Advanced Micro Devices

Designing Interrupt Systems with the Am9519A Universal Interrupt Controller

Technical Manual

The International Standard of Quality guarantees the AQL on all electrical parameters, AC and DC, over the entire operating range.

901 Thompson Place, P.O. Box 3453, Sunnyvale, California 94088
(408) 732-2400 TWX: 910-339-9280 TELEX: 34-6306

Printed in U.S.A

TABLE OF CONTENTS

INTRODUCTION

General

Processors exist as tools for the implementation of information system transfer functions. All useful processor systems include at least one peripheral device in order to communicate with the user of the system. The processor not only manipulates information once it is in the system, but also handles the transfer of information to and from the user via the peripherals. Often several devices are integral parts of the overall system. All peripherals must be serviced in one way or another by the system processor. The basic parameters that influence the design of peripheral servicing algorithms are the frequency of service required, the service latency allowed and the service duty cycle of the devices.

There are two general methods used to initiate and coordinate this activity: Program controlled service and Interrupt driven service. In program controlled transfers, the processor schedules all peripheral events; an Interrupt driven system, on the other hand, allows modification of the system activities by external devices.

With no interrupt capability, processors must depend on software polling techniques to service peripheral devices. As the number of such devices grows and/or as the complexity of service increases, the polling program becomes very time consuming and the overhead devoted to polling becomes a significant fraction of the available processing resource. When this limits system performance, the use of interrupts can often provide substantial improvement.

Interrupts are used to enhance processor system throughput and response time by minimizing or eliminating the need for software polling procedures. Interrupts are hardware mechanisms that allow devices external to the processor to asynchronously modify the instruction sequence of the processor program being executed. An elementary single interrupt could be used simply to alert the processor to the fact that some kind of service is desired and thus to initiate a polling routine. More complex systems may have multiple interrupts and vectoring protocols which can be used to further improve performance and eliminate all polling requirements. Vectoring allows direct identification of the interrupting device and its associated service routine.

Figure 1 illustrates the essential functioning of a typical interrupt procedure. As the main program is executing instructions, an external interrupt arrives, in this example during instruction M+2.The processor completes M+2 and then, instead of executing M+3, it performs some kind of interrupt acknowledge procedure, often involving execution of an additional interrupt instruction. The result will usually be that the address of instruction M+3 is saved for future reference, and the location of instruction N is determined. The processor then proceeds to execute the interrupt service routine starting with instruction N. The service routine may save, and later restore, the processor status as well as perform tasks requested by the interrupting device. The last instruction in the routine (N+K) directs the processor to resume the main program at instruction M+3.

Notice that the presence of the hardware interrupt has caused a modification of the sequence of instruction execution; an additional block of instructions has been inserted in the main program. Interrupts provide the system designer with a significant capability that can help optimize his cost/performance tradeoffs.

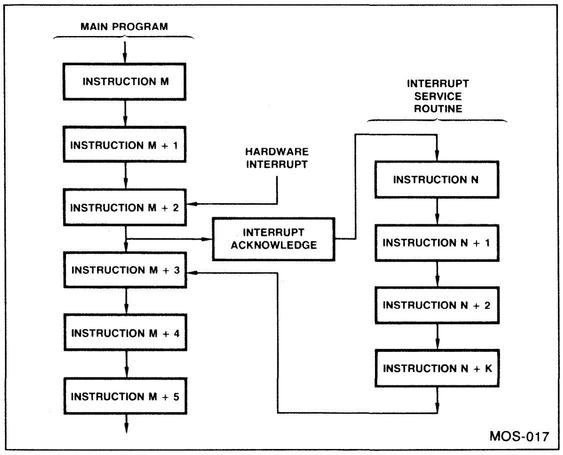


Figure 1. Basic Interrupt Procedure.

Features

The Am9519A Universal Interrupt Controller is a processor support device designed to enhance the interrupt handling capability of a wide variety of processors. A single Am9519A manages the masking, priority resolution and vectoring of up to eight interrupts. It may be easily expanded by the addition of other Am9519A chips to handle a nearly unlimited set of interrupt inputs. It offers many programmable operating options to improve both the efficiency and versatility of its host system operations. The Am9519A is well adapted to a wide range of uses including small, simple, as well as large, sophisticated, interrupt systems.

The Am9519A provides any mix of one, two, three and four byte responses to the host processor during the interrupt acknowledge process. The response bytes are all fully programmable so that any appropriate addressing, vectoring, instruction or other message protocol may be used. Contention among multiple interrupts is managed internally using either fixed or rotating priority resolution circuitry. The direct vectoring capability of the Am9519A may be bypassed using the polled mode option.

An internal mask register permits individual interrupts to be disabled. It may be loaded in parallel by the host processor with any bit pattern, or mask bits may be individually controlled. The interrupt inputs use "pulse-catching" circuitry so that an external register is not needed to capture interrupt pulses. Narrow noise pulses, however, are ignored. The interrupt polarity may be selected as either active-high or active-low.

Another important feature of the Am9519A is its ability to generate software interrupts. The host processor can set interrupt requests under program control, thus permitting hardware to resolve the priority of software tasks. This is often a powerful system asset, especially for sophisticated operating software, as well as an aid for system testing, diagnostic, debugging and maintenance procedures.

The Am9519A is implemented with AMD's n-channel silicon gate MOS technology. The chip contains 4500 transistors within a total chip area of 28,765 square mils. It is packaged in a standard 28-pin dual in-line package.

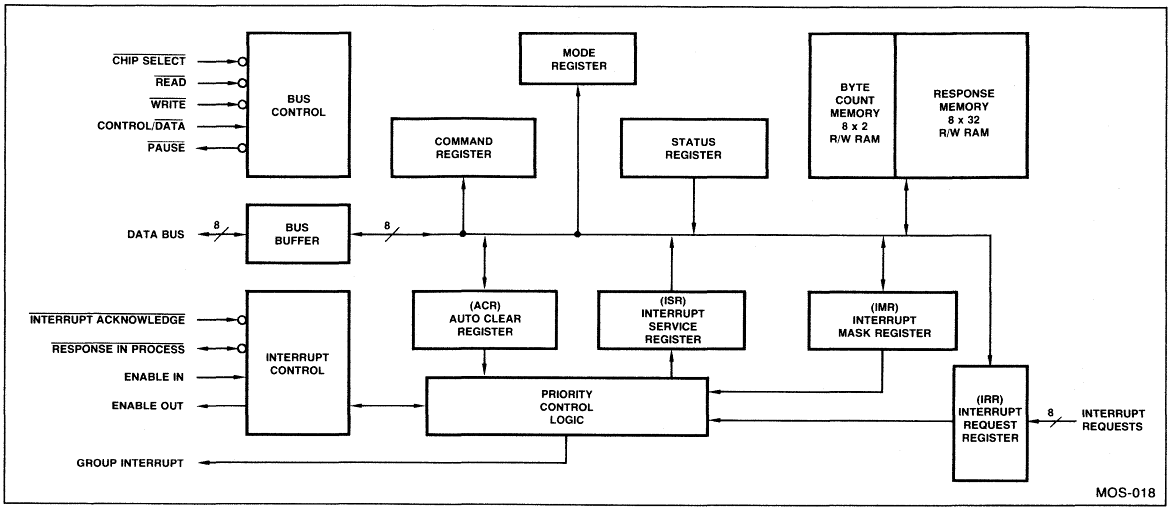


Figure 2. Am9519A Block Diagram.

HARDWARE INTERFACE

Block Diagram

The block diagram of the Am9519A shown in Figure 2 indicates the interface signals and the basic internal information flow. Interrupt Request inputs are captured and latched in the Interrupt Request register. Any requests not masked by the Interrupt Mask register will cause a Group Interrupt output to the host processor if the unit is enabled. When the processor is ready to handle the interrupt it issues an Interrupt Acknowledge pulse which causes (a) the priority of pending interrupts to be resolved and (b) a byte from the response memory associated with the highest priority interrupt to be transferred to the data bus. The transfer of additional response bytes is controlled by additional Interrupt Acknowledge signals. Other interrupt management functions are controlled by the Auto Clear register, the Interrupt Service register and the Mode register. The host processor controls the Am9519A by using the Command register. The Status register reports on the internal condition of the part.

The Am9519A is addressed by the host processor as two distinct ports: a control port and a data port. The control port provides direct access to the Status register and the Command register. The data port is used to communicate with all other internal locations.

Interface Signal Description

Figure 3 summarizes the interface signals. Figure 4 shows the interface signal pin assignments.

Data Bus (DB)

The eight three-state bidirectional data bus lines are used to transfer information between the Am9519A and the system data bus. The direction of information flow is controlled by the $\overline{CS}$, $\overline{RD}$, $\overline{WR}$ and $\overline{IACK}$ input signals. Data and command information are written into the device; status, data and response information are output by it.

Description	Abbreviation	Type	Pins
+5 Volts	VCC	Power	1
Ground	VSS	Power	1
Data Bus	DB	I/O	8
Response In Process	RIP	I/O	1
Interrupt Request	IREQ	Input	8
Chip Select	CS	Input	1
Read	RD	Input	1
Write	WR	Input	1
Control/Data	C/D	Input	1
Interrupt Acknowledge	IACK	Input	1
Enable In	EI	Input	1
Enable Out	EO	Output	1
Group Interrupt	GINT	Output	1
Pause	PAUSE	Output	1

Figure 3. Am9519 Interface Signal Summary.

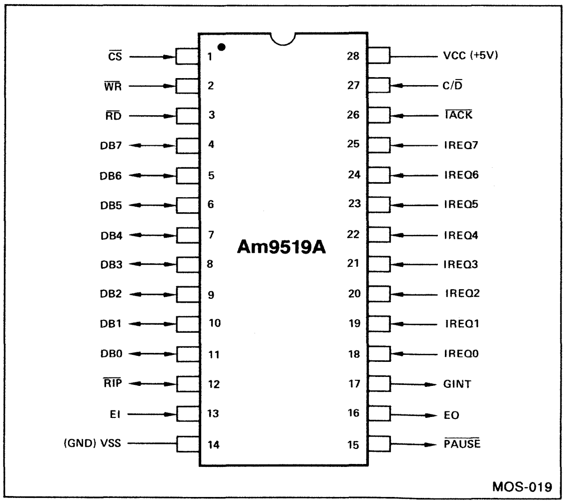


Figure 4. Connection Diagram.

Chip Select ($\overline{CS}$)

The Chip Select input is an active low signal used to condition the chip for read and write operations on the data bus; Read/Write transfers will not take place unless the $\overline{CS}$ input is low. Chip Select does not condition Interrupt Acknowledge operations. However $\overline{CS}$ must be high for a minimum of 100nsec prior to an interrupt acknowledge cycle or the response memory may be altered. Chip Select is usually derived by decoding an address output by the host processor; the negative-true polarity matches outputs from typical decoder circuits.

Read ($\overline{RD}$)

The Read input is an active low signal conditioned by Chip Select that indicates information is to be transferred from the Am9519A to the data bus. Read is usually a timed pulse issued by the host processor.

Write ($\overline{WR}$)

The Write input is an active low signal conditioned by Chip Select that indicates information is to be transferred from the data bus to the Am9519A. Write is usually a timed pulse issued by the host processor.

Control/Data (C/$\overline{D}$)

The Control/Data input acts as the port address line and is used to select source and destination locations for read and write transfers. Data transfers (C/$\overline{D}$=0) are made to or from preselected internal memory or register locations. Control transfers (C/$\overline{D}$=1) write into the command register or read from the status register.

Interrupt Request (IREQ)

The eight Interrupt Request inputs are used by external devices to indicate that service is desired. The Interrupt Request Register associated with the inputs uses asynchronous pulse-catching circuitry to latch any active requests that occur. The input polarity may be programmed to capture either positive-going or negative-going transitions. Reset selects the active low option.

Response In Process ($\overline{RIP}$)

The Response In Process signal is a bidirectional line designed to be used when two or more Am9519A circuits are connected together. $\overline{RIP}$ is used to prevent new higher priority interrupts from interfering with an Interrupt Acknowledge process that is underway. An Am9519A that is responding to a selected interrupt will treat $\overline{RIP}$ as an output and will assert $\overline{RIP}$ low until the acknowledge response is complete. An Am9519A without a selected interrupt will treat $\overline{RIP}$ as an input and will ignore $\overline{IACK}$ pulses as long as $\overline{RIP}$ is low. The $\overline{RIP}$ lines from multiple Am9519A circuits may be wired directly together. $\overline{RIP}$ is an open drain signal, and requires an external pullup resistor (3.3K or larger) to V_{CC} in order to establish the logic high level. When using the Am9519A with other devices, a $\overline{RIP}$ must be generated to force $\overline{PAUSE}$ inactive when they are responding.

Group Interrupt (GINT)

When active, the Group Interrupt output indicates that at least one bit is set in the Interrupt Register, (IRR) which is not masked by the Interrupt Mask Register or the Interrupt Service Register. GINT is used to notify the host processor that service is desired. It may be programmed for either active high or active low polarity in order to simplify the interface with the host circuitry. Reset selects active low. When active high is selected the output is a standard two-state buffer configuration. When active low is selected the output is open drain and requires an external pullup resistor (3.3K or larger) to V_{CC} in order to establish the logic high level. The open drain configuration is useful for wired-or connections in systems with more than one Am9519A. GINT may glitch after the last interrupt acknowledge pulse of an acknowledge cycle. This is not a problem for most CPUs since vectored interrupts are level sensitive.

Interrupt Acknowledge ($\overline{IACK}$)

The Interrupt Acknowledge input is an active low signal generated by the host processor and used to request interrupt response information. One response byte will be transferred by the Am9519A for each $\overline{IACK}$ pulse received and up to four bytes may be transferred during each interrupt acknowledge sequence. The first $\overline{IACK}$ pulse following a GINT output also initiates the internal selection of the highest priority unmasked interrupt.

Many processors provide interrupt acknowledge signals directly, including the 8085, the 8080A and the 2650. For others, such as the Z80, 6809, Z8000, 68000 and 8086, it can be generated quite easily with simple gating.

Pause

The Pause output is an active low signal used during $\overline{IACK}$ cycles to indicate that the Am9519A has not completed the data bus transfer operation presently underway. The $\overline{IACK}$ pulse should be extended by the host processor at least until the $\overline{PAUSE}$ output goes high. The width of active $\overline{PAUSE}$ pulses is a function of several variables; it will be quite short in some systems and longer in others. $\overline{PAUSE}$ is an open drain output and requires an external pullup resistor to establish the high logic level. $\overline{PAUSE}$ signals should be wired together in multiple chip interrupt systems. In some systems $\overline{RIP}$ is used in place of pause. This can save logic in some systems. Be sure to check timing to system requirements.

Enable In (EI)

The Enable In input is an active high signal used to implement a "daisy chain" expansion capability with other Am9519A chips. EI may also be used as a hardware disable/enable input for the interrupt system. When EI is low, the daisy chain is disabled. However, PAUSE is still driven low in response to $\overline{IACK}$. Internally, a relatively high impedance resistor is connected between EI and V_{CC} so that an unused EI requires no external pullup resistor.

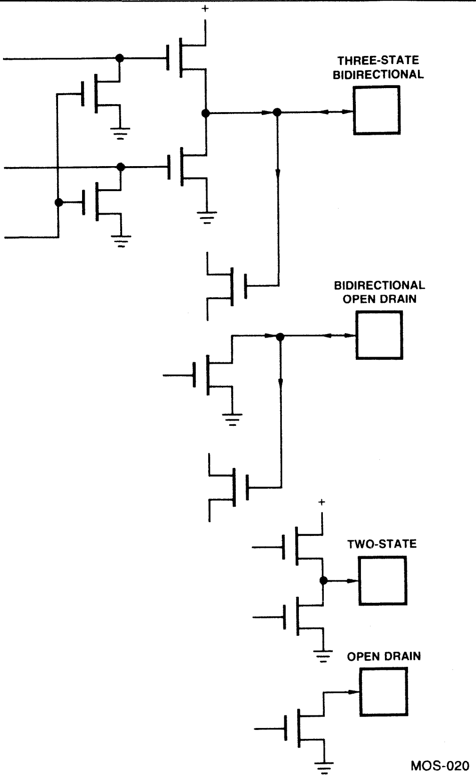
Enable Out (EO)

The Enable Out output is an active high signal used to implement a "daisy chain" expansion capability with other Am9519A chips. When the $\overline{IACK}$ input goes low, EO goes low until EI goes high and the chip determines that no unmasked request is pending. EO is a two-state output with relatively modest drive capability.

Interface Considerations

All of the input and output signals for the Am9519A are specified with logic levels identical to those of standard TTL circuits. The worst-case input logic levels are 2.0V high and 0.8V low. Except for the open drain signals, the worst-case output logic levels are 2.4V high and 0.4V low. Thus, for TTL interfacing, the normal worst-case noise immunity of at least 400mV is maintained. The logic level specifications take into account all combinations of the three variables that affect the logic level threshold: ambient temperature, supply voltage and processing parameters. A change in any of these toward nominal values will improve the actual operating margins.

The $\overline{\text{PAUSE}}$ and $\overline{\text{RIP}}$ outputs are open drain with no active pullup transistors; their output high levels are established by the external circuitry. The GNT output, when programmed for active low polarity ($\overline{\text{GINT}}$), is also an open drain output that does not control its output high level.



Signal	Output Description
Data Bus (DB0-DB7)	Bidirectional, Three-State
Response In Process (RIP)	Bidirectional, Open-drain
Pause	Open-drain
Group Interrupt (GINT)	Two-State when active high Open-Drain when active low
Enable Out (EO)	Two-state, low drive

Figure 5. Am9519A Output Buffer Summary and Circuitry.

All of the output buffers except EO and the open drain outputs can source at least 200μA worst-case and can sink at least 3.2mA worst-case while maintaining TTL output logic levels. EO normally only drives EI of another Am9519A chip and is specified with less drive capability in order to improve the priority resolution speed in multi-chip interrupt systems. The open drain outputs all sink at least 3.2mA as the other outputs do. Current sourcing for the open drain outputs is determined by the external circuitry. Figure 5 summarizes the types of outputs on the Am9519A.

Unprotected open gate of high quality MOS transistors exhibit very high resistances on the order of 10^{14} ohms. It is easy in many circumstances for charge to enter the gate node of such an input faster than it can be discharged and consequently for the gate voltage to rise high enough to break down the oxides and destroy the transistor. All inputs to the Am9519A include protection networks to help prevent damaging accumulations of static charge. The protection circuitry is designed to slow the transitions of incoming current surges and to provide low impedance discharge paths for voltages beyond the normal operating levels. Please note, however, that input energy levels can nonetheless be too high to be successfully absorbed. Conventional design, storage, and handling precautions should be observed so that the protection networks themselves are not overstressed.

Within the limits of normal operation, the input protection circuitry is inactive and may be modeled as a lumped series RC as shown in Figure 6. The functionally active input connection during normal operation is the gate of an MOS transistor. Except for EI, no active sources or drains are connected to the inputs so that neither transient nor steady-state currents are impressed on the driving signals by the Am9519A other than the charging or discharging of the input capacitance and the accumulated leakage associated with the protection network and the input circuit. Lumped input capacitances are usually around 6pF and leakage currents are usually less than 1μA.

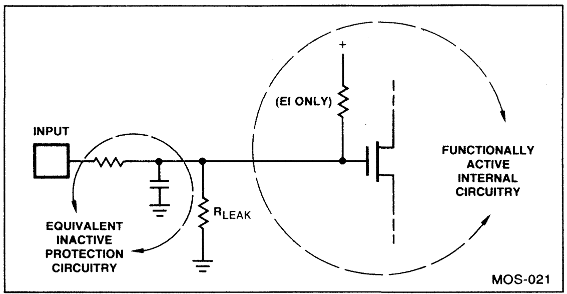

INPUT
(EI ONLY)
FUNCTIONALLY ACTIVE INTERNAL CIRCUITRY
R_{LEAK}
EQUIVALENT INACTIVE PROTECTION CIRCUITRY
MOS-021

Figure 6. Input Circuitry.

Fanout from the driving circuitry into the Am9519A inputs will generally be limited by transition time considerations rather than DC current limitations when the loading is dominated by MOS circuits like the Am9519A. In an operating environment, all inputs should be terminated so they do not float and accumulate stray static charges. Unused inputs should be tied directly to Ground or to V_{CC}, as appropriate. An input in use will have some type of logic output driving it and termination during operation will not be a problem. Where inputs are driven from logic external to the card containing this chip, however, on-board termination should be provided to protect the chip when the board is unplugged and the input would otherwise float. A pullup resistor or a simple inverter or gate will suffice.

IREQ Timing

The circuitry at the IREQ inputs is quite straightforward and is illustrated in Figure 7. Inverters 1 and 2 buffer the input and shift the logic voltages to the somewhat wider swing used internally. The exclusive-or gate is used to select the sense of the active transition edge that will set the IRR. Mode register bit M4 is used directly for control of the exclusive-or gate. The selected interface edge will always produce a negative going transition at output 3. Inverters 4, 5, 6, 7 and 8 form a delay chain. Nor gate 9 has three inputs and the IRR bit will be set when all three inputs to 9 are low. As shown in the timing diagram of Figure 8, the input to gate 9 from inverter 8 is normally low when there is no active IREQ signal at the interface. When a transition occurs, the output of gate 3 will go low and only the signal from inverter 5 prevents the immediate setting of the IRR bit. As shown in the left portion of the timing diagram, if the output from 3 has returned high be-

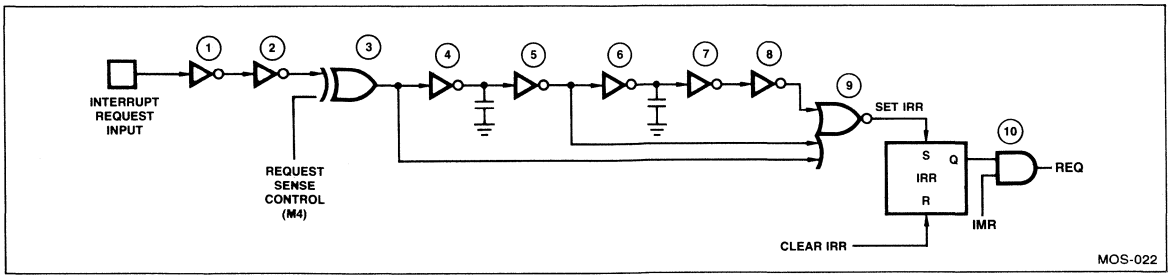


Figure 7. Interrupt Request Logic.

fore the output from 5 goes low, the IREQ transition will be ignored and the IRR bit will not be set. On the other hand, the right side of the timing diagram shows that if the active IREQ input is present long enough, then the output from both 3 and 5 will become low at the same time, and output 9 will go high. Output 8 is used to turn off Nor gate 9 after the IRR bit is set.

In summary, the input circuitry for the IREQ signals provides these characteristics:

1. Polarity for IREQ inputs is controlled;
2. Narrow IREQ pulses are ignored;
3. Wide IREQ pulses are captured;
4. Transitions to active levels are captured just once;
5. New transitions are required to generate new interrupts.

The IRR thus acts in a "pulse-catching" mode with respect to the IREQ inputs. Figure 9 shows the types of IREQ waveforms that will be recognized and latched by the IRR. Note that a transition to a level may be used although only a pulse is required; it is not necessary to maintain an IREQ input active level. Further, a continuously active level on IREQ will not cause a new interrupt each time IRR is cleared. There must be a new active transition on IREQ after IRR is cleared in order to generate a new interrupt. An active level must go inactive for about 40nsec before its new active edge will be recognized.

To minimize noise sensitivity, all active IREQ pulses narrower than about 20nsec will be ignored by the IRR. To maintain the pulse-catching characteristics, all active IREQ pulses wider than the specified data sheet minimum will be captured by the IRR. The results for intermediate pulse widths will depend on charactertics of the particular part being used and its operating conditions, especially temperature.

Power Supply

The Am9519A requires only a single +5V power supply. The commercial temperature range parts have a voltage tolerance of ±5 the military temperature range tolerance is ±10%. Maximum supply currents are specified in the data sheet at the high end of the voltage tolerance and the low end of the temperature range. In addition, the current specifications take into account the worst-case distribution of processing parameters that may be encountered during the manufacturing life of the product. Typical supply current values, on the other hand, are specified for a nominal supply of +5.0 volts, nominal ambient temperature of 25°C, and nominal processing parameters. Supply current always decreases with increasing ambient temperature; thermal run-away is not a problem.

Although supply current will vary from part to part, a given unit at a given operating temperature will exhibit a nearly constant power drain. There is no functional operating region that will cause more than a few percent change in the supply current. Decoupling of VCC, then, is straightforward and will generally be used simply to isolate the Am9519A from external VCC noise.

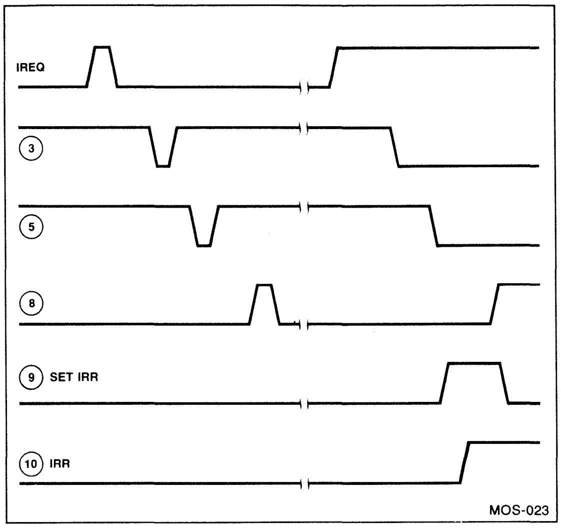


Figure 8. IREQ Internal Timing.

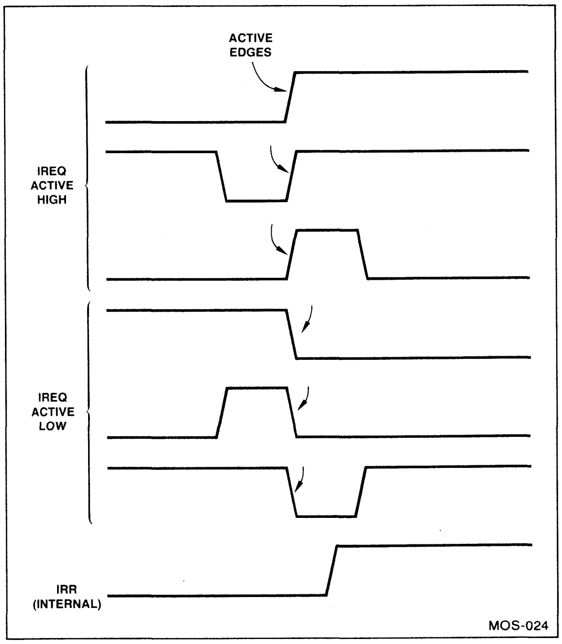


Figure 9. IREQ Waveforms.

OPERATING DESCRIPTION

Reset

The Am9519A does not include an external hardware reset input. The reset function is accomplished either by software command or automatically during power-up. The reset may be initiated by the host processor at any time simply by writing all zeros into the command port. Power-up reset circuitry is internally triggered by the rising V_{CC} voltage when a predetermined threshold is reached, generating a brief internal reset pulse.

The response memory and byte count registers are not affected by resets. Their content after power-up are unpredictable and all 8 channels must first be initialized by the host processor. A software reset does not disturb previous response memory and byte count contents.

The Interrupt Mask register is set to all ones by a reset, thus disabling recognition of interrupts by the chip. The Status register continues to reflect the internal condition of the chip and is not otherwise directly affected by a reset. All other registers are cleared to all zeros by a reset. The polarities of the Mode register control bits are assigned to provide a reasonable operating option environment when cleared by a reset.

Register Description

The Am9519A uses several control and operation registers plus the response memory to perform and manage its many functions. Figure 10 lists these elements and summarizes their size and number.

Description	Abbreviation	Bit Size	Quantity
Interrupt Request Register	IRR	8	1
Interrupt Service Register	ISR	8	1
Interrupt Mask Register	IMR	8	1
Auto Clear Register	ACR	8	1
Status Register	–	8	1
Mode Register	–	8	1
Command Register	–	8	1
Byte Count	–	2	8
Response Memory	–	32	8

Figure 10. Am9519A Register and Memory Summary.

Interrupt Request Register (IRR)

The IRR is eight bits long and is used to recognize and store active transitions on the eight Interrupt Request input lines. A bit in the IRR is set whenever the corresponding IREQ input makes an inactive-to-active transition and meets the minimum active pulse width requirements. IRR bits may also be set by the host processor under program control using two types of commands. This capability allows software initiated interrupts, and is a significant tool for system testing and for sophisticated software designs.

All IRR bits are cleared by a reset. Individual IRR bits are cleared automatically when their interrupts are acknowledged by the host processor. Four types of commands, in addition to reset, allow the host program to clear IRR bits.

The IRR may be read onto the data bus by preselecting it in Mode register bits M5 and M6, followed by a read operation at the data port.

Interrupt Service Register (ISR)

The ISR is eight bits long and is used to store the acknowledge status of individual interrupts. When an $\overline{\text{IACK}}$ pulse arrives, the Am9519A selects the highest priority request that is pending, then clears the associated IRR bit and sets the associated ISR bit. When the ISR bit is programmed for automatic clearing, it is reset by the internal hardware before the end of the acknowledge sequence. When the ISR bit is not programmed for automatic clearing, it must be reset by command from the host processor.

Internally, the Am9519A uses the ISR to erect a "masking fence". When an ISR bit is set and fixed priority mode is selected, only requests of higher priority will cause a new GINT output. Thus, requests from lower priority interrupts (and from new requests associated with the set ISR bit) will be fenced out and ignored until the ISR bit is cleared. In the rotating priority mode, all requests are fenced by an ISR bit that is set, and no new GINT outputs will be generated until the ISR is cleared. When auto clear is specified, no fence is erected since the ISR bit is cleared.

When fixed priority is selected and an unmasked interrupt arrives from a device of higher priority than the current ISR, GINT will go true and the host processor will be interrupted if its interrupt input is enabled. When the new interrupt is acknowledged, the associated higher priority ISR bit is set and the fence moves up to the new level. When the new ISR bit is cleared, the fence will then fall back to the previous ISR level.

The ISR may be read onto the data bus by preselecting it in Mode register bits M5 and M6, followed by a read operation at the data port.

Interrupt Mask Register (IMR)

The IMR is eight bits long and is used to enable/disable the processing of individual interrupts. Only unmasked IRR bits can cause a Group Interrupt to be generated. The IMR does not otherwise affect the operation of the IRR. An IRR bit that is set while masked will cause a GINT when its IMR bit is cleared.

All eight IMR bits may be set, cleared, read or loaded in parallel by the host processor. In addition, individual IMR bits may be set or cleared by command. This allows a control routine to directly enable and disable an individual interrupt without disturbing the other mask bits and without knowledge of their state or the system context.

The IMR polarity is active high for masking; a zero enables the interrupt and a one disables it. The power-on reset and the software reset cause all IMR bits to be set, thus disabling all requests. Care should be taken when disabling a channel. An infinite wait or wrong data will result if the channel being disabled is currently causing GINT to be active. To avoid this, disable CPU interrupts prior to writing to the mask register or gate GINT with $\overline{\text{CS}}$ externally.

Auto Clear Register (ACR)

The ACR is eight bits long and specifies the automatic clearing option for each of the ISR bits. When an auto clear bit is set, the corresponding ISR bit that has been set in an $\overline{\text{IACK}}$ cycle is cleared by the internal hardware before the end of the $\overline{\text{IACK}}$ sequence. When an auto clear bit is not set, the corresponding ISR bit that has been set in an $\overline{\text{IACK}}$ cycle is cleared by command from the host processor.

The auto clear option, when selected, provides two effects. First, it eliminates the need for the associcated interrupt service routine to issue a command to clear the ISR bit. Secondly, it eliminates the masking fence that would otherwise have been erected, allowing lower priority interrupts to cause a new GINT output.

The ACR is loaded in parallel from the data bus by issuing the ACR load preselect command followed by a write into the data port. The ACR may be read onto the data bus by preselecting it in Mode register bits M5 and M6, followed by a read operation at the data port.

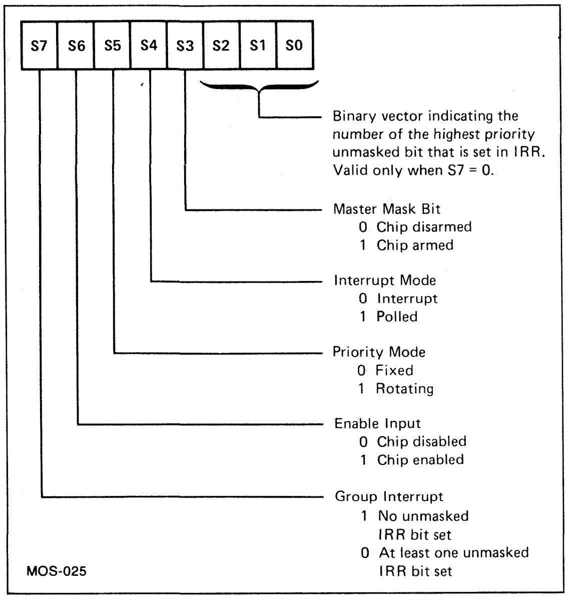


Figure 11. Status Register Bit Assignments.

Status Register

The Status Register is eight bits long and contains information describing the internal state of the Am9519A chip. The Status register is read directly by executing a read operation at the control port. Figure 11 shows the Status bit assignments. The Status register is $C7_H$. After power up, it is not affected by reset.

The high order status bit, S7, reflects the information state of the Group Interrupt signal. Note that the polarity definition of S7 is independent of the defined polarity of GINT (Mode bit M3). Bit S7 remains valid when GINT is disabled by the polled mode option, thus permitting the host processor to check for "interrupts" by reading the Status register.

Status bit S6 reflects the state of the Enable In input signal and is used to indicate, in a multiple chip interrupt structure, which chips in the chain are disabled. When S6 is high, the chip can generate a GINT output and operation of its EO signal proceeds. When S6 is low, no GINT will be generated and EO will be forced low.

Status bit S5 reflects the state of the Priority Mode option, as specified by bit M0 of the Mode register. When S5 is high, rotating priority has been selected. When S5 is low, fixed priority has been selected.

Status bit S4 reflects the state of the Interrupt Mode option, as specified by bit M2 of the Mode register. When S4 is high, the polled mode has been selected and GINT disabled. When S4 is low, the interrupt mode has been selected.

Status bit S3 reflects the state of the Master Mask bit as specified by bit M7 of the Mode register. When S3 is low, the chip has been disarmed and IRR bits that are set will not generate GINT outputs. When S3 is high, the chip has been armed and interrupts can occur.

Status bits S2, S1 and S0 form a three bit field indicating the encoded binary number of the highest priority unmasked bit that is set in the IRR. This field should be considered invalid except when bit S7 of the Status register is low, indicating that at least one unmasked interrupt request is present. The binary coding of the field corresponds to the zero through seven numbering of the IREQ inputs. When more than one unmasked IRR bit is set, the S2, S1, S0 field will indicate the one unfenced request that is the highest priority as determined by the priority mode being used. Thus, the number of the dominant interrupt after all masking, fencing and priority resolution, is encoded into the Status register. This field is quite useful in the polled mode since it can act as a psuedo-vector for the host processor software.

Command Register

The Command Register is eight bits long and is used to store the most recently entered command. It is loaded directly from the data bus by executing a write operation at the control port. Depending on the specific command opcode that is entered, an immediate internal activity may be initiated or the part may be preconditioned for subsequent data bus transfers. The "Command Description" section of this note explains each command operation. The commands are summarized in Figure 17.

Mode Register

The Mode register is eight bits long and controls the operating modes and options of the Am9519A. Figure 12 shows the bit assignments for the Mode register. No single command or interface operation will load all bits of the Mode register in parallel. The five low order bits (M0 through M4) are loaded in parallel directly from the command register. Mode bits M5, M6, and M7 are controlled by separate commands. The Mode

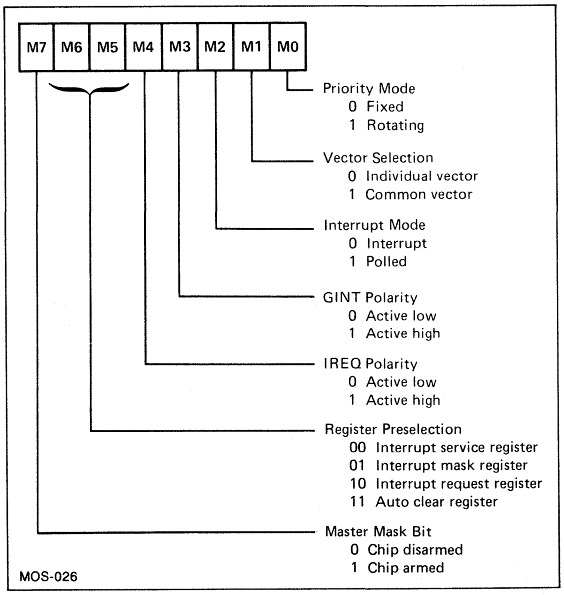


Figure 12. Mode Register Bit Assignments.

register cannot be read out on the data bus. The data in Mode bits M0, M2, and M7 are available as part of the Status register. The Mode register is cleared by a software reset or a power-up reset. The "Operating Options" section of this note describes the detailed functions associated with each Mode bit.

Information Transfers

Figure 13 summarizes the control signal configurations for all information transfers on the Am9519A data bus. The interface control logic assumes the following conventions:

1. $\overline{RD}$ and $\overline{WR}$ are never active at the same time.
2. $\overline{RD}$, $\overline{WR}$ and C/$\overline{D}$ are ignored unless $\overline{CS}$ is low.
3. $\overline{IACK}$ will not go low when the chip is being selected by the host.

Control Input					
$\overline{CS}$	C/$\overline{D}$	$\overline{RD}$	$\overline{WR}$	$\overline{IACK}$	Data Bus Operation
0	0	0	1	1	Transfer contents of data register specified by Mode bits M5, M6, to data bus. ISR, IMR, IRR, ACR
0	0	1	0	1	Transfer contents of data bus to data register specified by Command register.
0	1	0	1	1	Transfer contents of Status register to data bus.
0	1	1	0	1	Transfer contents of data bus to Command register.
1	X	X	X	0	Transfer contents of selected response memory location to data bus.
1	X	X	X	1	No information transferred; data bus outputs off.

Figure 13. Summary of Data Bus Transfers.

When $\overline{IACK}$ is low, internal logic disables the $\overline{CS}$ input. This prevents signals on the address bus from inadvertently selecting the chip.

The host processor may read the Status register directly by simply performing a read operation with the control port selected. When a read is executed at the data port, the information transferred will be the contents of the ISR, IMR, IRR or ACR, depending on the state of Mode register bits M5 and M6.

The host processor may write directly into the command register by simply performing a write operation with the control port selected. When a write is executed into the data port, the contents of the data bus will be transferred to the ACR, IMR or response memory, depending on which command preceded the data write. Note that Mode bits M5 and M6 do not preselect the location for data write operations; only a command can do so.

When the response memory preselect command is issued, it should be followed by an appropriate number of data write operations to load 1, 2, 3, or 4 bytes of response information. If more than four bytes are written, the response memory addressing will "wrap around" and overwrite the information already entered. Response bytes are output by the Am9519A during IACK operations in the same order they were entered. Entry of response information into each new level must be preceded by a new response memory preselect command. Every channel must have at least one response byte written into the response memory, even if that channel is unused.

Interrupt Acknowledge operations are initiated by the host processor and occur following recognition of a GINT signal from the Am9519A. When $\overline{IACK}$ signal arrives, the interrupt system selects the highest priority unmasked pending interrupt request and then outputs a response byte associated with the selected interrupt. The selection process and the access of the response byte will take a variable amount of time that depends on several parameters, including:

1. the operating temperature,
2. the actual internal logic delays,
3. the number of Am9519A chips cascaded together,
4. the priority level of the interrupt being acknowledged,
5. the Mode register operating options,
6. the byte position within the response sequence.

The worst-case $\overline{IACK}$ pulse widths must be long enough to accommodate the accumulated delays that can occur in large interrupt systems operating in worst-case situations. Yet small systems operating under typical conditions will require only relatively narrow $\overline{IACK}$ pulses. The $\overline{PAUSE}$ output from the Am9519A is designed to provide interactive feedback to the host processor so that the $\overline{IACK}$ pulse width may be adaptively adjusted to meet the requirements of the actual interrupt being processed. $\overline{PAUSE}$ will go low following the falling edge of IACK, and will return high when IACK is no longer required.

During the first $\overline{IACK}$ of a complete acknowledge sequence, the $\overline{PAUSE}$ output remains low until the highest priority interrupt has been selected and the $\overline{RIP}$ output goes low. On subsequent $\overline{IACK}$ pulses for additional responses bytes associated with the same interrupt ($\overline{RIP}$ still low), $\overline{PAUSE}$ will remain high. The Am9519A expects the first $\overline{IACK}$ input to remain low at least until the $\overline{PAUSE}$ output goes high. Subsequent $\overline{IACK}$ inputs should meet the specified input pulse width requirements as called out in the data sheet.

Operating Options

The Mode register bits are used to establish the operating modes and conditions for the many functional features of the Am9519A. The Mode register allows the host processor to personalize the interrupt system for the application at hand.

Priority Selection

Bit M0 in the Mode register specifies the priority operating mode for the Am9519A. When M0=0, fixed priority is selected and the

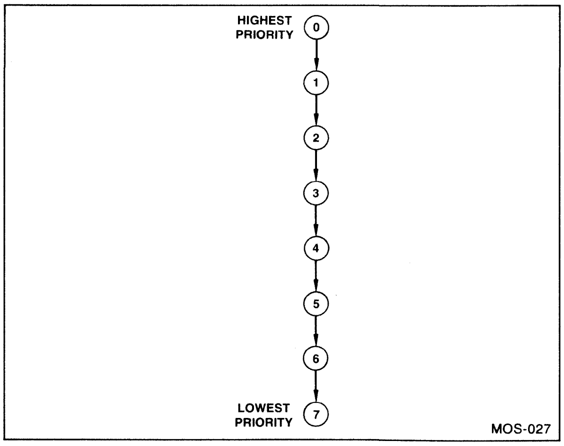


Figure 14. Fixed Priority Mode.

eight IREQ inputs are assigned a priority based on their physical location at the chip interface. IREQ0 has the highest priority and IREQ7 has the lowest. See Figure 14.

Priority is not resolved until the host processor initiates the interrupt acknowledge sequence. Thus, for example, an IREQ5 input may cause a GINT output to the host, but if an input on IREQ2 arrives before the falling edge of $\overline{\text{IACK}}$, then it is IREQ2 that will be selected and serviced. Notice that inherent in the fixed priority structure is the possibility that IREQ5 might never be selected and serviced as long as there are higher priority interrupts pending. IREQ2 could end up being serviced many times before IREQ5 is acknowledged. In many systems this is an appropriate method for handling the interrupting devices. Where circumstances permit, the masking capability of the Am9519A can be used by the host processor to modify the effective priority structure, perhaps by masking out recently serviced high priority devices, thus allowing lower priority inputs to be recognized.

Alternatively, where the eight interrupts have similar priority and service bandwidth requirements, the rotating priority mode may be selected (Mode register bit M0=1). As shown in Figure 15 the relative priorities remain the same as in the fixed mode; that is, IREQ2 is higher than IREQ3 which is

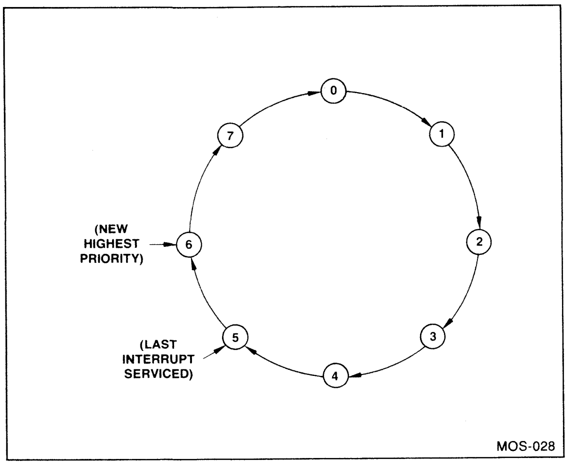


Figure 15. Rotating Priority Mode.

higher than IREQ4, etc. However, in rotating priority mode, the lowest priority position in the circular chain is assigned by the hardware to the most recently serviced interrupt.

The example illustrated in Figure 15 assumes that IREQ5 has just finished being serviced and has therefore been assigned the lowest priority. Thus, IREQ6 occupies the new highest priority position, IREQ7 next-to-highest, etc. If two new interrupts then arrive at level 1 and level 4, IREQ1 will be selected and serviced, and will become the lowest priority. IREQ4 will then be acknowledged unless an active input on IREQ2 or IREQ3 has arrived in the meantime.

This rotating priority scheme prevents any request from dominating the system. It assures that an input will not have to wait for more than seven other service cycles before being acknowledged. Rotation occurs when the ISR bit of the presently selected interrupt is cleared.

In the rotating priority mode, inputs other than the one currently being serviced are fenced out and will not cause interrupts until the ISR bit is cleared. Thus, only one bit at a time will be set in the ISR. Care should be used when selecting the rotating mode to keep from doing so at a time when more than one ISR is set.

Vectoring

Bit M1 of the Mode register specifies the vectoring option. When M1=0 the individual vector mode is selected and each interrupt is associated with its own unique four-byte location in the response memory. When M1=1, on the other hand, the common vector mode is selected and all response information is supplied from the location associated with IREQ0, no matter which request is being acknowledged. This operating option will be useful in situations where several similar devices share a common service routine and direct individual device identification is not important. This may be true simply because of the nature of the peripheral/system interaction, or it may be a transient system condition that only uses the common vector option temporarily, perhaps to save the overhead involved in filling the response memory twice.

Polled Mode

Bit 2 of the Mode register allows the system to disable the GINT output. When M2=0 the interrupt mode is selected with the GINT output enabled. This might be considered the "normal" interrupt mode and makes full use of the interrupt control and management capabilities of the Am9519A. When M2=1 the polled mode is selected which prevents the GINT output from going true by forcing it to its inactive state. In this condition, since no interrupts are supplied to the host processor, there will usually not be any $\overline{\text{IACK}}$ pulses returned to the Am9519A. Consequently, ISR bits are not set, fences are not erected and IRR bits will not be automatically cleared. In the polled mode the host processor may read the Status register to determine if a request is pending and which request has the highest priority. IRR bits may be cleared by the host software. When the polled option is selected, the EI input is connected directly to the EO output thus functionally removing the polled chip from the external priority hierarchy.

Effectively, the polled mode of operation bypasses the hardware interrupt, inter-chip priority resolution, vectoring and fencing functions of the Am9519A. What remains is the request latching, masking and intra-chip priority resolution.

GINT Polarity

Bit 3 of the Mode register specifies the sense of the GINT output. When M3=0, Group Interrupt is selected as active low ($\overline{\text{GINT}}$) and becomes an open drain output. This allows simple wired-or connections to other similar Am9519A outputs as well as to other sources of interrupts, and matches the polarity required by many processors. When M3=1, Group Interrupt is selected as active high (GINT) and becomes a two-state push-pull output, simplifying the interface to processors with active high interrupt inputs.

IREQ Polarity

Bit 4 of the Mode register specifies the sense of the IREQ inputs. When M4=0 the Interrupt Request signals are selected as active low ($\overline{\text{IREQ}}$) and a negative-going transition is required to set the IRR. When M4=1 the Interrupt Request signals are selected as active high (IREQ) and a positive-going transition is required to set the IRR. This sense option helps simplify the interface to interrupting devices.

Register Preselection

Bits 5 and 6 of the Mode register specify the internal data register that will be output by the Am9519A on any read operation at the data port ($\overline{\text{CS}}$=0, $\overline{\text{RD}}$=0, C/$\overline{\text{D}}$=0). These bits do not affect destinations for write operations. The four registers available for reading are the IRR, ISR, IMR and ACR. Preselect coding for each register is shown in Figure 12. The preselection remains in effect for all data read transfers until the contents of M5 and M6 are changed.

The ability to examine these important operating registers, combined with the information available in the Status register, provides significant insight into the internal conditions of the Am9519A. This allows the host processor not only enhanced dynamic operating flexibility, but also access to important diagnostic/testing/debugging information.

Master Mask

Bit 7 of the Mode register specifies the armed status of the Am9519A by way of the Master Mask control bit. When M7=0 the chip is disarmed just as if all eight bits in the IMR had been set. That is, IREQ inputs will be accepted and latched but will not cause GINT outputs to the host. In addition, the EO output is brought low, disabling any lower priority chips that may be attached. When M7=1, the chip is armed and any active unmasked interrupt inputs will be able to cause GINT outputs to the host processor.

The Master Mask capability permits the host system to disarm a chip and prevent processing of the interrupts without disturbing the contents of the IMR. Thus when the chip is re-armed, the old IMR conditions remain in effect and need not be reloaded. Note that a single command to the Master Mask bit of the highest priority interrupt chip is able to shut down the complete interrupt system, no matter how large.

Mode Reset

When a power-up or software reset occurs, the Mode register is cleared to all zeros. This means that after reset the following Mode register operating options will be in effect:

Fixed priority
Individual vectoring
Interrupt (non-polled) operation
GINT active low sense
IREQ active low sense
ISR preselected for reading
Chip disarmed by Master Mask

Operating Sequence

The management of interrupts by the Am9519A is illustrated below with a description of a fairly typical sequence of events. The Am9519A has already been initialized and enabled and is ready to run. The host processor has enabled its internal interrupt structure.

1. One (or more) of the IREQ inputs becomes active indicating that service is desired.
2. The requests are captured and latched in the IRR asynchronously. The latching action of the IRR cannot be disabled and active requests will always be stored unless a previous request at the same IRR bit has not been cleared.
3. If the active IRR bit is masked by the corresponding bit in the IMR, no further action takes place. When the IRR bit is not masked, an active Group Interrupt output will be generated if the Am9519A is not in its polled mode.
4. The GINT output from the Am9519A is used by the host processor as an interrupt input. When GINT is recognized by the host, it normally will complete the execution of its current instruction and will then execute some form of interrupt acknowledge sequence instead of the next program instruction. As part of the acknowledge cycle, the processor usually automatically disables its interrupt input. The Am9519A expects to receive one or more $\overline{\text{IACK}}$ signals from the processor during the acknowledge sequence.
5. When $\overline{\text{IACK}}$ is received, the Am9519A brings its $\overline{\text{PAUSE}}$ output low and begins selection of the highest priority unmasked active IRR bit. All interrupts that have become active before the falling edge of $\overline{\text{IACK}}$ are considered. When selection is complete, the $\overline{\text{RIP}}$ output is pulled low by the Am9519A and the contents of the first byte in the response memory associated with the selected request is accessed. $\overline{\text{PAUSE}}$ stays low until $\overline{\text{RIP}}$ goes low. RIP stays low until the last byte of the response has been transferred.
6. After $\overline{\text{PAUSE}}$ goes high, the host processor accepts the response byte on the data bus and brings the $\overline{\text{IACK}}$ line high. If another byte of response is required, another $\overline{\text{IACK}}$ pulse is output and is used by the Am9519A to access the next byte.
7. In parallel with the transfer of the first response byte, the Am9519A automatically clears the selected IRR bit and automatically sets the selected ISR bit. If the auto clear function is not in force for the selected interrupt, the ISR bit will cause a masking fence to be erected and GINT will be disabled until a higher priority interrupt arrives or until the ISR bit is cleared. The interrupt service routine will usually clear the ISR bit, often near the end of the routine.
8. If a higher priority request arrives while the current request is being serviced, and if the fixed priority mode is in effect, then GINT will be output again by the Am9519A. The GINT signal will be recognized by the host processor only if the host has enabled its interrupt input. If this new request is acknowledged, the Am9519A will clear the corresponding IRR bit and set the corresponding ISR bit.
9. When the host processor has completed all interrupt service activity to satisfy the interrupting devices, it will normally clear the remaining ISR bit, if any, enable its internal interrupt system, if it has not already done so, and then return to the main program.

COMMAND DESCRIPTIONS

The Am9519A command set allows the host processor to customize and alter the interrupt operating modes and features for particular applications, to initialize and update the response locations, and to manipulate the internal controlling bit set during interrupt servicing. Commands are entered from the data bus directly into the Command register by writing into the Am9519A control port ($\overline{CS}$=0, $\overline{WR}$=0, C/$\overline{D}$=1). All the available commands are described below and are summarized in Figure 17. In the binary coding of the commands, "X" indicates a do-not-care bit position.

RESET

Coding:

C7	C6	C5	C4	C3	C2	C1	C0
0	0	0	0	0	0	0	0

Description: The Reset command allows the host processor to establish a known internal condition. The response memory and byte count registers are not affected by the software reset. The IMR is set to all ones. The ISR, IRR, ACR and Mode registers are cleared to all zeros.

CLEAR IRR AND IMR

Coding:

C7	C6	C5	C4	C3	C2	C1	C0
0	0	0	1	0	X	X	X

Description: All bits in the IMR and all bits in the IRR are cleared at the same time. Thus all interrupts are enabled and the previous history of all IREQ transitions is forgotten. If GINT was active when the command was entered, it will go inactive.

CLEAR SINGLE IMR AND IRR BIT

Coding:

C7	C6	C5	C4	C3	C2	C1	C0
0	0	0	1	1	B2	B1	B0

Description: The same single bit position is cleared in both the IMR and the IRR. Other bits are not changed. If the specified IRR bit was generating an active interrupt output, GINT may go inactive upon entry of the command. The bit position cleared is specified by the B2, B1, B0 field as shown in Figure 16.

CLEAR IMR

Coding:

C7	C6	C5	C4	C3	C2	C1	C0
0	0	1	0	0	X	X	X

Description: All bits in the IMR are cleared to zeros. All IRR bits will therefore be unmasked and any IRR bits that had been set will be able to cause an active GINT output after the command is entered.

CLEAR SINGLE IMR BIT

Coding:

C7	C6	C5	C4	C3	C2	C1	C0
0	0	1	0	1	B2	B1	B0

Description: A single bit in the IMR is cleared. Other bits are not changed. If the corresponding bit in the IRR was set, it will be unmasked and will be able to cause an active GINT after entry of the command. The IMR bit cleared is specified by the B2, B1, B0 field as shown in Figure 16.

SET IMR

Coding:

C7	C6	C5	C4	C3	C2	C1	C0
0	0	1	1	0	X	X	X

Description: All bits in the IMR are set to ones. All IRR bits will therefore be masked and unable to generate an active GINT. If GINT had been active, it will go inactive after the command is entered.

SET SINGLE IMR BIT

Coding:

C7	C6	C5	C4	C3	C2	C1	C0
0	0	1	1	1	B2	B1	B0

Description: A single bit in the IMR is set. Other bits are not changed. If the corresponding bit in the IRR was active and generating a GINT output, GINT will become inactive after the command is entered. The IMR bit set is specified by the B2, B1, B0 field as shown in Figure 16.

CLEAR IRR

Coding:

C7	C6	C5	C4	C3	C2	C1	C0
0	1	0	0	0	X	X	X

Description: All bits in the IRR are cleared to zeros. GINT will become inactive. New transitions on the IREQ inputs will be necessary to cause an interrupt.

CLEAR SINGLE IRR BIT

Coding:

C7	C6	C5	C4	C3	C2	C1	C0
0	1	0	0	1	B2	B1	B0

Description: A single bit in the IRR is cleared to zero. It will not cause an active GINT until it is set. The IRR bit cleared is specified by the B2, B1, B0 field as shown in Figure 16.

SET IRR

Coding:

C7	C6	C5	C4	C3	C2	C1	C0
0	1	0	1	0	X	X	X

Description: All bits in the IRR are set to ones. Any that are unmasked will be able to cause an active GINT output. This command allows the host CPU to initiate eight interrupts in parallel.

SET SINGLE IRR BIT

Coding:

C7	C6	C5	C4	C3	C2	C1	C0
0	1	0	1	1	B2	B1	B0

Description: A single bit in the IRR is set to a one. If it is unmasked it will be able to generate an active GINT. This command allows the host processor to simulate with software the arrival of a hardware interrupt request. It also gives the software access to the hardware priority resolution, masking and control features of the Am9519A. The bit set is specified by the B2, B1, B0 field as shown in Figure 16.

CLEAR HIGHEST PRIORITY ISR BIT

Coding:

C7	C6	C5	C4	C3	C2	C1	C0
0	1	1	0	X	X	X	X

Description: A single bit in the ISR is cleared to zero. If only one bit was set, that is the one cleared. If more than one bit was set, this command clears the one with the highest priority. This command is useful in software contexts where the service routine does not know which device is being serviced. It should be used with caution since the highest priority ISR bit may not really be the bit intended. When using the auto clear option on some interrupts and/or when a subroutine nesting hierarchy is not priority driven, the highest priority ISR bit may not correspond to the one being serviced.

CLEAR ISR

Coding:

C7	C6	C5	C4	C3	C2	C1	C0
0	1	1	1	0	X	X	X

Description: All bits in the ISR are cleared to zeros. Mask fencing is eliminated.

CLEAR SINGLE ISR BIT

Coding:

C7	C6	C5	C4	C3	C2	C1	C0
0	1	1	1	1	B2	B1	B0

Description: A single bit in the ISR is cleared to zero. If the bit was already cleared, no effective operation takes place. The bit cleared is specified by the B2, B1, B0 field as shown in Figure 16. This will be the most useful command for service routines to use in managing the ISR without the help of the auto-clear option.

LOAD MODE BITS M0 THROUGH M4

Coding:

C7	C6	C5	C4	C3	C2	C1	C0
1	0	0	M4	M3	M2	M1	M0

Description: The five low order bits of the Command register are transferred into the five low order bits of the Mode register. This command controls all of the Mode options except the master mask and the register preselection.

CONTROL MODE BITS M5, M6, M7

Coding:

C7	C6	C5	C4	C3	C2	C1	C0
1	0	1	0	M6	M5	N1	N0

Description: The M6, M5 field in the command is loaded into the M6, M5 locations in the Mode register. This field controls the register preselection bits in the Mode register. The N1, N0 field in the command controls Mode bit M7 (Master Mask) and is decoded as follows:

N1	N0	
0	0	No change to M7
0	1	Set M7
1	0	Clear M7
1	1	(Illegal)

Thus, this command may be considered as three distinct commands, depending on the coding of N1 and N0:

1. Load M5, M6 only
2. Load M5, M6 and set M7
3. Load M5, M6 and clear M7

The Command Summary in Figure 17 lists all three versions.

PRESELECT IMR FOR WRITING

Coding:

C7	C6	C5	C4	C3	C2	C1	C0
1	0	1	1	X	X	X	X

Description: The IMR is targeted to be loaded from the data bus when the next write operation occurs at the data port. All subsequent data write operations will also load the IMR until a different command is entered. Read operations may be successfully inserted between the entry of this command and the subsequent writing of data into the IMR. The Mode register is not affected by this command.

PRESELECT ACR FOR WRITING

Coding:

C7	C6	C5	C4	C3	C2	C1	C0
1	1	0	0	X	X	X	X

Description: The ACR is targeted to be loaded from the data bus when the next write operation occurs at the data port. All subsequent data write operations will also load the ACR until a different command is entered. Read operations may be successfully inserted between the entry of this command and the subsequent writing of data into the ACR. The Mode register is not affected by this command.

PRESELECT RESPONSE MEMORY FOR WRITING

Coding:

C7	C6	C5	C4	C3	C2	C1	C0
1	1	1	BY1	BY0	L2	L1	L0

Description: One level in the response memory is targeted for loading from the data bus by subsequent data write operations. The byte count register for that level is loaded from the BY1, BY0 field in the command. The L2, L1, L0 field specifies which of the eight response levels is being selected. This command should be followed by one to four data write operations to load response bytes. Field coding:

BY1	BY0	Count
0	0	1
0	1	2
1	0	3
1	1	4

L2	L1	L0	Level
0	0	0	0
0	0	1	1
0	1	0	2
0	1	1	3
1	0	0	4
1	0	1	5
1	1	0	6
1	1	1	7

The byte count controls the number of bytes entered into the response memory as well as the number of bytes read from the memory by $\overline{\text{IACK}}$ pulses. Response bytes are output by the Am9519A in the same order they were entered.

B2	B1	B0	Bit Specified
0	0	0	0
0	0	1	1
0	1	0	2
0	1	1	3
1	0	0	4
1	0	1	5
1	1	0	6
1	1	1	7

Figure 16. Coding of B2, B1, B0 Field of Commands.

Command Code								Command Description	
7	6	5	4	3	2	1	0		
0	0	0	0	0	0	0	0	Reset (Clear IRR, ISR, ACR; set IMR)	
0	0	0	1	0	X	X	X	Clear all IRR and all IMR bits	
0	0	0	1	1	B2	B1	B0	Clear IRR and IMR bit specified by B2, B1, B0	
0	0	1	0	0	X	X	X	Clear all IMR bits (Enable)	IMR
0	0	1	0	1	B2	B1	B0	Clear IMR bit specified by B2, B1, B0	
0	0	1	1	0	X	X	X	Set all IMR bits (Disable)	
0	0	1	1	1	B2	B1	B0	Set IMR bit specified by B2, B1, B0	
0	1	0	0	0	X	X	X	Clear all IRR bits	IRR
0	1	0	0	1	B2	B1	B0	Clear IRR bit specified by B2, B1, B0	
0	1	0	1	0	X	X	X	Set all IRR bits	
0	1	0	1	1	B2	B1	B0	Set IRR bit specified by B2, B1, B0	
0	1	1	0	X	X	X	X	Clear highest priority ISR bit	ISR
0	1	1	1	0	X	X	X	Clear all ISR bits	
0	1	1	1	1	B2	B1	B0	Clear ISR bit specified by B2, B1, B0	
1	0	0	M4	M3	M2	M1	M0	Load Mode register bits 0–4 with specified pattern	
1	0	1	0	M6	M5	0	0	Load Mode register bits 5, 6 with specified pattern	
1	0	1	0	M6	M5	0	1	Load Mode register bits 5, 6 and set Mode bit 7	
1	0	1	0	M6	M5	1	0	Load Mode register bits 5, 6 and clear Mode bit 7	
1	0	1	1	X	X	X	X	Preselect IMR for subsequent loading from data bus	
1	1	0	0	X	X	X	X	Preselect ACR for subsequent loading from data bus	
1	1	1	BY1	BY0	L2	L1	L0	Load BY1, BY0 into byte count register and preselect response memory level specified by L2, L1, L0 for subsequent loading from data bus	

Figure 17. Am9519A Command Summary.

SYSTEM INTERFACE

Expansion

Several Am9519A chips may be cascaded to expand the number of interrupts than can be handled by the system. A two-chip configuration is shown connected to an 8080A/9080A microprocessor in Figure 18. In general, expansion past a single Am9519A will require simply an added Chip Select signal for each extra chip, and perhaps an inverter for the GINT signal if the processor interrupt input is active-high. The GINT, PAUSE, and RIP signals are all designed to be wire-OR'ed in expanded systems.

Priority management in expanded systems is controlled by the Enable In, Enable Out and Response In Process signals. Figure 19 shows the basic interconnections for an example interrupt system that can accept up to 40 interrupts, using five Am9519A chips. Notice that IACK is wired in parallel to all five circuits, and that the GINT, RIP and PAUSE lines are respectively tied together. The three pullup resistors are used to establish the high logic levels for the open-drain outputs. Enable In of the first chip (A) is allowed to float, or may be tied high. Each Enable Out signal is connected to the next lower level Enable In input. Each chip accepts eight IREQ inputs; for purposes of this example it is assumed that an active interrupt arrives at chip D in the chain.

Figure 20 shows the timing relationships for the configuration of Figure 19. When the IREQ arrives, a GINT output is generated by chip D and is used to interrupt the host processor. When the host returns an IACK pulse, all the EO lines are brought low in parallel. PAUSE also goes low, and is used to extend the IACK pulse.

After the fall of IACK, all chips wait until a brief internal delay elapses and then examine EI. If EI is low, internal activity is suspended until EI goes high. If EI is high, then the internal circuitry is checked to see if an unmasked request is pending. If so, RIP is brought low, PAUSE is brought high, EO is kept low, and the first response byte is output on the data bus. In this example, there is no request in chip A and therefore the EO(A) line is brought high. This then allows chip B to see if it has an unmasked request waiting for service. If not, EO(B) goes high also and, with no interrupts at C, EO(C) goes high, driving EI(D) high. Since chip D finds a waiting request, it does not bring EO(D) high but it does bring RIP low. When RIP goes low it allows all the PAUSE outputs to switch high which permits the termination of the IACK pulse.

It can be seen, then, that the PAUSE output will automatically adjust the position of its rising edge to accommodate the exact functional and operational conditions that occur for each particular IACK cycle. For larger systems, like that in Figure 19, operating at high temperatures with slow versions of the Am9519A and servicing low priority interrupts, the processor delay caused by PAUSE may be quite long and a few processor wait cycles may be required to extend the IACK pulse. On the other hand, when a system like Figure 18 is running at typical room temperatures with typical parts and the interrupt is a high priority one, the PAUSE output width will be quite narrow and no wait cycles will be necessary.

The RIP output serves two basic functions within the interrupt system. First, its falling edge informs the other connected chips that an interrupt request has been selected and PAUSE may, therefore, be released. Secondly, as long as RIP is low, only the single chip that is pulling RIP down is allowed to respond to IACK inputs. RIP stays low until all response bytes for the selected interrupt have been transferred.

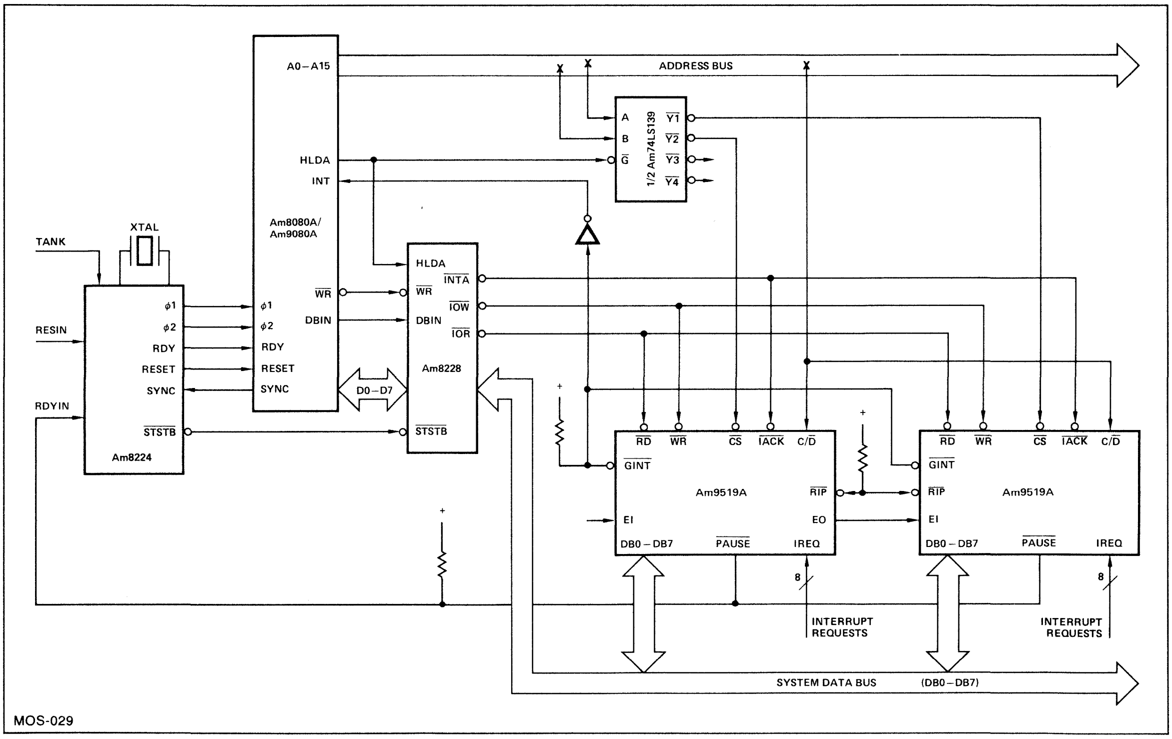


Figure 18. 16 Interrupt Configuration with 8080A/9080A.

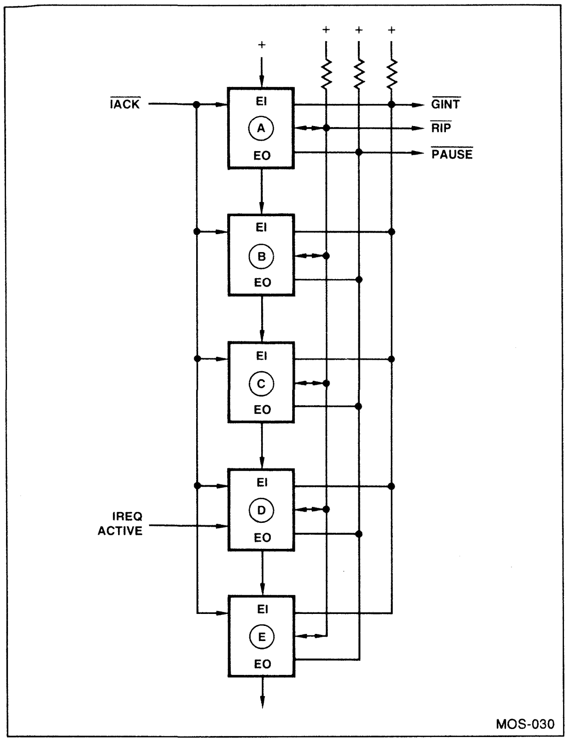


Figure 19. Five-Chip Expansion Example.

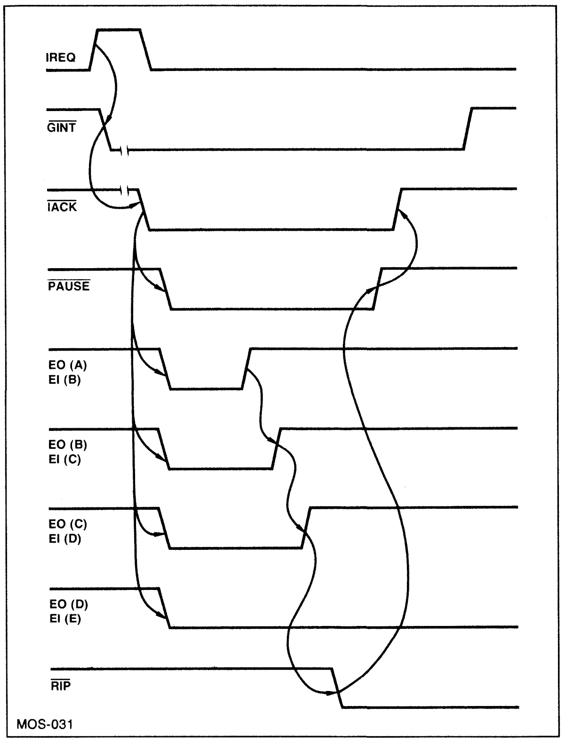


Figure 20. Inter-Chip Priority Resolution.

Assume that a new interrupt arrives at chip B in Figure 19 during the time that the first byte of a multibyte response for the interrupt at chip D is being transferred. Without the $\overline{\text{RIP}}$ signal there would be confusion when the second $\overline{\text{IACK}}$ pulse arrives. Both chips might try to send out response bytes since the interrupt at chip B is a higher priority, yet chip D is in the midst of a response sequence. With $\overline{\text{RIP}}$ present, however, no problem arises. Chip D pulls $\overline{\text{RIP}}$ low when it is selected and keeps $\overline{\text{RIP}}$ low until its response is complete. Chip B treats $\overline{\text{RIP}}$ as an input and will not respond to $\overline{\text{IACK}}$ until $\overline{\text{RIP}}$ goes high.

Initialization and Support

Before the Am9519A can perform useful work, it must be initialized to customize it for a particular application and to load it with appropriate data values. During active operation, control options may be changed and response data may be modified. Because of the many ways it might be used, the Am9519A can be programmed using many different approaches. The following sequence description shows only one of several possible methods for constructing a basic service routine:

1. Disable processor interrupts.
2. Execute software reset at Am9519A
3. Transfer commands and response data from a control table into the Am9519A
4. Transfer operating options into the Mode register.
5. Transfer the operating Mask conditions into the IMR.
6. Clear the IRR.
7. Clear Master Mask.
8. Enable processor interrupts.
9. Return.

Figure 21 shows an example listing for such a routine using 8080A/9080A coding. Several assumptions are made about the hardware and software environment in which the routine will function:

1. When the routine is entered, register pair H, L contains the address of the first location of a control table, and register B contains a count that indicates how many entries the table contains.
2. One Am9519A is in the system. Its data port is decoded by the hardware as hex I/O address C2. Its control port is hex I/O address C3.
3. Only the first five interrupts will be in use by the main program. The others will be used later to support other processes.
4. Main program options: Fixed priority, Individual vectoring, Interrupt mode, GINT active high, IREQ active low, IRR selected for reading, Auto clear not used.

The control table is an important part of the routine and contains command information as well as the response data itself. The table consists of up to eight entries, each up to five bytes long, with all entries the same length. The first byte of each entry contains the response memory preselect command code with fields for the BY1, BY0 byte count and the L2, L1, L0 level pointer. The next one to four bytes of each entry contain the data loaded into the response memory. In this example the table has entries of four bytes each and is illustrated in Figure 22.

This type of table organization may contain extra bytes, but it compensates for this by allowing a brief, simple program to handle it. The table is fairly general and allows any length response to be programmed independently for each interrupt. It

```
LOC   OBJ          SEQ        SOURCE STATEMENT

                    0 ;
                    1 ;        * * * * * * * * * * * * * * * * * * *
                    2 ;        *                                   *
                    3 ;        *          EXAMPLE PROGRAM          *
                    4 ;        *  FOR CONTROL AND INITIALIZATION   *
                    5 ;        *          OF THE AM9519A           *
                    6 ;        *  UNIVERSAL INTERRUPT CONTROLLER   *
                    7 ;        *                                   *
                    8 ;        * * * * * * * * * * * * * * * * * * *
                    9 ;
                   10 ;
                   11 ;
3000               12          ORG 3000H
                   13                              ;
00C3               14 CPORT    EQU 0C3H            ; CONTROL PORT ADDRESS.
00C2               15 DPORT    EQU 0C2H            ; DATA PORT ADDRESS.
00B0               16 LMASK    EQU 10110000B       ; LOAD MASK COMMAND.
00E0               17 MASK1    EQU 11100000B       ; MASK VALUE TO ENABLE THE FIRST
                   18                              ;   FIVE INTERRUPTS.
0088               19 MODE1    EQU 10001000B       ; MODE COMMAND FOR 'NORMAL'
                   20                              ;   OPERATION. (M0 - M4 ONLY)
00A9               21 MODE2    EQU 10101001B       ; MODE COMMAND TO CLEAR MASTER
                   22                              ;   MASK AND PRESELECT IRR.
0040               23 CLIRR    EQU 01000000B       ; CLEAR IRR COMMAND.
                   24                              ;
                   25                              ;
3000 F3            26 ENTRY:   DI                  ; DISABLE CPU INTERRUPTS
3001 3E00          27          MVI A,00000000B     ; GET SOFTWARE RESET COMMAND AND
3003 D3C3          28          OUT CPORT           ; SEND TO CONTROL PORT.
                   29                              ;
3005 7E            30 AAA:     MOV A,M             ; GET CONTROL BYTE FROM TABLE &
3006 D3C3          31          OUT CPORT           ; SEND TO CONTROL PORT.
3008 23            32          INX H               ; INCREMENT TABLE POINTER.
3009 0E03          33          MVI C,3             ; INITIALIZE VECTOR BYTE COUNT.
300B 7E            34 BBB:     MOV A,M             ; GET VECTOR BYTE AND
300C D3C2          35          OUT DPORT           ; SEND TO DATA PORT.
300E 23            36          INX H               ; POINT TO NEXT TABLE BYTE.
300F 0D            37          DCR C               ; DECREMENT BYTE COUNT.
3010 C20B30        38          JNZ BBB             ; ENTRY DONE?  NO:BACK TO BBB.
3013 05            39          DCR B               ; YES:DECREMENT ENTRY COUNTER.
3014 C20530        40          JNZ AAA             ; TABLE DONE? NO:BACK TO AAA.
                   41                              ;
3017 3E88          42          MVI A,MODE1         ; YES: PROCEED WITH MODE BYTE.
3019 D3C3          43          OUT CPORT           ; SEND MODE TO CONTROL PORT.
301B 3EB0          44          MVI A,LMASK         ; GET MASK LOAD COMMAND AND
301D D3C3          45          OUT CPORT           ; SEND TO CONTROL PORT.
301F 3EE0          46          MVI A.MASK1         ; GET OPERATING MASK AND
3021 D3C2          47          OUT DPORT           ; SEND TO DATA PORT.
3023 3E40          48          MVI A,CLIRR         ; GET CLEAR IRR COMMAND AND
3025 D3C3          49          OUT CPORT           ; SEND TO CONTROL PORT.
3027 3EA9          50          MVI A,MODE2         ; GET CLEAR MASTER MASK COMMAND
3029 D3C3          51          OUT CPORT           ; AND SEND TO CONTROL PORT.
302B FB            52          EI                  ; ENABLE CPU INTERRUPTS.
302C C9            53          RET                 ; RETURN TO CALLING PROGRAM.
                   54                              ;
                   55          END
```

Figure 21. Example Routine.

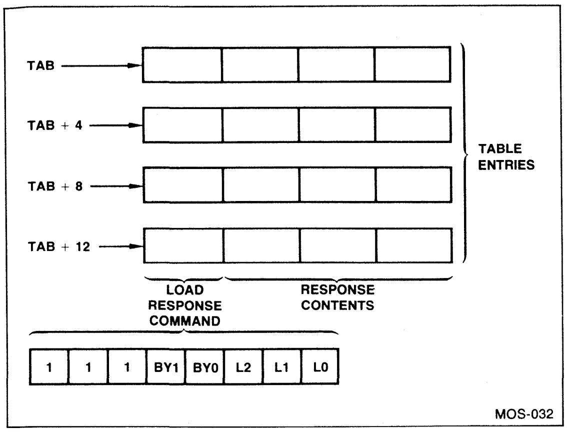


Figure 22. Example Control Table.

also allows any number of response locations to be updated in any order. The program driving the table simply assumes that every response level receives the same number of bytes as the level with the longest response.

Other table organizations are also possible. A more general table could contain the IMR value to be used, the ACR value, the table byte length, the operating mode values, etc. As more of the variable control information is added to the table, the software routine becomes more general and can be used not only for initialization, but for operational changes as well. Then there might be several tables in memory with an address supplied to the routine that points to the controlling table to be used. Note that the calling program can use just portions of an existing table if desired, simply by controlling the contents of the machine registers when the routine is entered.

Another approach is to omit the byte count/level command code from the table and compute its value in the driving routine. This may be especially appropriate when all the response entries are the same length and contiguous levels are being filled. The BY1, BY0 field need not change then, and a simple increment instruction will generate the proper command coding by changing the L2, L1, L0 field. To minimize the table length, which might become an important consideration for larger systems with many more interrupts, it is also possible to use the byte count to control the number of bytes transferred into each memory level.

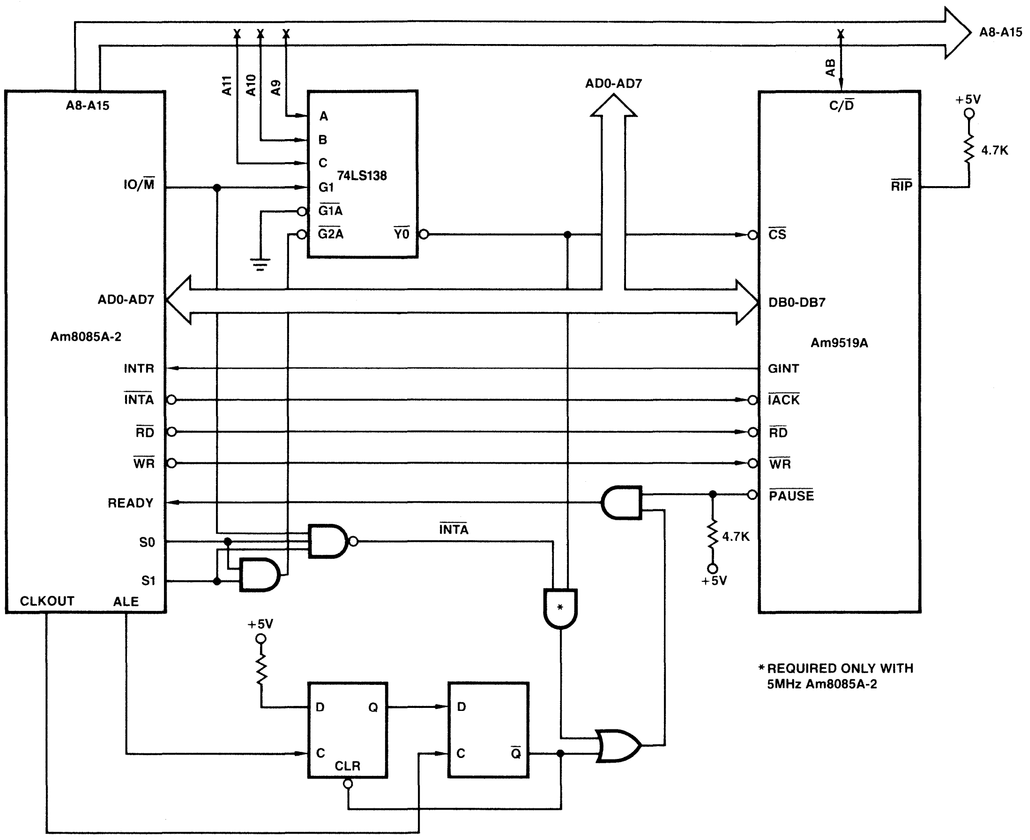
The Am9519A offers new levels of versatility and sophistication for interrupt systems. It represents interesting opportunities for both hardware and software engineers to enhance new designs and to take advantage of the features now available.

Other Applications

Since its introduction, the Am9519A has become popular in a variety of system environments. The following applications illustrate a few of them.

Am9519A to Am8085 Interface

CIRCUIT DIAGRAM:



*REQUIRED ONLY WITH 5MHz Am8085A-2

Figure 23. MOS-564

DESCRIPTION OF INTERFACE:

The two D-type flip-flops in conjunction with S0, S1 and IO/M̅ insert a wait state to the Am8085A whenever the microprocessor acknowledges an interrupt. The circuitry is required since the Am8085A samples a READY (wait) signal on the rising edge of CLK OUT during T2, and ready requires a 100nsec minimum set up time prior to the rising edge of T2. The 3-input NAND gate predecodes an I̅N̅T̅A̅ active signal using the two status outputs (S0, S1) and the IO/M̅ address pin. This is required since the I̅N̅T̅A̅ output from the Am8085A becomes valid too late to meet the 100nsec setup time of READY. The Am9519A's P̅A̅U̅S̅E̅ output takes over control of the ready line following T2.

Insertion of a wait during the read and write operation to the Am9519A with C̅S̅ is required only when the 5MHz Am8085A-2 part is used. The wait state is needed because the minimum R̅D̅ and W̅R̅ pulse width of the Am8085A-2 is 230nsec, while the Am9519A-1, as example, requires 250nsec minimum and the Am9519A, 300nsec minimum.

Am9519A to Z80 Interface

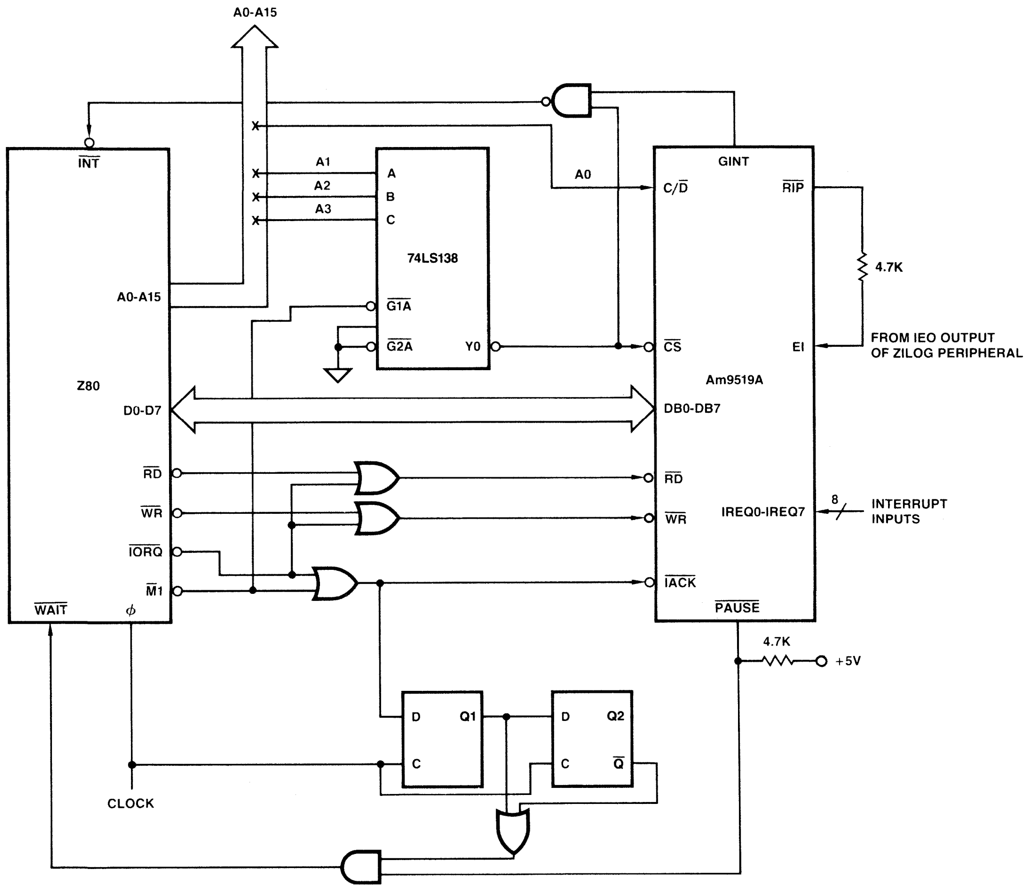


Figure 24.

DESCRIPTION OF INTERFACE:

The Am9519A can be configured as the sole interrupting device to the Z80 or in conjunction with other devices (such as the SIO, CIO, CTC, etc.) in a daisy chain interrupt nest. When the Am9519A is part of a nested daisy chain, it must be the lowest priority device because the Am9519A activates its Enable Out (EO) output after receipt of the interrupt acknowledge signal (IROQ • M1) from the microprocessor, while the Zilog parts require their IEI (Interrupt Enable In) signal to setup at least 200nsec (2.5MHz Z80) prior to the Interrupt Acknowledge going active.

A wait state is inserted whenever an Interrupt Acknowledge sequence begins since the Am9519A begins resolving its interrupt priorities after the interrupt acknowledge signal goes active. The flip-flops assert a wait state at the second TW falling clock edge and allow the PAUSE output to extend the wait state as required.

These flip-flops are required and since pause output will not be active soon enough to meet the Z80's setup time on WAIT input, M1 is ANDed with CS to prevent CS being active during interrupt request. Either an Am9519A or Am9519A-1 can be used with the Z80A.

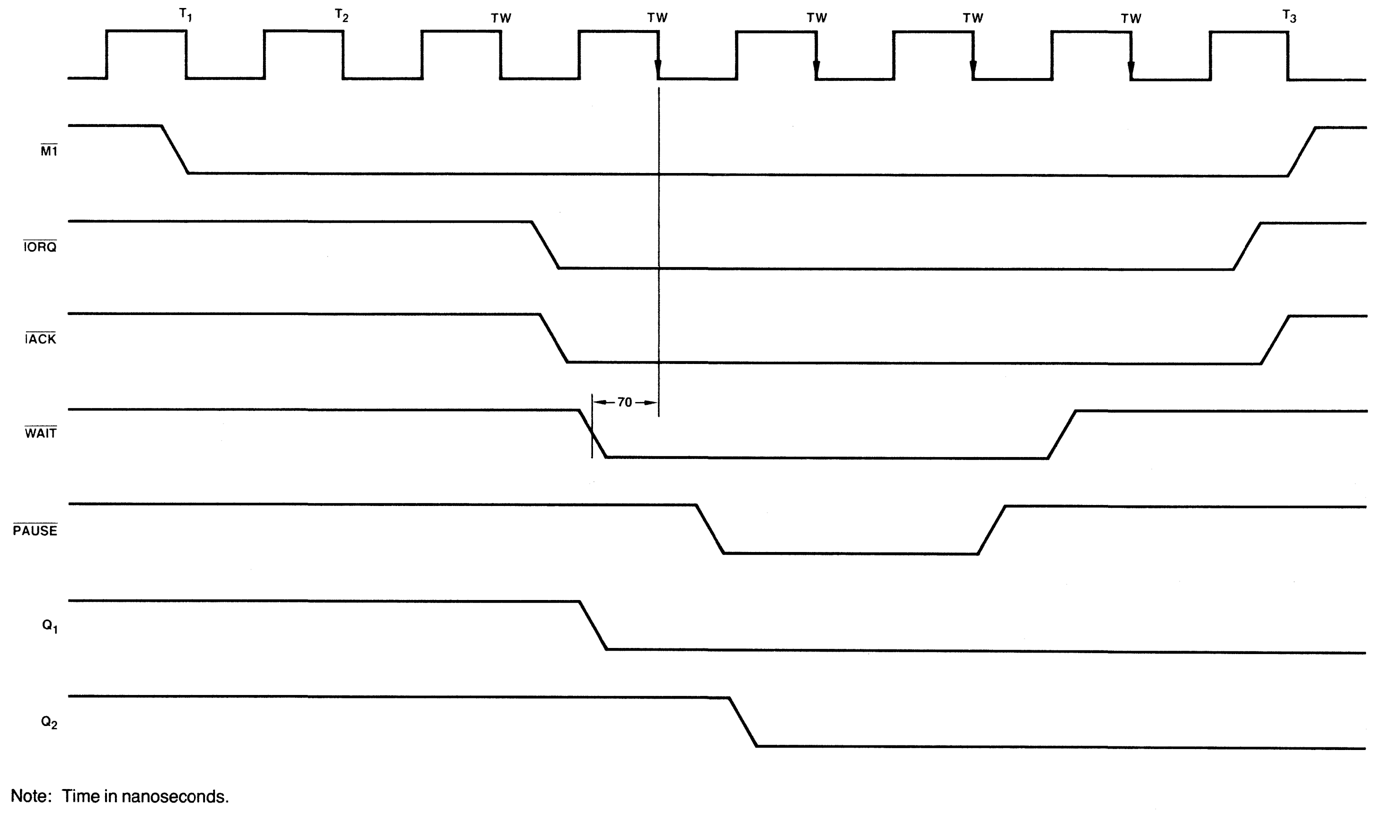


Figure 25. Z80 – Am9519A Interrupt Acknowledge Timing

Am9519A to 68000 Interface

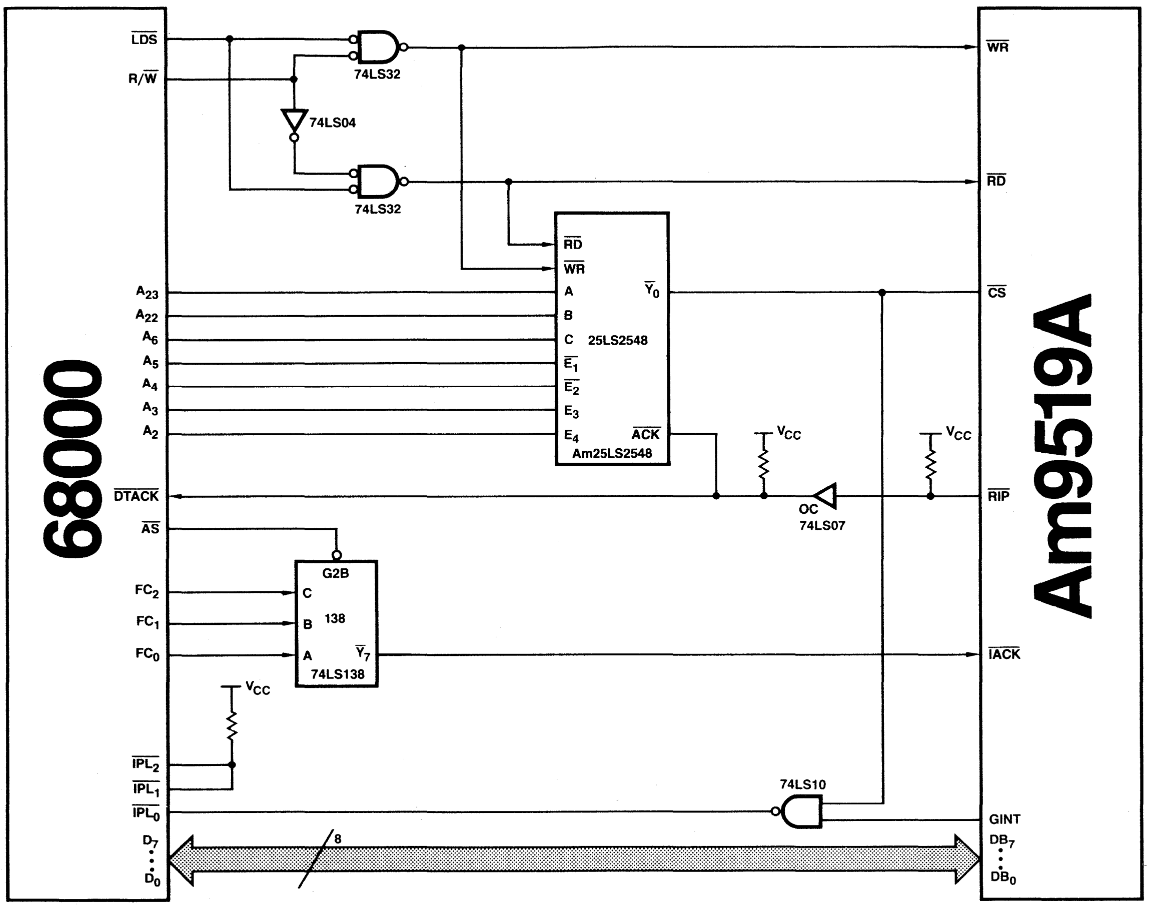


Figure 26.

This interface for the Am9519A will work for the 4MHz and 6MHz 68000s. It will also work the faster CPUs if wait states are inserted during programming. /RIP will automatically insert the appropriate number of wait states during the interrupt acknowledge cycle.

Additional decoding of the upper address bits should be done to reduce the address space occupied by this part. An I/O space separate from the memory address space should be generated by appropriately decoding the address bits. This will prevent spurious glitches on chip select and meet the Am9519A's requirement that chip select be high for at least 100nsec prior to an interrupt acknowledge cycle. If their are no other interrupting devices in the system /GINT can be connected directly to one of the /IPL pins thus eliminating the 74LS148 shown.

/AS is used to deglitch the 74LS138's output to give a clean /INTA. Note that the Am9519A must be connected to the lower data bus and /AS must not be used to deglitch /CS.

One last thing to watch out for is the recovery time. Do not use the MOVEP instruction to program the Am9519A.

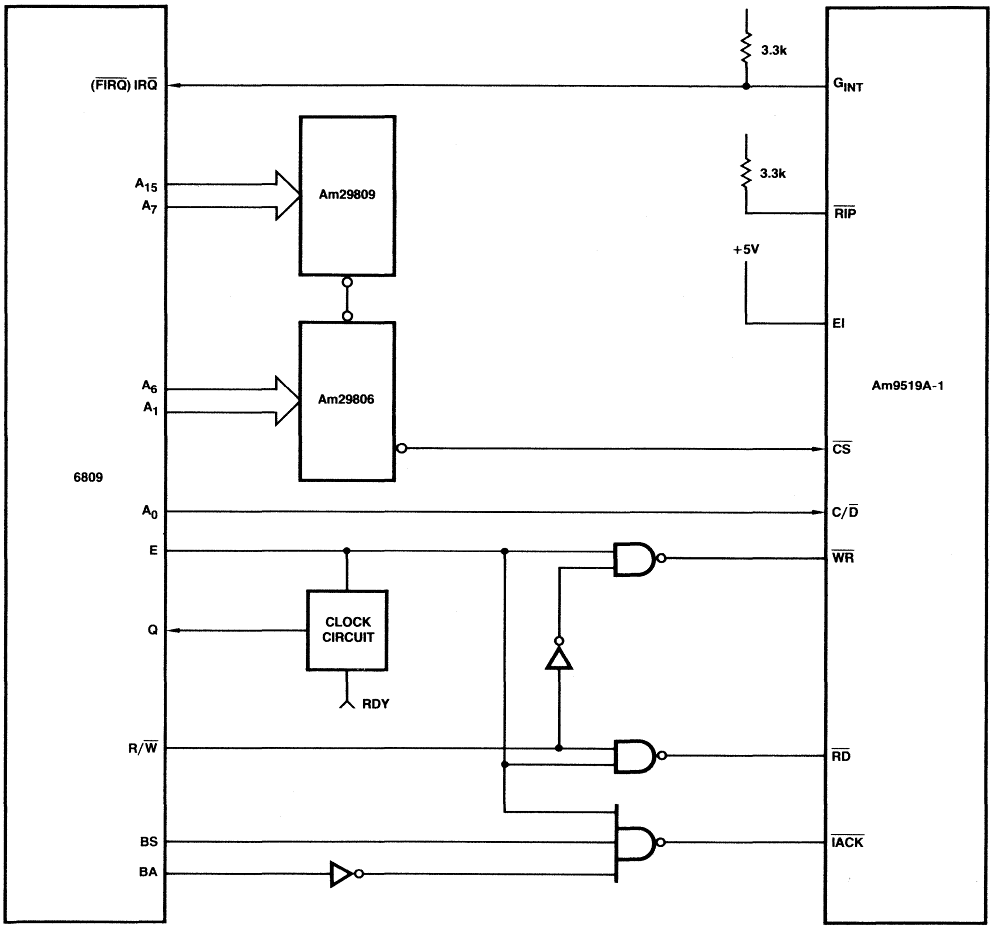
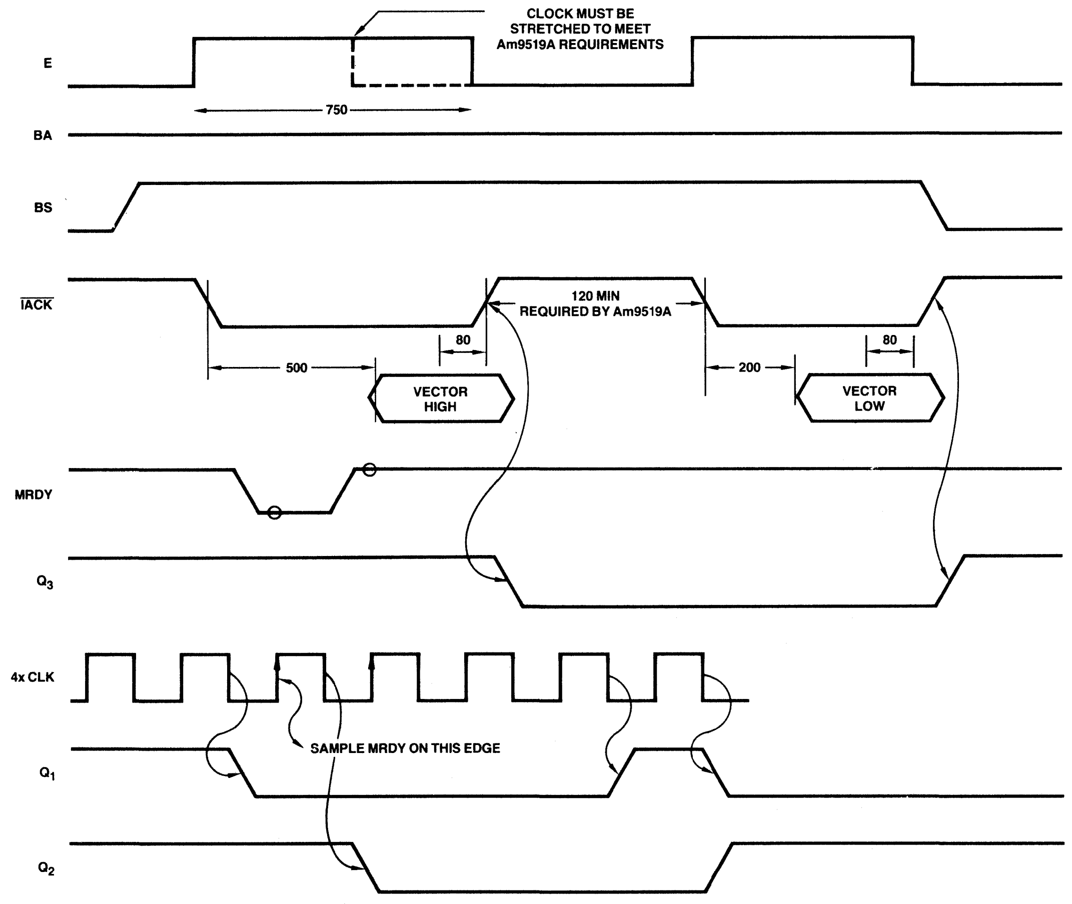

(FIRQ) IRQ
A15
A7
Am29809
A6
A1
Am29806
6809
A0
E
Q
CLOCK CIRCUIT
RDY
R/W
BS
BA
3.3k
3.3k
+5V
GINT
RIP
EI
Am9519A-1
CS
C/D
WR
RD
IACK

Figure 27a.



Note: Time in nanosecods.

Figure 27b.

Am9519A with the Am8086

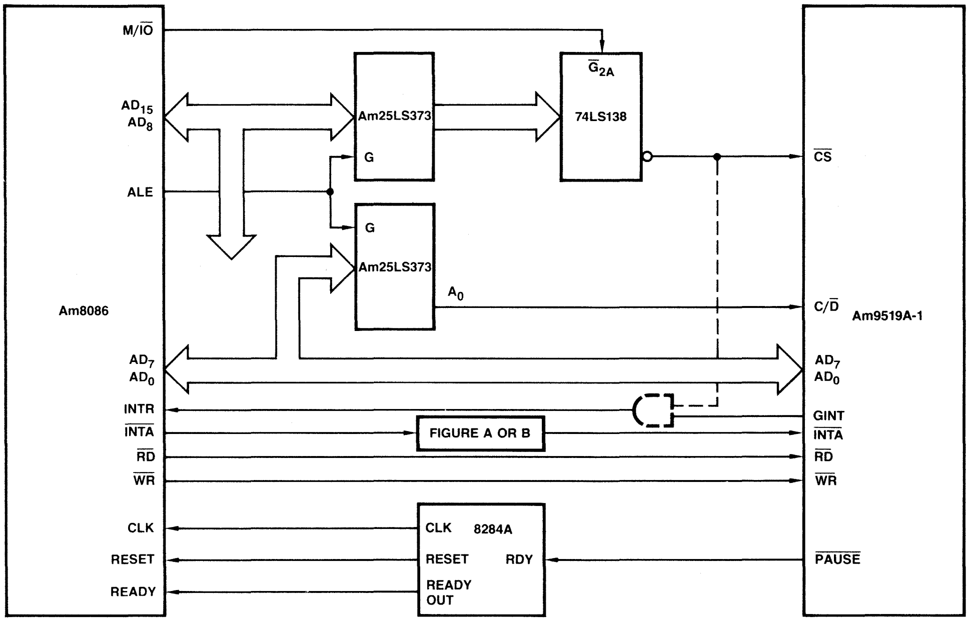
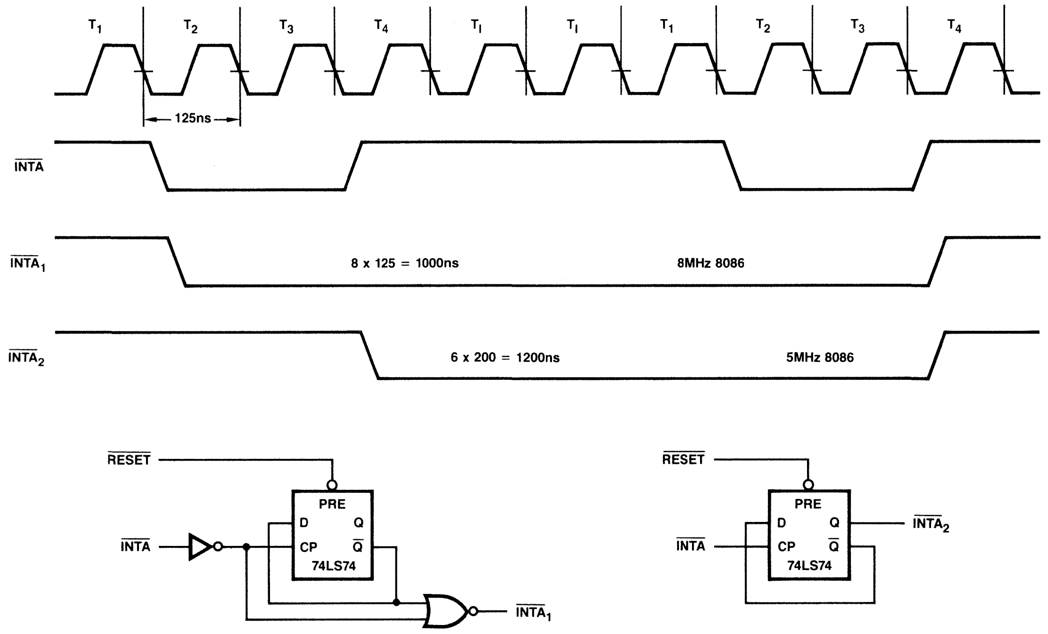


Figure 28.

This interface illustrates the use of the Am9519A interrupt controller with the Am8086. The Am9519A has many advantages over the Am8259A. However, some external logic is required. The purpose of this logic is to convert the two interrupt acknowledge cycles from the Am8086 into one. Those familiar with the Am9519A may question this since the Am9519A can be programmed to accept two interrupt acknowledge pulses. The reason for choosing this approach is faster response time. The Am9519A requires the first interrupt acknowledge pulse to be 900nsec in order to resolve priority. Thus WAIT states would need to be inserted. The simple logic shown eliminates the need for wait states and meets the Am9519A timing requirements.

The gate shown by dotted lines is optional. Its purpose is to prevent interrupts when the CPU does a write to the mask register. This can be done in software by disabling interrupts whenever a write to the Interrupt Mask Register (IMR) is done.

The discussion so far has centered on the interrupt acknowledge response. When programming the Am9519A a careful timing analysis shows that for a 5MHz CPU the Am9519A-1 should be used or a WAIT state inserted. For the 8MHz CPU a WAIT state will be required. This is due to the uncertainty of where the /RD pulse occurs with respect to the clock. This will result in the data to clock setup time not being met in some cases. The best fix is to insert a WAIT state for programmed I/O.



8MHz 8086
Figure A

5MHz 8086
Figure B

Figure 29.

U.S. AND CANADIAN SALES OFFICES

NORTHEAST AREA

Advanced Micro Devices
6 New England Executive Park
Burlington, Massachusetts 01803
Tel: (617) 273-3970

Advanced Micro Devices (Canada) Ltd.
2 Sheppard Avenue East
Suite 1610
Willowdale, Ontario
Canada M2N5Y7
Tel: (416) 224-5193

Advanced Micro Devices (Canada) Ltd.
AMD
4019 Carling # 301
Kanata, Ottawa
Canada K2K2A3

Advanced Micro Devices
290 Elwood Davis Road
Suite 316
Liverpool, New York 13088
Tel: (315) 457-5400

MID-ATLANTIC AREA

Advanced Micro Devices
40 Crossways Park Way
Woodbury, New York 11797
Tel: (516) 364-8020

Advanced Micro Devices
Waterview Plaza, Suite 303
2001 U.S. Route #46
Parsippany, New Jersey 07054
Tel: (201) 299-0002

Advanced Micro Devices
110 Gibralter Road #110
Horsham, Pennsylvania 19044
Tel: (215) 441-8210
TWX: 510-665-7572

Advanced Micro Devices
Commerce Plaza
5100 Tilghman Street, Suite 320
Allentown, Pennsylvania 18104
Tel: (215) 398-8006
FAX: 215-398-8090

Advanced Micro Devices
205 South Avenue
Poughkeepsie, New York 12601
Tel: (914) 471-8180
TWX: 510-248-4219

Advanced Micro Devices
10 Main Street South
Southbury, Connecticut 06488
Tel: (203) 264-7600

Advanced Micro Devices
7223 Parkway Drive #203
Dorsey, Maryland 21076
Tel: (301) 796-9310
FAX: 796-2040

SOUTHEAST AREA

Advanced Micro Devices
4740 North State Road #7
Suite 102
Ft. Lauderdale, Florida 33319
Tel: (305) 484-8600

Advanced Micro Devices
7850 Ulmerton Road, Suite 1A
Largo, Florida 33541
Tel: (813) 535-9811

Advanced Micro Devices
15 Technology Parkway #200
Norcross, Georgia 30092
Tel: (404) 449-7920

Advanced Micro Devices
8 Woodlawn Green, Suite 220
Woodlawn Road
Charlotte, North Carolina 28210
Tel: (704) 525-1875

Advanced Micro Devices
303 Williams Avenue Southwest
Suite 118
Huntsville, Alabama 35801
Tel: (205) 536-5505

Advanced Micro Devices
6501 Six Forks, Suite 150
Raleigh, North Carolina 27609
Tel: (919) 847-8471

MID-AMERICA AREA

Advanced Micro Devices
500 Park Boulevard, Suite 940
Itasca, Illinois 60143
Tel: (312) 773-4422

Advanced Micro Devices
9900 Bren Road East, Suite 601
Minnetonka, Minnesota 55343
Tel: (612) 938-0001

Advanced Micro Devices
3592 Corporate Drive, Suite 108
Columbus, Ohio 43229
Tel: (614) 891-6455

Advanced Micro Devices
16985 West Blue Mound Road, Suite 201
Brookfield, Wisconsin 53005
Tel: (414) 782-7748
FAX: (414) 782-8041

NORTHWEST AREA

Advanced Micro Devices
1245 Oakmead Parkway
Suite 2900
Sunnyvale, California 94086
Tel: (408) 720-8811

Advanced Micro Devices
One Lincoln Center, Suite 230
10300 Southwest Greenburg Road
Portland, Oregon 97223
Tel: (503) 245-0080

Advanced Micro Devices
Honeywell Ctr., Suite 1002
600 108th Avenue N.E.
Bellevue, Washington 98004
Tel: (206) 455-3600

MID-CALIF AREA

Advanced Micro Devices
360 N. Sepulveda, Suite 2075
El Segundo, California 90245
Tel: (213) 640-3210

Advanced Micro Devices
21600 Oxnard Street, Suite 675
Woodland Hills, California 91367
Tel: (213) 992-4155

SOUTHERN CALIF AREA

Advanced Micro Devices
4000 MacArthur Boulevard
Suite 5000
Newport Beach, California 92660
Tel: (714) 752-6262

Advanced Micro Devices
9619 Chesapeake Drive #210
San Diego, California 92123
Tel: (619) 560-7030

MOUNTAIN WEST AREA

Advanced Micro Devices
6750 LBJ Freeway, Suite 1160
Dallas, Texas 75240
Tel: (214) 934-9099

Advanced Micro Devices
8240 MoPac Expressway
Two Park North, Suite 385
Austin, Texas 78759
Tel: (512) 346-7830

Advanced Micro Devices
1873 South Bellaire Street
Suite 920
Denver, Colorado 80222
Tel: (303) 691-5100

Advanced Micro Devices
40 W. Baseline Road # 206
Tempe, Arizona 85283
Tel: (602) 242-4400

Advanced Micro Devices
1955 W. Grant Road # 125
Tucson, Arizona 85745
Tel: (602) 792-1200

INTERNATIONAL SALES OFFICES

BELGIUM
Advanced Micro Devices
Belgium N.V. – S.A.
Avenue de Tervueren, 412, bte 9
B-1150 Bruxelles
Tel: (02) 771 99 93
TELEX: 61028
FAX: 7623712

FRANCE
Advanced Micro Devices, S.A.
Silic 314, Immeuble Helsinki
74, rue d'Arcueil
F-94588 Rungis Cedex
Tél: (01) 687.36.66
TELEX: 202053
FAX: 686.21.85

GERMANY
Advanced Micro Devices GmbH
Rosenheimer Str. 139
D-8000 Muenchen 80
Tel: (089) 40 19 76
TELEX: 05-23883
FAX: 406 490

Advanced Micro Devices GmbH
Feuerseeplatz 4/5
D-7000 Stuttgart 1
Tel: (0711) 62 33 77
TELEX: 07-21882
FAX: 625 187

Advanced Micro Devices GmbH
Zur Worth 6
D-3108 Winsen/Aller
Tel: (05143) 53 62
TELEX: 925287
FAX: 5553

HONG KONG
Advanced Micro Devices
1303 World Commerce Centre
Harbour City
11 Canton Road
Tsimshatsui, Kowloon
Tel: (852) 3 695377
TELEX: 50426
FAX: (852) 123 4276

ITALY
Advanced Micro Devices S.R.L.
Centro Direzionale
Via Novara, 570
I-20153 Milano
Tel: (02) 3533241
TELEX: 315286
FAX: 656878

JAPAN
Advanced Micro Devices, K.K.
Dai 3 Hoya Building
8-17, Kamitakaido 1 chome
Suginami-ku, Tokyo 168
Tel: (03) 329-2751
TELEX: 2324064
FAX: (03) 326 0262

SWEDEN
Advanced Micro Devices AB
Box 7013
S-172 07 Sundbyberg
Tel: (08) 98 12 35
TELEX: 11602
FAX: 298087

UNITED KINGDOM
Advanced Micro Devices (U.K.) Ltd.
A.M.D. House,
Goldsworth Road,
Woking,
Surrey GU21 1JT
Tel: Woking (04862) 22121
TELEX: 859103
FAX: 22179

Advanced Micro Devices (U.K.) Ltd.
The Genesis Centre
Garrett Field
Science Park South
Birchwood
Warrington WA3 7BH
Tel: Warrington (0925) 828008
TELEX: 628524
FAX: 827693

Advanced Micro Devices maintains a network of representatives and distributors in the U.S. and around the world. For a sales agent nearest you, call one of the AMD offices above.