
INTERFACING THE ISCC™ TO THE 68000 AND 8086

INTRODUCTION

The ISCC™ uses its flexible bus to interface with a variety of microprocessors and microcontrollers; included are the 68000 and 8086.

The Z16C35 ISCC is a Superintegration form of the 85C30/80C30 Serial Communications Controller (SCC). Super integration includes four DMA channels, one for each receiver and transmitter and a flexible Bus Interface Unit (BIU). The BIU supports a wide variety of buses

including the bus types of the 680X0 and the 8086 families of microprocessors.

This Application Note presents the details of BIU operation for both slave peripheral and DMA modes. Included are application examples of interconnecting an ISCC to a 68000 and a 8086 (These examples are currently under test).

ISCC BUS INTERFACE UNIT (BIU)

The following subsections describe and illustrate the functions and parameters of the ISCC Bus Interface Unit.

Overview

The ISCC™ contains a flexible bus interface that is directly compatible with a variety of microprocessors and microcontrollers. The bus interface unit adds to the chip by allowing ease of connection to several standard bus configurations; among others are the 68000 and the 8086 family microprocessors. This compatibility is achieved by initializing the ISCC after a reset to the desired bus configuration.

The device also configures to work with a variety of other 8- or 16-bit bus systems and is used with address/data multiplexed or non-multiplexed buses. In addition, the wait/ready handshake, the interrupt acknowledge, and the bus high byte/low byte selection are all programmable. Separate read/write, data strobe, write, read, and address strobe signals are available for direct system interface with a minimum of external logic.

Modes Description

There are basically two bus modes of operation: multiplexed and non-multiplexed. In the multiplexed bus mode, the ISCC internal registers are directly accessible as separate registers with their own unique hardware addresses. By contrast, in the non-multiplexed mode, all

registers access through an internal pointer which first loads with the register address. Loading of the pointer is done as a data write. In either case, there are some external addressing signals.

Chip Enable (CE) allows external selection through the decode of upper order address bits like accessing separate chips. A separate input (not part of the AD15-AD0 bus connection) selects between the internal SCC and DMA sections of the chip. This input is A0/SCC/DMA and provides direct transfers to the appropriate chip subsystem; either multiplexed or non-multiplexed bus mode.

A second separate input (not part of the AD15-AD0 bus connection) provides for a selection between the internal SCC; both channels A and B (Table A-1). This input is A1/A/B and provides direct transfers to the appropriate SCC channel when A0/SCC/DMA selects the SCC; either multiplexed or non-multiplexed bus mode. Note that these two signals, A1/A/B and A0/SCC/DMA, are inputs when

ISCC BUS INTERFACE UNIT (BIU) (Continued)

the ISCC is a slave peripheral; they become outputs when the ISCC is a bus master during DMA operations.

Table 1. Accessing the ISCC Registers

A0/SCC/DMA	A1/A/B	ACCESS
1	1	SCC Channel A
1	0	SCC Channel B
0	x	DMA

The following discussions assume knowledge of the SCC Serial Communications Controller operations and refer to internal register designations. For a detailed explanation, refer to the SCC Technical Manual.

Non-Multiplexed Bus Operation

When the ISCC initializes for non-multiplexed operation, Write Register 0 (WR0) takes on the form of WR0 in the Z8530, Write Register Bit Functions (Figure A-1). Register addressing for the SCC section is (except for WR0 and RR0) accomplished as follows. Programming the write registers requires two write operations. Reading the read registers requires both a write and a read operation.

The first write is to WR0 which contains three bits that point to the selected register (note the point high command). The second write is the actual control word for the selected register. If the second operation is a read, the selected register is accessed. When in the non-multiplexed mode, all registers in the SCC section of the ISCC, including the data registers, access this way.

The pointer register automatically clears after the second read or write operation so WR0 (or RR0) addresses again. There is no direct access to the data registers. They are addressed through the pointer (this is in contrast to the Z8530 which allows direct addressing of the data registers through the C/D pin).

When the ISCC starts for non-multiplexed operation, register addressing for the DMA section is (except for CSAR) accomplished as follows. It is completely independent of the SCC section register addressing. Programming the write registers requires two write operations and reading the read registers requires both a write and a read operation. The first write is to the Command Status Address Register (CSAR) which contains five bits that point to the selected register (CSAR bits 4-0). The second write is the actual control word for the selected register. If the second operation is a read, the selected register is accessed. The pointer bits automatically clear after the second read or write operation so CSAR addresses again. When in the non-multiplexed mode, all registers in the DMA section of the ISCC are accessed.

Multiplexed Bus Operation

When the ISCC initializes for multiplexed bus operation, all registers in the SCC section are directly addressable with the register address occupying AD5 through AD1 or AD4 through AD0 (Shift Left/Shift Right modes).

The Shift Left/Shift Right modes for the address decoding of the internal registers (multiplexed bus) are separately programmable for the SCC and DMA sections. For the SCC section, the programming and operation is the same as the SCC; programming occurs through Write Register 0 (WR0), bits 1 and 0, and Write Register Bit Functions (Figure A-2). The programming of the Shift Left/Shift Right modes for the DMA section occurs in the BCR, bit 0. In this case, the shift function is similar to the SCC section; with Left Shift, the internal register addresses decode from bits AD5 through AD1. In Right Shift, the internal register addresses decode from bits AD4 through AD0.

During multiplexed bus mode selection, Write Register 0 (WR0) becomes WR0 in the Z8030, Write Register Bit Functions (Figure A-2).

Write Register 0 (non-multiplexed bus mode)

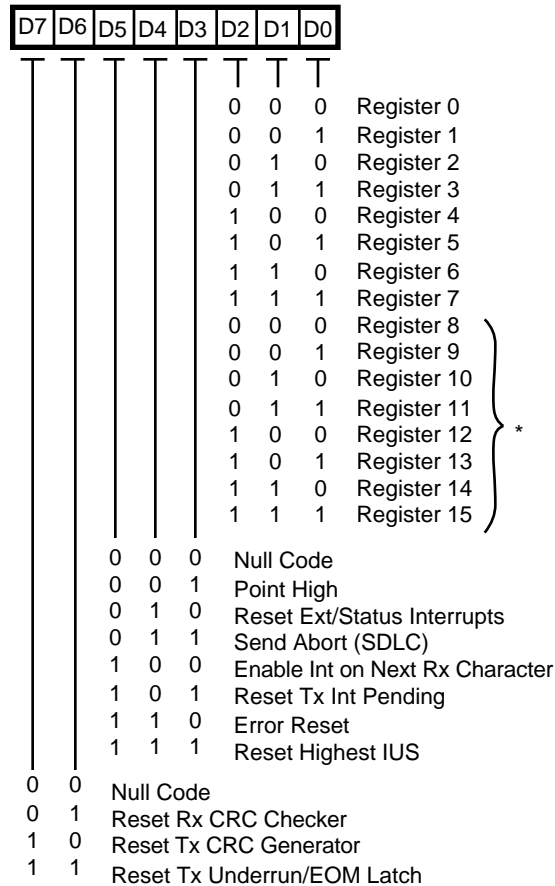


Figure 1. Write Register 0 Bit Functions (Non-Multiplexed Bus Mode)

Register 0 (multiplexed bus mode)

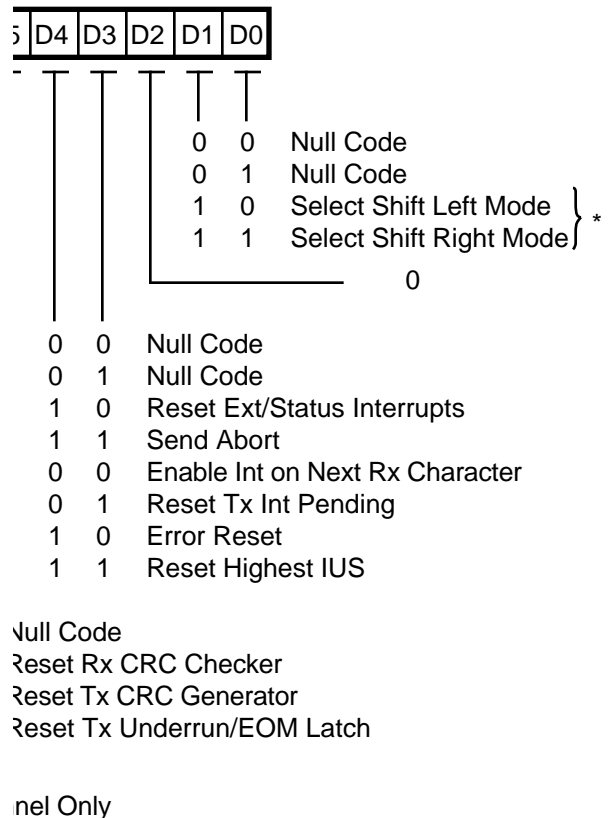


Figure 2. Write Register 0 Bit Functions (Multiplexed Bus Mode)

BUS DATA TRANSFERS

All data transfers to and from the ISCC™ are done in bytes regardless of whether data occupies the lower or upper byte of the 16-bit bus. Bus transfers as a slave peripheral are done differently from bus transfers when the ISCC is the bus master during DMA transactions. The ISCC is fundamentally an 8-bit peripheral but supports 16-bit buses in the DMA mode. Slave peripheral and DMA transactions appear in the next sections.

Data Bus Transfers as a Slave Peripheral

When accessed as a peripheral device (when the ISCC is not a bus master performing DMA transfers), only 8 bits transfer. During ISCC register read, the byte data present on the lower 8 bits of the bus is replicated on the upper 8 bits of the bus. Data is accepted by the ISCC only on the lower 8 bits of the bus.

ISCC™ DMA Bus Transfers

During DMA transfers, when the ISCC is bus master, only byte data transfers occur. However, data transfers to or from the ISCC on the upper 8 bits of the bus or on the lower 8 bits of the bus. Moreover, odd or even byte transfers activate on the lower or upper 8 bits of the bus. This is programmable and explained next.

During DMA transfers to memory from the ISCC, only byte data transfers occur. Data appears on the lower 8 bits and replicates on the upper 8 bits of the bus. Thus, the data is written to an odd or even byte of the system memory by address decoding and strobe generation.

During DMA transfers to the ISCC from memory, byte data only transfers. Normally, data appears only on the lower 8 bits of the bus. However, the byte swapping feature

BUS DATA TRANSFERS (Continued)

determines which byte of the bus data is accepted. The byte swapping feature activates by programming the Byte Swap Enable bit to a 1 in the BCR. The odd/even byte transfer selection occurs by programming the Byte Swap Select bit in the BCR. If Byte Swap Select is a 1, then even address bytes (transfers where the DMA address has A0 = 0) are accepted on the lower 8 bits of the bus. Odd address bytes (transfers where the DMA address has A0 = 1) are accepted on the upper 8 bits of the bus. If Byte Swap Select is a 0, then even address bytes (transfers where the DMA address has A0 = 0) are accepted on the upper 8 bits of the bus. Odd address bytes (transfers where the DMA address has A0 = 1) are accepted on the lower 8 bits of the bus.

Bus Interface Handshaking

The ISCC™ supports data transfers by either a data strobe (DS) combined with a read/write (R/W) status line, or separate read (RD) and write (WR) strobes. These transactions activate via chip enable (CE).

ISCC programming generates interrupts upon the occurrence of certain internal events. The ISCC internally prioritizes its own interrupts, therefore, the ISCC presents one interrupt to the processor even though lower priority internal interrupts may be pending. Interrupts are individually enabled or disabled. Refer to the sections on the SCC core.

Interrupt Acknowledge (INTACK) is an input to the ISCC showing that an interrupt acknowledge cycle is progressing. INTACK is programmed to accept a status acknowledge, a single pulse acknowledge, or a double pulse acknowledge. This programming activates in the BCR. The double pulse acknowledge is compatible with 8X86 family microprocessors and the status acknowledge is compatible with 68000 family microprocessors.

During an interrupt acknowledge cycle, the SCC and DMA interrupt priority daisy chain internally resolves. Thus, the highest priority internal interrupt is presented to the CPU.

The ISCC can return an interrupt vector that encodes with the type of interrupt pending enabled during this acknowledge cycle. The ISCC may request an interrupt but not return an interrupt vector [note that the no vector bit(s) in the SCC section (WR9 bit 1) and in the DMA section (ICR bit 5) individually control whether or not an interrupt vector returns by these cores]. The interrupt vector can program to include a status field showing the internal ISCC source of the interrupt. During the interrupt acknowledge cycle, the ISCC returns the interrupt vector when INTACK, RD or DS go active and IEI is high (if the ISCC is not programmed for the no vector option).

During the programmed pulsed acknowledge type (whether single or double), INTACK is the strobe for the interrupt vector. Thus when INTACK goes active, the ISCC drives the bus and presents the interrupt vector to the CPU. When the status acknowledge type programs, the ISCC drives the bus with the interrupt vector when RD or DS are active.

WAITRDY programs to function either as a WAIT signal or a READY signal using the BCR write. When programmed as a wait signal, it supports the READY function of 8X86 family microprocessors. When programmed as a ready signal, it supports the DTACK function of 680x0 family microprocessors.

The WAIT/RDY signal functions as an output when the ISCC is not a bus master. In this case, this signal serves to indicate when the data is available during a read cycle, when the device is ready to receive data during a write cycle, and when a valid vector is available during an interrupt acknowledge cycle.

When the ISCC is the bus master (DMA section has taken control of the bus), the WAIT/RDY signal functions as a WAIT or RDY input. Slow memories and peripheral devices use WAIT to extend the data strobe (/DS) during bus transfers. Similarly, memories and peripheral devices use RDY to indicate valid output or that it is ready to latch input data.

CONFIGURING THE BUS

The bus configuration programming is done in two separate steps (actually it is one operation), to enable the write to the Bus Configuration Register (BCR). The first operation that accesses the ISCC after a device reset must be a write to the BCR since this is the only time that the BCR is accessible. Before and during the write, various external signals are sampled to program bus configuration parameters. During this write, the A0/SCC//DMA pin must be Low.

Address strobe programs multiplexed/non-multiplexed selection. In a non-multiplexed bus environment, address strobe (as an input) is not used but tied high through a suitable pull-up resistor. Thus, no address strobe is present before the BCR write. Then, when write to the BCR takes place, the non-multiplexed mode is programmed because there is no address strobe before this first write to the device. Note that address strobe becomes an output during DMA operations so it is not tied directly to V_{CC}.

During the write operation to the BCR, the A1/A/B input is sampled to select the function of the WAIT/RDY pin (Table A-2). When the BCR Write is to the SCC Channel A (A1/A/B High during the BCR write), the WAIT/RDY signal functions as a wait. When the BCR Write is to Channel B (A1/A/B Low during the BCR write), the WAIT/RDY signal functions as a ready.

Table 40. Signals Sampled During the BCR Write

A1/A/B	WAIT/RDY Function
1	WAIT (8086 RDY compatible)
0	READY (68000 DTACK compatible)

This programming affects the function of the WAIT/RDY signal both as an input, when the ISCC is bus master during DMA operations, and as an output when the ISCC is a bus slave.

With this programming, the ISCC is immediately configured to function successfully on this first and subsequent bus transactions. The remaining bus configuration options are programmed by the value written to the BCR.

Bit 0 of the BCR controls the Shift Left/Shift Right address decoding modes for the DMA section. In this case, the shift function is similar to the SCC section. During Left Shift, the internal register addresses decode from bits AD5 through AD1. During Right Shift, the internal register addresses are decode from bits AD4 through AD0. This function is only applicable in the multiplexed bus mode.

Bits 1 and 2 of the BCR control the interrupt acknowledge type as shown in the Table A-3.

Table 41. BCR Control of Interrupt Acknowledge

BCR bit 2	BCR bit 1	Interrupt Acknowledge
0	0	Status Acknowledge
0	1	Pulsed Acknowledge (single)
0	1	Reserved (action not defined)
1	1	Double Pulsed Acknowledge

The Status Acknowledge remains active throughout the interrupt cycle and is directly compatible with the 680x0 family interrupt handshaking. The Status Acknowledge signal latches with the rising edge of AS for multiplexed bus operation. It latches by the falling edge of the strobe (RD or DS) for non-multiplexed bus operation. The Pulsed Acknowledges are timed to be active during a specified period in the interrupt cycle. The Double Pulsed Acknowledge is directly compatible with the 8x86 family interrupt handshaking. Refer to the timing diagrams in the ISCC Product Specification for details on the Acknowledge signal operation.

Reserve bits 3, 4, and 5 of the BCR program as zeros. Bits 6 and 7 of the BCR control the byte swap feature (Table A-4). Byte swap is applicable only in DMA transfers when the ISCC is the bus master and only affects ISCC data acceptance (transfers from memory to the ISCC).¹

Table 42. Byte Swap Control

Enable (BCR bit 7)	DMA Data Read by the ISCC
0	lower 8 bits of bus only
1	upper or lower 8 bits of bus

Swap Select*	A0	DMA Data read by the ISCC
0	0	upper 8 bits of bus
0	1	lower 8 bits of bus
1	0	lower 8 bits of bus
1	1	upper 8 bits of bus

* BCR bit 6

APPLICATIONS EXAMPLES

The following application examples explain and illustrate the methods of interfacing the ISCC to a Motorola 68000 and an Intel 8086.

68000 Interface to the ISCC

Figure A-3 shows a connection of the ISCC to a 68000 microprocessor. The 68000 data bus connects directly, or through bus transceivers, to the ISCC address/data bus. R/W and RESET also directly connect. In this example, the ISCC is on the lower half of the bus; DS of the ISCC connects to LDS of the 68000. The processor address lines decode to produce a chip enable for the ISCC. In addition, processor addresses A1 and A2 connect to A0/SCC/DMA and A1/A/B, respectively, through a tri-state driver.

The driver is normally ON (enabled) but turns OFF by BGACK to grant the bus to ISCC for DMA transfers. This is done since the A0/SCC/DMA and A1/A/B pins become outputs during DMA transfers and should not drive the system address bus. RD and WR tie high through independent pull-ups. They are not used in this application but become active outputs during DMA transfers and are not tied directly to V_{CC} .

Although not shown in Table A-5, the A0/SCC/DMA and A1/A/B pins may be decoded during DMA transfers to identify the active DMA channel.

Table 43. DMA A/B Channel Decode

A1/A/B	A0/SCC/DMA	DMA Channel
1	1	Receiver Channel A
1	0	Transmitter Channel A
0	1	Receiver Channel B
0	0	Transmitter Channel B

External logic can use this information to abort a DMA in progress.

For normal slave device bus interaction, a DTACK is generated. WAIT/RDY is programmed for ready operation and INTACK programs for the status type. WAIT/RDY generates a DTACK for normal data transfers and interrupt responses. Additional logic may be required when other interrupt sources are present.

During DMA transfers, the ISCC becomes bus master. Becoming bus master is done through the BUSREQ output and BUSACK input signals of the ISCC. They connect to an external bus arbitration circuit. This circuit

performs bus arbitration for multiple bus master requests and generates bus grant acknowledge (BGACK) which controls certain bus drive signal sources.

When the ISCC becomes the bus master, a 32-bit address generation by the DMA section is output on the ISCC address/data bus. The lower 16 bits of this address store in an external latch by AS (Address Strobe). Also, the upper 16 bits of this address store in an external latch by UAS (Upper Address Strobe). With BGACK low (active) and with the processor address lines tri-stated, the latch outputs drive the system address bus.

AS is pulled high by an external resistor. This pull-up insures an inactive AS (at a logic high level) when the ISCC is not driving this signal. Therefore, on power up or after a RESET, AS is inactive and programs the non-multiplexed bus mode on BCR write.

In this application, the outputs of the address latches are connected to the address bus so that A1 through A23 of the ISCC drives the system address bus (the ISCC provides a total of 32 address lines). A0 from the address latch is diverted to logic which generates UDS and LDS bus signals from the ISCC data strobe (DS). UDS is generated when A0 is low and LDS is generated when A0 is high. The lower and upper data strobes are applied to the system bus through tri-state drivers which are enabled only when BGACK is active. Bus direction is now controlled by the ISCC R/W signal which is now an output.

For initialization, the BCR write (the first write to the ISCC after RESET) is done with A2 = 0 (A1/A/B ISCC input at logic low). This selects the ready option of the WAIT/RDY signal to conform to the 68000 bus style. The AS signal programming of the non-multiplexed bus has already been discussed. The BCR is written with C0H to enable byte swapping. It also selects the sense of byte swapping with respect to A0 appropriate to this bus style and selects the STATUS type of interrupt acknowledge.

8086 Interface with the ISCC

Figure A-4 shows the connection of the ISCC to an 8086 microprocessor and companion clock state generator. In this application, the ISCC connects for multiplexed address access to the internal ISCC registers. AD15 through AD0 of the 8086 connect directly, or through a bus transceiver, to the corresponding AD15 through AD0 address/data ISCC bus pins. RD and WR are directly compatible and tie together to form the read and write bus signals.

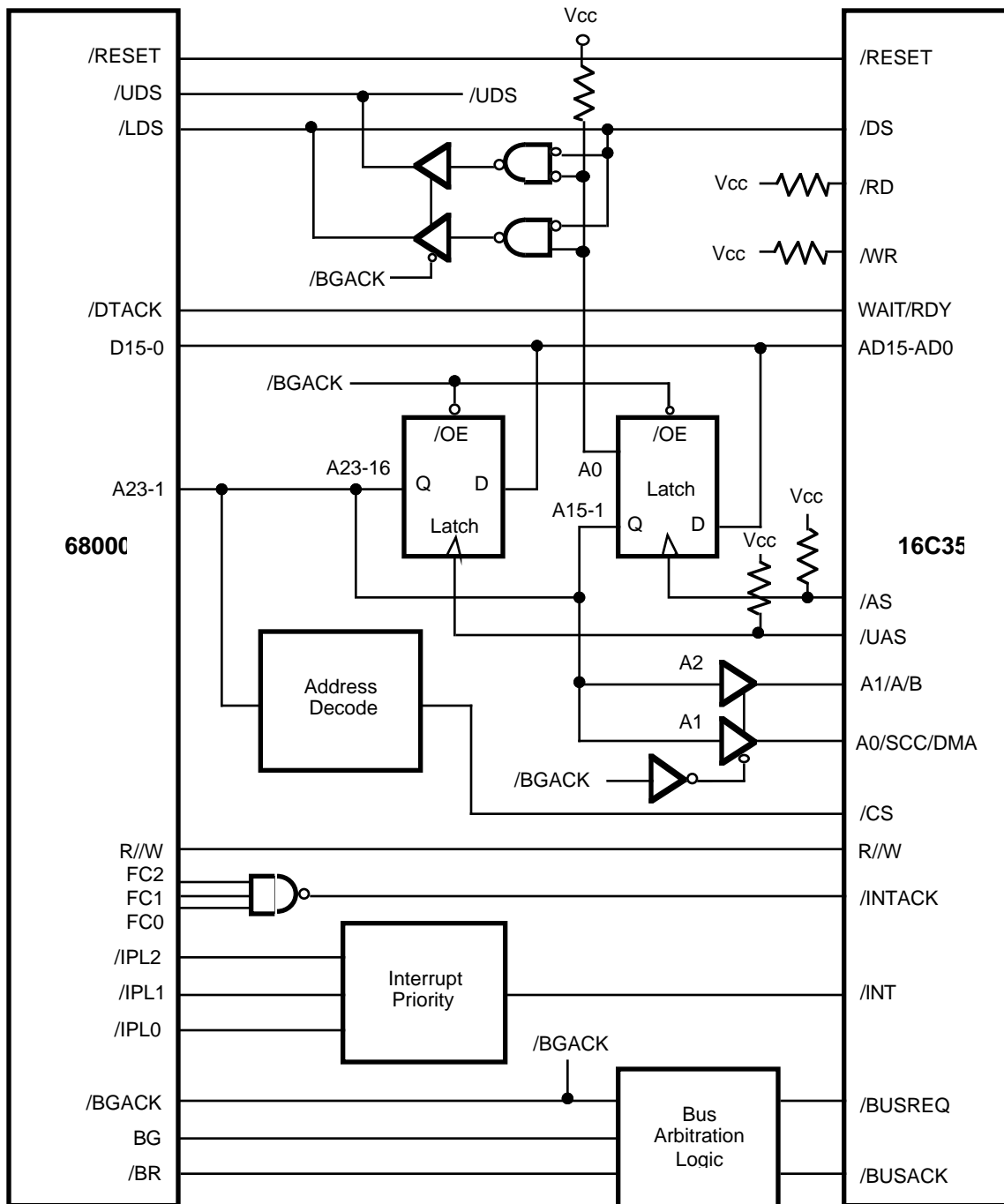
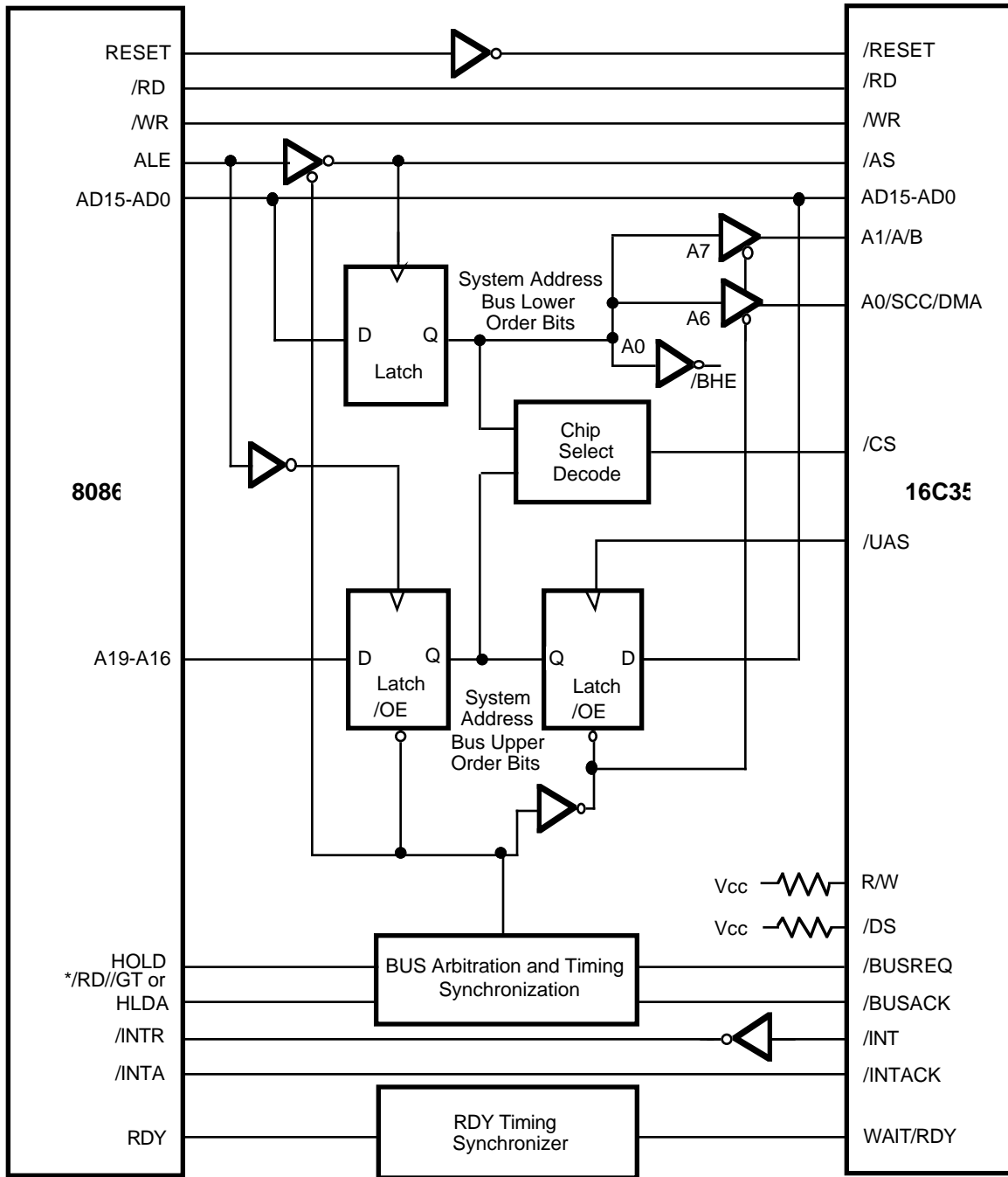


Figure 3. ISCC Interface to a 68000 Microprocessor

APPLICATIONS EXAMPLES (Continued)



* maximum mode

Figure 4. ISCC Interface to an Intel 8086 Microprocessor

When the ISCC becomes a bus master during DMA operations, RD and WR of the 8086 are tri-stated which allows the corresponding ISCC signals to control the bus transactions. The sense of RESET reverses, so the ISCC RESET signal inverts from the reset applied to the 8086 from the clock state generator.

RD/WR and DS of the ISCC are inactive in this application and tie high. They tie high through independent pull-ups since these signals become active when the ISCC is bus master during DMA transactions.

Assuming other devices in the system, the ISCC chip enable input (CE) activates from a decode of the address. In this example, the ISCC internally decodes addresses A1 through A5 and uses A6 and A7, externally. Thus, the address decode circuitry decodes address lines A0 and A8 and above. The decode of A0 for chip enable places the ISCC as an 8-bit peripheral on the lower byte of the bus. A0 and the upper level address lines (including A6 and A7) demultiplex from the 8086 address/data bus through a latch strobed by ALE.

The demultiplexed addresses A6 and A7 connect to A0/SCC/DMA and A1/A/B, respectively, of the ISCC to control selection of the DMA and SCC channels A and B. This connects through the tri-state drivers. They enable when the 8086 is the bus master and disable when the ISCC is bus master. This prevents the ISCC from improperly driving the system address bus since A0/SCC/DMA and A1/A/B become active outputs when the ISCC is the bus master.

The address map for the ISCC appears in Table A-6 for this application.

Table 44. ISCC Address Map

A0	A1-A5	A6	A7	Registers Addressed
1	x	x	x	ISCC not enabled
0	-	0	x	DMA Registers per A1 - A5
0	-	1	1	SCC Core Channel A Registers
0	-	1	0	SCC Core Channel B Registers

Since A0 specifies the lower byte of the bus and includes the chip enable decode, the internal ISCC register addresses decode without A0. Thus, Table 6 implies that the Left Shift address decode selection is made for both the SCC and DMA sections of the ISCC. The left shift selection is the default selection after reset. Left/Right Shift selection programming is discussed later.

The ALE signal of the 8086 applies to AS of the ISCC through an inverting tri-state buffer. The buffer disables when the ISCC becomes a bus master during DMA transactions. This prevents conflicts since ALE remains active even when the 8086 is in the HOLD mode during DMA transfers. Now, the ISCC AS is an active output. The address strobe for the demultiplexing latch of addresses A0 through A15 connects on the ISCC side of the ALE tri-state buffer. This allows the latch to serve two functions; to hold either the 8086 or the ISCC address when it is bus master.

After reset, ALE is active and the tri-state buffer enabled. This supplies address strobes to the ISCC. The presence of one of these address strobes, before writing to the BCR, programs the ISCC to the multiplexed bus mode of operation. The ISCC chip enable (CE) can be inactive and still recognize an address strobe (AS) before the BCR write (Figure 4 shows open latches when the input strobe is low).

When the ISCC is bus master during DMA transactions, BHE generates from A0. This is done from the output of the lower order address latch through an inverting tri-state driver. This driver enables only when the ISCC is the bus master. Whole word transfers are not done by the ISCC DMA, thus, BHE generated for the ISCC is always the inverse of A0.

The upper bus system address lines demultiplex from the 8086 and the ISCC in separate latches. Like the 68000 example, high order address lines from the ISCC latch via UAS (upper address strobe). The separate latches drive the same upper order address lines. A16 from the ISCC connects to the corresponding A16 address bus line as derived from the 8086. The output of the two latches alternately enable depending upon bus mastership.

The diagram shows INT from the ISCC connected to the 8086 INTR input via an inverter since these signals are of opposite sense. In actual practice, the ISCC interrupt request is first processed by an interrupt priority circuit. INTA (Interrupt Acknowledge) of the 8086 connects directly to the INTACK input of the ISCC. Conforming to the 8086 style of interrupt acknowledge, the ISCC is programmed to the Double Pulse Interrupt Acknowledge type. When this selection occurs, the ISCC responds to two interrupt acknowledge pulses. The first pulse is recognized but no action follows. The second pulse causes the ISCC to go active on the data bus and return the interrupt vector to the CPU. This action also takes place with the Single Pulse Interrupt Acknowledge type selection, except that the bus goes active with the first and only interrupt acknowledge pulse.

To start, the BCR write (first write to the ISCC after RESET) is done with A7 = 1 (A1/A/B ISCC input at logic high). This selects the wait option of the WAIT/RDY signal to conform to the 8086 bus style. The AS signal programming of the multiplexed bus was covered earlier. The BCR is written with 86H to enable byte swapping, select the sense of the byte swapping with respect to A0 (appropriate to this bus style), and select the Double Pulse type of interrupt acknowledge.

When the ISCC™ begins DMA transfers, it communicates requests for the bus through BUSREQ and BUSACK. The 8086 receives and grants bus requests through HOLD and HLDA in the minimum mode and through RQ/GT in the maximum mode. Depending upon the system requirements, there could be more than one potential bus

master. Therefore, there is a requirement for a bus arbitration circuit.

The minimum mode connection is relatively straightforward. The maximum mode configuration requires a translation of the ISCC BUSREQ and BUSACK signals into/from the 8086 RQ/GT timed pulse style of handshake. Refer to the information on the 8086 for detailed application information.

The ISCC™ WAIT/RDY output is compatible with the 8086 clock generator RDY input except that one edge of the signal must be synchronous with the 8086 clock. The synchronization occurs through external circuitry. Refer to the information on the 8086 for detailed application information.