diagram to assign gates within the array and develop masks for the metal interconnection layers.

Standard cells, when compared to gate arrays, offer a more flexible internal architecture at the expense of higher prototype costs. Similar to gate arrays, standard cells utilize predefined macro cells from the manufacturer's library providing logic functions ranging from simple gates to complete random access and read only memories. Cell size is variable and relates to the implementation of the specific logic function. Since cell placement is not limited to a fixed row architecture, efficient packing of cells into a dense structure generally produces a device with smaller overall die size dimensions when compared to a gate array. As a result, cost per device for a standard cell is usually less than a gate array implementation of an identical logic circuit. The penalty of lower device costs is apparent in higher prototype cost and longer prototype turnaround time associated with the additional fabrication steps of the flexible architecture. Standard cells do offer

the ability to incorporate logic functions such as memories and controllers far more efficiently than gate arrays.

ASIC technology provides designers with a tool for increasing the functionality and performance of their digital systems. Stemming from the integration power of ASICs, designers are able to implement an increasing number of logic functions into less PCB area. Not possible before ASIC technology, complex digital systems can now operate in shrinking electronic enclosures requiring less power and producing less heat. ASIC technology was a driving force in the development of the VMEbus Industrial Control System. Using the PCB mechanical constraints specified by the VMEbus Eurocard format, approximately 70% of the allotted PCB area is required by I/O control point conditioning circuitry. This circuitry converts the AC/DC currents and voltages, common in switches and relays of industrial field devices, to the low voltage logic signal levels required by the PLC. As a result, only 30% of the PCB area is available for the digital control and bus interface logic which provides the interface between the I/O module and the PLC as defined in the functional specifications. ASIC technology provided the best strategy for suppling the logic capacity necessary to implement the I/O module functional specifications in the least amount of PCB area.

The ASIC Development Cycle

The development of an ASIC device begins with the adoption of a design specification. This specification, which is developed following functional partitioning of the digital system, contains all information required to design and test the device. Partitioning is the process of dividing

the digital system implementation, generally along functional lines, into a set of blocks or modules. Partitioning is a trial and error process based on factors such as packaging constraints, circuit timing considerations, and available logic technology. In general, a complex digital system is partitioned into PCB assemblies which in turn are partitioned into functional logic blocks. Partitioning of the digital system continues until each logic block is described as an ASIC device or collection of standard logic components. Completion of the partitioning process results in specification information concerning I/O pin signal assignments, signal interfaces, and timing requirements for each ASIC device in the system.[6]

From the device specification, the ASIC design is then internally partitioned into functional blocks. Partitioning continues within each functional block creating a hierarchical path down to the complete logic gate implementation of each specified device function. This hierarchical structure forms the basis for creating schematic diagrams describing the ASIC design. Using high-level design tools developed by the manufacturer and operating on an electronic workstation platform, the designer selects logic macros from the ASIC device library, generates the schematic diagrams, and performs verification of the ASIC design. The signal net lists, created by the workstation design tools, supply the manufacturer with all required data to perform layout and routing in the target ASIC device.

A simulation utility, available in the workstation environment, provides the designer with a tool for verifying the ASIC design. The first step in the simulation process is to develop a set of test vectors containing input signal stimulus, in a chronological time sequence, which emulates

operation of the digital system at the ASIC input pins. An analysis of the resulting signal logic states produced at the ASIC output pins during simulation determines if the logic implementation performs the desired operation. Initial functional simulations are generally for design verification only without regard for circuit performance or signal timing. Unit propagation delays are assumed by the simulator for all internal logic devices when establishing logic states on the design output signals. The simulation phase following functional verification, performance simulation, incorporates the effects of gate propagation delay, output loading, and signal interconnect timing into the resulting output signal states. Since the physical device layout for the design is not available from the manufacturer at this point, the timing delays produced by loading effects of signal interconnect wiring are best estimations calculated by the simulator. Design performance with regard to signal timing is verified during this stage of simulation for both best and worst case logic device delay scenarios.

At the completion of functional and performance simulations, the designer develops a test program for the ASIC design. The test program is used by the manufacturer to verify operation of the device following fabrication. For this reason, it is extremely important that the test program is designed to not only exercise all logic functions within the device completely but also provide detection of internal faults. Internal device faults include logic gate interconnections stuck at either the LOW or HIGH logic state. While the test program provides a complete functional test incorporating fault coverage, the test program length must be designed efficiently to require the least amount of test time. High volume ASIC devices should not require

significant amounts of test time which create bottlenecks at the manufacturer's test equipment in the production facility.

The test program is developed to provide device simulation input stimulus which conforms to strict timing specifications outlined by the manufacturer's test equipment. The test program is built as a chronological sequence of equal, fixed length test periods. Within each test period, input signals are driven to defined logic states, a clock pulse is provided on all clock inputs, and output signals are strobed for specified logic states. The test period length is a fixed value, generally on the order of one thousand nanoseconds, providing device verification at one megahertz operation. Although the ASIC device may be designed to operate at system speeds greater than one megahertz, the test program does not simulate the design under those conditions. The test program is intended as a tool for ensuring correct device functionality without faults and does not evaluate signal timing performance.

Upon completion of the test program, the designer submits the ASIC design to the manufacturer for device layout and interconnect routing. The manufacturer uses an automated CAD system to assign individual logic functions to specific gates in the target ASIC device. Logic functions are strategically placed inside the device to simplify signal routing and provide the most efficient utilization of available gates. If the design does not fit into the specified device, the designer is advised to select the next larger available device offering greater gate density and possibly a larger I/O pin count. All signal routes and interconnections are then completed within the constraints

of the selected device. At this point, all required data to fabricate the device is available to the manufacturer.

Before the prototype devices are fabricated, the manufacturer supplies the layout and routing information back to the designer. This data allows the simulator to account for the physical device layout effects on design performance. Using the supplied data, the simulator incorporates the capacitive loading effects, calculated for the interconnect wiring, into logic gate delays. This method provides simulation results which emulate the actual fabricated device. The designer verifies the results from both performance and test program simulations which utilize the device layout parasitics. When complete confidence with the design performance is achieved, the designer approves the manufacturer to begin fabrication of the prototype devices. Within approximately four to six weeks, the designer receives a small quantity of prototype devices, fabricated and tested by the manufacturer. These devices are subjected to a series of tests which verify their conformance to all design specifications and ensure reliable operation in the target system. Upon successful completion of system testing, the designer accepts the prototype devices and releases the ASIC design to production.

The ASIC development cycle just described is a detailed sequence of steps requiring close interaction between the designer and manufacturer at key points in the process. Completion of all steps outlined in the ASIC development cycle and adherence to established design guidelines is critical to a first pass ASIC design success. The schedule impact and engineering costs associated with additional passes through the development cycle to fix

design problems can have disastrous effects on system time to market and cost constraints. The burden lies with both the manufacturer and the designer to ensure the proper steps are taken to produce a quality ASIC device in a timely and efficient manner. For the manufacturer, providing a high quality ASIC development environment, for popular workstation platforms, with accurate logic macro libraries is extremely important. It is the designer's responsibility to produce a reliable logic design which conforms to established guidelines. The designer must also verify the design for accuracy in terms of both functionality and performance using simulation options available in the ASIC development environment.

CHAPTER 2

OPERATION OF A SLAVE DEVICE IN A VMEBUS SYSTEM

The term "VMEbus" describes an interfacing system used to interconnect data processing, memory, and peripheral control equipment into a compact hardware configuration. The VMEbus system provides a 16/32-bit backplane through which multiple devices can communicate without disturbing the internal activities of other devices located on the bus. The VMEbus is classified as an asynchronous bus since data throughput performance rates are based on device response times rather than bus cycle timing restrictions.[3]

The VMEbus Specification from Motorola® outlines mechanical and electrical design specifications for master and slave devices operating in the VMEbus system. In addition, the VMEbus backplane signal definitions are presented along with illustrated timing specifications. Devices that adhere to these specifications are assured reliable operation with other devices located on the VMEbus. The VMEbus backplane contains signals grouped into four sub-buses; the DTB, the priority interrupt bus, the utility bus, and the arbitration bus. Each sub-bus provides the signals required to perform specific functions within the VMEbus system. The sections that follow detail the design specifications for each sub-bus as they apply to a slave device operating in the VMEbus system. The arbitration bus provides a tool for transferring control of the DTB between VMEbus masters and is transparent to slave device operation on the bus. For this reason, the arbitration bus is not included in this discussion.

VMEbus DTB

The VMEbus DTB is an asynchronous, parallel data bus used by VMEbus masters to access bus devices for data transfer operations. The DTB contains 31 address lines (A31 to A1), 32 data lines (D31 to D0), six address modifiers (AM5 to AM0), and seven bus control signals. The control signals consist of two data strobes (DS0*,DS1*), long word (LWORD*), bus error (BERR*), data transfer acknowledge (DTACK*), read/write control (RD/WR*), and address strobe (AS*). A signal name followed by an asterisk represents a signal whose active logic state is LOW. Master and slave devices utilize all or part of the available control signals during DTB cycles based on the defined system address/data bus widths and VMEbus cycle type.

The DTB is an asynchronous bus since bus performance is determined by the response time of the slave device during the data transfer cycle. A VMEbus master waits for an acknowledgement from the responding slave before terminating a bus cycle. This mode of operation allows slave devices to control the amount of time required to complete the data transfer cycle and permits slaves with both fast and slow response times to operate in the same VMEbus system. As a precaution, a bus timer is present in the VMEbus system to generate a bus error condition if an acknowledgement is not received within a specified time period. This time period, or bus timeout period, is generally on the order of a few microseconds. In most cases, a bus timeout is the result of a slave device malfunction or a DTB cycle error. Often, the master clears the bus error and attempts the data transfer cycle again before a system error condition is reported.

At the beginning of a data transfer cycle, a VMEbus master drives all or part of the 31 DTB address lines with a valid binary address value. Distinguished by the width of the address during a data transfer cycle, three addressing modes are possible on the DTB. Extended addressing mode is used by a master that drives all 32 address lines during a DTB cycle. The lower 24 address lines are valid during standard addressing while only the least-significant 16 address lines are driven in the short addressing mode. During standard and short addressing modes, slave devices are not responsible for decoding the unused address lines. Independent of the addressing mode, the address value present on the VMEbus points to a memory location unique to only one device in the system. This prevents two slaves from responding to the same data transfer cycle, creating contention on the bus.

In conjunction with the address, the master drives the six address modifier lines along with control signals LWORD* and IACK* at the beginning of the bus cycle. The address modifier lines provide information to the slave devices concerning the type of data transfer cycle currently underway on the bus. This information includes the addressing mode and the type of master initiating the cycle. LWORD* signals all devices addressed by the cycle that the master is requesting a 32-bit data transfer operation. IACK* indicates the presence of an interrupt acknowledge cycle on the VMEbus. The interrupt acknowledge cycle is a special cycle that utilizes a small subset of DTB signals and is described in the discussion of the priority interrupt bus. Slaves do not acknowledge normal data transfer cycles if IACK* is active. Furthermore, slaves operating on the VMEbus should

respond only to DTB cycles which are within the specified limitations of the slave device.

The VMEbus master produces a falling edge on signal AS* to indicate the presence of valid logic levels on the DTB address lines, DTB address modifiers, LWORD*, and IACK* for the current cycle. Slaves are often designed to latch the state of these signals with AS*, but this is not a requirement since the master continues to drive these signals until the cycle is acknowledged by the slave. Following the assertion of AS*, the master establishes signal RD/WR* to a logic level that corresponds to the direction of data flow for the DTB cycle. A logic HIGH on RD/WR* indicates data is read from the slave while a logic LOW signals data is written to the slave during the DTB cycle. RD/WR* remains valid until the DTB cycle acknowledgement is received from the slave device.

At this point all address and control information is available to the slave device and the data transfer portion of the DTB cycle is ready to begin. Data is transferred as a long word when all 32 data lines are driven; as a word when the least-significant 16 data lines are driven; and as a byte when only eight data lines are driven onto the VMEbus data bus. Signal LWORD*, when active, accompanies cycles transferring either all 32 data lines or an unaligned series of data bytes in the same cycle. The master drives one or both of the data strobes to initiate the data transfer in the DTB cycle and provide the slave with two types of information. Data strobes indicate whether a word or byte of data is to be transferred during the cycle and which byte, ZERO or ONE, if a byte cycle. A logic LOW level present on data strobe ZERO, DS0*, indicates a DTB cycle to byte ZERO. In a similar manner, a logic

LOW level on data strobe ONE, DS1*, signals a DTB cycle to byte ONE. For a device data bus containing 16 lines, data byte ZERO is comprised of the most significant eight lines labeled eight through fifteen while data byte ONE is represented by lines seven to zero. A logic LOW level present on both data strobes indicates a DTB cycle transferring a complete word of data. When a slave detects an active data strobe(s), the slave drives data onto the VMEbus if the cycle is a read cycle or latches data from the VMEbus if a write cycle.

For the slave, two signals are available on the VMEbus for providing DTB cycle status information to the master. Signal DTACK* is driven by the responding slave when data is transferred successfully during the DTB cycle. DTACK* indicates to the master that the slave has latched data from the VMEbus data bus during a write cycle or is driving valid data onto the VMEbus data bus if a read cycle. Upon reception of DTACK*, the master terminates the DTB cycle. If an error occurs during a DTB cycle, the responding slave drives BERR* signaling the master to terminate the cycle. When no slave device responds to a DTB cycle, the bus timer drives BERR* after a fixed timeout delay.

DTB Cycle Timing Specifications

In addition to functional definitions of signals available on the DTB, The VMEbus Specification outlines timing specifications detailing signal transition events that occur during a DTB cycle. These timing specifications or rules are necessary for designing master and slave devices which operate correctly in the VMEbus system. Devices that are designed to meet all VMEbus timing specifications provide reliable communication with other

devices located in the system. The timing specifications applicable to slave device operation are illustrated in Figure 2.1 for a DTB read cycle and Figure 2.2 for a DTB write cycle. Timing parameters are numbered and listed in Table 2.1. In the diagrams, a special notation is used to illustrate the data strobe timing. Since the data strobes do not always perform logic level transitions simultaneously, DSA* is used to indicate the data strobe which is first to transition active and/or inactive during the cycle. DSB* represents the data strobe which follows DSA* when both data strobes transition in the cycle. The areas labeled with a repeating series of capital letter X represent the unknown state of the data bus when not driven by any device on the VMEbus. This condition is commonly referred to as a tristate bus. The term "driven" is used to describe the bus condition when a device is in the initial or final stages of driving the data bus but the logic levels are not yet stable. "Valid" represents the bus when driven with stable logic levels on all required data lines during the DTB cycle. Understanding the timing specifications of read and write cycles on the DTB provides valuable insight into the design of a slave device operating in the VMEbus system.

A DTB cycle begins with the master establishing valid logic levels on the VMEbus address lines, address modifiers, LWORD*, and IACK*. These signals are guaranteed to be stable at least ten nanoseconds before the detection of a falling edge on AS* or DSA* by a slave. Although the master does not assert DSA* before AS*, it is possible for a slave to detect the falling edge of DSA* before the falling edge of AS* due to bus loading effects giving rise to signal skew on the VMEbus. If this situation occurs, the slave is guaranteed that the assertion of AS* occurs no later than ten nanoseconds

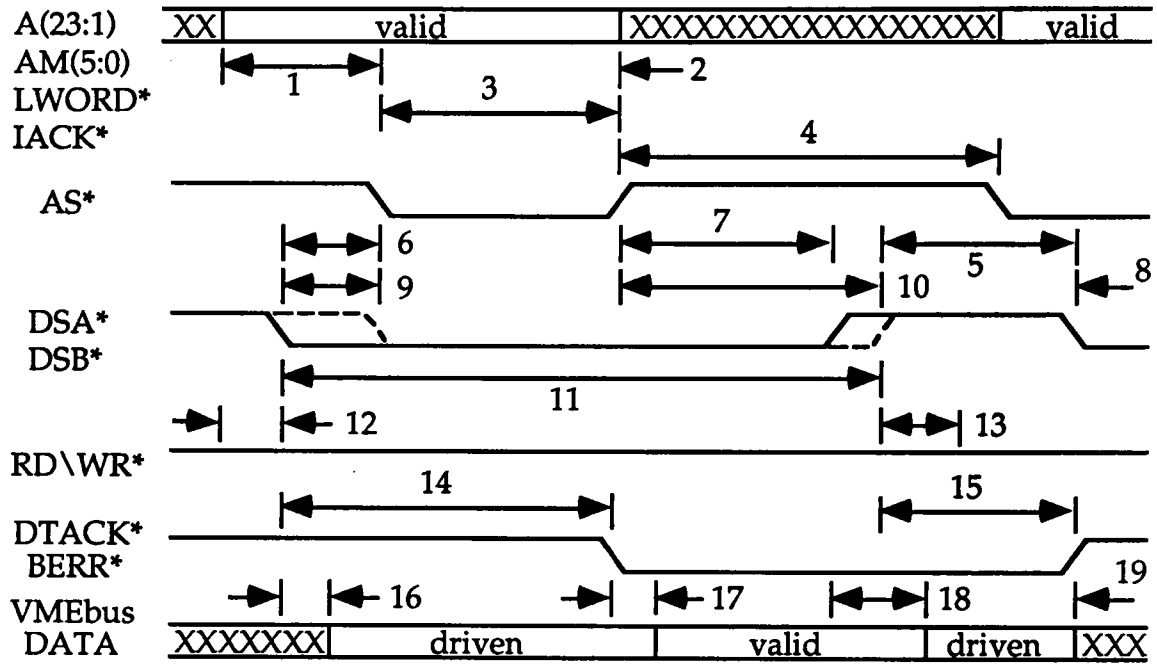

Figure 2.1. DTB Read Cycle Timing Diagram

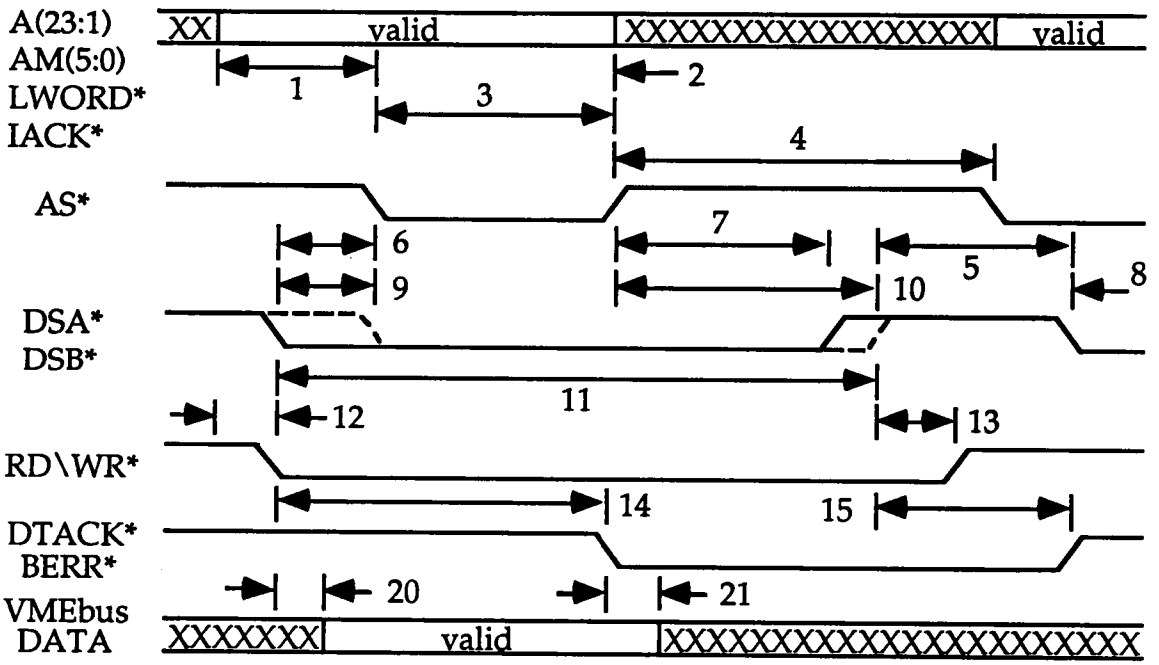

Figure 2.2. DTB Write Cycle Timing Diagram

Table 2.1. Timing Parameters for DTB and Priority Interrupt Bus

Timing Parameter	Description	Delay at VMEbus Slave [a] Minimum (ns)	Maximum (ns)
1	A(23:1) valid to AS*	10	
2	DTACK* to AS*	0	
3	Active hold time AS*	30	
4	Release hold time AS*	30	
5	Inactive hold time DSX*	30	
6	DSA* to AS*		10
7	AS* to DSA*	0	
8	DTACK* to DSA*	0	
9	DSA* to DSB*		20
10	DTACK* to DSB*	0	
11	Active DS region	30	
12	RD/WR* to DSA*	10	
13	DSB* to RD/WR*	0	
14	DSA* to DTACK*/BERR*	30	
15	DSB* to DTACK*/BERR*	0	
16	DSA* to VMEbus DATA	0	
17	VMEbus DATA to DTACK*	0	
18	DSA* to VMEbus DATA	0	
19	VMEbus DATA to DTACK*	0	
20	VMEbus DATA to DSA*	10	
21	DTACK* to VMEbus DATA	0	
22	AS* to IACKIN*	40	
23	AS* to IACKIN*		40
24	IACKIN* to DTACK*	0	
25	Inactive hold time IACKIN*	30	
26	IACKIN* to IACKOUT*	0	
27	IACKOUT* to AS*	30	
28	AS* to IACKOUT*		30

[a] Symbol ns represents delay in nanoseconds (10^{-9} seconds).

after the falling edge of DSA*. In addition, signal RD/WR* is also stable at least ten nanoseconds prior to the assertion of DSA*. To compensate for the potential skew between the detection of falling edges on DSA* and DSB*, a slave must allow a minimum of 20 nanoseconds after the assertion of DSA* to detect the possible assertion of DSB*. This timing specification ensures that a slave does not respond to a DTB cycle with only a data byte when the master is actually requesting a data word.

As mentioned previously, the falling edge of DSA* provides an indication to the slave that the data transfer portion of the DTB cycle is beginning. For a DTB read cycle, the slave is required to drive the VMEbus data bus with valid data and assert signal DTACK* at any time following the assertion of DSA* but before the bus timeout period. The slave asserts DTACK* during a DTB write cycle only after the data, present on the VMEbus, is successfully written into the appropriate memory device. A slave must not assert DTACK* for at least 30 nanoseconds from the falling edge of DSA* to ensure correct decoding of the data strobes. If an error occurs during the DTB cycle, the slave drives signal BERR* instead of DTACK* within the same timing restrictions.

Upon detection of either DTACK* or BERR*, the master begins the process of terminating the DTB cycle. The master releases the address lines, address modifiers, IACK*, LWORD*, AS*, DSA*, and DSB* within a minimum of zero nanoseconds from the falling edge of DTACK* or BERR*. A slave is guaranteed that signal RD/WR* remains valid until both DSA* and DSB* are inactive. The slave is required to maintain its assertion of DTACK* or BERR* until both DSA* and DSB* are released by the master.

During a DTB read cycle, the slave must drive valid data on the VMEbus until the release of DSA*. The master cannot initiate the data transfer portion of a new DTB cycle by reasserting DSA* until the slave has released DTACK* or BERR* from the previous cycle.

VMEbus Priority Interrupt Bus

In conjunction with the DTB, the VMEbus backplane provides a priority interrupt bus used for the generation and service of interrupt requests in the VMEbus system. The VMEbus interrupt structure consists of seven, prioritized interrupt request levels represented by seven interrupt request lines, IRQ1* to IRQ7*, contained within the priority interrupt bus. Interrupts are serviced through a priority structure with highest priority assigned to level seven, IRQ7*. Interrupt acknowledge priority decreases from level seven to the lowest priority interrupt, IRQ1*. In addition to the interrupt request lines, the priority interrupt bus also contains the interrupt acknowledge signal, IACK*, and the interrupt acknowledge daisy chain. The interrupt acknowledge daisy chain structure allows the VMEbus system to have multiple devices connected to a single interrupt request line. A slave device interrupts operation of the VMEbus system by asserting an interrupt request line through the use of an open collector driver. If two slave devices are asserting a single interrupt request line simultaneously, the interrupt acknowledge daisy chain ensures that only one device responds to each interrupt acknowledge cycle. The interrupt acknowledge daisy chain, illustrated in Figure 2.3, consists of a single wire which spans the entire length of the VMEbus backplane passing through each device located on the

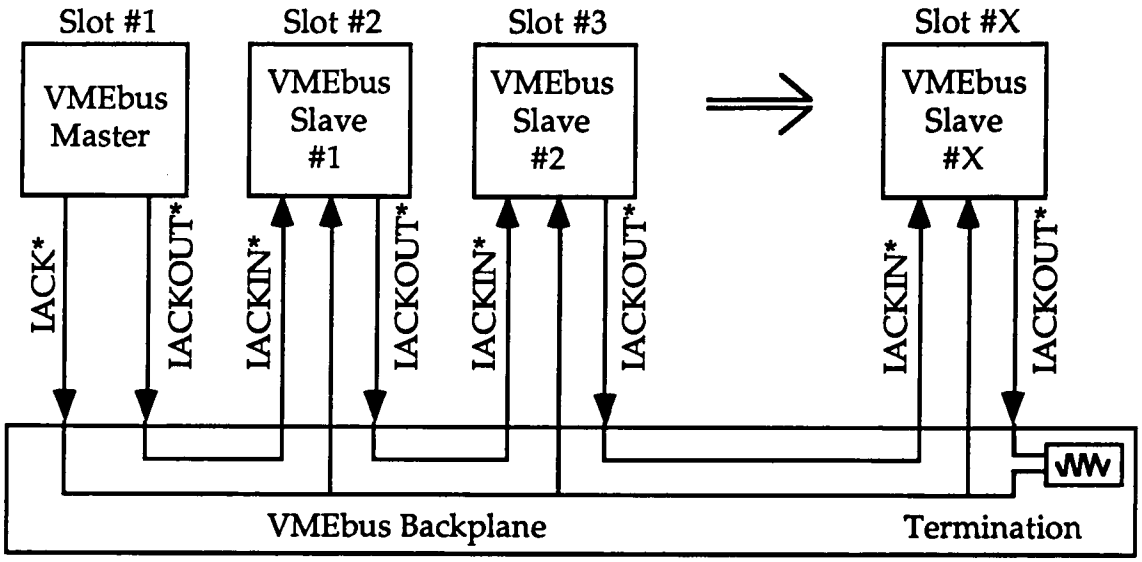

Figure 2.3. Interrupt Acknowledge Daisy Chain

bus. The daisy chain enters each device as signal IACKIN* and exits each device as IACKOUT*. At the beginning of an interrupt acknowledge cycle, the master asserts a falling edge onto its IACKOUT* line. This falling edge moves down the daisy chain, propagating through each device not requesting an interrupt, until it reaches the device that is requesting the interrupt. The requesting device does not propagate the falling edge to its IACKOUT* line and responds to the interrupt acknowledge cycle by driving an interrupt service routine vector onto the VMEbus. This vector provides the master with address information concerning where tasks, required for servicing the interrupt request, are located. If an unpopulated slot exists in the VMEbus backplane, a jumper is used to connect IACKIN* to IACKOUT* directly thus maintaining the integrity of the daisy chain for the entire length of the backplane.

The interrupt acknowledge cycle is a special cycle on the VMEbus which utilizes a subset of the DTB signals in addition to the interrupt acknowledge daisy chain. Timing specifications for an interrupt acknowledge cycle are illustrated for a responding device in Figure 2.4 and for a non-responding device in Figure 2.5. The numbered timing parameters are included with other DTB parameters in Table 2.1 on page 23. The interrupt acknowledge cycle begins in a fashion similar to the DTB cycle with the three least-significant address lines, LWORD*, and IACK* driven to valid logic levels before the falling edge of AS*. The remaining address lines and address modifiers are not used and should not be decoded during the interrupt acknowledge cycle. IACK* is driven to a logic LOW level distinguishing the interrupt acknowledge cycle from a normal DTB cycle. The interrupt level being serviced during the cycle is represented as a binary value on the three least-significant address lines.

Following the initialization of the interrupt acknowledge cycle, the master produces a falling edge on its IACKOUT* line a minimum of 40 nanoseconds from the assertion of AS*. Each device that is not requesting an interrupt of the level being acknowledged propagates the falling edge to its IACKOUT* line and maintains a logic LOW level on IACKOUT* until AS* is released at the end of the cycle. Each non-requesting device is also required to release its IACKOUT* line no later than 30 nanoseconds after AS* is released by the master. The device that responds to the interrupt acknowledge cycle does not assert its IACKOUT* line but drives the VMEbus data bus with an interrupt vector during the data transfer portion of the cycle. The responding device also asserts either DTACK* or BERR* to

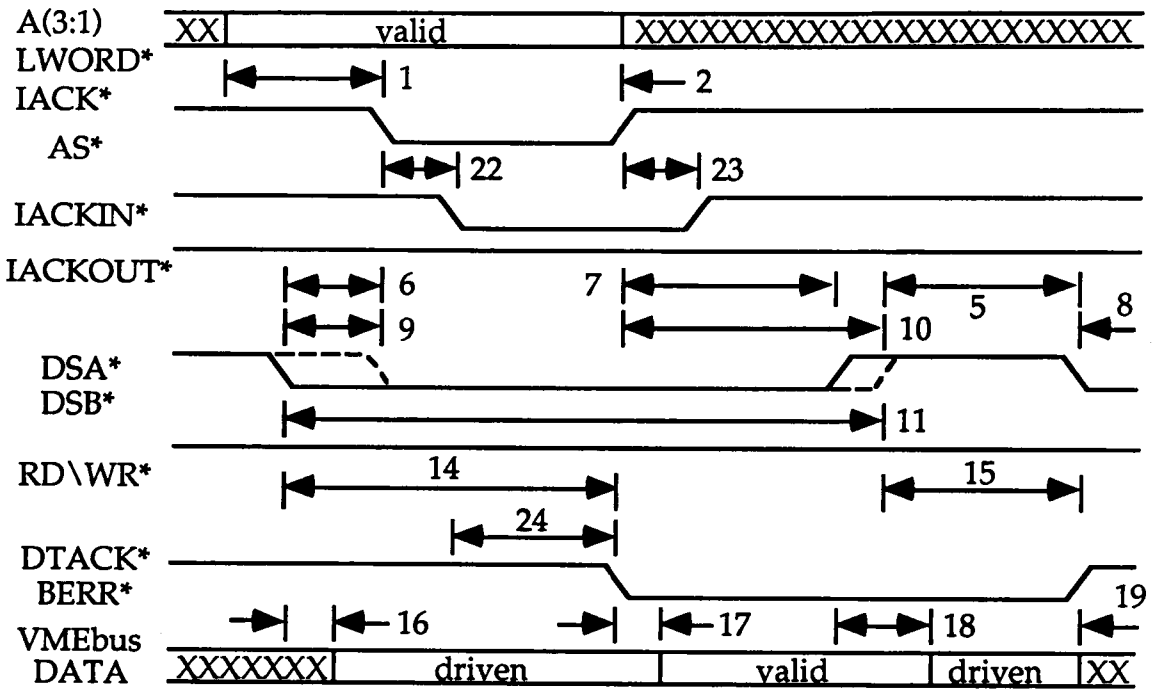

Figure 2.4. Interrupt Acknowledge Cycle for a Responding Device

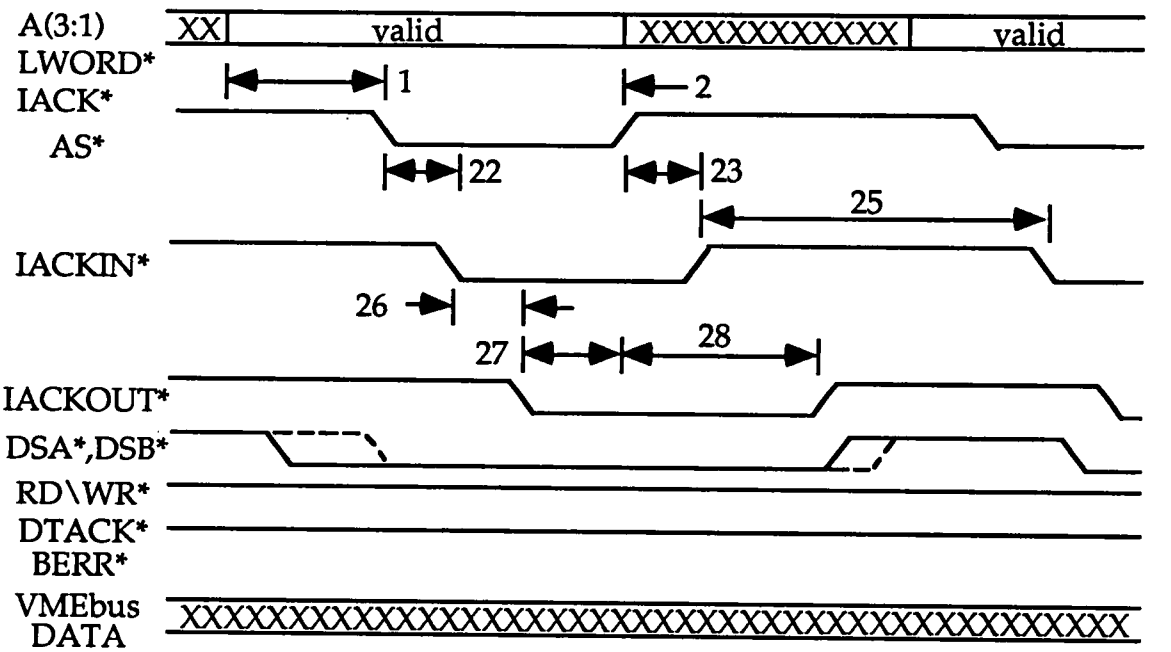

Figure 2.5. Interrupt Acknowledge Cycle for a Non-Responding Device

initiate termination of the cycle. Timing specifications for the data transfer portion of the interrupt acknowledge cycle are basically identical to a normal DTB cycle and can be referenced in the previous discussion.

VMEbus Utility Bus

Utility functions such as periodic timing, initialization, and diagnostic capabilities are provided by the VMEbus utility bus. Two utility bus signals are of specific importance to slave devices operating in the VMEbus system. These signals are labeled system reset, SRESET*, and system clock, SYSCLK.

SRESET* is an open collector VMEbus backplane signal used for system initialization. Any device which is capable of asserting SRESET* is required to maintain a logic LOW level on SRESET* for a minimum of 200 milliseconds. This time interval allows all devices in the VMEbus system to complete their unique initialization requirements before SRESET* is released and normal operation begins. SRESET* is asserted during system shut-down and power-up events to ensure that devices in the system begin and cease operation in an orderly fashion. SRESET* is also asserted by bus devices in response to push-button inputs or system operation malfunctions.

SYSCLK is a fixed frequency input clock signal provided on the VMEbus backplane. The frequency of SYSCLK is 16 megahertz ± 1.6% with a 50 ± 10% duty cycle. SYSCLK is an independent clock signal exhibiting no fixed timing or phase relationships with any DTB or priority interrupt bus signal events. SYSCLK provides a time base for the VMEbus system that is always present independent of the SRESET* line.[3]

Additional DTB Features

The VMEbus DTB contains additional features which enhance bus performance and control in systems with distributed processing capabilities. Two DTB cycles with special characteristics coupled with a technique for compressing consecutive DTB cycles are discussed in the paragraphs that follow. These features are outlined as requirements in the VMEbus Industrial Control System Specification and warrant consideration in the design process of devices operating in the system.

Inside a DTB cycle, the master uses AS* to strobe address and control information onto the bus along with DS0*and DS1* to control data transfer. The master is allowed to release AS* immediately following the assertion of DTACK* or BERR* by the responding slave. The master may also drive new address and control information onto the bus, and after a minimum delay, reassert AS*. This sequence of events is described as "address pipelining" and allows a master to provide address and control information for an upcoming DTB cycle while the data transfer portion of the current cycle is in progress. Address pipelining enhances bus performance by overlapping DTB cycles during system operation. When using the address pipelining technique, the master is not allowed to violate the minimum "off" time specification for AS*, DS0*, and DS1*, which is 30 nanoseconds at the slave device. In addition, the master must not assert DS0* or DS1* for an upcoming cycle until the responding device has released DTACK* or BERR* in the current cycle.

For VMEbus systems with multiple master devices, a DTB cycle exists which performs a read cycle immediately followed by a write cycle to the

same system address without arbitration of bus control between the cycles. This cycle is referred to as a read-modify-write cycle and is characterized by a read cycle followed by a write cycle where AS* remains asserted between the cycles. When AS* is active, control of the VMEbus cannot be transferred between VMEbus master devices. The read-modify-write cycle allows a master device operating in systems with multiple bus masters to share memory and I/O devices on the VMEbus without the potential of interruption during the status and update of a system address.

The address-only cycle is a DTB cycle which does not involve the transfer of data. The cycle begins as a normal cycle with address, address modifiers, LWORD*, and IACK* driven with valid logic levels followed by a falling edge on AS*. The master asserts AS* for a minimum hold-time of 30 nanoseconds, measured at the slave device, before releasing AS* and terminating the cycle. The master does not assert DS0*, DS1*, or control signal RD/WR* nor does the master wait for an acknowledgement from the slave. Address-only cycles are used primarily for the broadcasting of information from a master to all listening devices present on the VMEbus. This information is generally used to issue commands and/or update system status to slave devices.

CHAPTER 3

I/O MODULE ASIC SPECIFICATION

During the initial planning phase of the VMEbus Industrial Control System, development and system engineering personnel outline performance, packaging, and cost alternatives for the system. As specifications evolve, specific design requirements and constraints for I/O modules operating in the system are assembled into a document. This document, referred to as the I/O Module Specification, forms the basis from which hardware and ASIC development engineers design and test I/O module PCB assemblies. An integral part of the I/O module design includes the incorporation of complex logic functions into ASIC technology to decrease cost and increase reliability. The I/O Module ASIC Specification is developed to outline the design requirements for the ASIC device. An overview of the specifications for the I/O Module ASIC device are presented in the sections that follow.

Design Cost Constraints

The most important design constraint in developing a series of I/O modules for operation in a modern industrial control system is the ability to provide the largest variety of I/O point functionality with the lowest possible cost. Cost per I/O control point in an industrial control system is a key competitive feature in the world market. For this reason, system designers are encouraged to develop PLCs with large I/O control point handling capabilities coupled with the ability to accommodate a variety of different I/O

control point types. Furthermore, the cost of system expansion resulting from the addition of control points is a significant factor in the cost per control point equation. Since system expansion costs are a direct reflection of I/O module costs, the I/O module design significantly impacts the final cost per control point calculations for the system. The market pressures to contain I/O module costs provides an incentive for incorporating ASIC technology into the I/O module design. Using an ASIC device to implement address decode and control logic minimizes PCB area required for these functions. This provides maximum utilization of limited board area for actual I/O point conditioning circuitry. In addition, the ASIC device improves I/O module reliability by reducing the component count and signal interconnections present on the PCB assembly.

Resulting from cost constraints and design schedule limitations, the development of a single, ASIC device for operation in the entire series of analog, discrete, and special function I/O modules provided a more feasible approach than the development of a unique ASIC device for each I/O module type. A single, general purpose ASIC reduces the development and inventory costs generally associated with multiple devices. However, the general purpose ASIC is required to perform I/O module control operations in a method which is transparent to the I/O point type. External programmable logic devices are used in conjunction with the ASIC to provide control signals for I/O point data storage devices specific to the module type. During VMEbus data transfer cycles, the ASIC device must generate control signals used by external devices to provide either buffer/latch enables for a discrete I/O module or static ram control signals in

the analog or special-function I/O module. In addition, a microcontroller operating on the I/O module necessitates the presence of bus arbitration logic in analog and special function I/O modules. The ASIC device is required to provide the proper handshaking signals and arbitration logic to prevent contention on the I/O module data bus.

The I/O Module ASIC Specification outlines design requirements for the ASIC device which achieve cost and functionality specifications of the I/O module PCB assemblies. The primary requirement of the ASIC is to reduce logic present on the I/O module PCB to only I/O point data storage devices, a module identification device, and data transceivers required for VMEbus communication. The only clock input available to the ASIC is the 16 megahertz VMEbus system clock present on the backplane. This limitation eliminates the need for an external clock device on the I/O module PCB assembly. Furthermore, the ASIC is confined to a TGC103 device which is the smallest member of the Texas Instruments Gate Array Family. The TGC103 device restricts the ASIC to a maximum of 3000 equivalent NAND gates of logic with 84 I/O pins housed in a plastic leaded chip carrier package. The specification of a gate array device as opposed to a standard cell offers a reduction in the device prototype cycle time and associated fabrication costs. These reductions are key components in the plan to remain within established development cost and schedule constraints for I/O modules in the VMEbus Industrial Control System. Although standard cell devices offer lower unit costs, the projected I/O module production volumes did not provide an adequate payback incentive to offset the higher costs and cycle time associated with device fabrication.

VMEbus Compatibility Requirements

The I/O Module Specification describes I/O module operation in the VMEbus Industrial Control System as a slave device accommodating both standard and short addressing modes allowing a maximum data bus width of 16 bits. The I/O module ASIC provides control signals conditioned by external programmable array logic devices for I/O point data storage devices and is also responsible for regulating data flow between the eight-bit I/O module data bus and the 16-bit VMEbus data bus. The ASIC supplies the necessary enable, direction, and control signals for the VMEbus data bus transceivers, located on the I/O module PCB, to ensure correct operation during byte and word data transfer cycles.

During VMEbus cycles, the ASIC is responsible for decoding the address and address modifiers to determine if I/O module participation in the cycle is required. At the beginning of each DTB cycle, the ASIC decodes the address modifier lines to determine the addressing mode for the cycle. Each I/O module operating in the VMEbus Industrial Control System is assigned a unique, eight-bit physical address by the PLC which locates the module in the system configuration map. The ASIC retains this physical address in memory and performs a comparison between the physical address and the binary value present on VMEbus address lines A24 through A16 at the beginning of each standard DTB cycle. If the two addresses are equal, the master is attempting to access a memory location present on the I/O module and participation in the DTB cycle is required. The ASIC performs the data transfer operation, specified by DTB control signals, to an I/O module memory location decoded from the remaining address lines, A15 through

A1. A15 through A1 contain a binary value which is used as an offset pointer into the I/O module memory map. The memory map is identical for all I/O module types and is outlined in the next section. The ASIC does not decode the VMEbus address lines during an extended addressing mode DTB cycle.

In the VMEbus Industrial Control System, short addressing mode DTB cycles are used to initiate events on and query status information of I/O modules in the system. The ASIC does not decode address lines A24 through A16 or use A15 through A1 as a memory map offset pointer during short mode DTB cycles. There are two short addressing mode DTB cycles defined for I/O modules operating in the VMEbus Industrial Control System. A short mode, address-only cycle broadcast on the VMEbus to hexadecimal address 0122 initiates all I/O modules in the system to immediately disable all output points. A short mode, data transfer cycle to an address representing the binary value equivalent of hexadecimal 0100 + 2 (I/O module logical address) either reads status information from the I/O module if a read cycle or initiates a module reset if a write cycle. The I/O module logical address is a unique, five-bit value assigned by the PLC during system initialization. Implementation of the I/O module logical address scheme allows masters without standard addressing capability to access I/O modules on an individual basis.

After decoding the address and address modifier lines, the ASIC determines if a valid VMEbus cycle to an I/O module memory map address offset exists. If the cycle is valid and involves data transfer, the ASIC decodes control signals DS0*, DS1*, and LWORD* to determine the amount of data

requested by the master. Since I/O modules provide a maximum of 16 data bits in a single DTB cycle, the ASIC verifies the logic level present on control signal LWORD* to ensure that the DTB cycle is not requesting a 32-bit data transfer operation. If the data transfer requested is valid for the memory offset accessed, the ASIC provides the control signals necessary for external I/O module components to complete the data transfer operation and acknowledges the cycle to the master by asserting DTACK* onto the VMEbus. If the cycle is a valid access to an I/O module address offset and an error results, the ASIC asserts signal BERR* onto the VMEbus. Bus errors are generated for cycles which request transfer of greater than 16 data bits or are invalid for the particular I/O module type. Also, a DTB write cycle to an I/O module address offset defined as a read-only memory location results in a bus error.

The logic design of the I/O module ASIC requires detailed analysis to ensure compliance with all VMEbus timing specifications during decoding, data transfer, and acknowledgement of VMEbus cycles. Logic required to implement the VMEbus features supported by I/O modules is contained within the ASIC device. These features include address pipelining, the read-modify-write cycle, and the interrupt acknowledge daisy chain. The ASIC monitors signal IACK* and IACKIN* to detect the presence of an interrupt acknowledge cycle. When the cycle is present, the ASIC decodes the appropriate address and control signals to determine if a response to the cycle is required. If a response is not required, then the ASIC asserts its IACKOUT* signal passing the cycle to the next device on the daisy chain. The interrupt

service routine vector transferred on the VMEbus during the interrupt acknowledge cycle is contained in ASIC device memory.

Memory Map

The memory map, shown in Figure 3.1, illustrates the configuration of I/O module memory for identification, status, I/O point status, physical address, logical address, and special function data storage. Status, identification, and address information areas are common to all I/O module types while I/O control point and special function data have distinct, assigned memory blocks. The I/O module type determines which areas of the memory map are accessible by a system master. Since the ASIC operates transparent to I/O module type, external programmable array logic devices are present for providing module type information to the VMEbus and for determining which areas of the I/O module memory map are accessible by the master. For example, an I/O module containing only discrete I/O points does not provide access to memory map areas dedicated for analog and special function I/O point data. Additional information from the external programmable array logic device outlines the number of I/O control points of each specific type that are present in the I/O module configuration. From this information, the VMEbus master is again only allowed access to address offsets in the memory map which correspond to the number of I/O points available of a specific point type. An I/O module containing 16 input discrete I/O points does not provide access to memory map offsets containing data for discrete I/O points 17 through 64. This implementation prevents a master from transferring invalid I/O point data to/from a system I/O module.

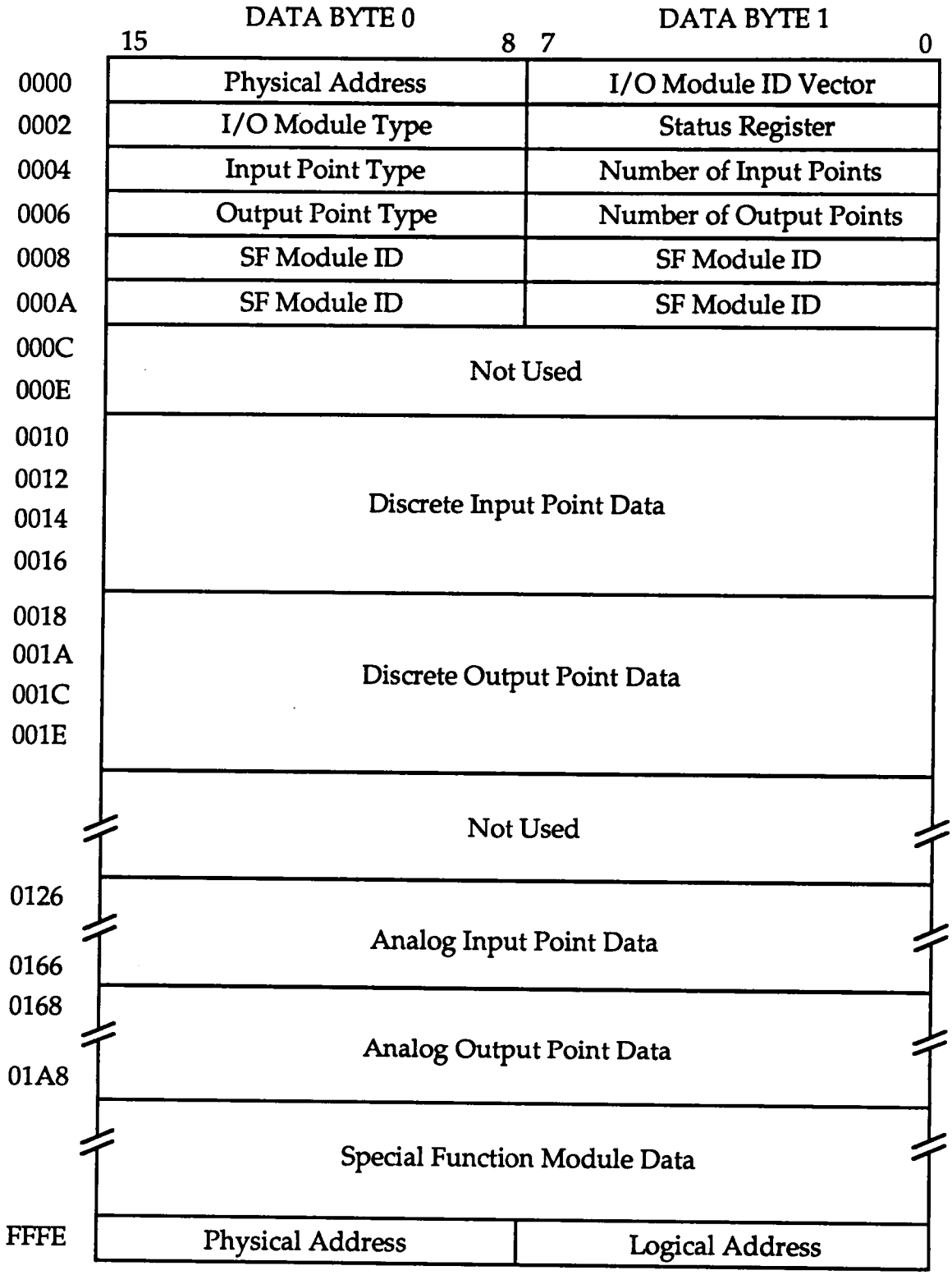

Figure 3.1. I/O Module Memory Map

The I/O module memory map is organized into 16-bit, word-memory offsets addressed by VMEbus address lines A15 through A1. The least-significant bit of the 16-bit address, always assumed as logic LOW in the memory map, provides a byte offset pointer and is replaced by DS0* and DS1* in the VMEbus system. Each address offset consists of two data bytes, ZERO and ONE. Data byte ZERO represents data bits zero through seven while byte ONE contains bits eight through fifteen. Three types of data transfer operations are possible to an offset location in the memory map. A VMEbus word cycle transfers the entire 16 bits of offset data while a byte cycle transfers only one of the two possible offset data bytes. The I/O module ASIC decodes control signals DS0* and DS1* to determine if a data word or specific data byte is being transferred during the current DTB cycle. The ASIC then provides the proper sequence of control signals required by the bus transceivers to transfer data between the VMEbus and the correct I/O module memory device(s).

In the I/O module memory map, hexadecimal address offset 0000 contains the I/O module physical address in data byte ZERO and the I/O module identification vector in data byte ONE. When read by a VMEbus master, the identification vector provides an eight-bit code identifying the device as an I/O module in the VMEbus Industrial Control System. The I/O module type vector, identifying the module as either analog, discrete, or special function, is read from data byte ZERO of address offset 0002. A status register containing current I/O module operation information is read at offset 0002, data byte ONE. Address offsets 0004, 0006, 0008, and 000A contain additional I/O module information including module input and output

point types, number of points present of each type, and special function module ID data if required. Although the ASIC stores the I/O module physical address and identification vector, module status and I/O point type information are contained within external PCB memory devices controlled by the ASIC. To a VMEbus master, address offsets 0000 through 000A are read-only memory locations with the exception of offset 0002. A write operation is permitted to offset 0002 as an indication to clear an existing module output disable and/or watchdog timeout. During this cycle, the data present on the VMEbus is ignored and a write operation is not actually performed to a memory map offset.

Discrete input control point data is located in the I/O module memory map beginning at address offset 0010 and continuing to offset 0016. A maximum of 64 discrete input control points are allowed on an I/O module organized as shown in Figure 3.2. Input points are numbered zero through 63 with point zero located at bit zero of offset 0010 and increasing sequentially to bit 15 of offset 0016. An input point read as logic LOW is considered active or ON during control operations. In a similar manner, discrete output

	DATA BYTE 0								DATA BYTE 1							
	15							8	7							0
0010	I15	I14	I13	I12	I11	I10	I9	I8	I7	I6	I5	I4	I3	I2	I1	I0
0012	I31	I30	I29	I28	I27	I26	I25	I24	I23	I22	I21	I20	I19	I18	I17	I16
0014	I47	I46	I45	I44	I43	I42	I41	I40	I39	I38	I37	I36	I35	I34	I33	I32
0016	I63	I62	I61	I60	I59	I58	I57	I56	I55	I54	I53	I52	I51	I50	I49	I48

Figure 3.2. Discrete Input Point Data

control point data, present at address offsets 0018 through 001E, also exhibit active LOW polarity characteristics. Discrete input point address offsets are considered as read-only memory to system masters while offsets containing discrete output point data are accessible by both read and write cycles. A read operation to a discrete output offset performs an I/O module output control point readback operation. The logic state of the output control points are driven onto the data bus by the I/O module during a readback operation.

Control data for analog type I/O points are located in address offsets 0126 through 01A8 of the I/O module memory map. Analog I/O point data is represented as a 13-bit, binary value contained inside a word address offset as illustrated in Figure 3.3. The 13-bit binary value consists of a digital representation of the analog voltage or current signal in bits zero through 11 accompanied by a sign bit 12. The sign bit provides a positive or negative polarity representation for the analog signal value. The remaining, most-significant three bits, 13 through 15, of each analog I/O point address offset are always read as logic level LOW during DTB cycles. The memory map allows a maximum of 64 analog inputs contained in offsets 0126 through

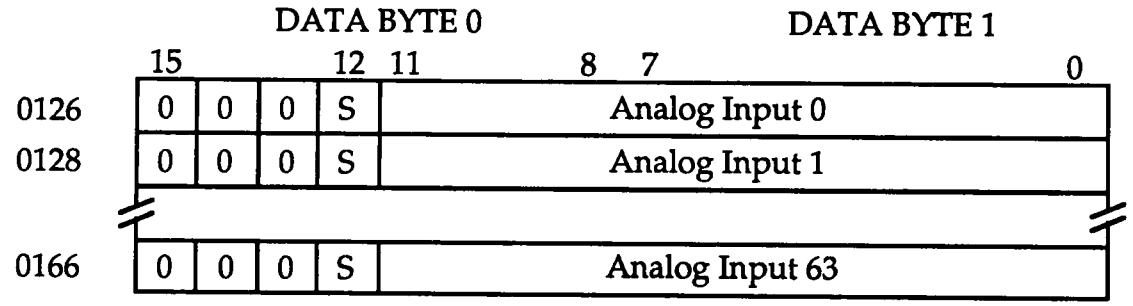

Figure 3.3. Analog Input Point Data

0166 and 64 analog outputs present at offsets 0168 through 01A8. As with discrete I/O point data, analog input point offsets are read-only memory while analog output point offsets are accessible by both read and write data transfer cycles. A read cycle to an analog output control point offset provides a readback feature by returning the most recent data value written into static RAM by the PLC.

Address offsets 01AA through FFFC in the I/O memory map provide scratchpad and I/O control point data memory for special function modules. The configuration for this area of memory is defined during system initialization by the PLC. The last address offset in the memory map, offset FFFE, is a reserved location containing the I/O module physical address in byte ZERO and the module logical address in byte ONE. Written during I/O module initialization, this offset is read-only during normal operation. The logical address is written once and then subsequently read as eight bits but only the least-significant five bits, zero through four, are used by the ASIC during address decoding.

Initialization Routine

As a power source is applied to the VMEbus Industrial Control System, SRESET* is held at a logic level LOW for a minimum of 200 milliseconds. During this time, I/O modules in the system power-up into an initial reset state. At the first rising edge on SYSCLK following the return of SRESET* to logic level HIGH, each I/O module completes an initialization routine to clear memory and establish all output control points to the inactive state. Each I/O module then asserts its interrupt request level six,

IRQ6*, onto the VMEbus requesting I/O module initialization. I/O module interrupts are serviced in a specified order beginning with the module located in closest proximity to the PLC and proceeding down the interrupt acknowledge daisy chain until all interrupt requests are cleared.

The PLC acknowledges the I/O module interrupt request by initiating an interrupt acknowledge cycle on the VMEbus. Each I/O module decodes address lines A3 through A1 along with other VMEbus control signals to ensure a level-six interrupt acknowledge is in progress. In addition, each I/O module waits for a falling edge on its IACKIN* input before responding to the interrupt acknowledge cycle. If a falling edge is not detected at IACKIN* before the interrupt acknowledge cycle is terminated, the I/O module takes no action and resets its logic to decode the next interrupt acknowledge cycle. An I/O module responds to an interrupt acknowledge cycle by placing an eight-bit vector onto the VMEbus data bus and asserting DTACK*. The PLC uses this vector to execute an interrupt service routine defined for I/O modules in the system. During an interrupt acknowledge cycle response, the I/O module releases its interrupt request line and does not assert IRQ6* again unless reinitialization is required.

During the interrupt service routine, the PLC reads identification and control point type information from the I/O module before assigning the unique physical and logical addresses. These addresses are written to the I/O module through short-mode DTB cycles to address 0120 as either separate byte cycles or a single word cycle. If the addresses are written with two separate byte cycles, the byte order in which the addresses are written is not important but I/O module initialization is not complete until both cycles are

terminated. This requires an I/O module to only respond to DTB cycles containing short-mode address 0120 until both the physical and logical addresses are written into memory. Following power-up or a system reset, the PLC initializes the I/O modules in a sequential manner until all I/O modules in the system are initialized. The I/O module ASIC contains all decode and control logic required to perform the I/O module initialization routine and also dedicates internal memory for storage of the assigned physical and logical addresses.

Watchdog Timer

The watchdog timer is a I/O module feature provided as a mechanism for alerting the PLC that an access of I/O control point data has not occurred within a specified time period. This mechanism is a very important part of an industrial control system controlling processes and/or machinery since continuous monitoring and updating of control point information is essential to minimizing the probability of a system malfunction. The I/O module ASIC contains all logic required to implement the watchdog timer and provides an output signal to disable output control points when a timeout occurs. The ASIC also asserts a bit in the I/O module status register to indicate the presence of a watchdog timeout condition. To clear the watchdog timeout and resulting output control point disable, a watchdog timeout clear command is required from the PLC. This command consists of either a data word or date byte ONE write cycle to address offset 0002 in the I/O module memory map. The data present during the cycle is irrelevant and is ignored by the I/O module. During normal I/O module operation, the

ASIC clears the watchdog timer with each PLC access of I/O control point data. This procedure is often referred to as petting the watchdog timer and prevents the watchdog timeout event.

CHAPTER 4

I/O MODULE ASIC DESIGN

The development of an ASIC device using gate array technology provides a method of combining multiple logical functions into a single, VLSI device offering lower power consumption, higher reliability, and requiring less PCB area. ASIC design practice follows established guidelines for logic design including the use of synchronous logic whenever possible, elimination of race conditions, avoidance of floating signals, and design for testability. Designing the ASIC device to implement the required logic functionality while providing controllability and observability of internal signal interconnections is essential to a successful ASIC development project. The design approach taken, tradeoffs considered, and theory of operation for the I/O module ASIC device are discussed in the sections that follow.

Metastability

An ASIC design which contains logic that is asynchronous in nature can result in a device that is unreliable and difficult to test. Two signals are described as asynchronous when no fixed timing relationship exists between them. A circuit containing asynchronous logic is more likely to exhibit unreliable operation due to glitches, race conditions, and metastability than a circuit which is synchronized to a system clock. In high-speed logic designs, the use of asynchronous logic is often unavoidable and requires the designer to perform detailed signal timing analysis in an effort to ensure correct circuit operation during worst-case conditions.

Metastability is the unpredictable operation of a memory device DATA output resulting from signal transitions on two or more device inputs within a specified time window. Manufacturers provide detailed timing specifications outlining minimum time intervals between transitions on device input signals to ensure reliable operation of memory components such as a flip-flop or latch. These timing specifications generally include a minimum time, before or after a device clock or latch enable input transition, in which other device input signal transitions are not permitted. Setup and hold timing specifications describe the minimum time that the DATA input to a memory device can not transition before or after the rising edge of the device clock. The inactive-state setup time specifies the minimum time before a rising clock edge that a CLEAR or PRESET input must not transition logic state. A logic design that does not adhere to these timing specifications for memory devices can result in a circuit which exhibits metastability.

When a device exhibits metastable operation, the behavior of the DATA output is unpredictable for an unspecified period of time from the initiation of the condition. Although the DATA output eventually settles to a valid logic level, this logic level may not be the state expected when the inputs are presented correctly to the device. During the time period of unpredictable operation, the DATA output can float at an illegal logic level or oscillate between logic levels. This condition is best described as "like a pencil balanced on its point, it cannot stay this way for long. It can, however, stay undecided for much longer than its specified settling time."[5] Although there is no fixed time that the DATA output is guaranteed to settle, the

settling time is best modeled by an exponential probability distribution curve. The curve describes the probability for a specified amount of time from the initiation of the metastable condition, that the DATA output has settled to valid logic state. Therefore, an infinite amount of time is required to ensure, with 100% probability, that the DATA output is stable after metastability occurs. Probability distribution curves are sometimes generated for specific memory devices through experimental setup, but data is generally not readily available. In design practice, a time delay is imposed on the propagation into circuit logic of a memory device DATA output that is susceptible to metastability. This time delay is selected to provide an acceptable probability that the DATA output has settled to a valid logic level. As a general rule, an acceptable probability is achieved by delaying the DATA output by three to four times the maximum memory device propagation delay from clock to DATA output.

To minimize the effects of metastability, asynchronous signals are often synchronized to a system clock. In most cases, metastable conditions are confined to synchronizing logic placed in the asynchronous signal path. The addition of synchronizing logic does add delay to the signal path which may not be tolerable in high-speed circuit designs. A common circuit used for synchronizing asynchronous signals consists of two D-type flip-flops and a inverter as illustrated in Figure 4.1. During operation, the asynchronous signal is clocked into the synchronizing flip-flop on the falling edge of the system clock. The DATA output of the synchronizing flip-flop is then clocked into the logic flip-flop DATA input at the next rising clock edge. When a metastable condition occurs in the synchronizing flip-flop, the

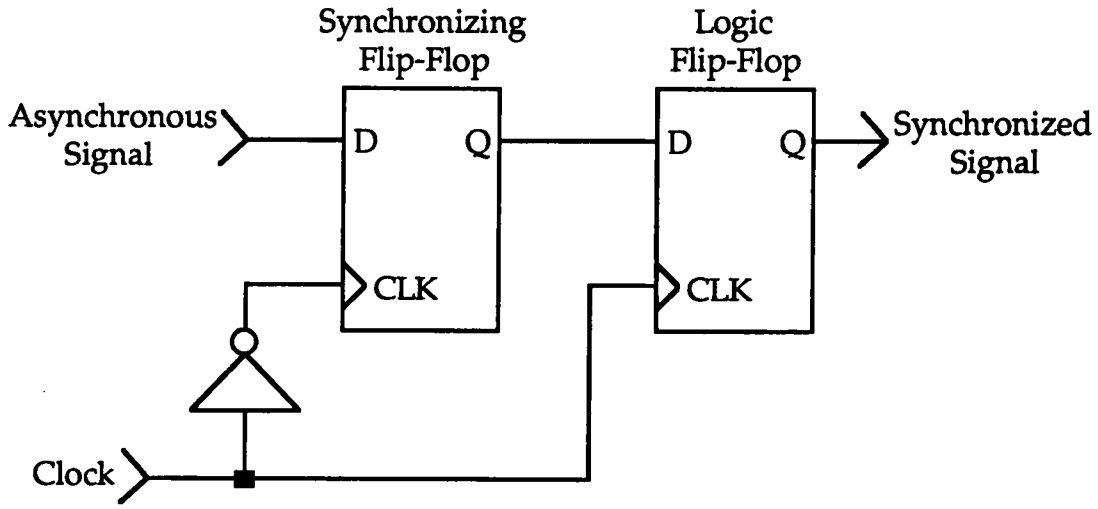

Figure 4.1. Synchronizing Logic Circuit

DATA output is allowed a half-clock cycle minus the data setup time of the logic flip-flop for settling to a valid logic state. Using the VMEbus system clock as the clock input, a minimum settling time of 25 nanoseconds is provided based on clock-skew specifications. This settling time averages six to seven times the maximum propagation delay of a flip-flop memory device in the TGC103 gate array library and provides a high probability that a stable input is present at the logic flip-flop DATA input when the rising clock edge occurs. The DATA output of the logic flip-flop is therefore propagated into the circuit as a signal synchronized to the system clock and free from metastability.

The penalty associated with using a synchronizing circuit results from sampling the asynchronous input signal one-half clock cycle before the logic flip-flop DATA output is propagated into the circuit. Circuit response time to logic transitions on asynchronous inputs is increased since a transition occurring after the falling edge of the clock is not detected until the

following clock cycle. This situation can delay circuit response to asynchronous input transitions by a maximum of one and one-half clock periods. When using the VMEbus system clock as the synchronizing clock, the circuit response delay approaches a maximum value of 101 nanoseconds which is calculated from clock duty cycle specifications.

In addition to the DATA input, CLEAR and PRESET inputs to memory devices are also synchronized for metastability protection. If a PRESET or CLEAR input to a memory device is released within a minimum time before a rising edge of the clock input, the inactive-state setup time is violated and unpredictable DATA output operation is possible. To reduce the probability of metastability, a partial synchronization is performed on the asynchronous CLEAR or PRESET input. The new, synchronized CLEAR or PRESET signal follows the asynchronous input during a transition to an active logic level but is synchronized with the clock when the input returns to the inactive state. Using the synchronized CLEAR and PRESET as inputs to a memory device significantly increases the probability that when a clock input edge arrives, the CLEAR and PRESET inputs are stable at a valid logic level.

The single, most important design specification for any I/O module in the VMEbus Industrial Control System is to provide reliable operation over time and temperature extremes. Frequent I/O module failure or inconsistency of operation can not be tolerated in a system where industrial processes and equipment are being monitored and controlled. As controller of all I/O module operations, the importance of a reliable ASIC design far exceeds speed and simplicity design considerations. For this reason,

extensive synchronizing logic in the I/O module ASIC is used to produce output signals with predicable timing relationships for controlling I/O module functions. Synchronizing logic is the key interface between external, asynchronous inputs from the VMEbus and the synchronous control logic present in the ASIC. Although the synchronous ASIC design provides enhanced reliability and testability, increased logic complexity and circuit response delay are the tradeoffs. Neither of these characteristics are considered as significant disadvantages in the I/O module ASIC design since adequate logic gate capacity to accommodate additional synchronizing logic is provided in the gate array device and I/O module cycle response speed is not a critical performance specification of the VMEbus Industrial Control System.

ASIC Design Considerations

The I/O module ASIC is designed into a TGC103 device from the Texas Instruments TGC100 Gate Array Family. This family contains gate arrays of various densities constructed using a one micron gate length, CMOS technology. Designing with CMOS logic presents additional precautions which the ASIC designer must address to avoid potential problems.

Logic gate inputs and tristate buses internal to the ASIC design which are not driven to valid logic levels and allowed to float can result in a ASIC device which draws excessive supply current. A logic gate input, not driven, can float to the logic threshold causing both output stage, MOS transistors to turn on at least partially which results in static current flow into the logic gate. This produces excessive current draw by the ASIC and can lead to

device failure. In addition, the output of a logic gate which has an undriven input may oscillate generating excessive noise in the device. The TGC100 Gate Array Family contains two logic structures or macros available to eliminate the potential problems associated with floating signal interconnections.

A tie-off macro is a TGC100 library device which contains no inputs but provides either a logic HIGH or logic LOW output. Connecting a tie-off macro to an unused logic gate input eliminates floating interconnections internal to the ASIC device. A TGC100 library bus-hold macro is placed on any signal interconnection capable of being driven by multiple logic devices and, when not driven, returns to a tristate condition. When a signal enters tristate, the bus-hold macro maintains a resistive drive at the same logic level present on the interconnection prior to the tristate condition. This resistive drive strength is easily overridden by the next device which drives the interconnection. The bus-hold macro provides a tool for eliminating floating data buses internal to the ASIC device.

Although there is not a specified limit to the load that a CMOS logic gate can drive, high fanouts on logic gate outputs internal to the ASIC is not recommended. As the logic gate output load increases, the switching time from one logic level to the other is extended. During the switching or transition time, there exists a brief time period where both the N- and P-channel output stage transistors of the logic device are in the active region. As the switching time is extended due to excess loading, the time period is increased and the logic gate draws more current. This results in excess power consumption by the ASIC device. The TGC100 Gate Array Family contains

several signal buffer devices with various drive strengths for distributing signal loading within the design. Signal buffering is incorporated throughout the I/O module ASIC design to minimize logic fanout and reduce potential loading problems.

ASIC Design Tradeoffs

During the development of the I/O module ASIC logic design, several tradeoffs were encountered and evaluated. These tradeoffs included the width of the I/O module data bus, number of I/O pins required for the ASIC device package, and the implementation strategy for I/O module identification information. The evaluation process for each tradeoff examined criteria for alternative selection such as cost, complexity, and flexibility. In the final analysis, the alternative chosen provided the strictest adherence to applicable I/O module ASIC design specifications.

The initial tradeoff associated with the ASIC design involved the implementation of the I/O module data bus as either an eight-bit or 16-bit bus. The I/O module data bus provides an interconnection medium between all memory devices present on the I/O module PCB and the VMEbus data bus. In either bus configuration, two data bus transceivers are resident on the I/O module PCB to interface the data bus with the 16-bit VMEbus data bus. The ASIC implements the control logic necessary to provide output enable, latch enable, and direction control signals for these transceivers. In addition, the ASIC contains memory devices associated with several address offsets in the I/O module memory map and is capable of driving data onto the I/O module data bus during VMEbus DTB cycles.

Therefore, the ASIC design must allocate enough device package pins to accommodate the selected I/O module data bus width.

Using a 16-bit wide I/O module data bus offers a potential reduction in the I/O module response time during data word, VMEbus cycles. This stems from the ability of the ASIC to latch 16 bits of data into the transceivers simultaneously as opposed to one data byte at a time as required by an eight-bit I/O module bus. Disadvantages associated with the 16-bit data bus include the dedication of a larger number of ASIC device I/O pins to accommodate the bus width in addition to increased PCB area for bus interconnection routing. Using 16 ASIC device I/O pins for the data bus connection reduces the number of pins available for control signals and may require use of a larger density gate array device to gain additional I/O pins. This is a violation of the I/O Module ASIC Specification and the use of a larger ASIC device package is not an option.

An eight-bit I/O module data bus requires fewer ASIC device I/O pins and less PCB area for interconnection routing, but does slow data word, DTB cycle response times. Using an eight-bit I/O module data bus during a word DTB cycle, the ASIC must route the data, one byte at a time, between the VMEbus and the I/O module. The additional time required to handle the data properly, approximately 300 to 400 nanoseconds, delays the assertion of the cycle acknowledgement to the PLC. Considering PLC scan times in terms of milliseconds, this cycle response delay does not present a significant degradation to system performance. In general, a PLC scan involves the reading of input control points, analyzing control decisions based on these inputs, and writing updated output control point data for all I/O modules

operating in the system. PLC scans occur with frequency on the order of hundreds of milliseconds and therefore, the increased I/O module cycle response delay does not add significant delay to system scan performance. Based on ASIC device packaging limitations and minimal impact of cycle acknowledgement performance, the I/O module data bus width was chosen as eight bits. The eight-bit data bus allows the ASIC to provide additional I/O pins for performing control operations which in turn reduces the amount of external logic present on the I/O module PCB.

A second design tradeoff evaluated during the early stages of the development process involved ASIC device packaging. The ASIC is limited to a TGC103 device from the Texas Instruments Gate Array Family which provides two package options, a 68 pin or 84 pin plastic leaded chip carrier. Use of a 68 pin device offers less cost and less required PCB area for the ASIC, but results in 16 fewer device I/O pins available for control operations. Fewer ASIC device I/O pins requires the removal of selected control logic functions which in turn must be implemented in external logic devices. An estimate of the external logic required to implement the functions displaced by the smaller package ASIC yielded two programmable array logic devices and a latch device. The component cost of these devices, approximately three dollars and 50 cents, is considerably higher than the cost delta between the ASIC package options of slightly over one dollar. In addition, the added PCB area required by the larger ASIC device package is decisively less than the area consumed by the smaller package ASIC with accompanying external logic devices. To achieve both cost and PCB real estate advantages, the 84 pin package option was selected for the I/O module ASIC device.

The final design tradeoff involved selection of an implementation strategy for the storage and transfer of I/O module identification information. This tradeoff resulted from the I/O Module ASIC Specification requirement for a single, ASIC device designed to operate in the entire I/O module series defined for the VMEbus Industrial Control System. Although the ASIC is designed to control data transfer operations to I/O point data memory devices transparent of I/O point type, the storage and transfer of I/O module identification information is a task which can not be independent of I/O module type. Two strategies were examined for the implementation of I/O module identification information.

In the first approach, the ASIC device contains a set of I/O pins dedicated as identification inputs from external logic devices on the I/O module PCB. Binary codes, present at these ASIC input pins, indicate the I/O module type, number of input points, number of output points, and the I/O point electrical specification information. The ASIC decodes these binary codes for developing the appropriate identification information to be transferred onto the VMEbus data bus when requested by the PLC. The binary codes provide the total number of I/O points present in the I/O module configuration so that a bus error is issued to a PLC which attempts to access I/O points which are not available. The identification decode logic operation, internal to the ASIC, is designed for a fixed set of I/O module configurations supported in the VMEbus Industrial Control System. These configurations specify the allowable combinations of input and output control point counts of each point type along with available options for electrical specifications of each I/O control point. Further evaluation of this

approach exposes several disadvantages to decoding identification information by logic contained within the ASIC device.

Flexibility is the primary disadvantage to the binary code implementation of identification information. At the completion of the ASIC design, the identification decode logic is designed for a fixed set of I/O module configurations. Any future I/O module configuration additions or revisions will require a revision to the ASIC design. This is not desirable since ASIC design revisions result in substantial prototyping costs and schedule delays. In addition, the volume of identification information and number of I/O module configurations is extensive. Therefore, the amount of decode logic and the number of ASIC device I/O pins required to implement the identification function is significant. In consideration of these requirements, use of a larger ASIC device package is probable which is an alternative not supported by the I/O Module ASIC Specification.

A second approach to the implementation of I/O module identification involves a programmable array logic device external to the ASIC. Using a small number of control signals from the ASIC, the programmable array logic device implements the I/O module identification information function and drives the information onto the data bus when requested by the ASIC. This approach isolates I/O module identification information storage to the programmable array logic device thus allowing the ASIC to operate with complete independence of the particular I/O module configuration. In addition, any future additions or revisions to the I/O module configuration can be implemented by the programmable array logic device without an ASIC design revision. Since the ASIC is not aware of

the number of I/O control points present in the I/O module configuration, the external programmable array logic device also determines if a PLC access to a particular memory map offset is valid and reports the decision back to the ASIC. Although an external logic device is required, this approach was chosen due to increased flexibility and efficient use of ASIC device I/O pins.

I/O Module Block Diagrams

In conjunction with the I/O module identification functionality, additional external programmable array logic device(s) are present to provide ASIC signal conditioning for the particular I/O module configuration. This signal conditioning consists of control signal decoding on discrete I/O modules and address line conditioning on analog or special function I/O modules. Hardware block diagrams illustrating the digital control circuitry and I/O point data storage devices located on the I/O module PCB are discussed in the following paragraphs. In both diagrams, signals illustrated on the left side represent VMEbus signals to/from the I/O module while signals shown on the right side represent signals to/from I/O control point conditioning circuitry.

The block diagram in Figure 4.2 illustrates the hardware configuration required for a 16 input point and 16 output point discrete I/O module. Any number combination of input and output points is permitted in the I/O module configuration up to the maximum of 64 for each type. Octal buffers drive input control point data onto the I/O module data bus while octal latches with readback capability store the output control point data. The number of buffers, latches, and programmable array logic devices present is

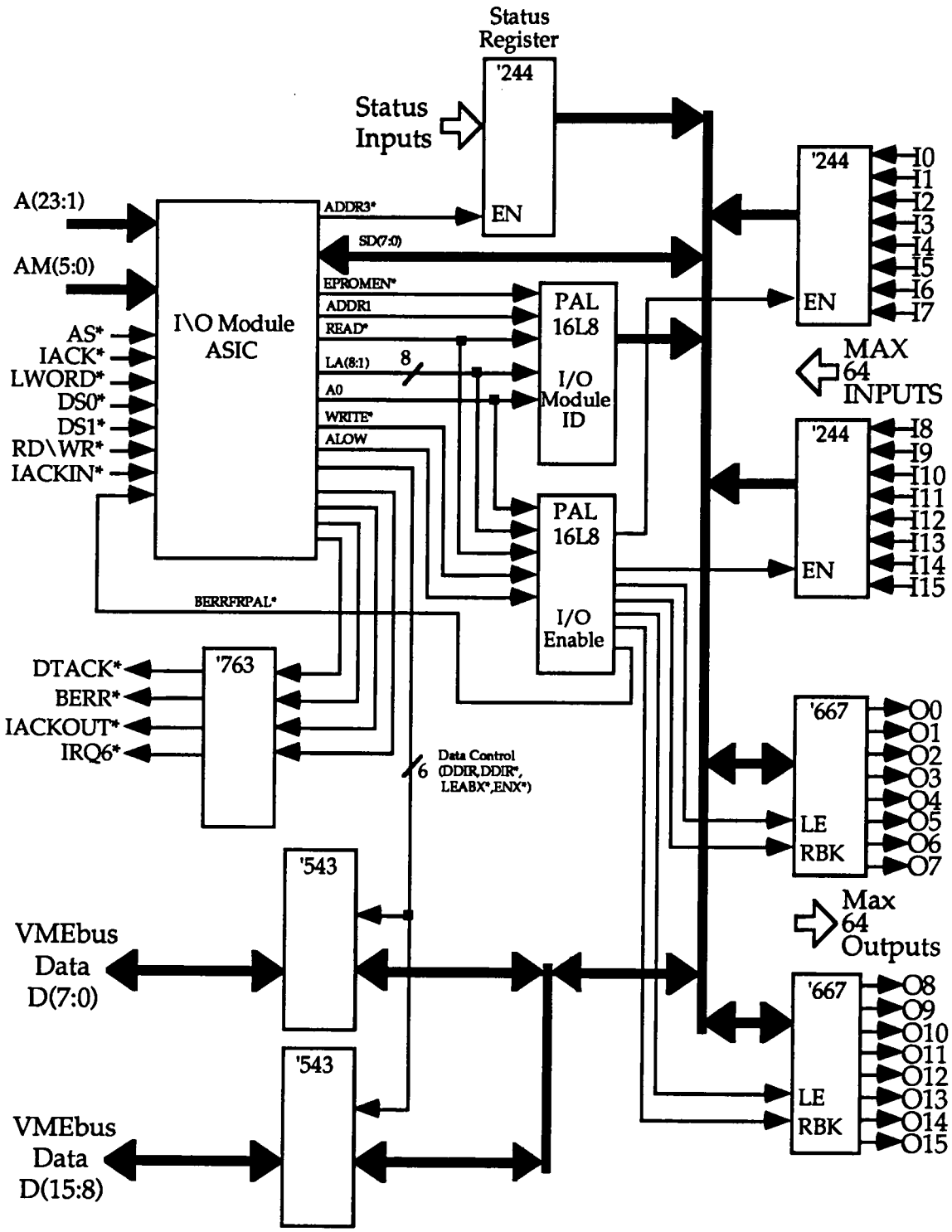

Figure 4.2. Discrete I/O Module Block Diagram

determined by the total number of control points in the I/O module configuration. Derived from ASIC control signals, the I/O point enable programmable array logic device provides the buffer enable, latch enable, and readback enable control inputs to the I/O control point data devices. The number of buffers and latches present on the I/O module determine the number of programmable array logic devices needed to supply all required control inputs. The I/O point enable programmable array logic device also provides a bus error signal back to the ASIC if the PLC attempts to access discrete I/O points which are not present in the I/O module configuration. The module identification programmable array logic drives identification information onto the I/O module data bus when requested by the ASIC.

An octal buffer, enabled by the ASIC, provides I/O module status information when requested by the PLC. The buffer drives the least significant seven bits of the data bus with status information derived from the I/O control point conditioning circuitry. The ASIC drives data bit seven directly with the watchdog timer fail indication. Additional components present on the I/O module PCB include an open-collector driver for VMEbus signals DTACK*, BERR*, and IACKOUT* along with two, octal bus latch/transceivers for interfacing the eight-bit I/O module data bus with the 16-bit VMEbus data bus.

The block diagram in Figure 4.3 illustrates the hardware configuration for an analog or special function I/O module. The octal buffers and latches of the discrete I/O module have been replaced by a microcontroller, two multiplexers, and a static ram device. The microcontroller provides control signals to D/A and A/D converters located within the I/O point conditioning

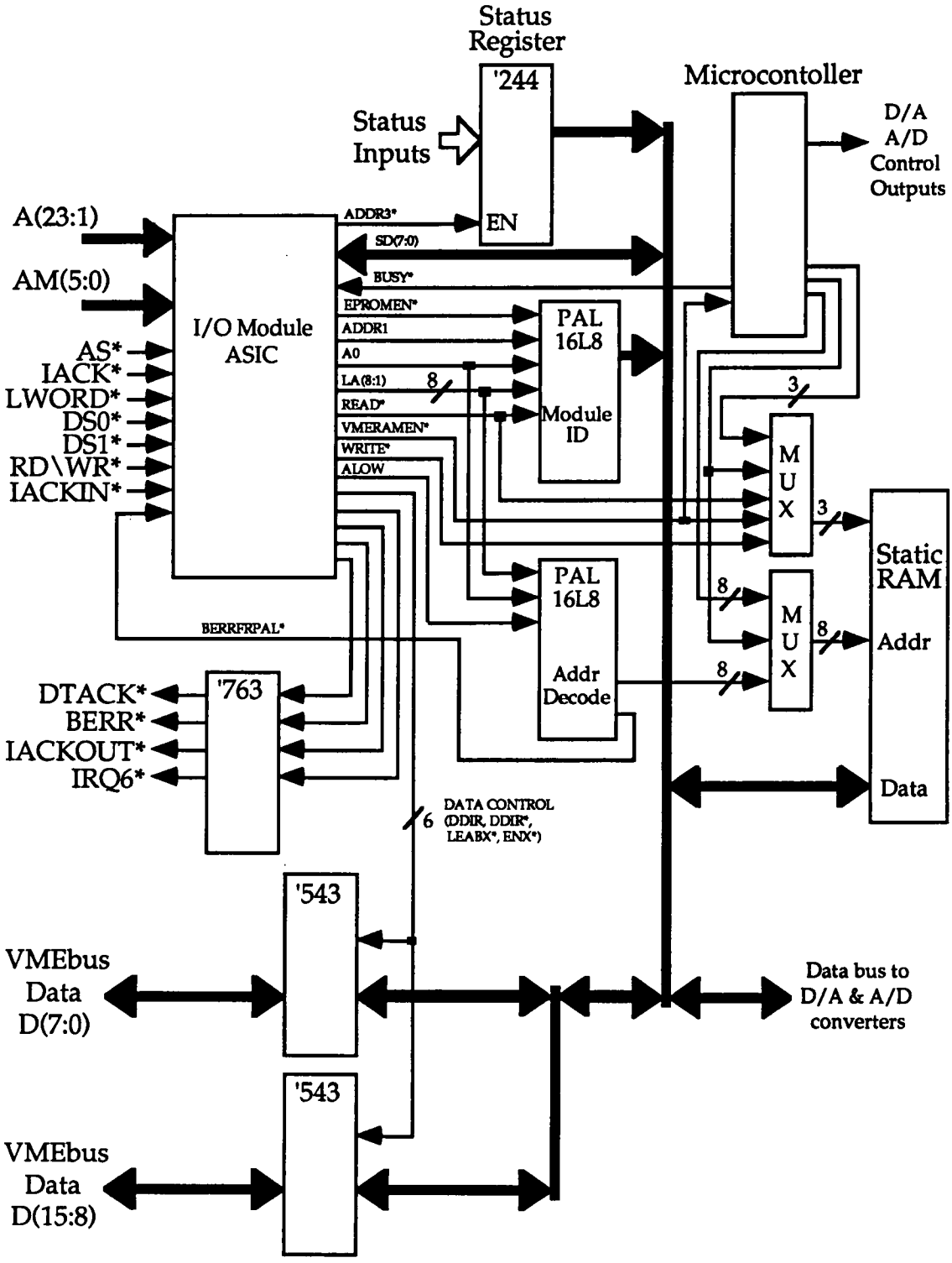

Figure 4.3. Analog I/O Module Block Diagram

circuitry in addition to performing data transfer operations between the static ram and the analog signal converters. The microcontroller also controls two multiplexers which switch static ram control/address inputs between the microcontroller and the ASIC. Since the microcontroller and ASIC share access to the static ram, bus arbitration of the I/O module data bus is performed, internal to the ASIC, using two handshaking signals. The I/O point enable programmable array logic device(s) present on the discrete I/O module are replaced by a single address decode device on the analog or special function I/O module. The address decode programmable array logic device decodes the latched VMEbus address lines output by the ASIC and generates a corresponding eight-bit address to the static ram. The address decode programmable array logic device also provides a bus error signal to the ASIC when analog I/O points not present on the I/O module are accessed by the PLC.

ASIC Pin Description

The I/O module ASIC device contains 42 input, 28 output, eight bidirectional, two power, and four logic ground pins. ASIC inputs consist of 38 VMEbus address/control signals and four signals from external I/O module logic devices including TEST, SIMIN, BUSY*, and BERRFRPAL*. TEST is a input signal driven only by test equipment during ASIC device verification. The test input is tied to a logic LOW level on the I/O module PCB disabling the test logic during normal operation. Input signal, SIMIN, is asserted by external logic to bypass the output control point disabling function present in the ASIC. Special applications defined for the VMEbus

Industrial Control System do not allow the disabling of output control points. BUSY* is a handshaking input signal, used only in analog and special function I/O module configurations, which indicates if the microcontroller has control of the I/O module data bus. If the BUSY* signal is active when a VMEbus DTB cycle to the I/O module is initiated, the ASIC delays acknowledgement of the cycle until the microcontroller releases the bus. BERRFRPAL* is generated by an external programmable array logic device and signals the ASIC if the I/O control point information, targeted by the current VMEbus cycle, is present in the I/O module configuration.

Eight pins of the ASIC device are configured as bidirectional signals providing a path between the I/O module data bus and the ASIC internal data bus. These pins allow the ASIC internal memory devices to latch data from and drive data onto the I/O module data bus during data transfer operations. Each of the twenty-eight ASIC output pin signal names and corresponding functional descriptions are listed in Table 4.1.

Design Methodology

With device specifications outlined and design alternatives considered, the ASIC design is implemented into gate array macrocell logic. The ASIC design provides control signals to programmable array logic devices, the VMEbus data bus transceivers, and the VMEbus open-collector drivers during data transfer and interrupt acknowledge cycles to the I/O module. These control signals initiate a sequence of events which perform data transfer operations reliably for all types of I/O control point data devices present in the I/O module configuration. Control signal timing is designed

Table 4.1. ASIC Output Pin Description

Signal Name	Functional Description
SD(7:0)	Internal ASIC eight-bit bidirectional bus connected to the I/O module data bus
LA(8:1)	VMEbus address lines A(8:1) latched by a falling edge on signal AS* at the beginning of each VMEbus cycle
IACKOUT*	Interrupt acknowledge daisy chain output
DDIR, DDIR*	Data flow direction control signals for the VMEbus data bus transceivers
LEAB0*, LEAB1*	Enable signals for latching data from the I/O module data bus into the corresponding VMEbus data bus transceiver - 0 corresponds to data byte ZERO, 1 to data byte ONE
EN0*, EN1*	Signals enabling the VMEbus data bus transceivers to drive data in the direction established by DDIR, DDIR*
ALOW	Logical OR of VMEbus address lines A9 through A15 following the falling edge of AS*
A0	Signal indicating which data byte is active during a data byte transfer operation on the I/O module
MODRST*	Reset signal to all external I/O module components
OUTDIS	Signal used to disable I/O module output control points
WRITE*	Indicates a write operation in process on the I/O module
EPROMEN*	Signals a read of I/O module identification information in the external programmable array logic device
ADDR3*	Enable for the external status register to drive the data bus
ADDR1	Signals a read of address offset 0001 in the memory map

Table 4.1 (Continued)

Signal Name	Functional Description
READ*	Indicates a read operation in process on the I/O module
VMERAMEN*	A signal used during I/O module data bus arbitration to indicate the ASIC has control of the bus - also serves as a chip enable input to static ram
DTACK	Data transfer acknowledgement from the ASIC device
BERR	Bus error condition indicator for the current VMEbus data transfer cycle
IRQ6	Interrupt request level six to VMEbus backplane

not only to accommodate the setup and hold specifications for a discrete I/O module data latch or buffer but also emulate the read and write cycle specifications of a static ram device present on an analog or special function I/O module. Control signal generation logic, designed to accommodate the timing requirements of all possible I/O control point data device types, allows the ASIC device to operate independently of the I/O module configuration. Adherence to VMEbus timing specifications, pertaining to slave device operation during data transfer and interrupt acknowledge cycles, is another important consideration in the design of ASIC control signal generation logic.

The timeline of control output signal events, developed from VMEbus and I/O point data device specifications, describes the chronological

sequence of steps taken by the ASIC to complete a data transfer operation. Utilizing synchronous logic design practice, the ASIC produces control output signal events exhibiting fixed, predictable timing relationships with signal transitions occurring in close proximity to the VMEbus system clock edges. The heart of the ASIC control logic design is a synchronous logic state machine clocked by the VMEbus system clock. Based on inputs from the VMEbus backplane and external programmable array logic devices, the state machine generates a sequence of control signal outputs which completes the data transfer operation in accordance with both the VMEbus and I/O point data device timing specifications. Asynchronous inputs to the state machine are conditioned with synchronizing logic to minimize the effects of metastability. The advantages of this implementation are evident through increased reliability and testability of the ASIC design. Penalties including increased VMEbus cycle response times and the addition of synchronization logic are not significant factors in the I/O module ASIC design.

Theory of Operation

The I/O module ASIC design is a top-down, hierarchical design comprised of six logic blocks partitioned according to functional similarity. A block diagram is shown in Figure 4.4. Detailed schematic diagrams of the ASIC design, located in the Appendix (Figures A.1 - A.7), designate signals exhibiting active LOW logic levels by ending signal names with an uppercase letter N as opposed to an asterisk. Schematic sheet TOPLEVEL illustrates the interconnection between the six functional blocks in addition to the device pads for ASIC I/O pins. Output pin buffer drive strengths are selected based

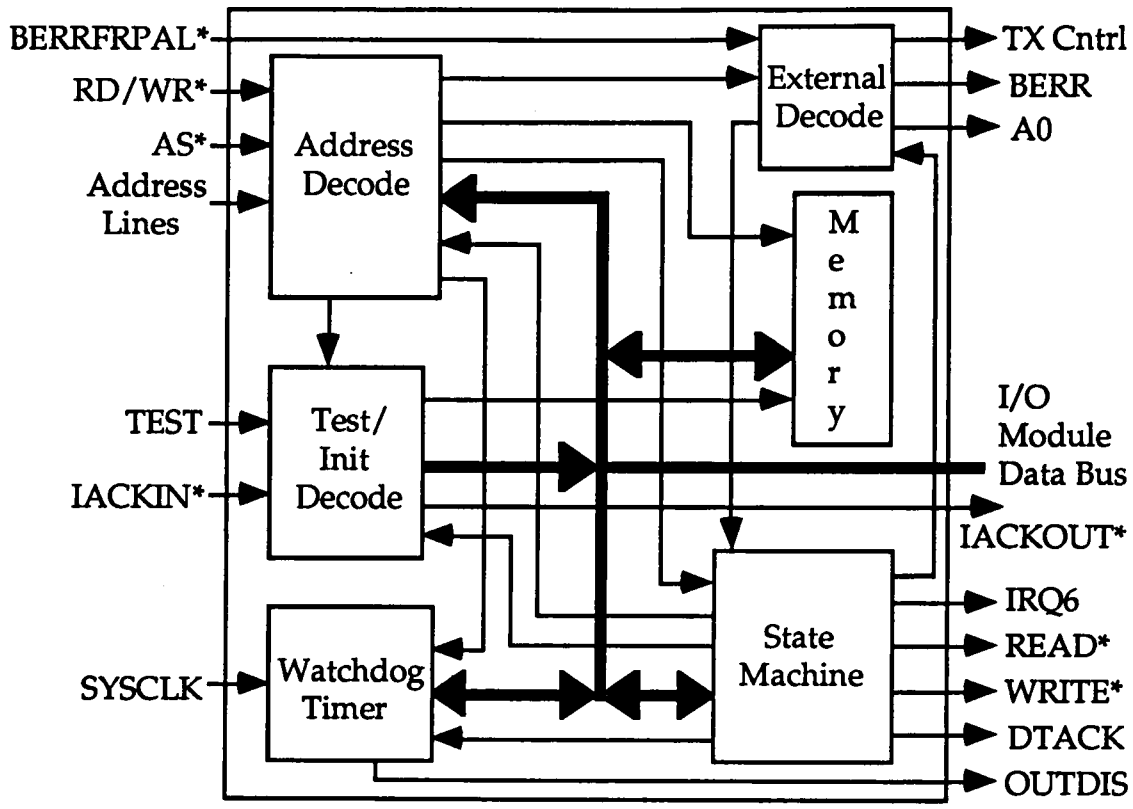

Figure 4.4. ASIC Block Diagram

on signal loading and ASIC device package capacity guidelines. Additional schematic sheets, illustrating the logic contained within each functional block, are discussed in detail in the paragraphs that follow.

The block, WATCHDOG, contains the logic required to implement the watchdog timer function in the I/O module ASIC design. The watchdog timer circuit is a chain of six, four-bit counters interconnected as a continuous, 24-bit counter of system clock edges. If not held in a reset condition by signal CLRDOG, a watchdog timeout condition is generated at the next rising clock edge after achieving a counter value of 8,388,608. Based on the VMEbus system clock period, this counter value corresponds to a

timeout period of 524 milliseconds. The watchdog timeout condition signals the PLC that the I/O control point status has not been read or written within a full timeout period. The ASIC immediately disables output control points and sets the appropriate status bit when the timeout condition is present.

The reset signal to the watchdog timer counters, CLRDOG, is generated by one of three sources; WDTRIG*, MODRST*, or WDCLR*. Upon completion of a successful data transfer cycle to the I/O module, the ASIC state machine generates a low pulse on signal WDTRIG* to clear the watchdog timer. An I/O module write cycle to address offset 0002 either as a data word or as data byte ONE generates a low pulse on signal WDCLR* which clears both the counters and the watchdog timeout condition. An I/O module reset condition, asserted by the state machine as signal MODRST*, also generates the CLRDOG signal. Device SYNCH0 provides synchronization of the CLRDOG signal release to the system clock. The watchdog timer begins counting at the next rising system clock edge following the release of CLRDOG. The J-K flip-flop, labeled as device G2, latches and holds a watchdog timeout condition, indicated by signal WDFL, independent of counter activity until a low pulse on signal WDCLR* is detected. Tristate buffer G2A drives the watchdog timeout condition onto bit SEVEN of the I/O module data bus during a read of the status register at address offset 0003. SYNCH1 and G16 provide a mechanism to hold the watchdog timer in a reset condition until the first VMEbus cycle access to a valid memory map address offset following I/O module initialization. The remaining logic contained in the WATCHDOG block provides extended capability for loading and reading the counters during ASIC device testing.

Twenty-three address lines, six address modifier lines, LWORD*, and IACK* are VMEbus input signals latched by the I/O module ASIC at the falling edge of AS*. Logic block, ADDDEC, contains these latches coupled with additional decoding logic for determining if a valid, VMEbus cycle to the I/O module is present. G100 is loaded with the I/O module physical address during initialization and is then compared, by device G84, to latched address lines A24 through A16 at the beginning of each standard, DTB cycle. The output of G84 is used in conjunction with other address decode outputs to generate signal VMEACC* indicating that a VMEbus cycle to a valid memory map address offset is currently underway. Device SYNCH0 provides the necessary synchronization of VMEACC* to serve as an input to the ASIC state machine.

Flip-flop devices G50 through G54 provide storage of the five-bit I/O module logical address. The logical address is clocked into the flip-flops by state machine signal ADDRLAT during initialization. At the beginning of a short addressing mode cycle on the VMEbus, the logical address is compared with address lines seven through one by device G39. The result of the compare operation is used in the generation of VMEACC* for short DTB cycles. When an I/O module reset is present, G50 through G53 are reset and G54 is preset which ensures the I/O module responds only to short mode cycles at address offset 0120 during initialization. Logic gates G81, G82, G97, and G98 decode the VMEbus address modifier lines to determine the addressing mode of the current cycle. Three address modifier codes are valid for I/O modules operating in the VMEbus Industrial Control System; hexadecimal 39 and 3d for standard addressing mode cycles and hexadecimal

2d for short addressing mode cycles. If the addressing mode of the VMEbus cycle does not match one of these codes, the ASIC decode logic ignores the cycle. Signal AHIGH is active HIGH when a HIGH logic level is present on each of the address lines, A15 through A1. AHIGH is used by the decode logic to detect a VMEbus cycle to address offset FFFE indicated by asserting signal ADDRE.

Logic block ADDEC also provides signal ALOW indicating that the current VMEbus cycle is pointing to an address offset between 0000 and 01FF in the I/O module memory map. External programmable array logic devices on analog and discrete I/O modules decode ALOW in conjunction with latched address lines A8 through A1 to determine if the address offset is valid for the I/O module configuration. Special function I/O modules require additional decoding logic for address lines A9 through A15. ADDEC generates signals ADDR0, ADDR1, ADDR2*, and ADDR3* when cycles to specific address offsets in the memory map are detected. ADDR0 corresponds to address offset 0000, byte ZERO and ADDR1 corresponds to offset 0000, byte ONE. In the same manner, ADDR2* and ADDR3* correspond to the data bytes at address offsets 0002 and 0003. ADDR1 and ADDR3* are provided on ASIC device output pins to control external devices containing the status/identification information defined at these offsets. Although address offset 0002 bytes ZERO and ONE are read-only data locations, write cycles are permitted as either a word or byte ONE data transfer to this offset. In either case, logic consisting of G26A, G35, G40, G45, G55, G60, and G62 generate a reset to the watchdog timer and disable the ASIC write enable output during a write to offset 0002.

After completion of the I/O module initialization routine, a short addressing mode write cycle to address offset 0100 + 2(logical address) initiates a I/O module reset by setting the output of device G3, RSTSM*. RSTSM* causes the ASIC state machine to transition into state zero at the next system clock edge and reset the I/O module. A short addressing mode read cycle to this same address offset reads the contents of offset 0000 in the memory map. The output of device G46 is set when a short addressing mode write cycle is detected to address offset 0122. This cycle issues the output control point disable command to the I/O module. OUTDIS is an ASIC output signal which immediately disables all I/O module output control points. G26 prevents the generation of OUTDIS until initialization is complete. Device G99 is a logic block symbol, labeled as DELAY30, which represents a delay element built with logic inverters. Based on the logic inverter device propagation delay specifications from the TGC103 library, the DELAY30 logic block provides a fixed delay ranging in value from a minimum of 14 nanoseconds to a maximum of 39 nanoseconds between the input and output. G99 provides a delayed and inverted version of VMEbus control signal AS*, labeled as ASCLKD. ASCLKD accommodates address decoding at the start of I/O module operations during VMEbus address-only cycles. ASCLKD is also used during interrupt acknowledge cycles to determine if the I/O module should respond to the falling edge on input IACKIN* or pass the edge to the IACKOUT* output.

A third functional block in the ASIC design labeled as the external decode block, EXTDEC, contains logic used in the generation of control signals to external I/O module logic devices and the VMEbus. These signals

include the direction, latch enable, and drive enable control signals to the VMEbus data bus transceivers. EXTDEC also contains logic which synchronizes and decodes the VMEbus data strobe inputs, DS0* and DS1*, providing synchronized decision inputs to the ASIC state machine. I/O module data bus arbitration handshaking logic, required by analog and special function I/O modules, is also located in the EXTDEC block.

Using selected output signals from the ADDDEC logic block and signal BERRFRPAL* from external programmable array logic, the EXTDEC logic block determines if a bus error condition is present during a VMEbus cycle to the I/O module. When a bus error is detected, the output of device G26 is set by a rising edge on signal SMTRIG. SMTRIG, generated by the state machine, provides an adequate delay from the beginning of a cycle to ensure all inputs to the bus error decode logic are stable. SMTRIG also ensures that the bus error is not asserted before the minimum specified time from activation of the VMEbus data strobes. During a VMEbus cycle to the I/O module, a bus error is detected by monitoring for a valid, internal ASIC address offset access and for an active level on input BERRFRPAL*. If the cycle does not access a valid internal ASIC address offset and the BERRFRPAL* is not active, the ASIC generates a bus error signal to the VMEbus. In this way, the external programmable array logic device controls which memory map I/O control point data address offsets are valid for the I/O module configuration. Device G26A clears the bus error condition when both synchronized data strobes return to an inactive state thus terminating the cycle.

Device G31 is a multiplexer which generate signal A0. A0 is a signal used by ASIC logic and external programmable array logic as an indication of

which VMEbus data byte is currently being transferred on the I/O module data bus. The output of device G44, decoded as a logical OR of the VMEbus data strobes, is labeled as SINGDOUB and serves as a select line to G31. SINGDOUB is also an input to the ASIC state machine indicating whether the current cycle is transferring a single data byte or a complete data word. During a data byte cycle, the logic state of A0 follows DS1* which is logic LOW during a transfer to data byte ZERO and logic HIGH for a transfer to data byte ONE. The ASIC state machine performs a data byte transfer cycle without any knowledge of which data byte is actually being transferred in the cycle. If a word data transfer is present, A0 follows state machine output A0SM which begins as logic LOW during the first half of the cycle to transfer byte ZERO and transitions logic HIGH during the cycle second half transferring byte ONE. Signal A0 is also a key element in the generation of drive, latch, and direction enable signals for the VMEbus data transceivers.

The drive enable signals for the VMEbus data bus transceivers, EN0* and EN1*, are asserted from state machine output signal DATADR using devices G38 through G41. The transceivers drive data onto the VMEbus data bus from the I/O module data bus or vice versa depending on the logic state of the direction control signals. Because of the difference in bus width between the VMEbus and I/O module data buses, decode logic is present to accommodate all VMEbus cycle data configurations and prevent bus contention. During a read cycle to the I/O module, either or both drive signals are enabled depending on the VMEbus data strobe configuration. If the read cycle is a single byte transfer, the data is placed on the I/O module data bus; latched into the correct byte transceiver, and the drive signal to that

transceiver is enabled. If the read cycle requests a data word transfer, the two data bytes are latched into the transceivers, one byte at a time from the I/O module data bus, and both drive signals are enabled providing a data word to the VMEbus. Device G50 ensures that only the drive enable signal for the byte ONE transceiver is activated during the module initialization interrupt acknowledge cycle independent of the data strobe configuration present on the VMEbus.

Write cycles, transferring a data word to the I/O module, require bus contention protection for the eight-bit I/O module data bus. Data bus contention is possible if both VMEbus transceiver drive enable signals are active simultaneously during a VMEbus data word write cycle. The state machine, controlling output A0, divides the data word write cycle into two parts, writing data byte ZERO during the first half of the cycle and data byte ONE during the second half. Signal A0 determines which VMEbus data byte is driven onto the I/O module data bus during the write cycle. As the state machine transitions the logic level of A0 to switch the transceiver drive enable signals from byte ZERO to byte ONE, the possibility exists for both transceivers to have active drive enables during the signal transitions. Devices G46 and G56 address this problem by preventing a transceiver drive enable signal from transitioning to an active state until the opposite drive enable signal is inactive. If the write cycle requires only a single data byte, signal A0 is decoded from the VMEbus data strobes and the appropriate data byte is driven onto the I/O module data bus.

The transceiver drive enable signals in conjunction with the VMEbus control signal, RDWR, are decoded to establish the transceiver direction

signals, DDIR and DDIR*. Although RDWR can change logic state before data strobes are released at the end of a DTB cycle, the logic circuit comprised of devices G18, G20, G22, and G23 establishes the correct transceiver direction at the beginning of the cycle and latches the logic state of the direction signals when one or both drive enable signals are active. This circuit prevents the transceiver direction signals from transitioning until both transceiver drive enable signals are inactive at the end of the cycle. ASIC logic prevents the data transceivers from changing direction and possibly corrupting the data before the cycle is complete.

Operating in an analog or special function I/O module configuration, devices G4, G7, G12, and G17 perform arbitration of the I/O module data bus between the ASIC device and a microcontroller. Device G4 generates signal VMERAMEN* indicating which device has control of the data bus. Following detection of a DTB cycle to the I/O module, the state machine drives signal BUSREQ* active as a request to gain control of the I/O module data bus. The microcontroller, when in control of the I/O module data bus, asserts signal BUSY* thus maintaining a HIGH logic level on output VMERAMEN*. After asserting BUSREQ*, the state machine monitors VMERAMEN* for a LOW logic level before proceeding with the DTB cycle. As long as a HIGH logic level remains on VMERAMEN*, the DTB cycle is not completed by the state machine since access to the I/O module data bus is not granted. The PLC inserts wait states into the DTB cycle until the cycle is acknowledged by the I/O module. If the microcontroller does not relinquish control of the bus for an extended time period, a bus timeout can occur resulting in a bus error to the PLC. To prevent this condition, the

microcontroller is required to periodically relinquish control of the data bus so that a pending VMEbus DTB cycle can be completed.

When the microcontroller releases the data bus by driving signal BUSY* logic HIGH and the state machine is asserting BUSREQ*, signal VMERAMEN* is driven logic LOW at the next falling clock edge of the system clock. The state machine detects the assertion of VMERAMEN*; takes control of the I/O module data bus; and completes the DTB cycle. When the cycle is complete, the state machine deasserts BUSREQ* and the bus arbitration circuit releases VMERAMEN* to logic HIGH. At this point, the microcontroller can regain control of the I/O module data bus. In discrete I/O module configurations, signal BUSY* is hardwired logic HIGH on the PCB since a microcontroller is not present. In this way, the state machine is always granted access to the data bus upon request.

A final output signal from the EXTDEC block, EPROMEN*, is generated by device G1 and provides an enable to programmable array logic containing the I/O module identification information. EPROMEN* is asserted when a VMEbus cycle is present to one of the address offsets containing the identification information. Enabled by signal EPROMEN*, external programmable array logic decodes several low order address lines to determine what identification information is required by the cycle.

Functional block, MEMORY, contains the required memory devices to provide storage of internal ASIC data including the interrupt acknowledge routine vector, physical address, and logical address. In addition, the decode logic controlling these memory devices is present in the MEMORY block. A state machine output signal, VECEN*, enables device G22 to drive the data

bus with the interrupt acknowledge vector during initialization. The interrupt acknowledge vector data pattern is specified in the I/O Module Specification as hexadecimal 41. Also during initialization, the logical address, assigned to the I/O module, is written into memory devices G2 and G26 along with the assigned physical address into device G7. Devices G8 and G12 provide a mechanism to indicate when both of these write operations have been completed. Signal WRBD flags the state machine that both the logical and physical addresses have been written to the I/O module. When WRBD is detected, the state machine completes the initialization routine and establishes the ASIC for normal I/O module operation. The output of G12 also serves as a clock signal to a memory device, located in the ADDDEC block, which holds a copy of the physical address. Following I/O module initialization, devices G22 and G26 are not accessible and devices G2 and G7 become read-only memory to VMEbus cycles. As a final note about the MEMORY block, device G3 generates an output signal which provides an enable signal to the eight, ASIC bidirectional data bus drivers. When an internal ASIC memory device is read by the VMEbus, the G3 output drives the requested internal data onto the I/O module data bus.

Logic required for I/O module initialization and for generating test mode signals during ASIC device testing is contained in the TSTINIDEC block. The output of device G5 is set from a rising edge of signal ASCLKD during an interrupt request level six acknowledge cycle when the ASIC is requesting interrupt service. If the G5 output is set when a falling edge appears on input, IACKIN*, a LOW level is produced at the output of device G3 which, in conjunction with at least one active data strobe, sets the output

of device G8 at the next rising edge of the system clock. Device SYNCH0 provides synchronization of the data input to G8 with the system clock. The output of G8 signals the state machine that an interrupt acknowledge cycle, requiring a response, is underway to the I/O module. After completing the interrupt acknowledge cycle and subsequent I/O module initialization routine, the state machine resets signal RESPOND by producing a pulse on signal ADDRLAT. If the G5 output is not set when a falling edge on IACKIN* is detected, a logic LOW is driven onto the output of device G2, IACKOUT*. This passes the interrupt acknowledge cycle to the next device located on the VMEbus interrupt acknowledge daisy chain.

The final block of the ASIC design contains the logic implementation of the state machine. The state machine, designed in accordance with the Moore model for sequential circuits, generates synchronized output signals that are a function of the state machine present state. State transitions occur at the system clock rising edge and are based not only on present state information but also on state machine input signals. Excluding test logic signals, the state machine contains 13 inputs and 14 outputs as illustrated in the symbol diagram located in the Appendix (Figure A.7). The state machine is comprised of 37 states and provides the control signals which sequence the I/O module through the initialization routine and VMEbus cycles. State assignments, conditional transitions from inputs, and active output signals in each state are illustrated in Figure 4.5. Output signals active during a state are labeled inside the state block. Six state variables, Q5 through Q0, provide a binary representation of the state value with Q0 as the most-significant digit. A gray code design approach to state variable assignment minimizes

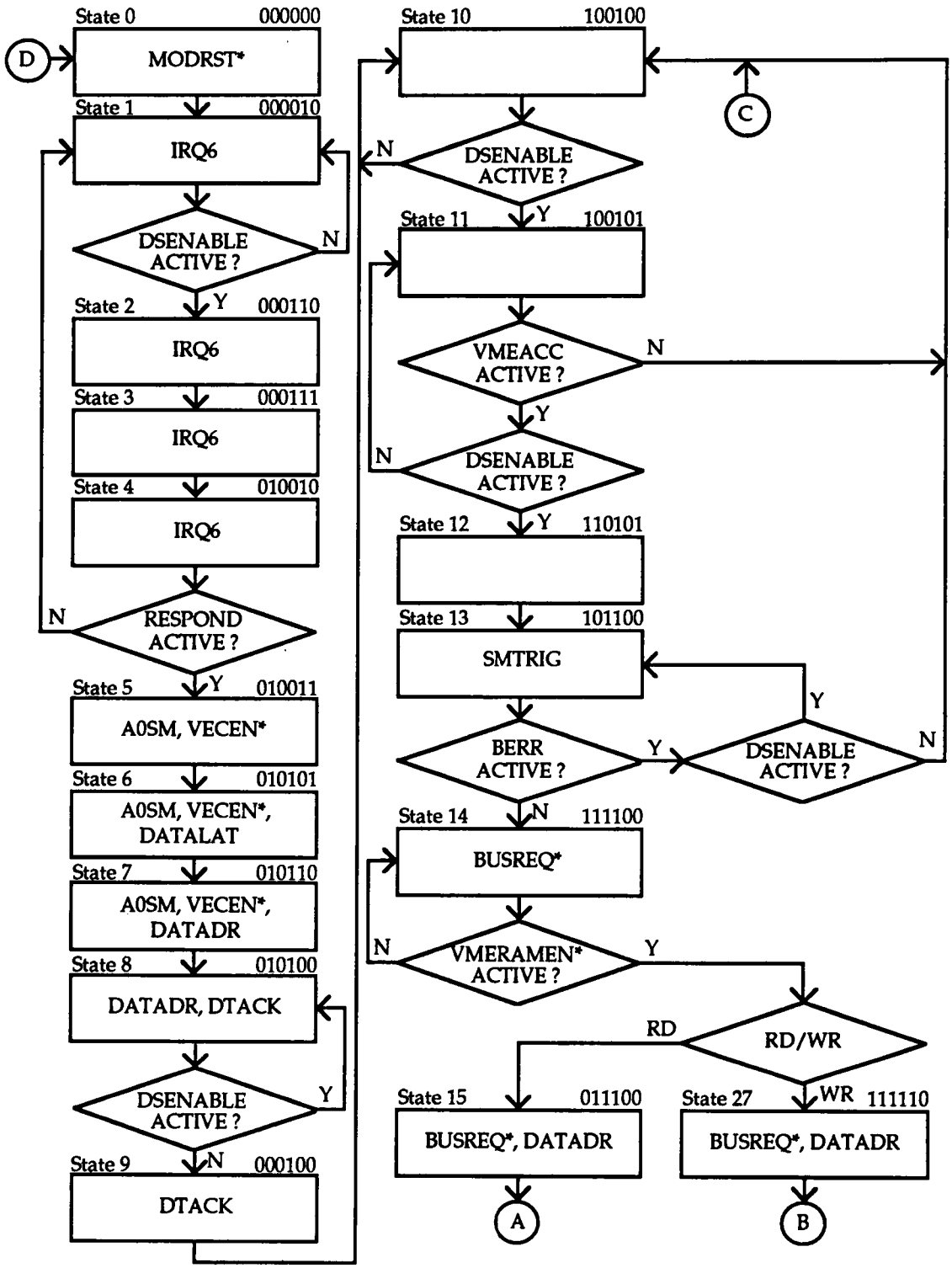

Figure 4.5. State Diagram for ASIC State Machine

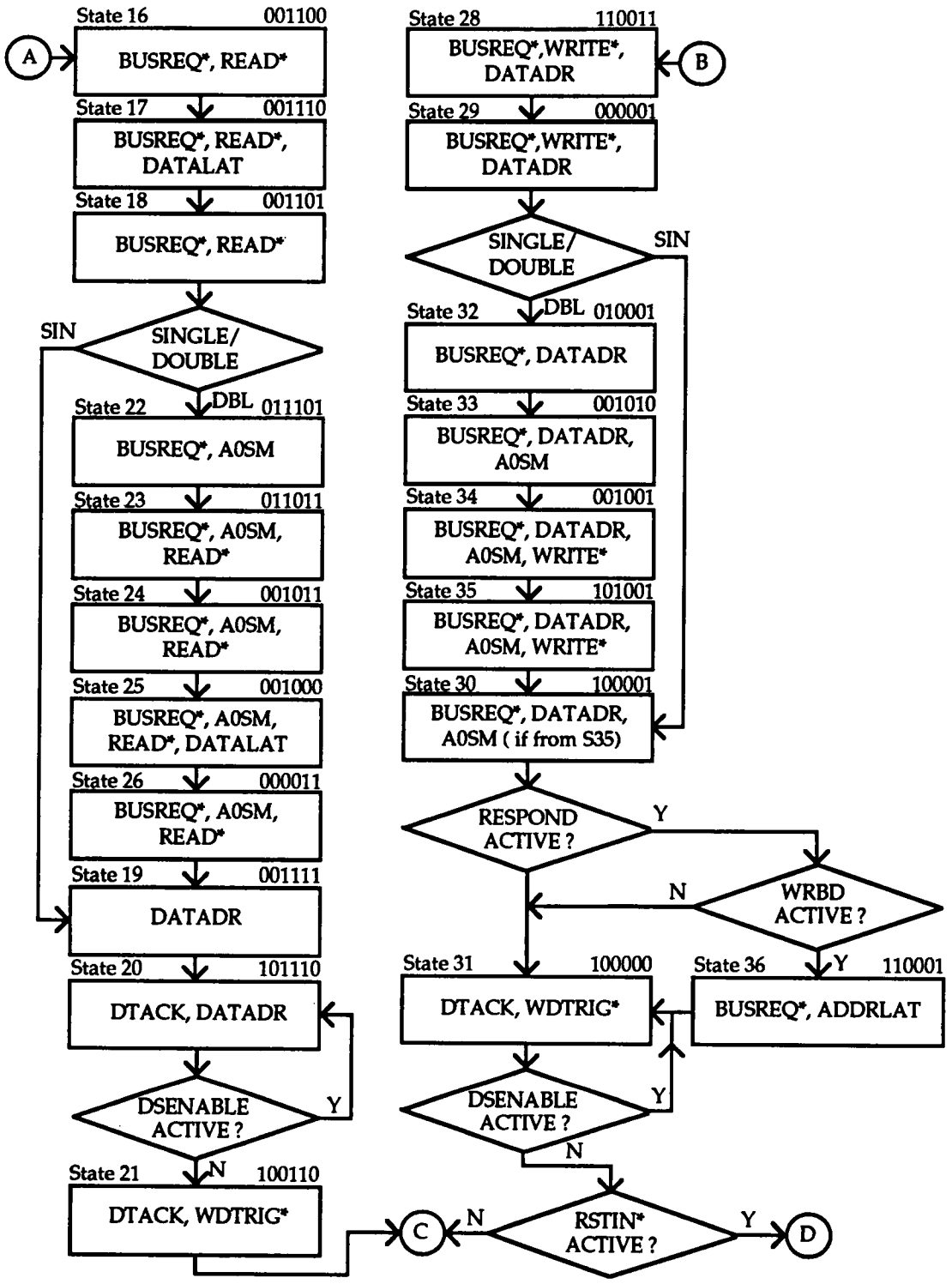

Figure 4.5 (Continued)

the possibility of a jump to an illegal state since only one state variable is changing logic level during a state transition. For additional protection, detection logic is present to reset the state machine if a transition into an illegal state occurs during normal operation. During a VMEbus reset, all memory devices internal to the state machine are preset which initializes the state machine to state zero. In state zero, the state machine generates a reset signal to the remaining ASIC logic. For metastability protection, synchronizing logic is present for release of the system reset input to the state machine. The defined I/O module reset command, generated by a short DTB write cycle to address offset 0100 + 2(logical address), transitions the state machine to state zero at the completion of the cycle.

As illustrated in the state diagram, the state machine begins the I/O module initialization routine with a transition into state one. In state one, the state machine asserts the interrupt request, IRQ6, to the VMEbus and waits for the indication of an active data strobe. When a data strobe transitions active, the state machine sequences through states two, three, and four providing a time delay for the decode logic to determine if an interrupt acknowledge cycle to the I/O module is in progress. Input signal RESPOND, when active, prompts the state machine transition through states five to nine. This sequence of states provides the necessary control signals to drive the interrupt acknowledge routine vector onto the VMEbus. When the VMEbus terminates the cycle, the state machine transitions to state 10 and monitors the active data strobe input, DSENABLE. DSENABLE signals that the data transfer portion of a VMEbus cycle is underway. If the RESPOND

input is not active during the interrupt acknowledge cycle, the state machine continues to assert the interrupt request and returns to state one.

As the data transfer portion of a VMEbus cycle begins, the state machine transitions to state 11 and provides an appropriate delay for the decode logic to determine if the cycle is addressing the I/O module. The state machine monitors input signal VMEACC as an indication from the decode logic. If the cycle is not addressing the I/O module, the state machine returns to state 10. Otherwise, the cycle is considered valid and the state machine proceeds to state 12. In state 13 the presence of a bus error condition is tested. If a cycle error is detected, a bus error signal is asserted and the state machine returns to state 10 upon termination of the cycle by the VMEbus. If no bus error exists, the state machine proceeds to state 14 where arbitration for control of the I/O module data bus is performed. The state machine remains in state 14 until control of the data bus is granted to the ASIC. By decoding the VMEbus control signal RD/WR*, the state machine determines if the cycle is a read or write to I/O module memory. Based on this decision, the state machine transitions through a sequence of states providing the necessary control signals to perform the specified cycle between the VMEbus and the I/O module. These state sequences are designed to generate control signals which perform read and write cycles independent of I/O control point type. The state machine transfers data reliably between the VMEbus and either static random access memory devices, with access times no greater than 120 nanoseconds, or Advanced Low-Power Schottky TTL data buffers and latches. In other words, the read and write state sequences are designed to provide control signals, decoded by external programmable array logic,

which perform read and write cycles to all possible I/O control point data memory device types in the VMEbus Industrial Control System.

In determining whether a cycle is a byte or word data transfer, the state machine tests the logic state of input, SINGDOUB. A read cycle, requiring the transfer of a data word, consists of states 15 through 26 while a read cycle of a single data byte consists of only states 15 through 21. Similarly, a write cycle of a data word involves the sequence of states 27 through 35 while a write cycle of a single data byte only requires states 27 through 31. For the write cycle state sequence, a special note concerning output A0SM in state 30 warrants discussion. If the write cycle is transferring a single byte, the state machine transitions from state 29 to state 30 and output A0SM is not active during state 30. If the write cycle involves a data word, the state machine transitions from state 35 to state 30 and A0SM remains active in state 30. In contrast to the other state machine outputs, A0SM is dependent not only on the present state but also previous state information. A final note concerning the state machine diagram involves the presence of state 36. During initialization, the state machine enters state 36 at the end of the write cycle that completes the assignment of both physical and logical addresses to the I/O module. A state machine transition through state 36 completes I/O module initialization by generating a pulse on output ADDRLAT and establishes the ASIC logic for normal I/O module operation.

ASIC Testability Logic

Testability is a measure of the effective test coverage provided to the ASIC internal logic through input stimulus of device input pins and

detection of logic signal levels present on device output pins. ASIC designs exhibiting a high level of testability contain high degrees of controllability and observability. Testability in ASIC designs is a key component for providing maximum fault coverage without requiring an excessive number of test vectors.[7] Controllability is a measure of the ability to establish desired signal levels on internal logic interconnections through stimulus of the device input pins. In a similar fashion, observability describes the ability to detect signal levels present on internal logic interconnections by observing device output pins. It is extremely important to consider device testability early in the ASIC design cycle so that any additional test logic required to enhance controllability and observability is incorporated into the design.

Testability logic, present throughout the I/O module ASIC design, is controlled by a dedicated input signal, TEST. The ASIC enters a special test mode, used only during device verification, when input TEST is driven logic HIGH. In test mode, selected device input pins are used to control internal memory devices and provide observability of internal logic interconnections. Utilizing the ASIC data bus, memory devices are read and/or written and the logic states of internal signal interconnections are probed during test mode operations. Input TEST is tied logic LOW on the I/O module PCB to provide normal ASIC device operation.

The most significant amount of test logic present in the I/O module ASIC design is located in the watchdog timer circuit. To enhance controllability, the timer circuit is comprised of parallel load, binary counters with inputs connected directly to the internal ASIC data bus. In test mode, the counter outputs can be driven onto the ASIC data bus using three, tristate

buffers, G3, G4, and G5. Verification of timer operation is accomplished by loading the counters with a binary value; suppling a predetermined number of clock signals; and reading the counter outputs. In this manner, counter rollover and watchdog timeout generation are tested using a minimal number of test vectors that do not require long sequences of clock cycles nor involve a large amount of test time.

To verify the functionality of internal ASIC memory devices for reset and latching operations, additional test logic is included in the TSTINIDEC block. G27 provides both latch enable and drive enable control signals, decoded from the address modifier inputs, to memory devices in the MEMORY block. Data patterns are read/written to/from these memory devices during testing without the need for emulating VMEbus cycles. Test logic also drives the ASIC data bus, when enabled, with the logic state of selected internal logic signal interconnections. The state machine variables and illegal state flag are two examples of internal signals observed through the ASIC data bus in test mode.

A final testability logic addition to the ASIC design is contained in the state machine. Using the ASIC data bus, the state machine variables are written to a desired binary value while the clock input is disabled. In this manner, the state machine can be established in any desired state by writing to a latch in test mode. A clock signal(s) is then supplied to the ASIC allowing verification of state machine operation. This technique simplifies state machine testing and improves fault coverage by exercising circuitry, such as illegal state detection logic, not generally used during normal operation.

Design Analysis

Analysis of the I/O module ASIC design provided important feedback during the development phase of the project including a measure of pre-layout device performance under minimum/maximum timing delay conditions; a guideline for device power pin requirements; and calculation of device power consumption. Design performance is measured prior to device layout by conducting a timing analysis. The timing analysis involves detailed propagation delay calculations for selected internal ASIC signal paths. Using calculation results, race conditions are identified for the internal logic and timing diagrams are constructed which illustrate ASIC output signal transition events in relationship to external PCB signal events. In general, the timing diagrams are constructed to emulate I/O module response during VMEbus cycles. From the diagrams, time intervals between signal transition events are verified, for both minimum and maximum delay scenarios, to ensure that ASIC control output signal timing provides reliable operation with external PCB logic components. By evaluating ASIC design performance at the propagation delay extremes, device operation is verified to be independent of fabrication process, temperature, and load variations. At this point, timing analysis results are best estimates providing a design performance evaluation prior to device layout by the manufacturer. Minimum and maximum delays, characteristic of the actual ASIC device, are described by layout parasitics available from the manufacturer following layout and routing. These device parasitics are used during post-layout simulations for timing performance verification prior to design production release.

In completing the timing analysis for the ASIC design, the maximum and/or minimum propagation delay for an internal signal path, from input pin to output pin, is calculated for both the high-to-low transition, t_{phl}, and the low-to-high transition, t_{plh}. For an ASIC output signal pin, the propagation delay, referenced from a predetermined input signal pin transition, is given by equation 4.1:

$$t_{pd}(\text{output}) = t_{pd}(\text{input buffer}) + t_{pd}(\text{logic path}) + t_{sw}(\text{out buffer}) \qquad (4.1)$$

Equation 4.1 states that the propagation delay of an ASIC output signal is equal to the sum of the delay of an associated input signal through the input pad buffer; the accumulated delay of an internal logic path used in the logic equation for the output signal; and the switching delay associated with the output buffer used to drive the signal onto the device output pin. Since the logic function which determines the output signal state often contains multiple internal paths, the logic path resulting in the smallest delay when using minimum propagation delays in the calculations provides the minimum delay for the output signal. The logic path producing the largest delay when using maximum propagation delays provides the maximum delay for the output signal.

The logic path delay, given in equation 4.1, is the accumulative propagation delay for each ASIC logic device located in the signal path as indicated by equation 4.2:

$$t_{pd}(\text{logic path}) = \text{SUM}[(t_{pd}(\text{device})) + (\text{delta} \mid t_{pd}(\text{device}))(C_{out})] \qquad (4.2)$$

For each logic device, the propagation delay is comprised of two components. The first component represents the logic device propagation delay assuming no load on the device output while the second component adjusts the delay

based on the estimated output capacitive load. From equation 4.2, the second component is the multiplication of the delta delay per unit capacitive load parameter for the device and the estimated device output load, C_{out}. C_{out} is calculated as the sum of the input capacitances, C_{in}, of all logic devices present as loads to the output signal in question. The parameters specified in equations 4.1 and 4.2 including t_{pd}(input buffer), t_{pd}(device), delta | t_{pd}(device), and C_{in} are tabulated in the CMOS Gate Array Design Manual for all TGC100 library logic function macros. These tabulated values are specified by the manufacturer to include the effects of variations in wafer fabrication in addition to propagation delay variations over the specified supply voltage and temperature range. The tabulated values also provide a guideline for adjusting logic macro propagation delay based on the capacitive load presented to the macro output.

The third and final component in equation 4.1 is the output buffer switching delay for the output signal is question. The output buffer switching delay is made up of two components as illustrated by equation 4.3:

$$t_{sw}(\text{out buffer}) = (t_{pd}(\text{out buffer})) + (\text{delta} \mid t_{pd}(\text{out buffer}))(C_{PCB}) \qquad (4.3)$$

The maximum propagation delay and the delta delay per unit capacitive load are tabulated for the selected output buffer. The PCB capacitance, C_{PCB}, is estimated by summing the signal path etch capacitance with the input capacitances of all external devices connected as loads to the output signal pin. Etch capacitances are derived from physical PCB characteristics while input capacitances are obtained from external logic device data sheets.

A second area of analysis in the ASIC development process involves a calculation to determine the minimum voltage supply and ground pin

configuration required by the device. The power pin count is influenced by several factors including output load current requirements, operating frequencies of I/O pin buffers, and the number of simultaneously switching device output signals.[7] For an 84 pin, PLCC device in the TGC103 family, the CMOS Gate Array Design Manual recommends a minimum primary power pin configuration of two supply voltage pins and four ground pins. The power pin calculation, which requires grouping of simultaneous switching output signals, is performed to determine if secondary power pins are required to minimize the effects of switching transients.

A group of output signals is considered as a simultaneous switching output group if the signals perform a logic state transition within a five nanosecond window of each other. In the I/O module ASIC design, the largest potential simultaneous switching output signal group occurs during a VMEbus read cycle as the state machine transitions from state 21 to state 22. If the VMEbus is reading a data word of internal ASIC memory, output signals READ*, SD(7:0), EN0*, EN1*, and A0 can potentially transition logic state at approximately the same point in the cycle. To determine the required number of supply voltage or ground pins, the output signal buffers in this group are separated according to output buffer current rating and applied to equation 4.4:

$$\#\text{ required device pins} = \text{SUM}[(\#\text{ of buffers})(\text{Constant})] \qquad (4.4)$$

Equation 4.4 multiplies the number of output buffers in the group with similar current ratings by a constant and sums these results for all buffer current ratings present in the group. The constants used in equation 4.4 are tabulated according to buffer current rating in the CMOS Gate Array Design

Manual. Constants are available for both the voltage supply and ground pin calculations. Applying equation 4.4 to the I/O module ASIC design yields the following results:

$$\text{\# of supply pins} = (9)(0.083) + (3)(0.044) = 0.879$$
$$\text{\# of ground pins} = (9)(0.166) + (3)(0.087) = 1.755$$

Rounding these values to the next highest integer indicates that no secondary power pins are required by the I/O module ASIC design.

The final area of analysis applied to the I/O module AISC design produces an estimation of the device power dissipation, P_{diss}. P_{diss} consists of four components, specified in milliwatts, as illustrated by equation 4.5:

$$P_{diss} = P_{dc} + P_{core} + P_{input} + P_{output} \quad (4.5)$$

P_{dc} is the current draw created by leakage current through reverse-biased P-N junctions. P_{dc} is calculated by equation 4.6 using the maximum value, including tolerance, allowed for the device supply voltage, V_{cc}, and the estimated leakage current specified for a TGC103 device, I_{cc}:

$$P_{dc} = V_{cc}\, I_{cc} = (5.5 \text{ volts})(0.225 \text{ mA}) = 1.24 \text{ mW} \quad (4.6)$$

P_{core} represents the power dissipation within the ASIC logic core and is estimated using equation 4.7:

$$P_{core} = (0.018)(N_{cell})(F_{core}) = (0.018)(2878)(5) = 259.0 \text{ mW} \quad (4.7)$$

N_{cell} represents the equivalent basic cell count of all logic devices in the ASIC design and F_{core} is the average operating frequency of the core logic in megahertz. For the I/O module ASIC design, F_{core} is estimated as five megahertz.

The average power dissipation in the input buffers, P_{input}, is estimated by equation 4.8:

$$P_{input} = (0.06)(N_{input})(F_{input}) = (0.06)(50)(5) = 15.0 \text{ mW} \quad (4.8)$$

N_{input} represents the total number of ASIC device inputs and F_{input} is the average transition frequency of the input signals. Based on the signal transition events outlined for VMEbus cycles, F_{input} is estimated as five megahertz for the I/O module ASIC design. The average power dissipation in the output buffers, P_{output} , is calculated using equation 4.9:

$$P_{output} = (0.03)(N_{out})(F_{out})(10 + C_{load}) \qquad (4.9)$$
$$= (0.03)(36)(5)(10 + 35) = 243.0 \text{ mW}$$

N_{out} corresponds to the total number of device output buffers; F_{out} represents the average operating frequency; and C_{load} is the average capacitive load presented to a device output pin buffer. For the I/O module ASIC design, F_{out} is estimated as five megahertz and C_{load} as 35 picofarads.

Substituting the results obtained from equations 4.6, 4.7, 4.8, and 4.9 into equation 4.5 yields the power dissipation estimate for the I/O module ASIC design:

$$P_{diss} = 1.24 + 259.0 + 15.0 + 243.0 = 518.24 \text{ mW}$$

The design power dissipation estimate is within the allowable maximum dissipation of 714 mW for an 84 pin, PLCC device package in the TGC103 family. This power dissipation estimate for the I/O module ASIC design provides a benchmark value used in I/O module PCB power consumption calculations.

CHAPTER 5

ASIC SIMULATION AND TEST VECTOR DEVELOPMENT

An extremely important and time consuming part of an ASIC development project includes functional simulation and test program generation for the design. Functional simulation provides a tool for verifying operation of the logic design in accordance with all specified performance criteria. Test program generation converts the device functional simulation into a format, used by the manufacturer's test equipment, to verify operation of the ASIC device following fabrication. Both tasks are completed using the Texas Instruments ASIC design environment running on a Mentor Graphics® workstation. Simulation and test program generation are essential tasks in the development of a reliable and manufacturable ASIC device. A description of these tasks as applied to the I/O module ASIC development project are described in the sections that follow.

Functional Simulation

Functional simulation of the an ASIC design begins with the development of input stimulus for the Mentor Graphics logic simulator, Quicksim®, followed by verification of the resulting simulation output. Functional simulations not only verify that all logic functions in the ASIC design are operating properly but also provide a method of measuring design performance. Each logic macro in the TGC100 library contains assigned parameters for simulating minimum, unit, and maximum propagation

delay scenarios. These parameters are utilized by Quicksim to perform timing simulations of the design using estimated signal interconnection delays. From the simulation results, critical timing paths are evaluated to ensure reliable operation during both minimum and maximum propagation delay scenarios. Timing simulations are also conducted with device interconnect layout parasitics, supplied by the manufacturer, which allow the designer to simulate the design with actual device signal interconnect delays. Results from these simulations give the designer an emulation of actual device performance for evaluation.

Input stimulus to the Quicksim simulator is developed as a sequence of vectors, each vector containing a series of logic bits. Each bit in the vector represents the logic state driven onto a corresponding ASIC device input pin during the time period that the vector is valid. All device input signals are represented in each test vector regardless of whether the logic state of the input signal is changing or not. Vectors are labeled with a time stamp and processed by the simulator in chronological order. Time interval resolution between vectors can be as little as a tenth of a nanosecond. Clock inputs to the design are either defined as free-running at a specified rate or manually toggled in successive vectors to produce the desired clock period.

Input vectors, comprising the functional simulation for the I/O module ASIC design, were developed to emulate VMEbus cycles as defined by The VMEbus Specification. Signal transitions on ASIC input signals are specified in a chronological sequence designed to emulate the signal events present during a VMEbus cycle. Time intervals between signal state transitions correspond to either the minimum or maximum delays specified

on the VMEbus. Transitions on ASIC input signals, originating from logic devices located on the I/O module PCB, are inserted into the input vector sequence based on expected signal events and estimated propagation delay calculations. Blocks of simulation vectors are developed to emulate not only defined VMEbus cycles but also power-up reset and I/O module initialization routines. Input, SYSCLK, is established as an independent, 16 megahertz clock defined to run independently for the duration of the functional simulation. It is possible to stop the clock, provide single transitions on the clock input pin, and restart the free-running clock at any point in the simulation. As a final note, input vector blocks were also developed to simulate error conditions in I/O module ASIC design. These vectors provide a method of design verification in the presence of error conditions.

During simulation of the input vector sequence, the simulator produces output vectors which, similar to input vectors, contain a series of logic bits representing the logic state of each ASIC output pin signal. In addition, any logic interconnection internal to the ASIC design can be assigned a label and its logic state included in the simulator output vector. This technique provides a powerful tool for problem isolation in the early phases of design verification. During simulation, the simulator generates an output vector when any signal(s), assigned to the output list, exhibits a logic state transition. The current logic state of all assigned signals are included in the output vector, not just the signal(s) which transitioned. The simulator also assigns a time stamp to the output vector establishing a correspondence between the output signal transition(s) and related input signal transition events. Output vectors are listed in a chronological sequence as the

simulation progresses and accompanied by applicable error and warning messages. These error messages generally serve as flags to the designer for potential timing violations involving setup and hold parameters on memory devices.

The simulator processes the input vector sequence and produces an output vector sequence based on the design netlist which is compiled form the schematic diagram. To generate the simulation output, the simulator is instructed to use a selected propagation delay scenario for all logic devices in the design coupled with either estimated or actual interconnect delays. During the initial phases of functional verification, simulations are often selected for unit propagation delay assigning all logic devices in the design a propagation delay of one nanosecond. Unit delay simulations provide an approach for evaluating logic design functionality without potential timing complications. In the I/O module ASIC design, unit delay simulations were conducted during the initial development of each functional block to verify individual block operation before incorporation into the ASIC design hierarchy.

Upon successful completion of unit delay simulations, separate simulation runs are conducted for both minimum and maximum delay selections using an input vector sequence designed to emulate device operation at the system clock speed. Before design submission to the manufacturer, the effects of minimum and maximum delays on device performance are evaluated from the simulation results. All simulation results are verified for conformance with ASIC signal timing specifications and all warning messages resolved before the design is approved for device

layout. Collectively these simulations are commonly referred to as pre-layout simulations since interconnect delays are only estimations calculated by the simulator. Following layout of the device by the manufacturer, actual signal interconnect parasitics are available to the simulator for conducting minimum and maximum delay simulations. At this point, these simulations are described as post-layout and provide the most accurate emulation of actual device performance. Verification of the simulation results from post-layout simulations is the design engineer's final approval mechanism to ensure the ASIC design is in accordance with all system functional and timing specifications before production release. Both pre-layout and post-layout simulations were completed in the manner just described for the I/O module ASIC design in accordance with all applicable manufacturer's requirements.

Test Vector Development

Following fabrication of the ASIC device, the manufacturer verifies device operation using a test program developed specifically for the ASIC design and in accordance with the manufacturer's test equipment specifications. The test program is comprised of a sequence of test vectors each containing a series of logic bits representing the logic state of each ASIC device input, output, and bidirectional pin during a fixed test period. The test program performs a functional test of the ASIC device by driving the input pins to the assigned logic states indicated in the test vector, producing a clock pulse on the clock pin, and strobing the output pins for expected logic states also contained within the test vector. Bidirectional pins are either

driven as inputs or strobed as outputs during the test period based on a control group direction signal provided in the test vector. As opposed to functional simulations modeling device performance at system speed, test vectors are developed in a strict signal event timing format that adheres to guidelines derived from test equipment limitations.

The I/O module ASIC device is tested with a Trillium™ tester programmed using the Texas Instruments Test Description Language. The test program is developed, in the workstation design environment, by restructuring the device functional simulations into the defined TDL format. TDL emulates device operation at one megahertz by completing simulation of a single test vector in one, 1000 nanosecond test period. TDL requires that each input pin is assigned to an unknown logic state and that bidirectional pins are assigned to a tristate condition during the initial test vector of a test program. This eliminates potential problems with signal spikes during the tester power-up sequence. TDL also outlines restrictions which specify time markers where input, output, and bidirectional device pin signals are allowed to transition within the test period. These timing restrictions are incorporated into the device functional simulations for modeling design response to the TDL test program. Using Quicksim, the TDL simulations are verified for correct operation and then converted into TDL test vector format for tester programming.

In the TDL format, the first 700 nanoseconds of a test period are reserved for input delay group signal transitions. An input delay group consists of a set of device input and bidirectional pin signals that transition logic state at the same time within the test period. All input and

bidirectional signals, with the exception of the clock input, are assigned to a single input delay group and allowed to make one logic transition during the test period. This transition occurs at the test period delay offset assigned to the input delay group. Signals cannot change delay group assignments during execution of the test program and therefore always perform logic transitions at the same point in the test period. A maximum of seven input delay groups, spaced a minimum of 50 nanoseconds apart, are provided in TDL with the first group always fixed at the zero offset of the test period.

Clock inputs are assigned to a input delay group defined to perform two logic transitions within the same test period. This configuration provides a complete clock pulse during the test period. The minimum pulse duration allowed by TDL is 100 nanoseconds and clock signal transitions are not allowed at the zero offset. Although the clock signal is defined in TDL, a pulse on the clock input is not required in every test vector. When the clock is defined as free-running, a clock pulse is inserted into all subsequent test vectors. If the clock is not specified to run continuously or is stopped during simulation, no clock transitions appear in subsequent test vectors. Input clock transitions are controlled by commands issued in the test program.

Each bidirectional pin of the ASIC device is assigned to an input delay group and also placed into a bidirectional control group. The tester accommodates a maximum of six bidirectional control groups each with a dedicated direction control signal. This signal serves as the common, output enable control for all bidirectional signals within the group. When the output enable control for a specific group is active, each bidirectional signal in that group is configured as an output signal for that test period.

Otherwise, bidirectional signals in the group function as input signals to the device . TDL enforces three restrictions on bidirectional signals to prevent contention during direction changes. First, the bidirectional signal must be driven as an high impedance input in the test period prior to the test period that a direction change from input to output occurs. Second, the bidirectional signal can not be strobed as an output during a test period that a direction change is present. Finally, the bidirectional signal must not be driven as an input during test periods where a direction change is present or where the signal is configured as an output.

The TDL configuration developed for the I/O module ASIC design consists of seven input delay groups which are spaced at 100 nanoseconds intervals and begin at offset zero. This format is illustrated in Figure 5.1. The I/O module ASIC input and bidirectional pin signals are assigned to input delay groups as shown in Table 5.1. Two bidirectional control groups are defined in the ASIC design with signal ESDOUTN as the output enable control for the group containing the seven, least- significant bits of the data bus, SD(6:0), and ESD7 as the output enable control for the group containing signal SD7. Clock input, SYSCLK, is assigned to input delay group seven which provides a 100 nanosecond, HIGH-LOW-HIGH pulse during the test period. The I/O module ASIC test period signal definition is structured to model signal events exhibited during system operation and ensure stable output signal states at the test period strobe.

In generation of the test program for the I/O module ASIC design, functional simulations were restructured to provide a chronological sequence of signal transitions conforming to established test period timing

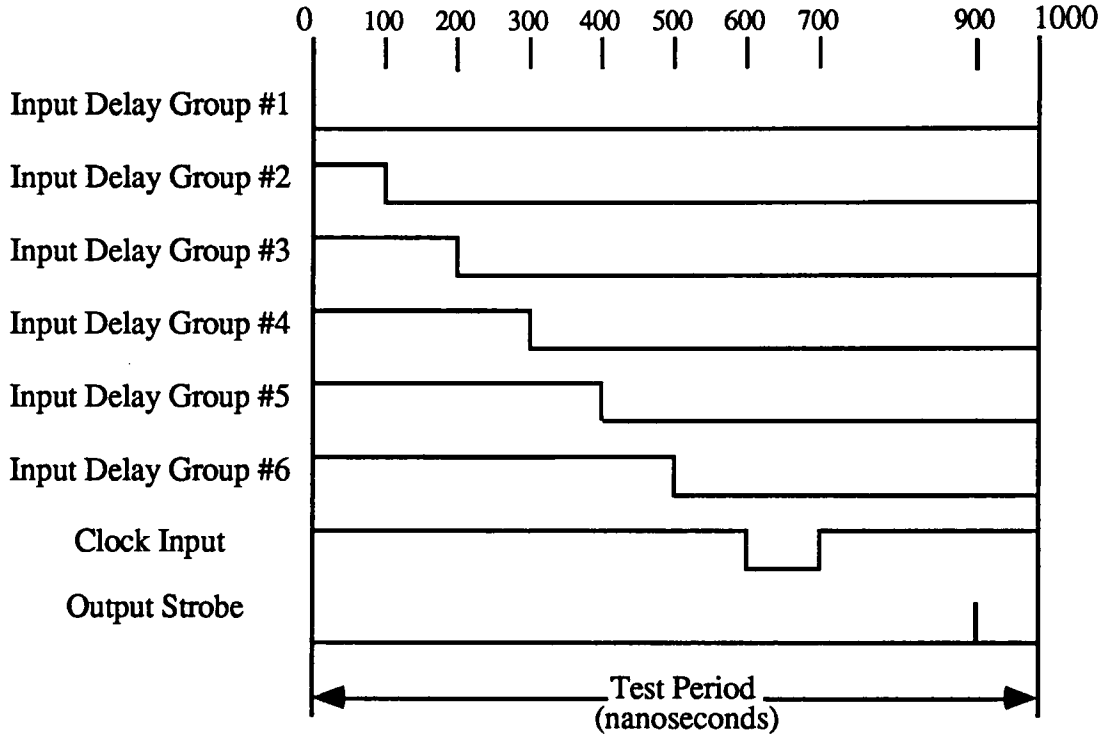

Figure 5.1. Input Signal Group Offset Delays

Table 5.1. Delay Group Signal Assignments

Group #1	Group #2	Group #3	Group #4	Group #5	Group #6
SD(7:0)	A(23:1)	AS*	RDWR	IACKIN*	BERRFRPAL*
	AM(5:0)		DS0*	TEST	
	SIMIN		DS1*	BUSY*	
	SYSRESET*				
	LWORD*				
	IACK*				

guidelines. Multiple simulation input vectors comprise a single test period by providing input signal transitions at the assigned input delay offsets. Each input vector contains logic state transitions for signals in a common input delay group. The time stamp associated with the input vector corresponds to the input group delay offset into the test period. The clock input is stopped and restarted at various points in the test program sequence to accommodate functional testing.

Simulations of the test program are conducted for both minimum and maximum propagation delay scenarios using standard tester capacitance loads modeled at each device output pin. The simulation results are verified not only for correct functional operation but also to ensure that all device output signals are stable at the output strobe point of each test period. To determine the expected output signal logic states in each test period, the logic states are taken from the simulation results at the output strobe point. These states must match in the simulation results from both propagation delay scenarios. This requirement ensures that the test program provides reliable verification of the ASIC over the specified operating temperature range and independent of device process variations. TDL also requires that all internal logic interconnections, input signal pins, and output signal pins toggle between both LOW and HIGH logic states at least once during the test program. To accomplish this 100% toggle rate, additional test vectors are accompanied by dedicated test logic in the I/O module ASIC design. In addition, ASIC output and bidirectional pins are also placed in the tristate condition, if possible, during the test program. When any of the above TDL

requirements are not satisfied during the test program generation, error messages are reported in the ASIC development environment.

A tool present in the ASIC design environment compiles the test program by combining the simulation input and output vectors corresponding to each test period into a sequence of TDL test vectors. Each TDL test vector represents one, 1000 nanosecond test period and contains a logic state bit for every device input, output, and bidirectional pin. Logic states, inserted into the TDL test vector for output signals and bidirectional signals, enabled as outputs, are taken from the simulation results. The assembled TDL test program is transferred to the manufacturer along with the design files for device fabrication and testing.

During testing, the manufacturer's tester reads the test vector and applies input signals to the device under test according to defined input group delay offsets. The tester strobes the output and bidirectional signal pins in each test period at the output strobe offset. The output logic states, read from the device under test, are compared with expected logic states contained within the test vector. Any mismatch is regarded as a test failure by the tester and the device is discarded by the manufacturer.

CHAPTER 6

ASIC PERFORMANCE RESULTS / DESIGN ENHANCEMENTS

Following submission of the design netlist and test program to the manufacturer, a quantity of five to ten prototype ASIC devices are fabricated and tested. These devices are delivered to the designer for verification in the target system. Approval of prototype device operation serves as the final step in the design verification process before ASIC production. For the I/O module ASIC project, five prototype devices were tested using a prototype 16 input/ 16 output discrete I/O module operating in the VMEbus Industrial Control System. The results of the ASIC device evaluation and recommendations for future design enhancements are presented in the sections that follow.

Performance Results

Testing of ASIC prototypes began with the verification of each logic function outlined in the I/O Module ASIC Specification. Using a debug utility in the PLC software, VMEbus cycles to the I/O module were performed as a series of single event steps. This technique facilitates I/O module response monitoring as events occur within the cycle. Data transfer during VMEbus read and write cycles to address offsets in the I/O module memory map was verified for the I/O module configuration. VMEbus cycles to invalid address offsets were produced to ensure that an incorrect response did not result. Cycles which initiate an I/O module reset and the disabling of output control points were also performed by the PLC. In addition,

completion of the I/O module initialization routine was verified following a system reset. ASIC device current consumption and I/O pin signal event timing were measured to ensure compliance with design specifications. At the completion of prototype functional testing, no specification violations were exhibited by the I/O module ASIC device.

In addition to the functional verification, prototype testing was conducted to ensure reliable operation over extended time periods of operation. This testing, labeled as reliability testing, exposed the target system to fluctuations in voltage supply and environmental conditions in an effort to identify design and/or component problems. Reports were continuously updated to indicate any I/O module ASIC failures during fluctuations in specified voltage and temperature ranges for the system. To perform the reliability testing, a test program was loaded into the PLC which simulates an actual control system application. This program enables the PLC to emulate a typical industrial control application by controlling system initialization, querying I/O module identification and status information, performing periodic updates of I/O point status, and reporting system errors.

For the I/O module ASIC reliability testing, the VMEbus Industrial Control System consisted of a PLC and multiple discrete, 16 input/ 16 output modules assembled into a VMEbus backplane chassis. The control application program was executed by the PLC over extended time periods, including nights and weekends, with error conditions accumulated into report summaries for evaluation. The system was subjected to environmental fluctuations by placing the VMEbus chassis into a temperature controlled chamber and allowing the ambient temperature to

cycle through the commercial temperature range. In addition, the backplane five volt power supply was adjusted within the limits specified by The VMEbus Specification during system operation. Variation of these parameters during testing provided a mechanism for determining if the effects of temperature and supply voltage fluctuations on I/O module ASIC device performance introduced errors into the system. At the completion of reliability testing, no failures were attributed to the I/O module ASIC prototypes.

Design Enhancements

Although the prototype devices performed flawlessly during functional and reliability testing, two potential problems surfaced in the I/O module ASIC design following extended analysis. Both potential problem areas are located in the logic circuit implementation of the I/O module initialization routine. An acceptable problem solution is provided in one case by a simple, external PCB hardware addition and in the other case by a PLC software modification. Neither identified problem area required an expensive redesign of the I/O module ASIC device.

The first potential problem results from the timing delay of a signal path located in the TSTINIDEC block. During an interrupt acknowledge cycle on the VMEbus, input signal, IACKIN*, is asserted logic LOW serving as an enable signal for devices G2 and G3. AS* is also asserted logic LOW, prior to IACKIN*, producing a logic HIGH level on signal ASCLK and ASCLKD. When both IACKIN*and AS* are logic LOW, the complementary outputs of device G5 allow either device G2 or G3 to produce a logic LOW level on its

corresponding output pin. If the ASIC is not requesting an interrupt, device G2 generates a LOW level on signal IACKOUT* passing the interrupt acknowledge cycle to the next device on the VMEbus daisy chain. When the ASIC is requesting an interrupt, a LOW level results on the output of device G3 which is used to generate signal RESPOND. The ASIC state machine responds to the interrupt acknowledge cycle when RESPOND is detected.

If the ASIC device is requesting an interrupt, the outputs of device G5 transition logic state during the next VMEbus interrupt acknowledge cycle to a level six interrupt. The transition of G5 outputs follow the rising edge on signal ASCLKD and must stabilize before a falling edge is detected on IACKIN* at the input to G2 or G3. From Table 2.1 on page 23, IACKIN* is asserted LOW a minimum of 40 nanoseconds from a falling edge on signal AS* at the slave device. When the ASIC is exhibiting maximum internal propagation delays, ASCLKD is delayed 39 nanoseconds from AS* and the production of stable G5 outputs from ASCLKD requires approximately four nanoseconds. This results in a total path delay from AS* to the input of G2 or G3 of approximately 43 nanoseconds. If IACKIN* is detected LOW exactly 40 nanoseconds from AS*, a LOW pulse of short duration is possible at the output of G2, IACKOUT*, since the outputs of G5 have not yet settled to their final logic state. When the outputs of G5 have stabilized, IACKOUT* is returned to a logic HIGH and the output of G3 is asserted. The unintentional pulse that results on IACKOUT* can produce unreliable system operation if the device located next on the interrupt acknowledge daisy chain is also requesting an interrupt and interprets the pulse as an acceptable interrupt acknowledge input signal. Although unlikely in normal operation, this

chain of events can result in two system devices responding to the same interrupt acknowledge cycle.

To eliminate the possibility of multiple responders to a single interrupt acknowledge cycle, a delay element is placed on the I/O module PCB between the IACKIN* signal from the VMEbus and the IACKIN* input pin of the ASIC device. This provides additional delay between the AS* and IACKIN* signals as seen at the ASIC device input pins and guarantees that G5 outputs are always stable before the falling edge of IACKIN*. The addition of a single, low cost delay element to each I/O module PCB prevents a costly second pass design of the ASIC device.

A second problem area is apparent when the ASIC device is operating in an analog I/O module configuration. During VMEbus read and write cycles to the I/O module, the ASIC state machine is designed to perform a handshaking operation with an external microcontroller before taking possession of the I/O module data bus. In contrast, the state machine does not request the data bus nor perform handshaking when responding to an interrupt acknowledge cycle in the initialization routine. As a result, VMERAMEN* is not asserted when the ASIC has control of the data bus inside the interrupt acknowledge cycle. If the microcontroller arbitrates for data bus control before the ASIC completes the interrupt acknowledge cycle, bus contention can result since the handshaking logic does not detect bus activity and grants bus access to the microcontroller.

As in the previous case, this potential problem area is resolved without an I/O module ASIC device redesign. Following completion of the interrupt acknowledge cycle, the PLC assigns physical/logical addresses and

performs the necessary write cycle(s) to transfer these addresses to the I/O module. The PLC then reads the I/O module identification vector at address offset 0001 in the memory map to determine the control point configuration. The potential for bus contention is eliminated by requiring the microcontroller to monitor the sequence of events during the I/O module initialization routine and restrict its use of the data bus until the interrupt acknowledge cycle is complete.

By configuring the ASIC device output pin, ADDR1, as an input signal to the microcontroller, the microcontroller detects a PLC read of address offset 0001 as a HIGH pulse on ADDR1. Since the read cycle corresponding to I/O module identification always follows the interrupt acknowledge cycle, the microcontroller prevents bus contention by not requesting control of the data bus until it detects the first active pulse on signal ADDR1. This action by the microcontroller ensures that the ASIC state machine has completed the interrupt acknowledge cycle and from this point will perform handshaking before taking control of the I/O module data bus. Requiring ADDR1 as an input to the microcontroller was incorporated into the specification for analog and special function I/O modules operating in the VMEbus Industrial Control System and therefore, the potential for bus contention during system initialization was eliminated.

BIBLIOGRAPHY

BIBLIOGRAPHY

1. Baker, Stan. "Merging Elements of ASIC Design." Electronic Engineering Times 2 Oct. 1989: 51.

2. LSI Logic Corporation. Logic Design Manual for ASICs. Second Edition, 1988.

3. Motorola, Incorporated. VMEbus Specification Manual. Revision C.1, 1985.

4. Read, John W., ed. Gate Arrays: Design Techniques and Applications. New York: McGraw-Hill, 1985.

5. Rubin, Kim. "Metastability Testing In Pals." Everything You Might Be Afraid To Know About Metastability. San Francisco: Wescon Electronic Show & Covention. 17-19 Nov. 1987.

6. Schroeter, John. Surviving the ASIC Experience. New Jersey: Prentice-Hall, 1992.

7. Texas Instruments, Incorporated. CMOS Gate Array Design Manual. Release 2.0, 1989.

8. Texas Instruments, Incorporated. Industrial Systems Division: Series 500 I/O System Product Profile. 1988.

9. Wong, Thomas, and Doug Page. "Can ASICs Remain Technology Drivers?" Electronic Engineering Times 2 Oct. 1989: 51.

APPENDIX

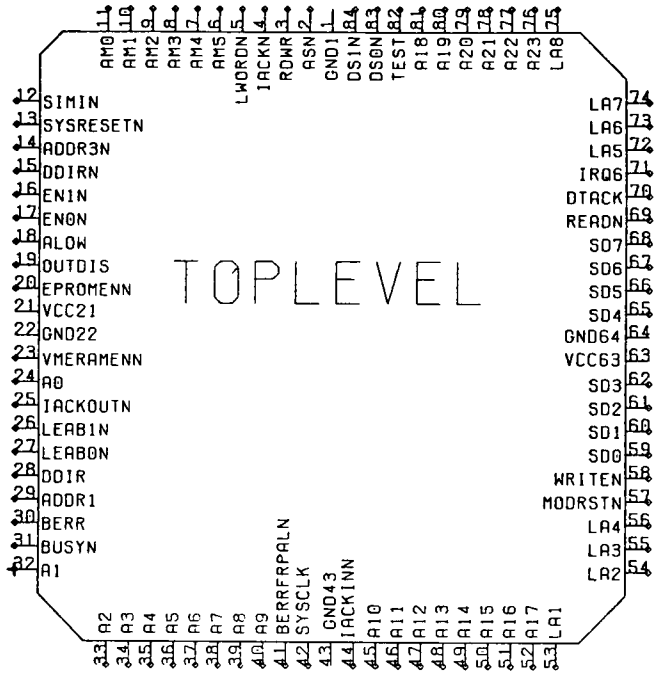

Figure A.1. Toplevel Schematic Diagram

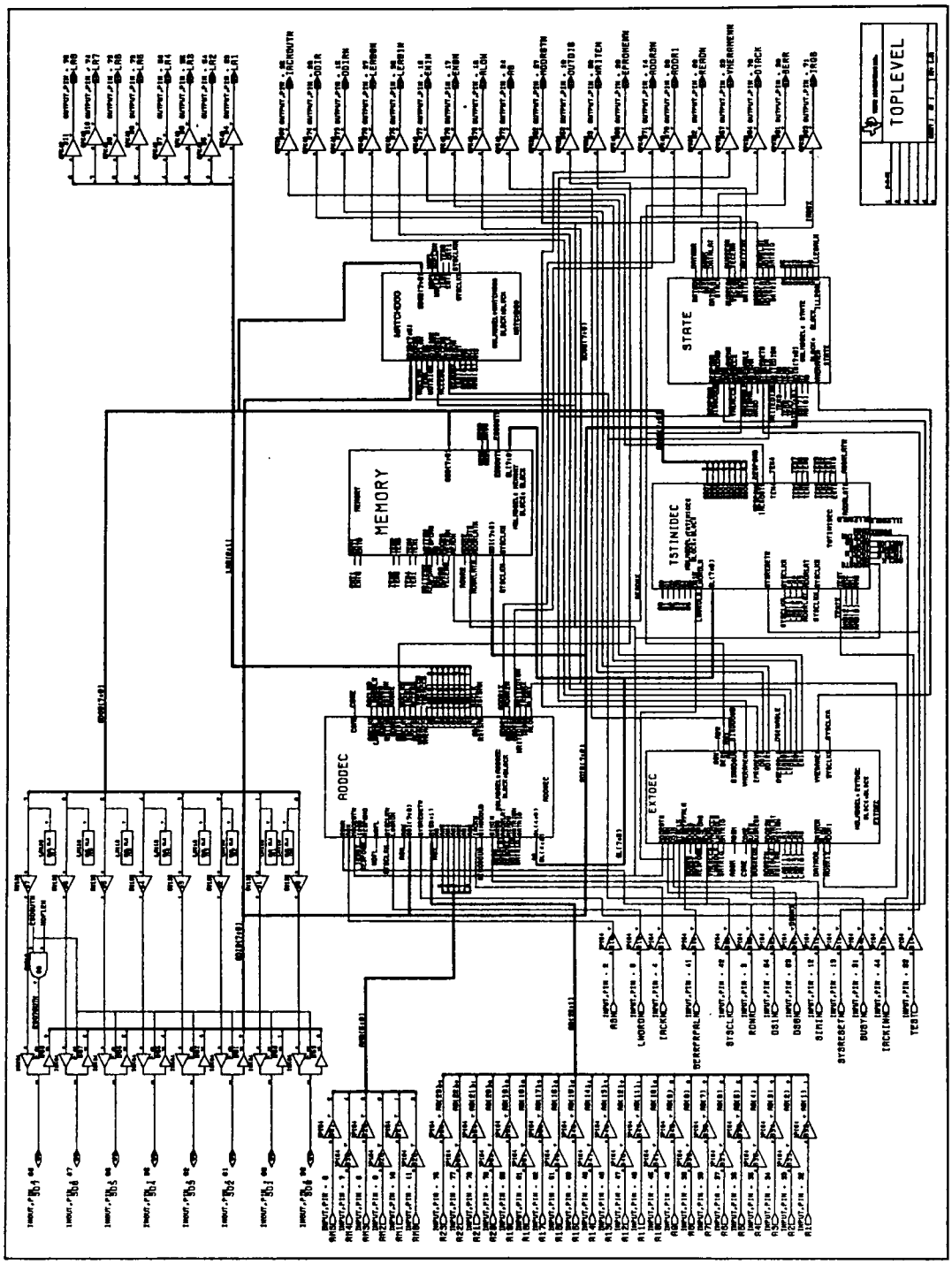

Figure A.1 (Continued)

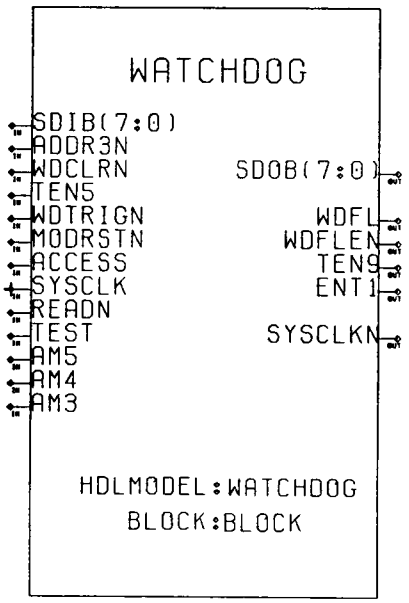

Figure A.2. Watchdog Timer Block Schematic Diagram

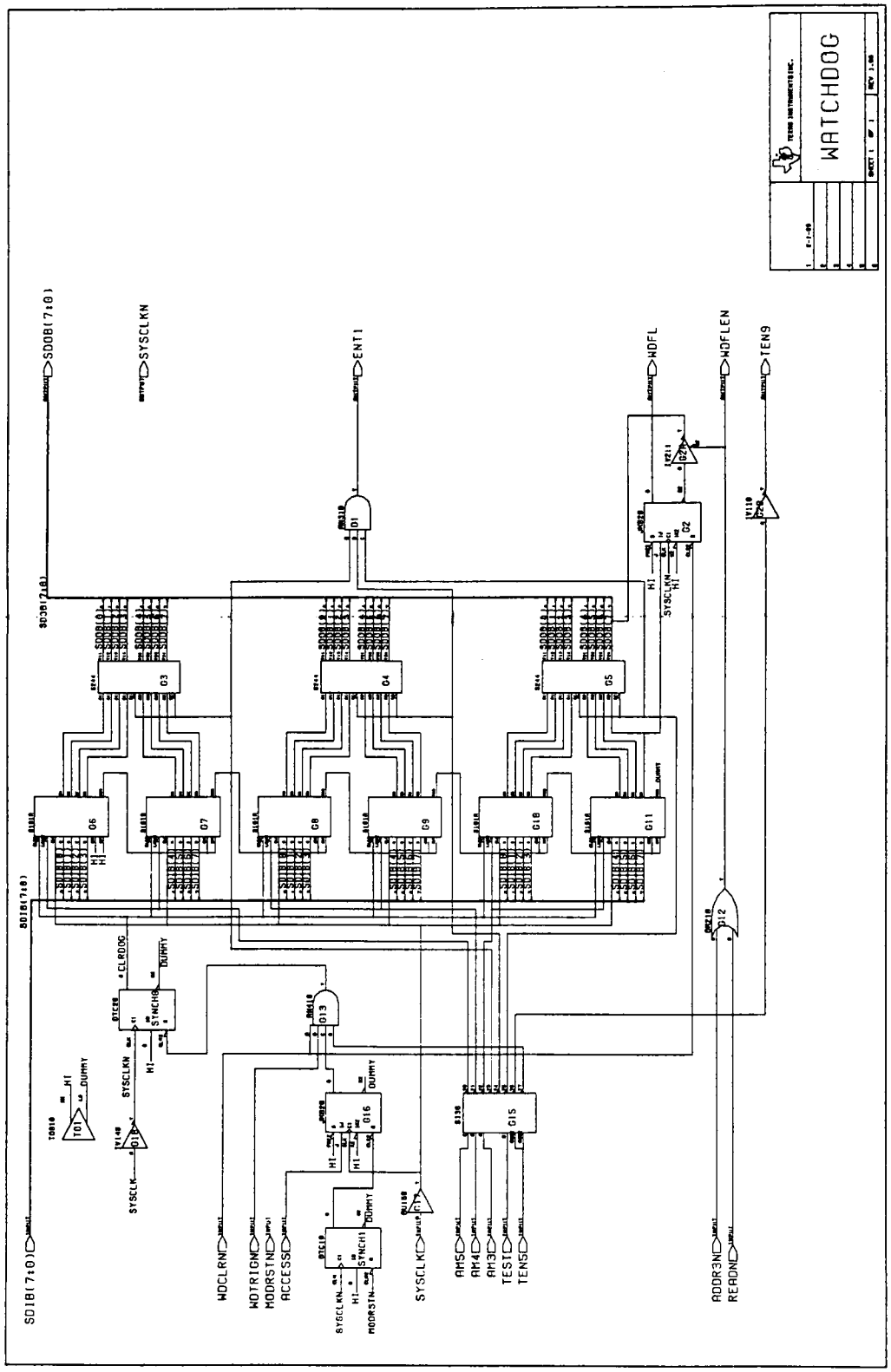

Figure A.2 (Continued)

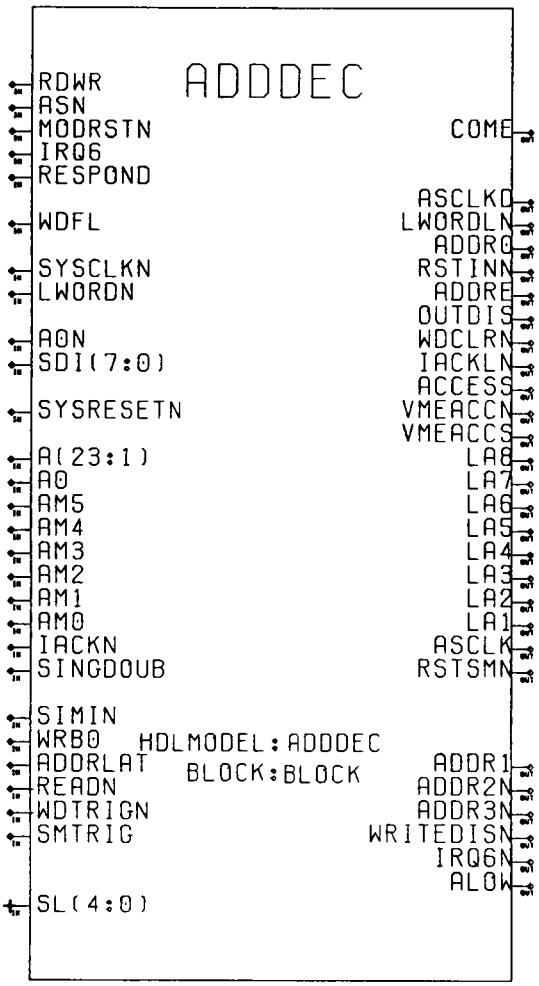

Figure A.3. Address Decode Block Schematic Diagram

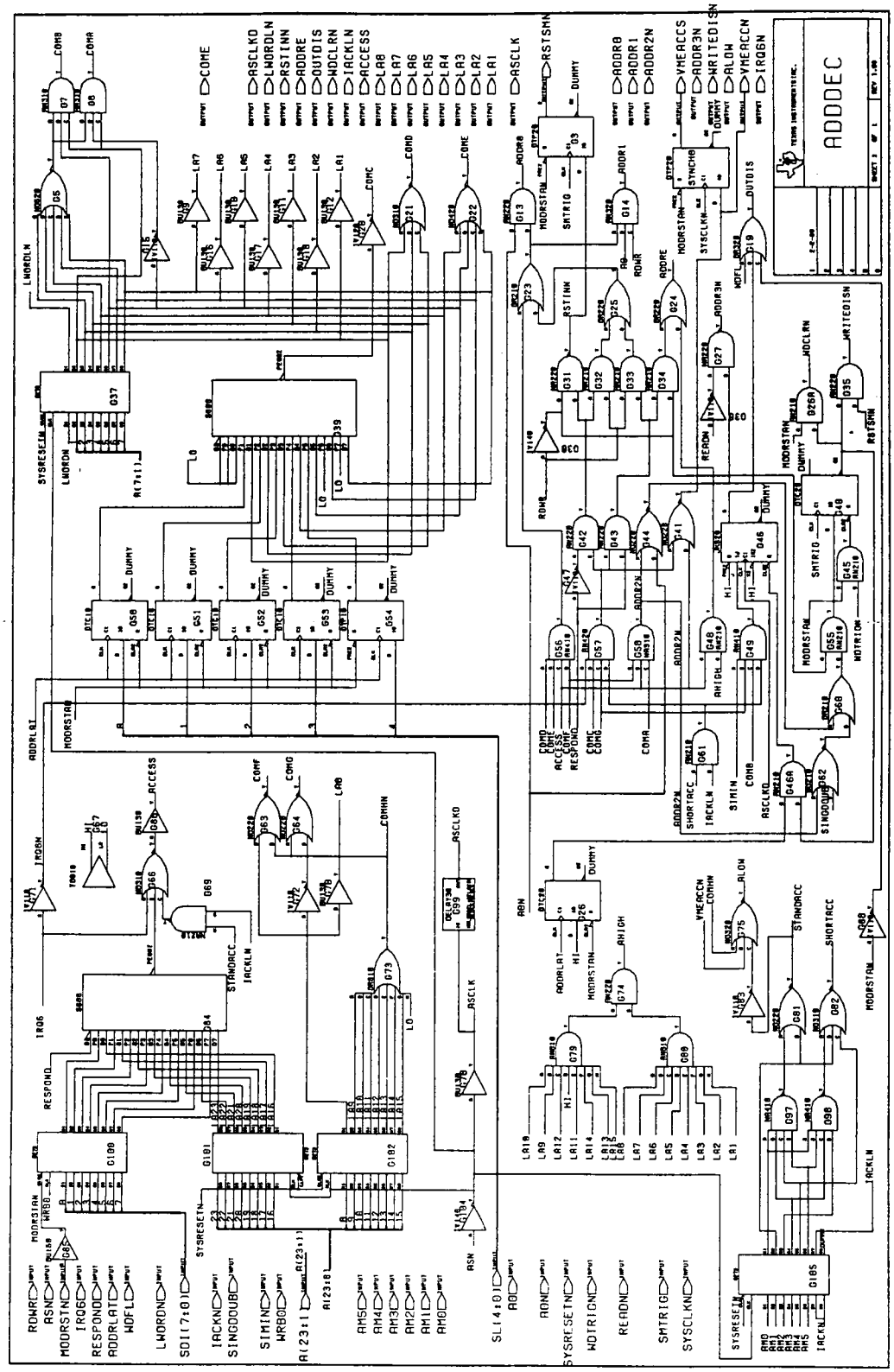

Figure A.3 (Continued)

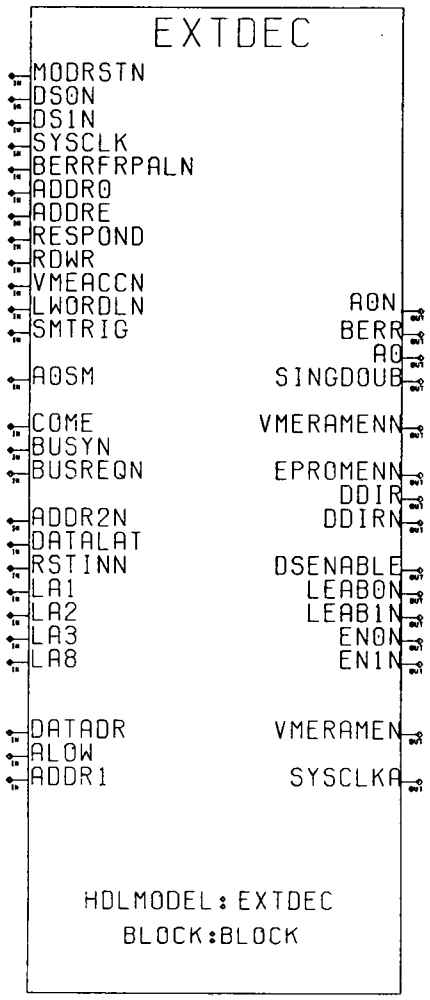

Figure A.4. External Decode Block Schematic Diagram

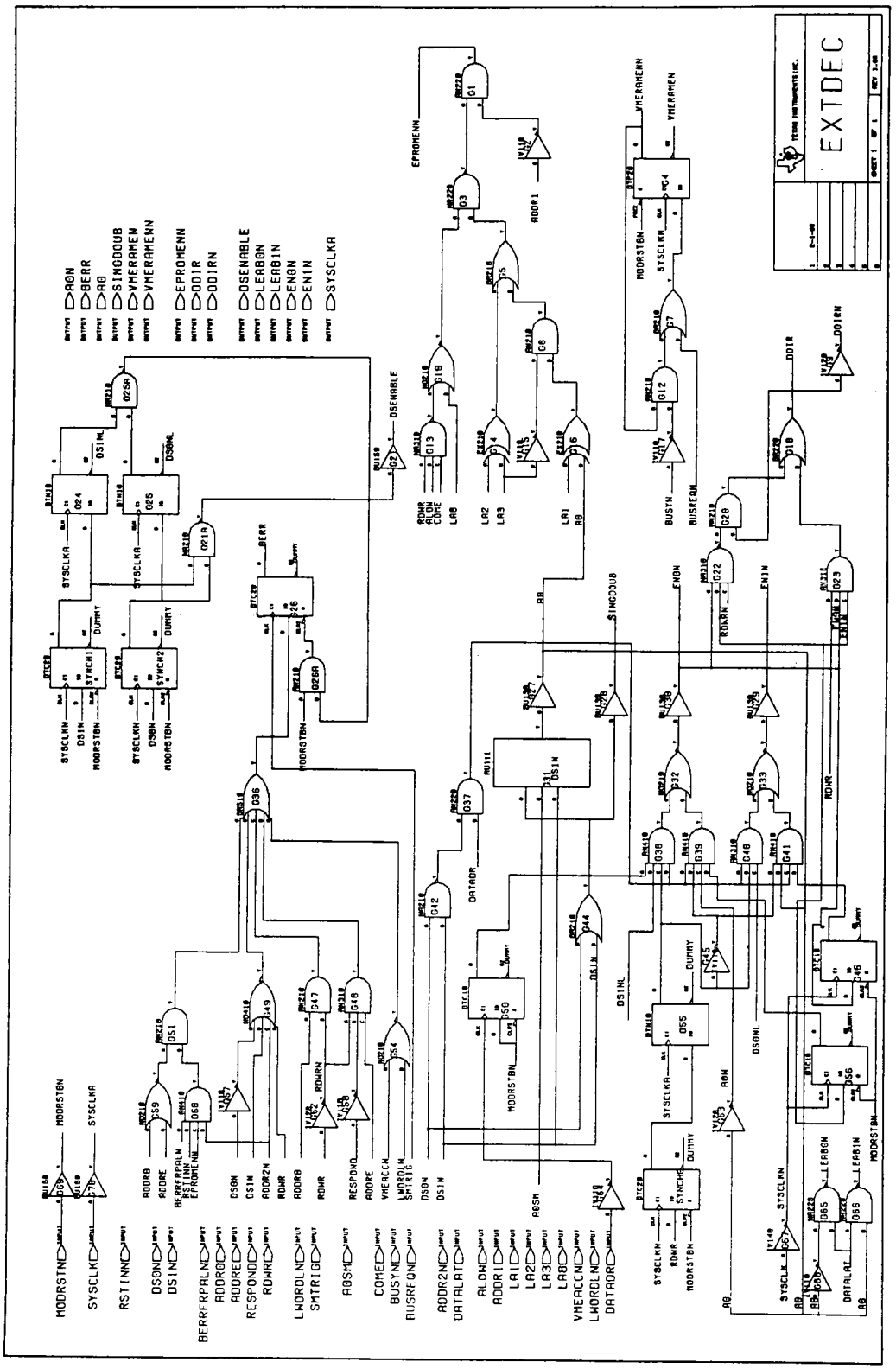

Figure A.4 (Continued)

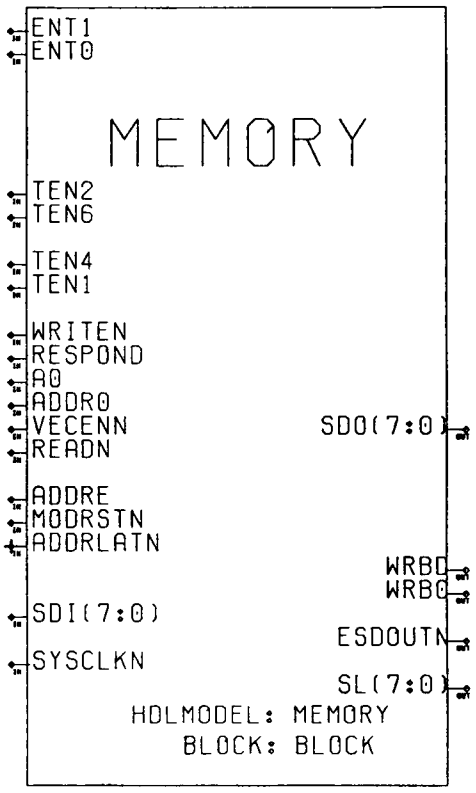

Figure A.5. Memory Block Schematic Diagram

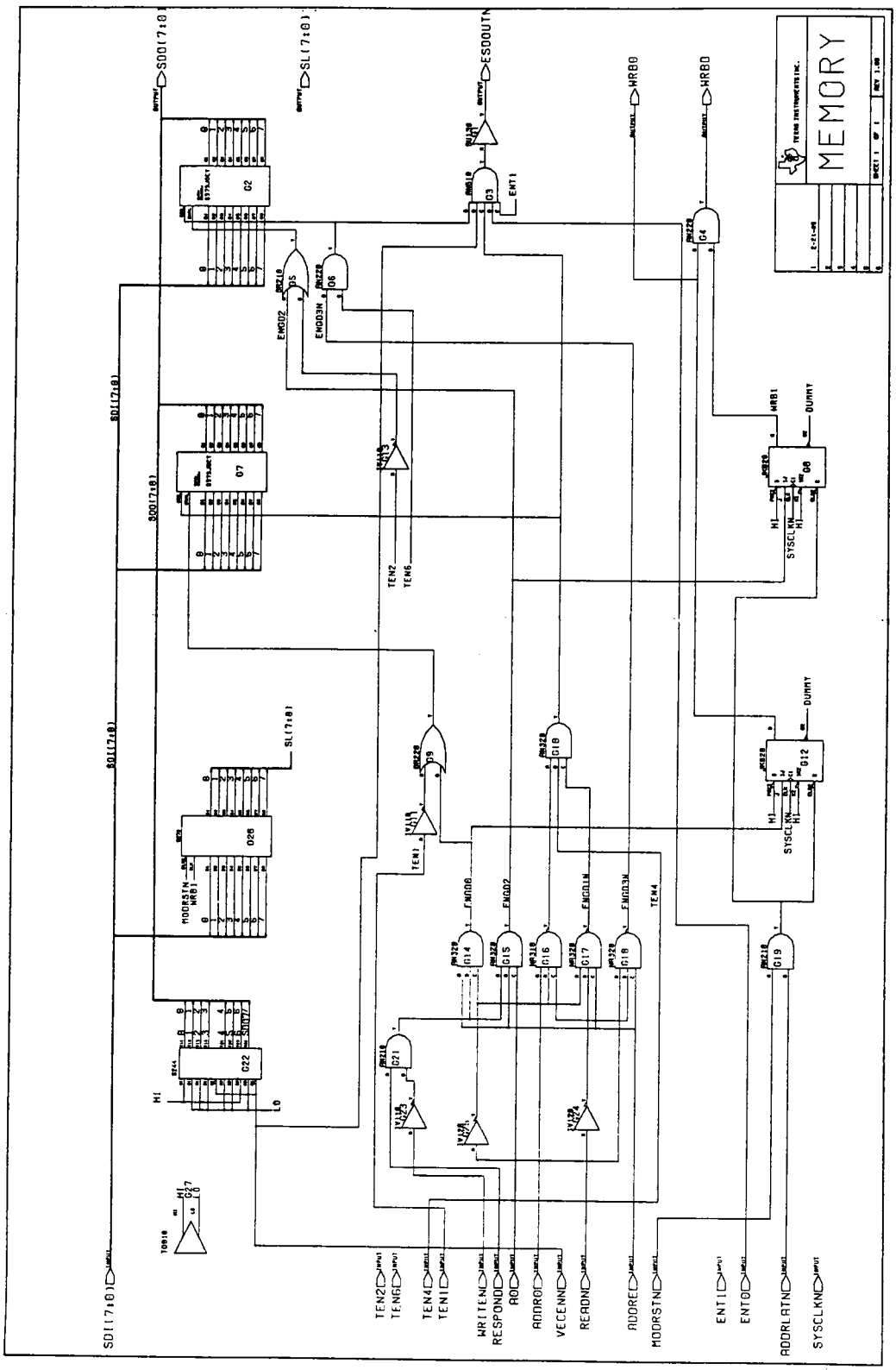

Figure A.5 (Continued)

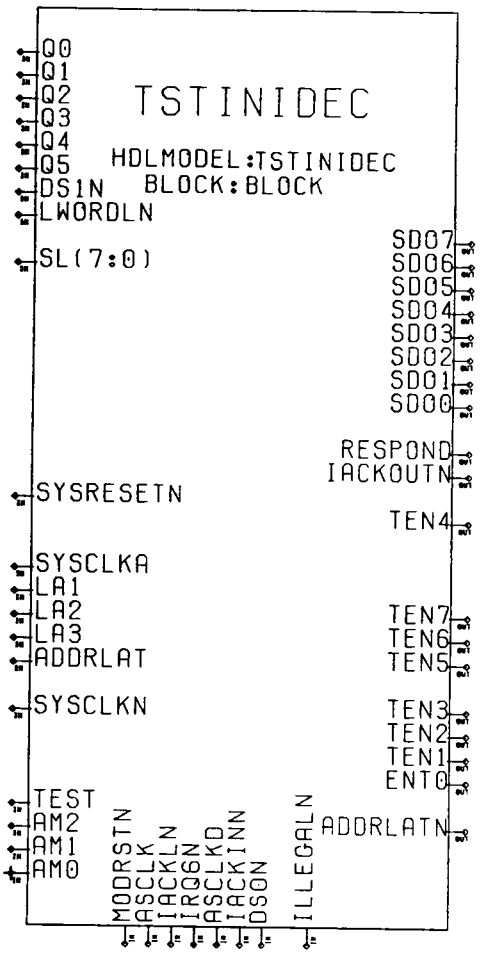

Figure A.6. Test/Initialization Block Schematic Diagram

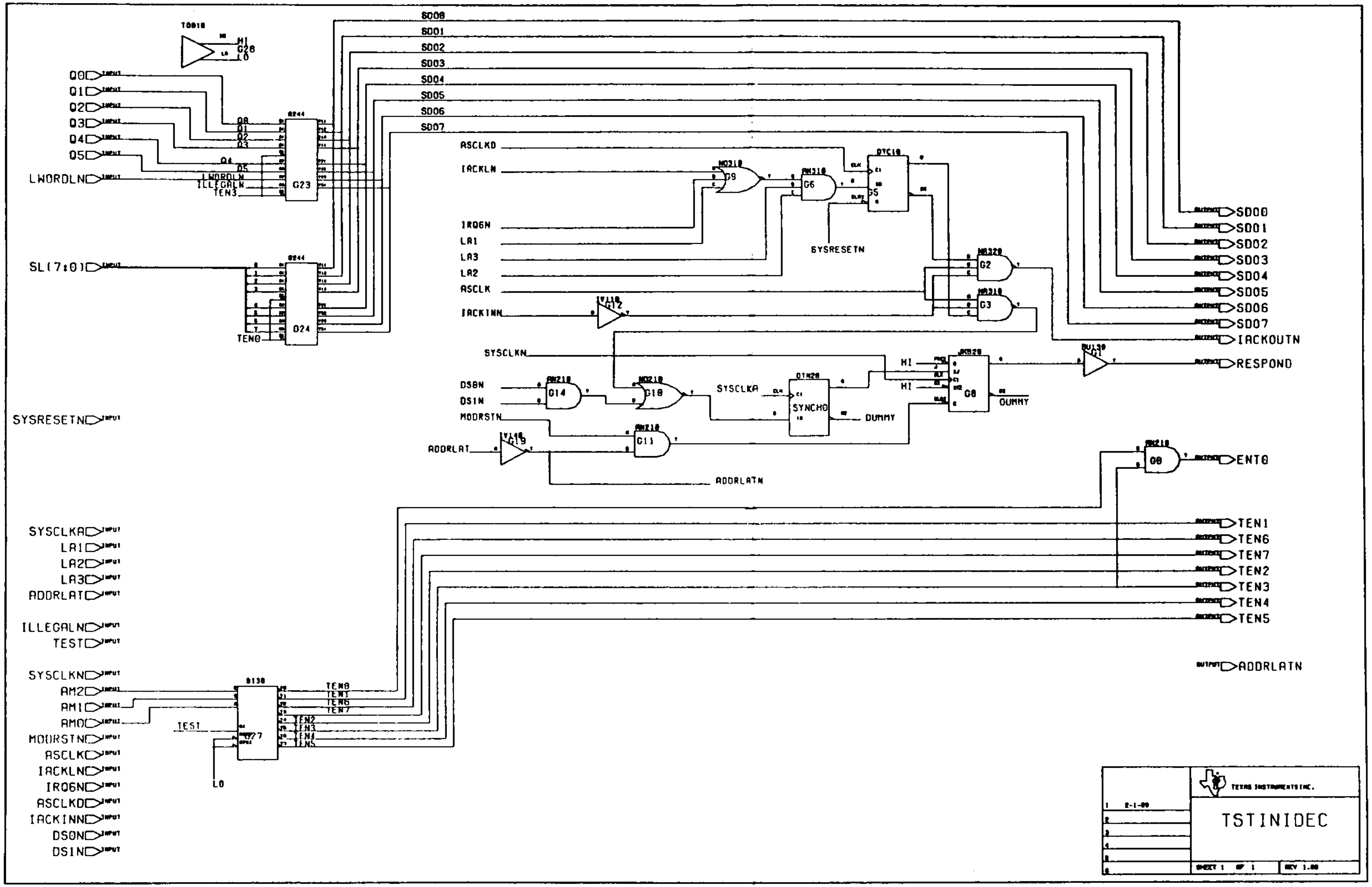

Figure A.6 (Continued)

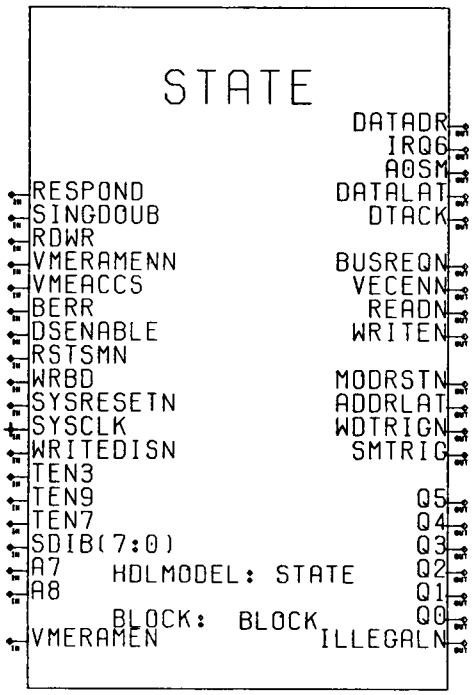

Figure A.7. State Machine Block Symbol Diagram

VITA

Thomas Cameron Perry was born in Johnson City, Tennessee on May 30, 1963. He attended elementary and middle schools in the Johnson City area before graduating from Science Hill High School in June, 1981. In September of 1981 he entered The University of Tennessee, Knoxville and in August, 1985 received the degree of Bachelor of Science in Electrical Engineering. He reentered The University of Tennessee on a part time basis in September of 1986 and after completing transfer credit from Georgia Tech, received a Master of Science degree in Electrical Engineering in December of 1992.

He is presently employed as a Senior Digital Design Engineer at AT&T Tridom Corporation in Atlanta, Georgia.