REVISIONS			
APPROVED	DATE	CHANGE NOTICE	REV
J.A.	5/9/85	PCN PDC-004	P1
[illegible]	11/12/85	PCN PDC-102	P2

```
**********************
*                    *
*     PRELIMINARY    *
*                    *
**********************
```

ICM-32xx MiniBus Interface Chip

WRITTEN BY: Jim Anderson	DOCUMENT NUMBER	REV
NATIONAL SEMICONDUCTOR CORPORATION 15201 GREENBRIER PARKWAY BEAVERTON, OREGON 97006	426600020-000	P2
	PAGE 1 OF 45	

1. Introduction

The NSC 6224 CMOS gate array provides the logic density required with appropriate power, package size, and cost parameters to implement an LSI solution to a modern, multi-processor system bus. The MiniBus Interface Chip (MBIC) exploits this technology to provide a peripheral chip to the NSC 32000 micro-processor family which mates the local processor bus to the more complex discipline of a system bus.

The reader should also refer to the MiniBus specification (426600019-000) for additional information of the complete bus as implemented on a system level.

As well as providing the MiniBus interface, the MBIC also provides a set of services to ease the implementation of the local processor:

- Address Decoders

  Distinguish Local and MiniBus address spaces

  Provide software controllable external access to local memory

- Memory control signals

  Output Enable and byte write signals to control local memory

- Interrupt Service

  Combines INT11-INT00, 4 addressable interrupts, and 6 error interrupts, with an interrupt mask to provide NMI and normal interrupt requests to the local circuitry.

- Wait state generation

  Generates the wait states necessary for proper synchronization of the CPU when performing MiniBus accesses.

- Dual Port Memory Support

  Provides for quasi dual-ported memory without external components. Allows support of full dual-ported memory if implemented on local board.

- Implements MBIC Physical Address Space

  Physical addressing of MiniBus masters allows for four software addressable interrupt flip-flops, two CSR registers, accessible from local or remote processors.

- Access restrictions user state

  User mode access to system memory may be allowed or denied based on the state of a bit in the CSR of the MBIC. Access to the internal MBIC registers and the I/O space of the bus is always restricted to supervisor mode.

The pinout configuration of the MBIC device is shown in Figure 1.1, and the basic internal data paths of the MBIC are shown in Figure 1.2. All data I/O to the device is done through transparent latches to allow the data to be latched when required.

The MBIC provides two distinct address decode mechanisms to allow simultaneous tracking of requests which may require service from either the local processor or an external unit accessing the board across the MiniBus. All necessary circuitry to tie-break and serialize simultaneous requests is included in the MBIC.

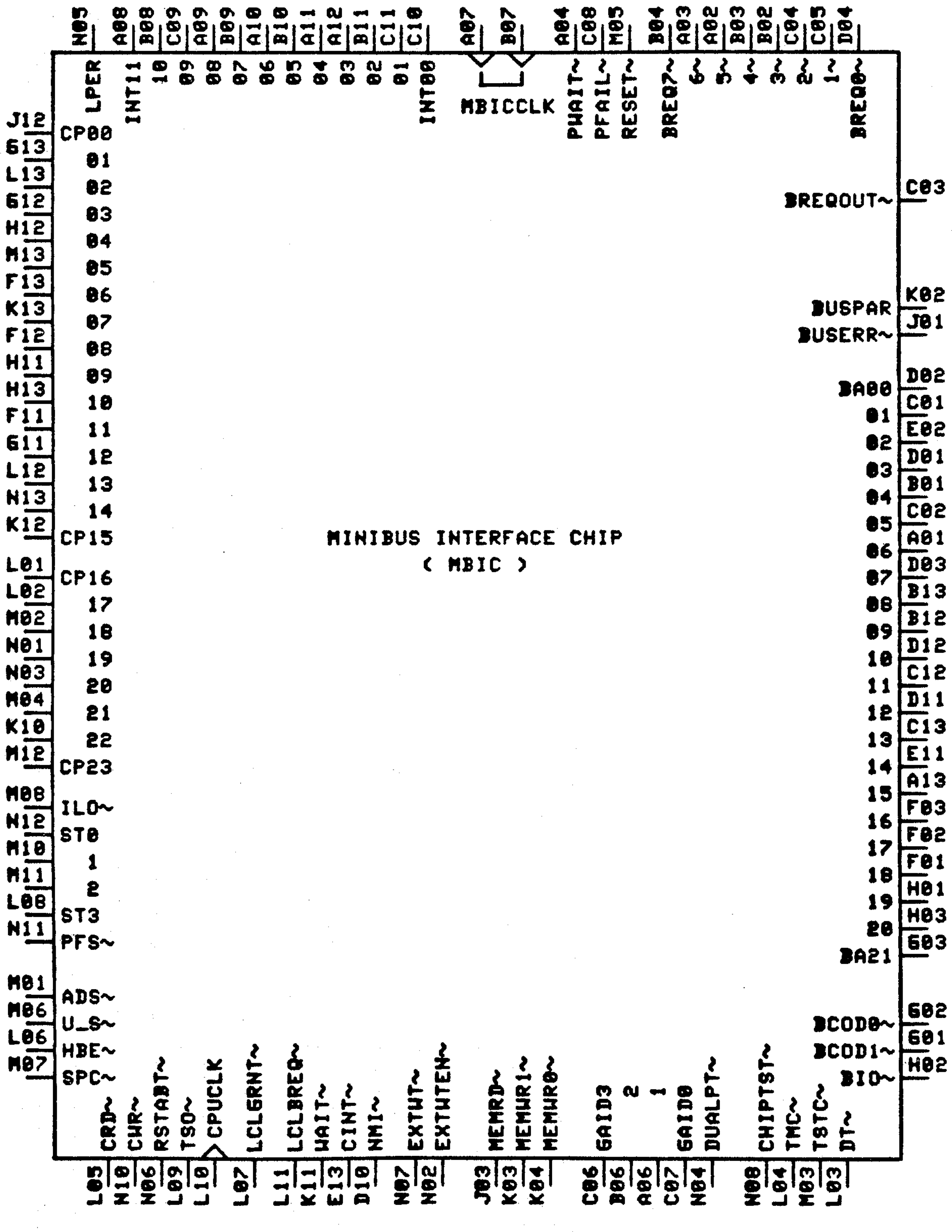
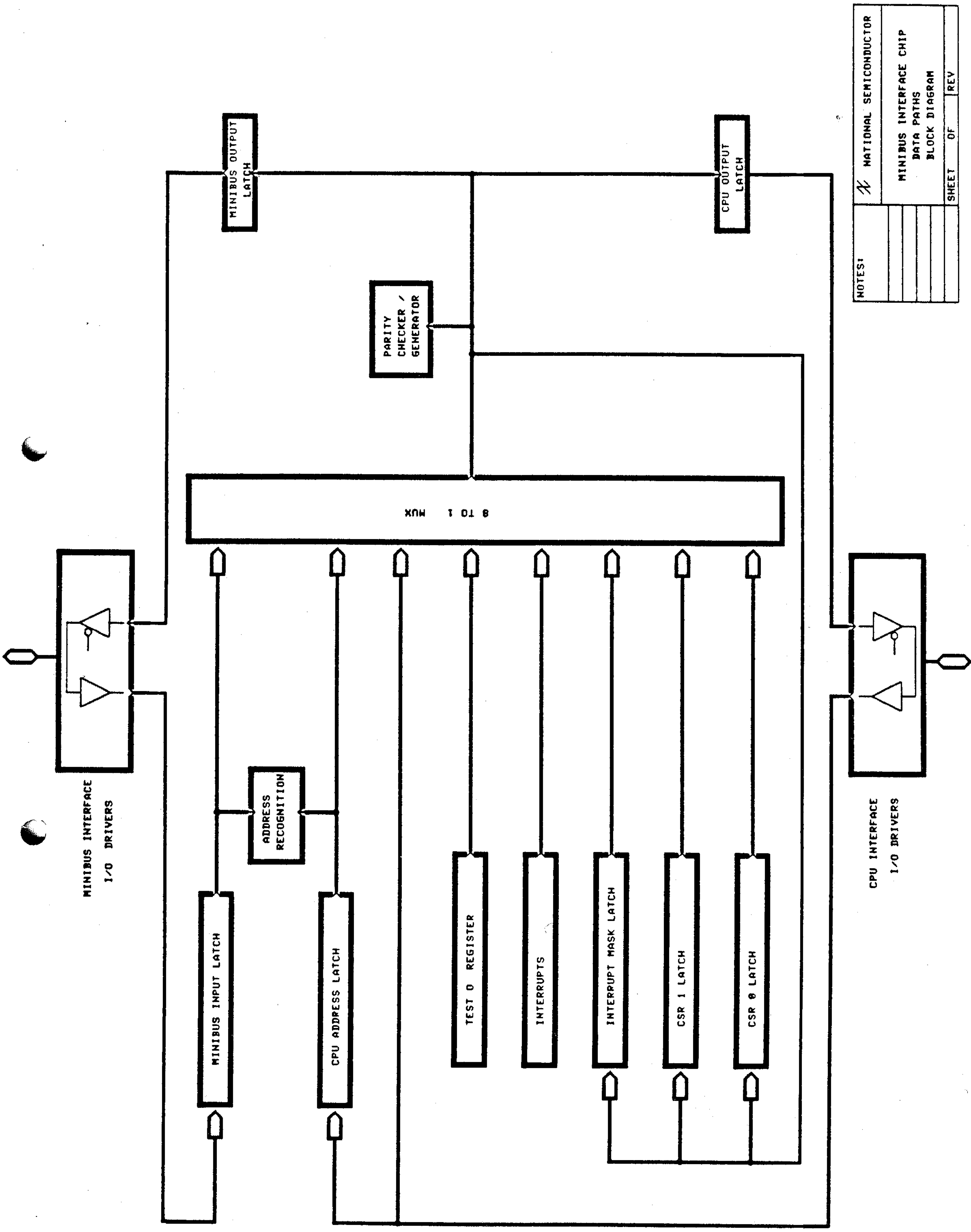


Figure 1.1

MiniBus Interface Chip


NATIONAL SEMICONDUCTOR
MINIBUS INTERFACE CHIP
DATA PATHS
BLOCK DIAGRAM
SHEET
OF
REV
NOTES:
MINIBUS OUTPUT LATCH
CPU OUTPUT LATCH
PARITY CHECKER / GENERATOR
8 TO 1 MUX
MINIBUS INTERFACE I/O DRIVERS
CPU INTERFACE I/O DRIVERS
ADDRESS RECOGNITION
MINIBUS INPUT LATCH
CPU ADDRESS LATCH
TEST 0 REGISTER
INTERRUPTS
INTERRUPT MASK LATCH
CSR 1 LATCH
CSR 0 LATCH

2. General Description

2.1. Signal Groups

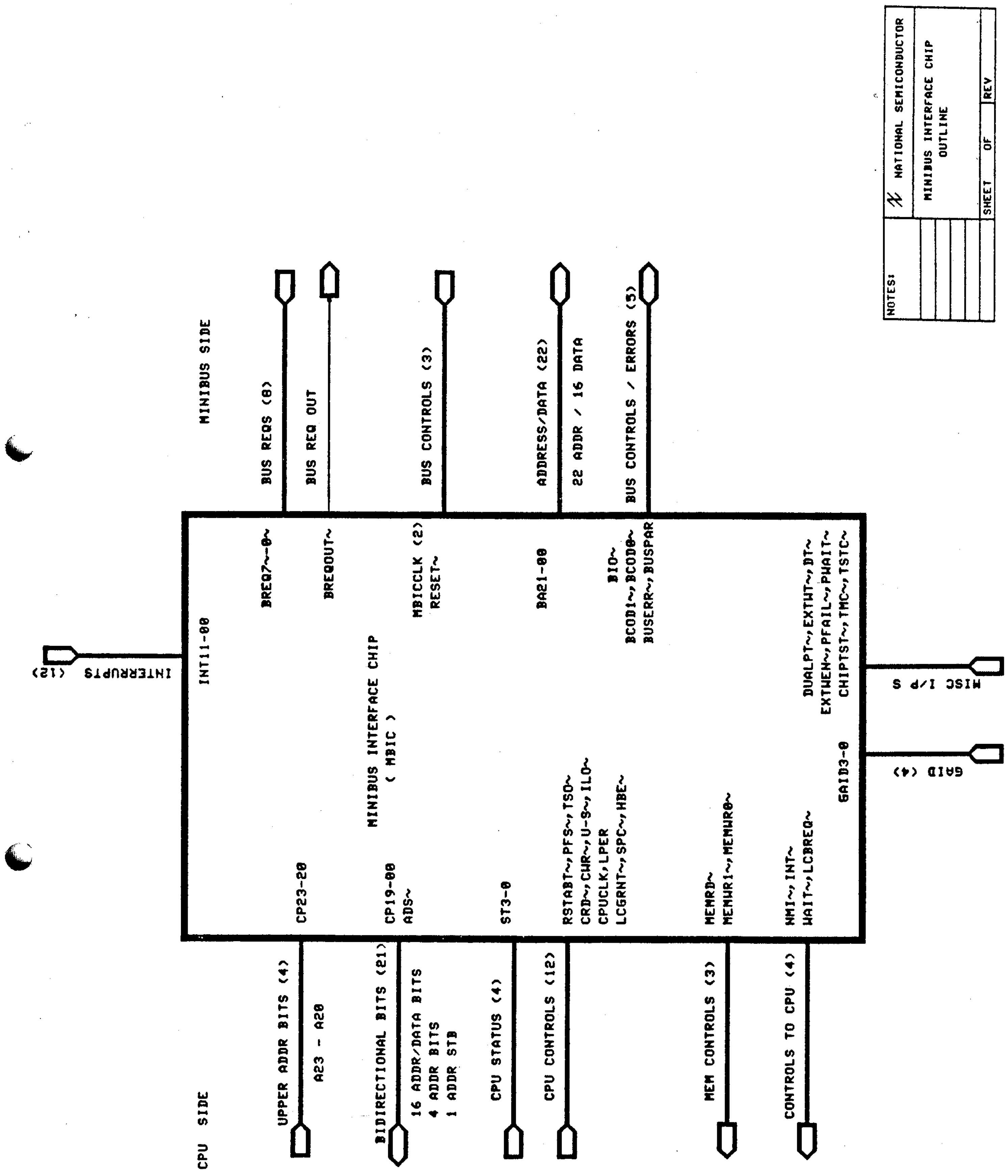
The MBIC signals fall into several distinct groupings related to their use in the bus implementation. These groupings are illustrated in Figure 2.1, and shown below in Table 2.1

BUS SECTION -		
	Inputs - (25)	MBICCLK (2) RESET~ PWAIT~ INT11-00 BREQ7~ - BREQ0~ PFAIL~
	Bi-Directional - (27)	BA21-00 BIO~ BCOD1~, BCOD0~ BUSPAR, BUSERR~
	Outputs - (1)	BREQOUT~
CONFIGURATION SECTION -		
	Inputs - (7)	GAID3 - GAID0 EXTWTEN~, EXTWT~ DUALPT~
LOCAL BUS SECTION -		
	Inputs - (20)	CPUCLK LCLGRNT~ HBE~ U_S~ ILO~, RSTABT~ CP23-20 CRD~, CWR~ SPC~, TSO~, PFS~ LPER ST3-ST0
	Bi-Directional - (21)	CP19-00 ADS~
	Outputs - (7)	MEMRD~ MEMWR1~, MEMWR0~ LCLBREQ~ CINT~, NMI~ WAIT~
MISCELLANEOUS -		
	Inputs - (2)	BDTEST CHIPTST~
TOTALS -		
	Inputs - 54, Bi-Directional - 48, Outputs 8	

Table 2.1

MiniBus Interface Chip Pinouts

The interpretation of the "system bus" signals has been covered in the MiniBus specification (426600019-000). Following is a brief description of the signals associated with the local implementation for which the MBIC supplies the MiniBus interface function.


NOTES:
NATIONAL SEMICONDUCTOR
MINIBUS INTERFACE CHIP
OUTLINE
SHEET
OF
REV
MINIBUS SIDE
BUS REQS (8)
BUS REQ OUT
BUS CONTROLS (3)
ADDRESS/DATA (22)
22 ADDR / 16 DATA
BUS CONTROLS / ERRORS (5)
BREQ7~-0~
BREQOUT~
MBICCLK (2)
RESET~
BA21-00
BIO~
BCOD1~,BCOD0~
BUSERR~,BUSPAR
INTERRUPTS (12)
INT11-00
MINIBUS INTERFACE CHIP
(MBIC)
DUALPT~,EXTWT~,DT~
EXTWEN~,PFAIL~,PWAIT~
CHIPTST~,TMC~,TSTC~
MISC I/P S
GAID3-0
GAID (4)
CP23-20
CP19-00
ADS~
ST3-0
RSTABT~,PFS~,TSD~
CRD~,CWR~,U-S~,ILO~
CPUCLK,LPER
LCGRNT~,SPC~,HBE~
MEMRD~
MEMWR1~,MEMWR0~
NMI~,INT~
WAIT~,LCBREQ~
CPU SIDE
UPPER ADDR BITS (4)
A23 - A20
BIDIRECTIONAL BITS (21)
16 ADDR/DATA BITS
4 ADDR BITS
1 ADDR STB
CPU STATUS (4)
CPU CONTROLS (12)
MEM CONTROLS (3)
CONTROLS TO CPU (4)

2.2. Pinouts - MiniBus Side

The pinout of the MBIC associated with the bus section of the component are listed below:

MiniBus Interface Chip Pinout (Bus Side)

Signal	Cell #	Type (D)	Die Pad	124 PCC & Daisy pin	124 PGA & Daisy /Number
BA00	M11I013	I/O	37	116	D02
BA01	M12I013	I/O	36	115	C01
BA02	M13I013	I/O	39	118	E02
BA03	M14I013	I/O	38	117	D01
BA04	M15I013	I/O	35	114	B01
BA05	M16I013	I/O	34	113	C02
BA06	M17I013	I/O	33	112	A01
BA07	M18I013	I/O	32	111	D03
BA08	M19I013	I/O	114	73	B13
BA09	M1AI013	I/O	117	76	B12
BA10	M1BI013	I/O	112	71	D12
BA11	M1CI013	I/O	115	74	C12
BA12	M1DI013	I/O	119	78	D11
BA13	M1EI013	I/O	113	72	C13
BA14	M1FI013	I/O	118	77	E11
BA15	M1GI013	I/O	116	75	A13
BA16	M1HI013	I/O	44	123	F03
BA17	M1II013	I/O	42	121	F02
BA18	M1JI013	I/O	43	122	F01
BA19	M1KI013	I/O	122	3	H01
BA20	M1LI013	I/O	48	5	H03
BA21	M1MI013	I/O	121	2	G03
BCOD0~	M13I02	I/O	46	1	G02
BCOD1~	M12I02	I/O	45	124	G01
BIO~	M11I02	I/O	47	4	H02
BREQ0~	M31I4	I	30	109	D04
BREQ1~	M32I4	I	29	108	C05
BREQ2~	M33I4	I	28	107	C04
BREQ3~	M34I4	I	27	106	B02
BREQ4~	M35I4	I	26	105	B03
BREQ5~	M36I4	I	25	104	A02
BREQ6~	M37I4	I	24	103	A03
BREQ7~	M38I4	I	23	102	B04
BREQOUT~	M11I04	I/O (O)	31	110	C03
BUSERR~	M1NI013	I/O	49	6	J01
BUSPAR	M1OI013	I/O	52	9	K02
INT00	M3BI4	I	1	80	C10
INT01	M3CI4	I	2	81	C11
INT02	M3DI4	I	3	82	B11
INT03	M3EI4	I	4	83	A12
INT04	M3FI4	I	5	84	A11
INT05	M3GI4	I	6	85	B10
INT06	M3HI4	I	7	86	A10
INT07	M3II4	I	8	87	B09
INT08	M3JI4	I	9	88	A09
INT09	M3KI4	I	10	89	C09
INT10	M3LI4	I	11	90	B08
INT11	M3MI4	I	12	91	A08
MBICCLK	M39I4	I	14	93	A07
MBICCLK	M3NI4	I	15	94	B07
PFAIL~	M41I4	I	13	92	C08
PWAIT~	M42I4	I	22	101	A04
RESET~	M3AI4	I	68	25	M05

The usage of these signals can best be understood from the MiniBus specification document. The only differences from the signals as specified in that document are as follows:

MBICCLK~	Two MBIC inputs are tied to this signal to minimize internal skew of clock distribution within the component.
BREQOUT~	The output line BREQOUT~ from the MBIC must be jumpered on the board to the MiniBus line BREQn~, where the "n" corresponds to the address set on the GAID2, GAID1, GAIDO inputs if the MBIC is on a processor board (i.e. becomes bus master).

2.3. Pinouts - Power and Test

MiniBus Interface Chip Pinout

Signal	Cell #	Type (D)	Die Pad	124 PCC & Daisy pin	124 PGA & Daisy /Number
DT	M4EI3	I	62	19	L03
CHIPTST~	M4MI3	I	77	34	N08
TMC~			61	18	L04
TSTC~			63	20	M03
+5V (VDD)			21	100	B05
+5V (VDD)			40	119	E01
+5V (VDD)			50	7	J02
+5V (VDD)			80	37	N09
+5V (VDD)			100	57	J13
+5V (VDD)			110	69	E12
GND (VSS)			20	99	A05
GND (VSS)			41	120	E03
GND (VSS)			51	8	K01
GND (VSS)			81	38	M09
GND (VSS)			101	58	J11
GND (VSS)			111	70	D13

2.4. Pinouts - Local Board Side

The pinouts of the MBIC associated with the in-board logic are listed below.

MiniBus Interface Chip Pinout - (Board Side)

Signal	Cell #	Type (D)	Die Pad	124 PCC & Daisy pin	124 PGA & Daisy /Number
CP00	M23I012	I/O	99	56	J12
CP01	M24I012	I/O	105	62	G13
CP02	M25I012	I/O	96	53	L13
CP03	M26I012	I/O	106	63	G12
CP04	M27I012	I/O	102	59	H12
CP05	M28I012	I/O	95	52	M13
CP06	M29I012	I/O	124	65	F13
CP07	M2AI012	I/O	98	55	K13
CP08	M2BI012	I/O	107	66	F12
CP09	M2CI012	I/O	104	61	H11
CP10	M2DI012	I/O	103	60	H13
CP11	M2EI012	I/O	108	67	F11
CP12	M2FI012	I/O	123	64	G11
CP13	M2GI012	I/O	94	51	L12
CP14	M2HI012	I/O	93	50	N13
CP15	M2II012	I/O	97	54	K12
CP16	M2JI012	I/O	53	10	L01
CP17	M2PI012	I/O	55	12	L02
CP18	M2QI012	I/O	57	14	M02
CP19	M2RI012	I/O	56	13	N01
CP20	M4GI1	I	65	22	N03
CP21	M4HI1	I	66	23	M04
CP22	M4II1	I	90	47	K10
CP23	M4JI1	I	87	44	M12
CPUCLK	M4II3	I	88	45	L10
ADS~	M2LI012	I/O	54	11	M01
WAIT~	M24I04	I/O (O)	92	49	K11
CINT~	M27I04	I/O (O)	109	68	E13
CRD~	M4NI1	I	70	27	L05
CWR~	M4OI1	I	82	39	N10
HBE~	M4LI1	I	73	30	L06
ILO~	M4MI1	I	78	35	M08
LCLBREQ~	M25I04	I/O (O)	91	48	L11
LCLGRNT~	M4KI1	I	76	33	L07
MEMRD~	M23I04	I/O (O)	58	15	J03
MEMWR0~	M22I04	I/O (O)	60	17	K04
MEMWR1~	M21I04	I/O (O)	59	16	K03
NMI~	M26I04	I/O (O)	120	79	D10
PFS~	M45I1	I	84	41	N11
TSO~	M4QI1	I	89	46	L09
DUALPT~	M4EI1	I	67	24	N04
EXTWTEN~	M48I1	I	64	21	N02
EXTWT~	M49I1	I	74	31	N07
GAID0	M41I1	I	16	95	C07
GAID1	M42I1	I	17	96	A06
GAID2	M43I1	I	18	97	B06
GAID3	M44I1	I	19	98	C06
LPER	M47I1	I	69	26	N05
RSTABT~	M46I1	I	72	29	N06
SPC~	M4FI1	I	75	32	M07
ST0	M4DI1	I	85	42	N12
ST1	M4CI1	I	83	40	M10
ST2	M4BI1	I	86	43	M11
ST3	M4AI1	I	79	36	L08
U_S~	M4PI1	I	71	28	M06

2.5. Signal Descriptions

CP15 - CP00

These 16 lines are multiplexed address/data from the MBIC to the local memory complex. If an on-board CPU is implemented, the lines would normally be directly wired to the CPU bus.

CPU control - These lines are inputs to the MBIC and provide the 16 LSB's of the address (latched internally by ADS). When a MiniBus cycle is requested, these lines form a bi-directional data bus appropriate for the transfer type requested by the CPU.

MBIC control -During the address portion of a MiniBus/MBIC initiated memory cycle, CPA15-01 contain the word address of the desired data and CPA00 will indicate the cycle type (i.e. 1 for read, 0 for write). During the data portion, these lines are again configured for the appropriate transfer direction.

(NOTE: The data bus transfers 16-bit quantities to/from the MiniBus. No byte alignment is performed. Byte operations are allowed via the control of the individual byte enables.)

CPA19 - CPA16

CPU control - These lines are address inputs to the MBIC and are NOT internally latched.

MBIC control -The MBIC will drive the corresponding address bits from the MiniBus onto these lines during the entire memory cycle.

(NOTE: The MBIC receives and decodes 22 bits of address on incoming bus transfers, but can only provide 20 bits of internal address lines. The 20 bits presented internally are formed from simple truncation of the incoming address, and the board designer must make provisions externally if any transformation is required.)

CPA23 - CPA20

These lines are input from a local CPU only, and are used to decode the memory requests from the CPU.

0 - 12 MB -> Local memory

The MBIC will provide the memory control lines (MEMWR1~, MEMWR0~, MEMRD~) only.

12 - 16 MB -> Bus Address space

below FE 0000	-> MiniBus Memory
FE pp00	-> 8-bit I/O to port pp (data will be sent / rcvd on bits 7-0)
FF 0000 - FF 7FFF	-> 16-bit I/O
FF 8grr - FF Bgrr	-> MBIC g, reg rr
FF C000 - FF FFFF	-> ignored completely

(NOTE: Since these address lines are NOT driven by the MBIC in MiniBus initiated cycles, and the CPU will have tri-stated the lines, the board designer must take the appropriate action to avoid incorrect action based on the state of these lines when the MBIC controls the internal bus.)

ADS~

CPU control - The MBIC uses the input strobe of ADS~ to latch the processor address internally and decode the address to determine if a MiniBus or local address is present.

MBIC control -During MiniBus initiated memory cycles, the MBIC will activate this signal to latch the address from CPA15-01 in the external latch.

MEMWR1~, MEMWR0~

These two output signals are intended to provide the write enable controls for each byte of a sixteen bit memory. They will be developed from the LSB of the CPU address, HBE~, and CWR~ inputs from the CPU if the address decode determines the CPU is accessing local memory. If the access is initiated by the MBIC, the information from the bus is used to generate these signals.

MEMRD~

This signal is intended to go directly to the output enable circuitry of the selected memory device. It will be activated either by a local memory decode of a CPU cycle, or a read access which is initiated by the MBIC.

LPER

This input line is used if the local memory has parity. It is sampled with the data on an external read, and will result in a local parity error being returned to the requestor on the MiniBus if active. No provision exists within the MBIC to detect or report parity errors on transactions between the local CPU and local memory.

EXTWTEN~, EXTWT~

These two input lines are used to inform the MBIC of the timing required by the local implementation of the memory. They are used only for timing accesses which originate from the MiniBus. If wait states are required during local accesses, it is the responsibility of the board designer to provide that functionality.

If EXTWTEN~ is 1, the MBIC will assume that the data is valid in time to transmit it down the bus in the cycle following the address out from the MBIC.

When EXTWTEN~ is activated (0), the timing is determined by the state of the EXTWT~ pin input. Activation of EXTWTEN~ will cause the MBIC to insert a clock period which samples the state of the EXTWT~ line at the leading edge of the clock. If the EXTWT~ line is found to be high, the MBIC will proceed to the return transfer at the next clock edge. If the EXTWT~ line is low, the MBIC will remain in the sampling state until it is found to be high.

In practice, allows the user the following options:

EXTWTEN~	EXTWT~	
1	x	Data can be returned to the bus in the bus clock following the address strobe
0	pullup	One bus clock will be allowed between the address out and the clock in which the data is returned.
0	logic	One bus clock is inserted without regard to the state of the EXTWT~ line. Additional bus clocks will be added as long as the logic input remains low.

WAIT~

This output is intended to go to the CWAIT~ pin of the TCU. It will be activated when the address decode from the CPU indicates other than a local memory access, and remain active until the external reference has been completed.

LCLBREQ~

This output will be activated by the MBIC when an incoming transfer on MiniBus has been latched which refers to the local memory array. It is expected to go to the HOLD~ pin of the CPU and the MBIC will await a local grant in response to use the internal bus.

NMI~ , CINT~

These output lines are expected to be tied to the corresponding lines on the CPU, and are activated when the MBIC must cause an interrupt to occur.

ST3, ST2, ST1, ST0

These four input lines are tied to the ST3-0 lines of the CPU to inform the MBIC of the bus state.

LCLGRNT~

This line is activated by the local processor circuitry to acknowledge the MBIC request for the local bus (requested by LCLBREQ~).

HBE~

This line is input from the appropriate CPU signal to distinguish word/byte write activity.

ILO~

This input signal indicates that the CPU desires an interlocked access. WHETHER OR NOT the associated request is for local memory, the MBIC will activate the WAIT~ signal until it can become bus master. It will keep the bus until the ILO signal returns to the inactive state.

Any slave selected on the bus must preserve integrity of the connection until the master allows bus arbitration to take place. If the slave has a local CPU, the CPU remains in HOLD mode until arbitration is allowed. This requirement is necessary to allow interlocked operations to occur as an indivisible operation. Since a processor board would normally allow arbitration on ALL BUT interlocked operations, this requirement has negligible impact on system performance.

CPUCLK~

This input signal should be tied to CTTL and is used to provide timing reference for the CPU signals.

CRD~ , CWR~

These input signals are expected to be tied to the TCU and will be used to determine the access type. The local memory control outputs will also derive their timing from these signals.

U S~

The user/supervisor input from the CPU is interpreted by the MBIC to limit the access rights of non-privileged users.

SPC~ , TSO~, PFS~, RSTABT~

These four CPU signals go directly to the kludge circuits in the interrupt and hold sections to avoid the deficiencies in certain early 32016 components.

GAID3 - GAID0

GAID3 should be pulled up for a memory board and grounded for a CPU board. GAID2 - GAID0 provide the remainder of the physical address for the MBIC.

ON A CPU BOARD, GAID2 - GAID0 MUST CORRESPOND TO THE JUMPERING OF BREQOUT~ TO THE APPROPRIATE BREQn~ LINE. EACH MiniBus MUST HAVE A CPU AT GAID ADDRESS "0111" SINCE MiniBus REQUIRES A DEFAULT MASTER AT POWER-UP TIME.

DUALPT~

This pin should be grounded if the board designer has implemented the necessary circuitry to provide full dual-port access to the local memory. In that case, the MBIC WILL NOT report a RETRY if a CPU request for bus access is pending when an external reference is received, it will simply activate the local bus request and expect to gain the access.

TMC, DT, TSTC

These pins are available for test purposes. TMC is not bussed on the backplane to facilitate individual board control during operation in burn-in cages.

TMC	DT	TSTC	
0	x	1	All MBIC outputs tri-state
0	1	0	All MBIC outputs 0
0	0	0	All MBIC outputs 1
1	x	x	Normal Operation

3. Operation

The MBIC provides both a local interface to connect the local CPU to the resources available on the MiniBus, and a bus interface which makes certain local resources available to other units within the MiniBus community.

The local CPU operates on whatever time base is appropriate for the device used, and the MBIC performs synchronization to the bus clock when required. A penalty of 1/2 BCLK must be paid in synchronizing signals from the CPCLK to the bus clock, and a penalty of 1/2 CPCLK is paid when synchronizing from the bus clock to the CPCLK. This overhead is only incurred when the address decode requires access to MiniBus resources.

3.1. Bus Section

The MBIC on a given board will continuously monitor the state of the MiniBus to detect any transfers directed at that board from other units within the MiniBus community.

Two distinct transfer types are recognized and acted upon:

1) References to the physical address space assigned to this MBIC (via jumpers of

GAID3-GAID0).

The MBIC will recognize its address in physical address space and allow external read/write of the internal registers.

All references to physical space will be satisfied without possibility of bus conflicts. Physical space is internal to the MBIC itself, and simultaneous requests from MiniBus and the processor are simply serialized by the MBIC.

2) References which match the current setting of the bus alias in CSR0 (i.e. local memory)

When an incoming address refers to the local memory, the MBIC will attempt to gain the internal bus by activating the LCLBREQ~ signal, latching the address in the Bus Input Latch (BIL) for use when the internal bus is available, and returning a "wait" status to the requesting master. When the local circuitry responds to LCLBREQ~ with LCLGRNT~, the MBIC will proceed to output the address to the local memory subsystem, latching it externally with the ADS~ strobe. The memory is controlled via the MEMRD~, MEMWR0~, and MEMWR1~ lines to complete the access.

In the event that the LCLBREQ~, LCLGRNT~ lines are tied directly to the HOLD~ and HOLDACK~ lines of conventional microprocessors, it is possible that the CPU may not relinquish the internal bus if it has initiated a MiniBus access. The recovery from this stalemate is discussed in the MiniBus specification.

Once the MBIC has obtained the local bus through the use of the LCLBREQ~, LCLGRNT~ sequence, it will maintain control of the bus until the requesting master allows arbitration to occur on the MiniBus. This will preserve the integrity of read-modify-write cycles externally applied to the local memory.

3.2. CPU Section

MBIC also monitors the 24 bit address of the CPU and appropriate CPU control signals to decode the requirements of the memory cycles initiated by the CPU.

CPU addresses in the first 12 MB are considered "local" memory addresses and the MBIC simply provides the appropriate MEMRD~, MEMWR1~ and MEMWR0~ signals to the local memory.

Addresses above 12 MB are decoded as follows:

CPU Addresses	Mapped Function/Addresses
C0 0000 -> FD FFFF	Bus addr 00 0000 -> 3D FFFF (memory)
FE 0000 -> FE FFFF	8-bit I/O access
FF 0000 -> FF 7FFF	16-bit I/O access
FF 8000 -> FF BFFF	Physical space access
FF C000 -> FF FFFF	(ignored)

3.3. Interrupts

The MBIC interrupt circuitry will respond to three distinct categories of interrupt stimuli:

- o Software Interrupts -

  Four internal flip-flops may be set or reset by the local CPU, or any other MiniBus Master.

- o Bus Interrupts -

  An active request on any of the twelve interrupt lines defined on the MiniBus backplane.

- o Error Interrupts

  The MBIC itself may detect certain error conditions and set an appropriate interrupt request to notify the processor of the error.

An MBIC reference to physical address space from either the local CPU, or another CPU across the MiniBus may cause one of the four "software interrupts" to become active (set). An active hardware interrupt line which is connected to one of the twelve (12) MBIC interrupt lines (INT11-INT00) will also cause a interrupt request to be active.

The set of active interrupts are masked by the current state of the INTMSK register as described in that section.

The top two interrupts (Bus conflict abort and Power fail) are non-maskable and cause immediate hardware interrupts since it is hazardous to continue processing when these situations have been detected.

The MBIC does not interpret a "int/ack" sequence from the CPU. It will activate the interrupt line and keep it active until the interrupt generating condition is cleared either by satisfying the external demand or clearing the latch for conditions detected within the MBIC. NMI indication is reset when CPU references the INTREG (which would be the normal action to determine which interrupt is occurring). When the reference to INTREG occurs, the MBIC will set its internal master interrupt lockout f/f which deactivates the NMI line.

The software handler has the sole responsibility to make the raw request go away (either by servicing the peripheral or resetting the software interrupt or error latch) and resetting the master interrupt f/f to re-arm the NMI of the MBIC.

4. Externally Initiated Sequences

Timing sequences for MBIC sequences which are initiated from another MiniBus master are given below. For simplicity, signal transitions are shown aligned with the rising or falling edge of the bus clock, although actual logic delays in the circuitry skew the transitions from these edges. For accurate timing refer to the timing section of this document.

4.1. External Read of Local Memory

```
=================================================
           External Read of Local Memory
=================================================
          1    2    3    4    5    6   7    8
BCLK     |addr|    |    | a  |  b > c > d |    |
               -------------------------
slave    -----|                         |
                                         --------------
                         ------------------------------
held     ---------------|                ?if arb allowed
                                         --------------
                         -------------
CP19-16=================|             |================
                         -------------
                         ---
CP15-00=================|A|==========================
                         ---
         --------------- -----------------------------
ADS~                    | |
                         ---
         -----------------          ----------------
CRD~                      |        |
                           ---------
         ----------------------   ----------------
bolld~                         | |
                                ---

                    Figure 4.1
=================================================
```

Clk 1 - During this bus clock, the address is being transmitted from the MiniBus master. The MBIC address recognition circuitry decodes the address as matching it's current setting in the CSR.

Clk 2,3 - The MBIC becomes selected as a slave unit at the beginning of clock 2 as a result of the address recognition during clock 1. Assuming that there is a local CPU, the MBIC can do nothing but reply with a wait on the bus, until it can gain access to the local memory.

During clock 2 and 3, the MBIC synchronizes to the CPU clock cycle and activates LCLBREQ~ to gain control of the internal bus. The CPU responds at it's own rate, the resulting LCLGRNT~ signal is synchronized to the bus clock and the MBIC now is in control of the resources required to satisfy the external request. Depending on the CPU response, this operation could require additional bus clock cycles.

Clk 4 - The "held" signal internal to the MBIC becomes active and the MBIC begins the manipulation of the internal bus. The 20 l.s. bits of the received address are driven out the internal data bus, and ADS~ activated to control the local address latch.

At mid-clock, the MBIC de-activates the ADS~ signal and floats the data lines, activating the CRD~ signal to enable the outputs of the memory array. The internal gating of the MBIC routes the data returning from the array to the bus output latch (BOL).

Clk 5 - The timing as shown assumes that EXTWTEN~ is low and EXTWT~ is high. EXTWT~ is sampled into a f/f at the beginning of Clk 5, and the MBIC will repeat Clk 5 until the value sampled is a "1".

Clk 6 - During the first 1/2 of Clk 6, the BOL of the MBIC is open (flow-thru) and the returning data is gated onto the bus with the appropriate "data now" status to inform the master of the transfer. At the mid-clock, the data is latched in BOL, and CRD~ de-activated since the memory activity is now complete.

The MBIC will continue to hold the data on the bus until the status received from the Master indicates "ready", inserting additional clocks as required and maintaining the data in BOL.

Clk 7 - The MiniBus transaction has been completed, and the MBIC drops the slave f/f. The internal bus is still controlled by the MBIC (held) and will not be released until arbitration is allowed on the MiniBus. This delay in releasing the internal bus is required to assure the indivisibility of interlocked operations from the master.

Certain internal signals in the MBIC remain active during this clock period as the MBIC proceeds in an orderly termination of it's internal states. This clock period is usable for the orderly shutdown, since another address transfer is required to re-select the MBIC as a slave, and the earliest that that transfer can occur is during this clock which would re-initiate a slave sequence beginning in Clk 8.

4.2. External Write of Local Memory

The external write transaction is very similar to the read operation. Again, the synchronization delays the start of the actual operation until clock a, when the address is moved to the external latch as before. As soon as the address is latched the data is placed on the CP15-00 lines, and the appropriate CWR~ signals activated. The timing shown has one "wait" state.

```
====================================================
           External Write of Local Memory
====================================================
          1   2    3    4    5    6    7    8
BCLK   |addr|    |    | a  >  b |  c |    |    |
            ---------------------
slave       |                   |
       ------                   --------------------
                      ------------------------------
held                  |         |  ?if arb allowed
       ----------------         --------------------
                      ----------------
CP19-16===============|              |==============
                      ----------------
                      ----------------
CP15-00===============| A |    D     |==============
                      ----------------
       ---------------- ----------------------------
ADS~                  | |
                      ---
       ---------------- ---- -----------------------
colld~                | |  | |
                      ---  ---
       ---------------------          --------------
CWR~                       |         |
                           -----------
```

Figure 4.2

==

The write operation tracks the read operation through the first 1/2 of Clk 4, since the nature of obtaining the internal bus and presenting the address to local memory is independent of the direction of transfer. The master would normally be driving the data onto the bus from Clk 2 onward, observing the wait indication from the slave to maintain the data on the bus.

Clk 5 - The MBIC will activate the CWR~ signal to write the data to the memory array, and replace the address on CP15-00 with the data to be written.

4.3. External Read of MBIC Register

The external access to physical address space of the MBIC is slightly different, since the LCLBREQ~ sequence is not used. Instead, the next CPU bus cycle is extended by a "wait" to suspend CPU activity long enough to perform the access without conflict.

With the exception of the "wait", no external signals on the CPU side of the MBIC are affected by the transaction.

```
================================================
       External Read of MBIC Register
================================================
          1   2    3    4     5   6    7    8
BCLK   |addr|    |    | a  |  b | c  | d  |    |
            --------------------------
slave       |                        |
       ------                         -----------
                      --------------------------
bsync                 |              ?if no cpu req
or held----------------              -----------
       --------------------------  -------------
bolld~                          |  |
                                ----
```

Figure 4.3

```
================================================
```

Clk 1 - During this clock period, the address is being sent to the MBIC by the current bus master as it was in the other transactions.

Clk 2,3 - The MBIC becomes selected as slave, and must perform a tie-breaking operation with the current processor activity to obtain exclusive use of the internal structure of the MBIC. This may be achieved by synchronizing to the CPCLK and momentarily suspending CPU operation thru use of a "wait" in the CPU's next internal bus cycle. Operation will also proceed if the processor is "held" from a previous external reference or executing a WAIT instruction.

The MBIC will ALWAYS be able to gain access to the internal structure by one of the necessary mechanisms so no possibility of a conflict will exist. The timing of this operation is variable depending on the synchronization and CPU state, but is shown as a 2 clock penalty in the diagram.

Clk 4 -	During the first 1/2 of clock 4, the MBIC transfers the address from the bus input latch (BIL) to the internal MBIC register address latch. At the mid-point of the clock, the internal structure is set to gate the appropriate information to the bus.
Clk 5 -	No real activity occurs in clock 5 in a read operation.
Clk 6 -	During clock 6, the MBIC is returning the requested data to the master across the MiniBus with the "data now" status. As with any other data return to the master, the data will be preserved on the bus for subsequent clocks if the master is not indicating a "ready" state.
Clk 7 -	When the MBIC enters clock 7, it is no longer selected as slave and performs the orderly termination process as above.

4.4. External Write of MBIC Register

```
================================================
        External Write of MBIC Register
================================================
         1    2    3    4    5    6    7    8
BCLK   |addr|    |     |   >  b |  c |     |    |
            ---------------------
slave       |                   |
       ------                   -----------------
                      ---------------------------
bsync                  |        |if no cpu req
or held-----------------        -----------------
------------------------------- -----------------
gawrstb~                     |  |
                             ----

                  Figure 4.4
================================================
```

The external write of an MBIC register is analogous to the read operation with the direction of the data flow changed. The actual write operation into the f/f's in the MBIC takes place in the second 1/2 of clock 5.

5. CPU Initiated Sequences

5.1. CPU Read of External Memory

```
=============================================
           CPU Read of External Memory
=============================================
              1      2      3      4      5
BCLK       |      |  a   |  b   > c    |      |
           ---------------------
bsync            ?              |
           -------              --------------
           -----------------------------------
master           ?              ?
           -------              --------------
sample berr                              --|
           ------------------------------------
BA21-19   idle   ? Addr ? rdy   ? idle
           ------------------------------------
           ------------------------------------
BA18-16   idle   ? Addr ? SLV   ? idle
           ------------------------------------
           ------------------------------------
BA15-00   bol    ? Addr ? SLV   ?
           ------------------------------------

                  Figure 5.1
==============================================
```

When the MBIC finds that the address latched by ADS~ requires an access to MiniBus, it immediately issues a wait to the CPU until the bus can be obtained and the request satisfied. The MBIC does not distinguish between MiniBus accesses to physical address space and those to memory, except to recognize that portion of physical space which resides within this particular MBIC, and treat that as a MBIC read as shown below.

The request is synchronized to the bus clock, and the MBIC will enter the arbitration procedure to obtain bus mastership if it is not already the master.

Clk 2 - The first clock period after synchronization and bus mastership has been achieved is the address transfer. The MBIC drives the BIO~, BCOD1~ and BCOD0~ lines to 101 to indicate that an address transfer is on the bus, and drives the address from the CPU on BA21-01, with BA00 high to indicate that the transfer being initiated is a READ operation.

Clk 3 - On the clock immediately following the address transfer, the MBIC floats the BA18-00 lines and the BCOD0~ line, expecting the responding slave to drive them in response. If BCOD0~ is not driven low during this clock and all subsequent clocks of this transaction, no other return signals are interpreted and a NO-RESPONSE sequence is initiated.

Normally, the slave would drive

BCODO~ and BA18-16 to 0's during the clock following the address transfer, since a finite time is required to access the requested data to return across the bus. The master MBIC will maintain the state of the bus until the slave responds with the appropriate status to end the transfer.

During this clock state, the master will be allowing bus arbitration unless the ILO signal is active so that other masters requiring the bus can perform the arbitration procedure in parallel with the current transfer.

Clk 4 - The transition to clk 4 occurs at the trailing edge of clock 3 if both the master and slave are indicating a "rdy" status on the bus. The incoming data is latched into the bus input latch and during Clk 4 another master would own the bus if arbitration had caused a switch of masters.

At this point, the MBIC transfers the returning data to the computer output latch (COL) and re-synchronizes to the CPU clock to release the CPU wait line synchronously to the CP clock.

5.2. CPU Write of External Memory

```
=============================================
          CPU Write of External Memory
=============================================
           1      2      3      4      5
BCLK    |      |  a   |  b   >  c   |      |
        -----------------------
cbreq,bsync    ?             |
        --------             ----------------
        -------------------------------------
master         ?             ?
        --------             ----------------
                             --------
mtc            when slvrdy   >      |
      ------------------------      ---------
sample berr                  -------|
        --------  ------ --------------------
bolld~         |  |   |  |
               ----   ----

        -------------------------------------
ba21-19  idle  ? cal  ? ibb  ? idle
        -------------------------------------
        -------------------------------------
ba18-16  idle  ? cal  ? SLV  ? idle
        -------------------------------------
        -------------------------------------
ba15-00  bol   ? cal  ? ci   ?
        -------------------------------------

               Figure 5.2
=============================================
```

The bus write transaction initiated by the CPU proceeds thru the initial synchronization, obtaining bus mastership, and address out transfer exactly as the read operation, the only distinction being that BA00 contains a 0 during the transfer of the address to indicate a write operation is being initiated.

Clk 3 - During clock 3, the MBIC floats only BA18-16 and BCOD0~ for the slave to drive, driving the data to be written on BA15-00. The MBIC must preserve the data on the bus until a return status from the slave indicates a "rdy" to inform the master that the data has been accepted.

The master is also driving the appropriate code on BA21-19 to control the write procedure.

Clk 4 - As for the read, the MBIC proceeds with the re-synchronization to the processor to release the "wait".

At the end of this clock, the MBIC samples the BERR~ line which would be activated by the slave unit at this point if the data received in the last clock had a parity error. An interrupt to the CPU would be generated if that occurred.

5.3. CPU Read of Local MBIC Register

```
======================================
   CPU Read of Local MBIC Register
======================================
            1      2      3      4
CPCLK   .      .      .   b  .      .
               ...............
waitc          |             |
        .......               .........

                   ...........
garegsel          |          |
        ..........            .........

                Figure 5.3
=======================================
```

The MBIC will detect references to MiniBus address space which are intended for the internal registers of the MBIC itself, addressed either by its physical address or by its local alias (physical address "F"). and perform those accesses as required. No bus mastership or synchronization is required, but the MBIC will introduce two wait clocks to the processor as the accesses are performed.

5.4. CPU Write of Local MBIC Register

```
======================================
   CPU Write of Local MBIC Register
======================================
            1      2      3      4
CPCLK   .      .      .   b  .      .
               ...............
waitc          |             |
        .......               .........
        ..............   ............
gawrstb~              |   |
                      ....

                Figure 5.4
=======================================
```

6. Asynchronous Bus Transfers

Not all potential members of the bus community require the sophistication of a complex transfer discipline as has been presented above. It is important to be able to implement simple devices requiring low bandwidth in a simple manner. ALL MiniBus MASTERS are required to implement the full synchronous discipline, but SLAVE ONLY members of the MiniBus family can be added as asynchronous devices.

MiniBus implements an asynchronous discipline which is modeled after the IN and OUT instruction sequences of popular 8-bit microprocessors. The user can connect a dumb slave to MiniBus as simply as to any of the low-performance busses of the past.

6.1. 8-bit I/O Read Cycle

```
==========================================================
                 8-Bit I/O Read Cycle
==========================================================
         1     2   3    4    5    6    7    8    9   10
BCLK   |    |    |    | c  | d  | e  | ex | f  | g  | h  |
       ----------------------------------------------
bsync       ?         ?                             |
       ----------------                              -----
       ---------------------------------------------------
master      ?         ?
       ----------------
       ----------------                                  -
BIO~                  |                                  |
                      ------------------------------------
       ---------------------------------------------------
BA21-19 idle?          <-----  "1",we1,we0 --------------->
       ---------------------------------------------------
       ---------------------------------------------------
BA18,BA17             ?
       ----------------
       -----------------------                       ------
BA16        ?         ?      |                      |
       ----------------      ------------------------
       ---------------------------------------------------
BA15-08               |<----------CPA 15-08--------------->
       ---------------------------------------------------
       ---------------------------------------------------
BA07-00               |   Input from slave device   ?
       ---------------------------------------------------
```

Figure 6.1

```
==========================================================
```

The 8-bit I/O cycle (Figure 6.1) is the result of the MBIC detecting the MiniBus 8-bit I/O address space as the target of the CPU initiated bus cycle. As with any other MiniBus access, the MBIC will issue a "wait" to the CPU and proceed to synchronize and gain bus mastership before proceeding any further.

Clk 4 -	The MBIC has gained the bus and activates the BIO~line placing bits 15-08 from the address latch on BA15-08. The MBIC floats BA07-00 since the I/O operation is a read. All other MBIC's will simply ignore the bus contents whenever BIO~ is

active.

Clk 5 -	At the mid-point of this clock, (1.5 bus clocks allowed for address setup), the MBIC will activate A16 which will be interpreted by the slave as the RD~ signal.
Clk 6 - Clk 7	No change occurs, this is simply a fixed timing to allow the slave to respond.
Clk 8 -	This is another timing clock, but at the trailing edge of this time period, the PWAIT~ line is sampled to determine if the addressed slave is indicating that it requires additional time to complete the transfer. The MBIC will continue in this state until it finds the PWAIT~ signal inactive.
Clk 9 -	The returning data will be latched in the MBIC at the end of this clock, and A16 will be de-activated at the trailing edge.
Clk 10 -	During clock 10, the MBIC continues the address and BIO~to assure that the RD~ signal (A16) is seen as inactive by all dumb slaves before the address is allowed to change. The re-synchronization to the CPU occurs to deliver the data.
Note 1:	No bus arbitration is allowed during an I/O cycle, the MBIC will allow arbitration (if ILO inactive) in the clock period following the I/O cycle (a bus idle clock).
Note 2:	No data or address swapping is performed in the MBIC, so the processor must address the I/O ports as every 256th address in the 64K range of 8-bit I/O. The CPU must also provide/receive the data on bits 07-00.

6.2. 8-bit I/O Write Cycle

```
==========================================================
           8-Bit I/O Write Cycle
==========================================================
         1    2    3    4    5    6    7    8    9   10
BCLK   |    |    |    | c  | d  | e  | ex | f  | g  | h  |
       ---------------------------------------------
bsync       ?         ?                             |
       ----------------                             ------
       --------------------------------------------------
master      ?         ?
       ----------------
       ----------------------                       ------
wrlatch~                     |                      |
                             ------------------------
       --------------------------------------------------
a21-19  idle?         ?<----- "1",we1,we0---------------->
       --------------------------------------------------
       --------------------------------------------------
a18              ?
       -------------
       ----------------------                       ------
a17         ?         |      |                      |
       ----------------      ------------------------
       --------------------------------------------------
BA15-08               |<-----------CPA 15-08 ------------>
       --------------------------------------------------
       --------------------------------------------------
BA07-00               |<-----------CPD 07-00 ------------>
       --------------------------------------------------

                       Figure 6.2
==========================================================
```

The 8-bit I/O read transfer is identical to the write, except for the activation of A17 (WR~) and the direction of BA07-00 during the transfer.

6.3. 16-Bit I/O Read Cycle

```
==========================================================
                16-Bit I/O Read Cycle
==========================================================
              1    2    3    4    5    6    7    8    9   10
BCLK       |    | a  | b  | c  | d  | e  | ex | f  | g  | h  |
           ------------------------------------------------
cbreq,bsync ?                                          |
           -----                                        -----
           ---------------------------------------------------
master         ?
           -----
                -------------------------------------------
mta            |                                           |
           -----                                           -
           ----- -------- -----------------------------------
bolld~          | |       | |
                ---       ---
           ----------------------                       -----
rdlatch~                        |                       |
                                -------------------------
           -------------- ------------------------------ ----
colld~                   | |                           | |
                         ---                           ---
           ---------------------------------------------------
a21-19      idle?<-------------- "1",we1,we0 --------------->
           ---------------------------------------------------
                ---    -----------------------------------
a18            ? |     |
           ----- -----
           ----------------------                       -----
a16            ?                |                       |
           -----                -------------------------
           ---------------------------------------------------
a15-00 bol  ?<- addr >?<--------- I/O board ----------->
           ---------------------------------------------------

                          Figure 6.3
==========================================================
```

To provide the same type of asynchronous interface for devices which require a 16-bit data path, the MBIC also allows the extended I/O cycles shown in figures 6.3 and 6.4. This mode of operation takes the first 2 clocks to transmit a multiplexed address which is expected to be latched by the slave device. During the remainder of the operation, the BA15-00 lines are then configured as a 16-bit data path.

Once the address has been transmitted and latched, the timing of the remainder of the operation is identical to the 8-bit I/O operations.

6.4. 16-Bit I/O Write Cycle

```
==========================================================
                  16-Bit I/O Write Cycle
==========================================================
          1    2    3    4    5    6    7    8    9   10
BCLK    |    | a  | b  | c  | d  | e  | ex | f  | g  | h  |
        ----------------------------------------------
cbreq,bsync ?                                        |
        -----                                        ------

        ---------------------------------------------------
master      ?
        -----

        -----  --------  ----------------------------------
bolld~       | |       | |
             ---       ---

        ----------------------                       ------
wrlatch~                      |                      |
                              ------------------------

        ---------------------------------------------  ----
colld~                                               | |
                                                     ---

        ---------------------------------------------------
a21-19   idle?<-------------- "1",we1,we0 --------------->
        ---------------------------------------------------
             ---    ---------------------------------------
a18          ? |    |
        ------ ------
        -----------------------                      ------
a17          ?                |                      |
        ------                ------------------------
        ---------------------------------------------------
a15-00   bol ?<- addr >?----------- data --------------->
        ---------------------------------------------------

                        Figure 6.4
==========================================================
```

6.5. Programmed I/O Assist Features

A processor which attempts to directly control a "dumb" peripheral unit on the MiniBus will require a significantly longer access time than with a local peripheral, due to arbitration, synchronization, and other penalties associated with a common bus. "Polling" of a peripheral unit will also congest the bus unnecessarily if the processor is looping to determine a change of state in a peripheral signal.

The MBIC provides two mechanisms to attempt to alleviate this problem.

1) Reading the TESTREG within the MBIC will allow software to directly examine the state of three interrupt lines (INT09, INT08, INT07). Testing the state of these interrupt lines does not require the MBIC to become MiniBus master, or to synchronize to the bus clock.

2) Writing to the TESTREG within the MBIC is a uniquely identified instruction which implements a form of "External Wait". The data pattern associated with the write instruction provides a mask to the MBIC and the MBIC will keep the CPU in a "wait" state (i.e. not completing the write

instruction) until one of the following occurs:

INT10 & D11 are true
OR
INT11 & D12 are true
OR
An unmasked interrupt becomes active
OR
The MBIC is addressed as a slave by another MiniBus master.

The MBIC does not become master or affect bus traffic while in this "wait". The intent of this mechanism is for software to be able to "wait" (not loop) on a particular signal, using an interrupt for timeout (or abort). The forced release when another MiniBus master selects the MBIC as slave is required to prevent erroneous software from locking up the MiniBus by placing its MBIC in this mode.

7. Physical Address Space Implementation

MBIC provides five (5) internal 16-bit registers that are accessible both from the local CPU and other MiniBus masters. All but one are both writeable and readable. The registers are:

Address				
CPU	BUS	Register		
FF 8g00	3F 8g00	INTREG	Interrupt register	r/w
FF 8g02	3F 8g02	IMSKREG	Interrupt mask reg.	r/w
FF 8g04	3F 8g04	CSR0	Control & Status reg.	r/w
FF 8g06	3F 8g06	CSR1	Control & Status reg.	r/w
FF 8g08	3F 8g08	TST0	Test register	r.o.

where "g" is 0 to E (MBIC ID) or "F" for local MBIC alias. The MBIC will respond to addresses directed at its internal registers from either side, and no conflict will exist if requests arrive at the same time. The resources internal to the MBIC are always under MBIC control and simultaneous accesses are simply serialized to satisfy both without difficulty.

8. Conflict Resolution

On any system which contains local memory in CPU cards and where the local memory is accessible to other system components, conflicts can occur when the CPU desires access to the system bus at the same time as the current bus master desires access to that CPU's local memory.

On MiniBus CPU cards which do not implement true dual-ported memory, these conflicts lead to a stalemate which must be resolved before system operation can continue. The conflicts are resolved as follows:

Conflicts on the bus are simultaneously detected by all system units (since all units continually monitor all bus transactions) by decoding the Retry code during a bus data cycle, as explained in the previous section.

Once a conflict has been detected, all units remove themselves from the conventional arbitration process until the conflict has been resolved. Bus mastership is transferred between units involved in the conflict by the simple rule that the unit issuing a retry code will become bus master in the next clock period.

During the conflict resolution, the arbitration lines are used to identify potential candidates for abort in the event that the conflict is irresolvable. A unit becomes a candidate for abort when it has given a retry code to another unit and has also received a retry itself within the same conflict sequence. Once identified as a candidate, the unit will drive its BREQ line to notify the other members of the MiniBus community of its candidacy. When a candidate unit is again referenced in the conflict sequence that unit will resolve the conflict by performing the local abort rather than re-issue the retry code if it is the highest priority candidate identified on the BREQ lines.

Normal arbitration resumes after the first successful data transfer over the bus (code other than "wait" or "retry" or an I/O transfer). A detailed description of the retry sequences is presented in the following section.

(Note: In the event of an abort, the highest priority CPU is the unit aborted to resolve the conflict. Since CPU's operate in a "fair arbitration" mode, the "priority" is really simply a tie-break mechanism only very loosely related to the service the processor receives.)

8.1. Retry Sequence

The actions of the MiniBus Interface Chips (MBICs) during a retry sequence are controlled by the state of 5 flip-flops within each MBIC:

RTSEQ -	Indicates a recovery sequence is in progress. This flip-flop sets when a retry status is sent from ANY slave to ANY master, and resets when a subsequent data now status occurs on the bus.
RRL -	Indicates that this MBIC was the master to which the LAST retry which occurred on the bus was directed. It sets when the particular MBIC is denied the requested transfer, and resets at the completion of the next transfer cycle (i.e. the next data now or retry).
RR -	Indicates that this particular MBIC has received a retry response at some point in the current recovery sequence. It sets simultaneously with RRL, and resets with the end of the recovery sequence (next data now).
RGL -	Indicates that this particular MBIC was the slave device which gave the last retry which occurred in this recovery sequence. It sets when the MBIC transmits the retry status on the bus and resets at the completion of the next transfer cycle.
RG -	Indicates that this particular MBIC was the originator of a retry response at some time within the current recovery sequence. It sets simultaneously with RGL, and resets with the end of the recovery sequence.

A list of the possible settings of these flip-flops is given in Table 8.1. The actions taken by individual MBICs in order to resolve the conflict either by re-ordering the accesses to allow amicable completion or (when necessary) by identifying the highest priority member of the conflict chain and forcing it to abort its request are discussed in the following paragraphs.

First, all MBICs which are in state <10> simply retire from active competition for bus resources until a data now status is returned (which changes their state to <00>). This is to simplify the recovery process and allow the bus request lines to be used to identify the bus masters which are candidates for abort if the conflict is unresolvable.

Since the normal arbitration for the bus is suspended during the retry sequence, the bus mastership is passed between participants according to the state of the flip-flops. To facilitate the resolution of the conflict, the MBIC which receives the retry response immediately gets off the bus to allow the issuer of the retry to satisfy its need and eliminate the cause of the conflict (if possible). The issuer of the retry will take control of the bus. If this transfer results in a data now, the bus mastership reverts to the original MBIC since the conflict is now resolved, and the receiving MBIC will be able to accommodate the incoming request. Alternatively, if the transfer results in another retry, bus mastership again transfers to the issuer of the retry, and the original MBIC loses any residual rights to the bus.

Thus the MBIC in state <13> or <17> is the bus master at any given point in the retry sequence. The MBIC in state <1C> or <1D> was the last master and has given up its mastership, but retains the right to re-take the bus if the current transfer is successful. An MBIC in state <14> or <15> is ineligible for bus mastership until the retry sequence completes (and then arbitrates normally), or until again drawn into the sequence by having a transfer directed to it (in which case, another state is assumed).

An MBIC becomes a candidate for abort when it has both given and been given a retry in the current sequence. The MBIC will begin to drive its bus request line when it enters state <15>, <17>, or <1D> to identify its candidacy for abort to the other bus members. Once identified as a candidate, whenever the MBIC recognizes a request directed at it, it will only issue the retry if another MBIC is driving a bus request at a priority level above it. If its bus request is the highest active request on the bus, the MBIC will instead perform the abort sequence to its CPU and satisfy the request, thus ending the retry sequence.

NOTE: In the above discussion, the references to data now include not only the data now status, but an observation of an I/O transfer which is equivalent to a satisfied bus transfer.

	RTSEQ	RRL	RR	RGL	RG	
<0x>	0	x	x	x	x	All x's are zero - The RTSEQ flip-flop sets in all MBICs simultaneously with the first retry observed on the bus.
<10>	1	0	0	0	0	This MBIC has observed a RETRY, but has not become involved in the sequence (yet anyway).
<11>	1	0	0	0	1	Invalid - when this MBIC gives a retry, it assumes bus mastership to attempt its transfer. During this transfer cycle RGL and RG are set, and the transfer will terminate with retry which would cause the MBIC to be in state 11101, or data now which resets all the flip-flops.
<12>	1	0	0	1	0	Invalid - RGL cannot be set without RG also set.
<13>	1	0	0	1	1	This MBIC has just denied access to a given unit - it now owns the bus and will attempt its transfer to resolve the conflict.
<14>	1	0	1	0	0	This MBIC has received a retry at some point, and the MBIC which gave that retry also was denied access, since the recovery is still in progress.
<15>	1	0	1	0	1	This MBIC has both given and received a retry, but neither was part of the last unsuccessful transfer.
<16>	1	0	1	1	0	Invalid - RGL cannot be set without RG also set.

	RTSEQ	RRL	RR	RGL	RG
<17>	1	0	1	1	1

This MBIC has just given the retry, and has received it at some point in the previous history of this sequence.

	RTSEQ	RRL	RR	RGL	RG
<18>	1	1	0	0	0
<19>	1	1	0	0	1
<1A>	1	1	0	1	0
<1B>	1	1	0	1	1

Invalid - RRL cannot be set without RR also set.

	RTSEQ	RRL	RR	RGL	RG
<1C>	1	1	1	0	0

This MBIC has just received the retry - the issuer of the retry now is the owner of the bus, attempting to resolve the conflict.

	RTSEQ	RRL	RR	RGL	RG
<1D>	1	1	1	0	1

This MBIC has just received the retry, and has given it at some point previously in this sequence.

	RTSEQ	RRL	RR	RGL	RG
<1E>	1	1	1	1	0

Invalid - RGL cannot be set without RG also being set.

	RTSEQ	RRL	RR	RGL	RG
<1F>	1	1	1	1	1

Invalid - The MBIC will not issue a retry to itself, so RGL and RRL cannot be set simultaneously

==
==

Table 8.1

Recovery Flip-Flop States

9. MBIC Internal Registers

9.1. Interrupt Register

Interrupt Register Local Addr=FF8f00 Bus Addr=3F8g00

15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
sw3	swx	sw0	h11	h10	h09	h08	h07	h06	h05	h04	h03	h02	h01	h00	err

INTREG - (Read)

After an interrupt to the CPU has occurred, this register is normally read to check which particular interrupt line has actually caused the interrupt. Each bit in the INTREG has one or more interrupt sources and an associated interrupt mask bit. The bit in the INTREG will be set active only when the interrupt source is active AND the interrupt mask bit has been set (enabling that interrupt).

The only exception to this rule is Power Fail and Retry Abort which are non-maskable (i.e. independent of the state of IMSK00).

The contents of INTREG, its associated masks and interrupt sources are as follow:

INTREG	Interrupt Mask	Interrupt source(s)
bit15	IMSK15	Software Interrupt 3
bit14	IMSK14	Software Interrupt 2
	or	Software Interrupt 1
bit13	IMSK13	Software Interrupt 0
bit12	IMSK12	Hardware Interrupt 11
bit11	IMSK11	Hardware Interrupt 10
bit10	IMSK10	Hardware Interrupt 09
bit09	IMSK09	Hardware Interrupt 08
bit08	IMSK08	Hardware Interrupt 07
bit07	IMSK07	Hardware Interrupt 06
bit06	IMSK06	Hardware Interrupt 05
bit05	IMSK05	Hardware Interrupt 04
bit04	IMSK04	Hardware Interrupt 03
bit03	IMSK03	Hardware Interrupt 02
bit02	IMSK02	Hardware Interrupt 01
bit02	IMSK02	Hardware Interrupt 01
bit01	IMSK01	Hardware Interrupt 00
bit00	- (no mask)	CSR1.15 (Retry abort Interrupt)
	- (no mask)	CSR1.14 (Power fail)
	IMSK00	CSR1.13 (Non-existent memory Interrupt)
	IMSK00	CSR1.12 (Bus error received)
	IMSK00	CSR1.11 (Memory parity error received)
	IMSK00	CSR1.10 (Bus parity error detected)

Interrupt Register Local Addr=FF8f00 Bus Addr=3F8g00

15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
clr	ulk	x	x	x	x	x	x	s3	s2	s1	s0	rs3	rs2	rs1	rs0

INTREG - (Write)

A write operation directed at INTREG will accomplish one or more actions depending on the data associated with the write operation.

CLR

when a "1" is written to this position, the error latches associated with the following errors are cleared:

CSR1.15	Retry Abort
CSR1.13	Non-existent memory
CSR1.12	Bus error rcvd
CSR1.11	Memory parity error rcvd
CSR1.10	Bus parity error detected

ULK

when a "1" is written to this bit position the master interrupt lockout flip-flop is reset to allow further interrupts. The clearing of this interrupt lockout is delayed until the address strobe following the "write" instruction to allow execution of one additional instruction prior to the interrupt circuit becoming "live" again.

Set Software Interrupts 3-0

Reset Software Interrupts 3-0

The least significant byte of the data word controls the software interrupt flip-flops. Any or all of the flip-flops may be altered in a single operation. The software f/f's may be set by individual bits from 04-07, or reset by bits 00-03. A "1" in the appropriate bit position will cause the set/reset action to occur, (if both set and reset are 1, the results will be indeterminate).

9.2. Interrupt Mask Register

Interrupt Mask Register Local Addr=FF8f02 Bus Addr=3F8g02

15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
sw3	swx	sw0	h11	h10	h09	h08	h07	h06	h05	h04	h03	h02	h01	h00	err

The interrupt mask register is a sixteen bit read/write register which contains the current interrupt mask values. The interrupt signal(s) associated with each bit are discussed under the INTREG discussion above.

9.3. Control and Status Register 0

```
CSR 0                   Local Addr=FF8f04         Bus Addr=3F8g04
-----------------------------------------------------------------
|15 |14 |13 |12 |11 |10 |09 |08 |07 |06 |05 |04 |03 |02 |01 |00 |
-----------------------------------------------------------------
|x17|x16|  CFADD21-16 (compl)   |GAtype |hld|lcl|xup|kli|klh| 0 |
-----------------------------------------------------------------
|        read/write             |  ro   | * |ro |r/w|r/w|r/w|ro |
-----------------------------------------------------------------
```

Bits 15 thru 08, and bit 03 of CSR0 contain the bits which control the response of the MBIC to incoming bus transfers to the MiniBus memory space as described below:

A21 thru A16 are compared to the one's complement of the value contained in bits 13-08 of CSR0. The comparison is modified by the current values of x17, x16, and xup as follows:

xup = 1	force a match on A19 and A18
x17 = 1	force a match on A17
x16 = 1	force a match on A16

If, after applying the forcing bits, the result of the comparison is an identical match, the MBIC becomes selected as a slave and begins the external reference sequence.

NOTE: The MBIC powers up with all these bits zero's so no external references are allowed until software sets a value in CSR0. (The complement of 000000 falls in the non-existent MiniBus address space, so no match can occur).

The most reasonable combinations of xup, x17, and x18 are shown below, other combinations are left to the reader to deduce.

xup	x17	x16
0	0	0

All bits of the CFADDR field are significant, The MBIC will respond to a 64 KB range of addresses positioned by CFADDR on a 64 KB boundary within the MiniBus address space.

xup	x17	x16
0	0	1

Only bits 21-17 of the CFADDR field are significant, both 0 and 1 in A16 will match. The MBIC will respond to a 128 KB range of addresses positioned by CFADDR on a 128 KB boundary within the MiniBus memory address space,

xup	x17	x16
0	1	1

Only bits 21-18 of the CFADDR field are significant, any combination of bits in positions A17 and A16 will be accepted. The MBIC will respond to a 256 KB range of addresses positioned by CFADDR on a 256 KB boundary within the MiniBus address space.

xup	x17	x16
1	1	1

Only A21 and A20 are significant in the comparison. Any bit configuration will be accepted in positions A19 - A16. The MBIC will respond to a 1 MB range of addresses positioned by CFADDR on a 1 MB boundary within the MiniBus address space.

In all cases, the address passed to the local memory array consists of the unmodified contents of the MiniBus lines A19 - A00. If the user has implemented a local memory size which is larger than the address range to which MBIC responds, the appropriate bits in CFADDR determine which segment of the memory is externally addressable.

In all cases, the address recognition range will not include the top 128 KB of the MiniBus address space which is not considered "memory" space.

<u>GAtype</u> - <u>bits</u> <u>07</u>,<u>06</u>

Read-only bits 07 and 06 of CSR0 combine to form a 2-bit field which may be used to distinguish different MBIC types. MBIC will supply a "00" in these bit positions when it is implemented on a CPU board (i.e. GAID3 = 0), and a "10" in these positions when implemented on a memory card (i.e. GAID3 = 1).

<u>HLD</u> - <u>bit</u> <u>05</u>

Bit 05 of CSR0 may only be written by an external reference originating from the MiniBus. When set, it will not release the local bus (de-activate LCLBREQ~) after the next external access to local memory, keeping the local CPU in the HOLD state until another external write to CSR0 resets bit 05.

<u>LCL</u> - <u>bit</u> <u>04</u>

Read-only bit 04 of CSR0 is implemented simply to allow software to determine the physical address of its local MBIC. References to the local MBIC may be made by using physical address "F", but it is necessary for software to be able to obtain the physical address of the local MBIC as well. Bit 04 will read as "1" to a local access, and as "0" to a bus access.

<u>KLi</u>,<u>KLh</u> - <u>bits</u> <u>02</u>,<u>01</u>

Read/Write bits 02 and 01 may be set to enable internal "kludge" circuits which correct for certain "bugs" in early 32016 devices. Bit 02 activates the kludge on the interrupt circuit, bit 01 activates the kludge on the HOLD circuit.

9.4. Control and Status Register 1

CSR 0 Local Addr=FF8f06 Bus Addr=3F8g06

15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
RTY	PF	NEM	BPR	MPR	BPD	INT	LCK	SW1	FIX	UM	MI	0	0	gp1	gp0
ro	ro	ro	ro	ro	ro	ro	ro	ro	r/w	r/w	r/w	ro	ro	r/w	r/w

CSR1 provides a collection of bits which assist in the identification of certain errors and interrupts, and provide several control bits which determine operating characteristics of the MBIC. Specific identification of the bits follows:

RTY	When set, the RTY bit indicates a retry abort interrupt is latched in the MBIC.
PF	When set, the PF bit indicates that the power fail signal is active on the MiniBus backplane.
NEM	Indicates a non-existent memory interrupt is latched in the MBIC.
BPR	Indicates a bus error received interrupt is latched in the MBIC (a slave MBIC selected by this unit activated the BERR line).
MPR	Indicates that a memory parity error has been received and is latched in the MBIC (a slave MBIC selected by this unit returned an internal parity error indication with the data).
BPD	Indicates that a bus parity error detected interrupt is latched in the MBIC (this MBIC detected a bus parity error during a return transfer from a selected slave).
INT	Indicates that the interrupt signal is active upstream of the master interrupt lockout. If the interrupt lockout were inactive, an interrupt would occur immediately.
LCK	Indicates the current state of the master interrupt lockout flip-flop.
SW1	Indicates the state of software interrupt 1 - needed to distinguish between interrupts caused by SW1 and

	SW2 since they share the same INTREG bit.
FIX	When set the MBIC operates in FIXED priority mode during bus arbitration.
UM	When set allows user-mode access to MiniBus memory address space. When reset, accesses are only allowed when the U_S~ signal is active.
MI	Maskable Interrupt Enable - When reset, the MBIC interrupt structure operates both the NMI and INT pins of the CPU. Interrupts associated with IMSK03-00 are directed to the NMI pin, the rest of the interrupts will use INT. If this bit is set, all interrupts from the MBIC are issued thru the NMI pin.
gp1,gp0	Two general purpose read/write bits for software use.

9.5. Test Register

Test register Local Addr=FF8f04 Bus Addr=3F8g04

15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
APE	0	0	0	H10	H09	H08	H07	0	0	0	0	0	0	0	MNX
read only								read only							

The test register allows software direct access to four of the interrupt lines on the backplane (INT08-11). To test the state of these wires (unmasked) does not require obtaining the MiniBus mastership, or even synchronization to the bus clock. This is intended as a very low overhead method of testing a signal from a peripheral card to synchronize programmed I/O operations.

The other 2 bits are of no interest in the operation of the device, but were brought out to facilitate device testing. They are:

APE	Address parity error signal
MNX	Master next signal