Preliminary

Am79C81A

CMOS RAM Buffer Controller (RBC)

Advanced Micro Devices

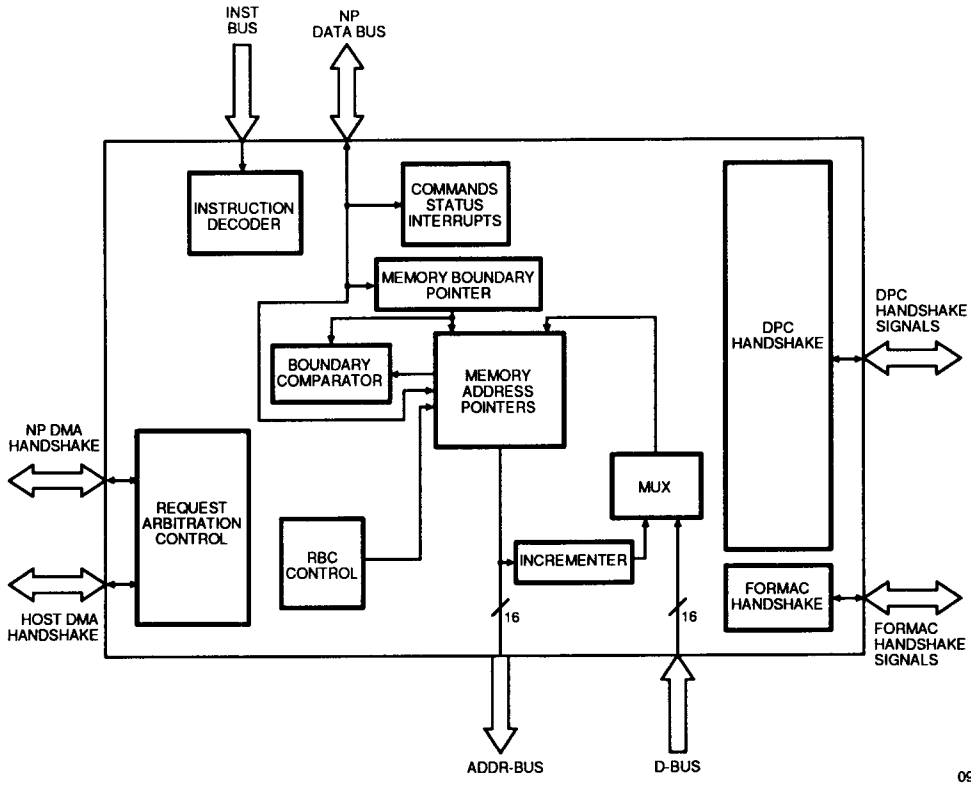
DISTINCTIVE CHARACTERISTICS

- **Total memory buffer management**
  - 16-bit address bus supports 64K words (32 bits wide) with the Am79C82A Data Path Controller (DPC)
  - Programmable registers and pointers
  - DMA arbitration between the Data Path Controller (DPC), Node Processor (NP), Memory full and empty notification and Host
- **Supports transmit link list addressing**
- **12.5-MHz byte clock**
- **TTL-Compatible I/O**
- **Single +5-V supply**
- **145-lead pin grid array package**

GENERAL DESCRIPTION

The Am79C81A RAM Buffer Controller (RBC), along with the Am79C82A Data Path Controller (DPC), manages Buffer Memory to implement a FIFO (first-in, first-out) data structure, simplifying the design of any high-speed interface. This CMOS device has TTL-compatible input and output pins which provide a straightforward interface to other external devices. Diagnostic information is also accumulated and reported by the RBC.

BLOCK DIAGRAM



09729-001A

Publication # 09729 Rev. C Amendment /0
Issue Date: February 1989

Note:

The word "frame" is used in the SUPERNET data sheets to describe three different groups of information.

1) One group is passed over the network media and has the following structure:

Frame Preamble	Start Delimiter	Frame Control	Destination Address	Source Address	Information	Frame Check Sequence	End Delimiter	Frame Status

2) The others are stored in buffer memory and structured as follows:
   A) Transmit frame:

Descriptor	Frame Control	Destination Address	Source Address	Information	Frame Check Sequence	Pointer

   B) Receive frame:

Descriptor	Frame Control	Destination Address	Source Address	Information	Frame Check Sequence

TABLE OF CONTENTS

CONNECTION DIAGRAM
PGA Bottom View (Pins facing up)

	A	B	C	D	E	F	G	H	J	K	L	M	N	P	R	
1	NC	VCC	$\overline{CSI}$	$\overline{DS}$	NP14	NP11	NP9	NP8	NP6	NP3	NP1	NP0	$\overline{NMINTR}$	NC	CLM/$\overline{BEC}$	1
2	NC	INST0	BMODE	NC	NP15	NP12	NP10	NC	NP7	NP4	NP2	BCLK	VCC	NC	NC	2
3	INST3	INST2	INST1	$\overline{READY}$	GND	NP13	GND	GND	VCC	NP5	GND	NC	NC	NC	RCVABT	3
4	DISRBC	HSRDRQ	R/$\overline{W}$	●									GND	FSHRCVF	NC	4
5	$\overline{HSRDACK}$	NPRDRQ	$\overline{RESET}$										DWRREQ	DRDREQS	NC	5
6	$\overline{NPRDACK}$	HSWRRQ	GND										DRDREQA	BRCVFRM	ERCVFRM	6
7	VCC	NPWRRQ	$\overline{HSWRACK}$										GND	LDRPXS	LDRPXA	7
8	$\overline{NPWRACK}$	GND	GND					**(Bottom View)**					DISNHRQ	VCC	NC	8
9	NC	NC	TESTEN										GND	DRDACKS	DRDACKA	9
10	$\overline{NPWACKE}$	TEST	GND										DWRACK	ACKONE	PARERR	10
11	NC	$\overline{HSWACKE}$	SO										GND	MDRDACK	MDWRACK	11
12	SI	NC	VCC										XBEERR	RBFERR	INICLBN	12
13	NC	D15	D12	D9	GND	D4	D1	GND	ADDR12	ADDR9	ADDR6	ADDR4	ADDR1	$\overline{WR}$	$\overline{RD}$	13
14	NC	D14	D11	D8	D6	D3	NC	GND	ADDR13	ADDR10	ADDR7	GND	ADDR2	VCC	$\overline{CS0}$	14
15	NC	D13	D10	D7	D5	D2	D0	VCC	ADDR14	ADDR11	ADDR8	ADDR5	ADDR3	ADDR0	ADDR15	15
	A	B	C	D	E	F	G	H	J	K	L	M	N	P	R	

09729–002A

PIN DESIGNATIONS

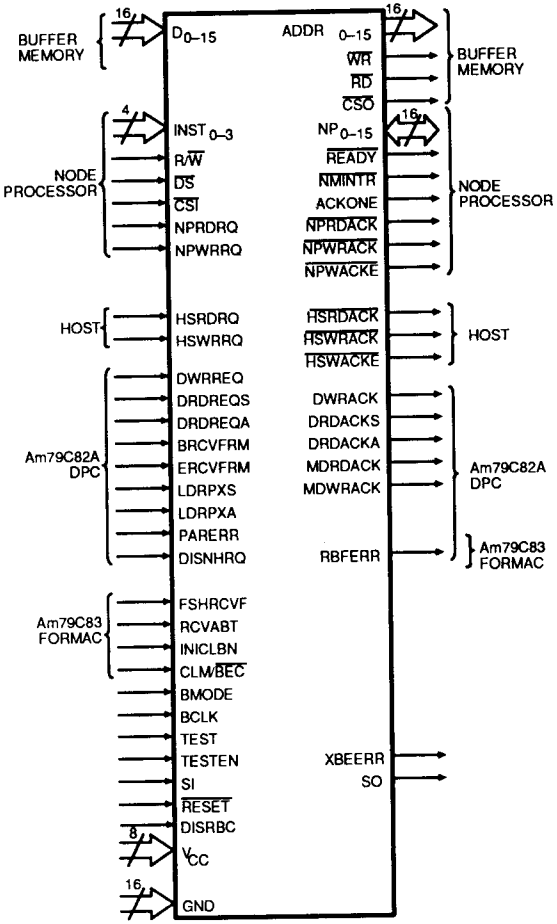
(Sorted by Pin Number)

Pin No.	Pin Name	Pin No.	Pin Name	Pin No.	Pin Name	Pin No.	Pin Name
A-1	NC	C-7	$\overline{\text{HSWRACK}}$	H-13	GND	N-10	DWRACK
A-2	NC	C-8	GND	H-14	GND	N-11	GND
A-3	INST_3	C-9	TESTEN	H-15	V_{CC}	N-12	XBEERR
A-4	DISRBC	C-10	GND	J-1	NP_6	N-13	ADDR_1
A-5	$\overline{\text{HSRDACK}}$	C-11	SO	J-2	NP_7	N-14	ADDR_2
A-6	$\overline{\text{NPRDACK}}$	C-12	V_{CC}	J-3	V_{CC}	N-15	ADDR_3
A-7	V_{CC}	C-13	D_{12}	J-13	ADDR_{12}	P-1	NC
A-8	$\overline{\text{NPWRACK}}$	C-14	D_{11}	J-14	ADDR_{13}	P-2	NC
A-9	NC	C-15	D_{10}	J-15	ADDR_{14}	P-3	NC
A-10	$\overline{\text{NPWACKE}}$	D-1	$\overline{\text{DS}}$	K-1	NP_3	P-4	FSHRCVF
A-11	NC	D-2	NC	K-2	NP_4	P-5	DRDREQS
A-12	SI	D-3	$\overline{\text{READY}}$	K-3	NP_5	P-6	BRCVFRM
A-13	NC	D-4	GUIDE PIN	K-13	ADDR_9	P-7	LDRPXS
A-14	NC	D-13	D_9	K-14	ADDR_{10}	P-8	V_{CC}
A-15	NC	D-14	D_8	K-15	ADDR_{11}	P-9	DRDACKS
B-1	V_{CC}	D-15	D_7	L-1	NP_1	P-10	ACKONE
B-2	INST_0	E-1	NP_{14}	L-2	NP_2	P-11	MDRDACK
B-3	INST_2	E-2	NP_{15}	L-3	GND	P-12	RBFERR
B-4	HSRDRQ	E-3	GND	L-13	ADDR_6	P-13	$\overline{\text{WR}}$
B-5	NPRDRQ	E-13	GND	L-14	ADDR_7	P-14	V_{CC}
B-6	HSWRRQ	E-14	D_6	L-15	ADDR_8	P-15	ADDR_0
B-7	NPWRRQ	E-15	D_5	M-1	NP_0	R-1	CLM/$\overline{\text{BEC}}$
B-8	GND	F-1	NP_{11}	M-2	BCLK	R-2	NC
B-9	NC	F-2	NP_{12}	M-3	NC	R-3	RCVABT
B-10	TEST	F-3	NP_{13}	M-13	ADDR_4	R-4	NC
B-11	$\overline{\text{HSWACKE}}$	F-13	D_4	M-14	GND	R-5	NC
B-12	NC	F-14	D_3	M-15	ADDR_5	R-6	ERCVFRM
B-13	D_{15}	F-15	D_2	N-1	$\overline{\text{NMINTR}}$	R-7	LDRPXA
B-14	D_{14}	G-1	NP_9	N-2	V_{CC}	R-8	NC
B-15	D_{13}	G-2	NP_{10}	N-3	NC	R-9	DRDACKA
C-1	$\overline{\text{CSI}}$	G-3	GND	N-4	GND	R-10	PARERR
C-2	BMODE	G-13	D_1	N-5	DWRREQ	R-11	MDWRACK
C-3	INST_1	G-14	NC	N-6	DRDREQA	R-12	INICLBN
C-4	R/$\overline{\text{W}}$	G-15	D_0	N-7	GND	R-13	$\overline{\text{RD}}$
C-5	$\overline{\text{RESET}}$	H-1	NP_8	N-8	DISNHRQ	R-14	$\overline{\text{CSO}}$
C-6	GND	H-2	NC	N-9	GND	R-15	ADDR_{15}
		H-3	GND				

PIN DESIGNATIONS
(Sorted by Pin Name)

Pin No.	Pin Name	Pin No.	Pin Name	Pin No.	Pin Name	Pin No.	Pin Name
P-10	ACKONE	B-15	D_{13}	B-2	$INST_0$	C-5	$\overline{RESET}$
P-15	$ADDR_0$	B-14	D_{14}	C-3	$INST_1$	A-12	SI
N-13	$ADDR_1$	B-13	D_{15}	B-3	$INST_2$	C-11	SO
N-14	$ADDR_2$	N-8	DISNHRQ	A-3	$INST_3$	B-10	TEST
N-15	$ADDR_3$	A-4	DISRBC	R-7	LDRPXA	C-9	TESTEN
M-13	$ADDR_4$	R-9	DRDACKA	P-7	LDRPXS	A-7	Vcc
M-15	$ADDR_5$	P-9	DRDACKS	P-11	MDRDACK	B-1	Vcc
L-13	$ADDR_6$	N-6	DRDREQA	R-11	MDWRACK	C-12	Vcc
L-14	$ADDR_7$	P-5	DRDREQS	N-1	$\overline{NMINTR}$	H-15	Vcc
L-15	$ADDR_8$	D-1	$\overline{DS}$	M-1	NP_0	J-3	Vcc
K-13	$ADDR_9$	N-10	DWRACK	L-1	NP_1	N-2	Vcc
K-14	$ADDR_{10}$	N-5	DWRREQ	L-2	NP_2	P-14	Vcc
K-15	$ADDR_{11}$	R-6	ERCVFRM	K-1	NP_3	P-8	Vcc
J-13	$ADDR_{12}$	P-4	FSHRCVF	K-2	NP_4	P-13	$\overline{WR}$
J-14	$ADDR_{13}$	B-8	GND	K-3	NP_5	N-12	XBEERR
J-15	$ADDR_{14}$	C-10	GND	J-1	NP_6	A-1	NC
R-15	$ADDR_{15}$	C-6	GND	J-2	NP_7	A-11	NC
M-2	BCLK	C-8	GND	H-1	NP_8	A-13	NC
C-2	BMODE	E-13	GND	G-1	NP_9	A-14	NC
P-6	BRCVFRM	E-3	GND	G-2	NP_{10}	A-15	NC
R-1	CLM/$\overline{BEC}$	G-3	GND	F-1	NP_{11}	A-2	NC
C-1	$\overline{CSI}$	H-13	GND	F-2	NP_{12}	A-9	NC
R-14	$\overline{CSO}$	H-14	GND	F-3	NP_{13}	B-12	NC
G-15	D_0	H-3	GND	E-1	NP_{14}	B-9	NC
G-13	D_1	L-3	GND	E-2	NP_{15}	D-2	NC
F-15	D_2	M-14	GND	B-5	NPRDRQ	G-14	NC
F-14	D_3	N-11	GND	B-7	NPWRRQ	H-2	NC
F-13	D_4	N-4	GND	A-6	$\overline{NPRDACK}$	M-3	NC
E-15	D_5	N-7	GND	A-10	$\overline{NPWACKE}$	N-3	NC
E-14	D_6	N-9	GND	A-8	$\overline{NPWRACK}$	P-1	NC
D-15	D_7	D-4	GUIDE PIN	R-10	PARERR	P-2	NC
D-14	D_8	A-5	$\overline{HSRDACK}$	C-4	R/$\overline{W}$	P-3	NC
D-13	D_9	B-4	HSRDRQ	P-12	RBFERR	R-2	NC
C-15	D_{10}	B-6	HSWRRQ	R-3	RCVABT	R-4	NC
C-14	D_{11}	B-11	$\overline{HSWACKE}$	R-13	$\overline{RD}$	R-5	NC
C-13	D_{12}	C-7	$\overline{HSWRACK}$	D-3	$\overline{READY}$	R-8	NC
		R-12	INICLBN				

LOGIC SYMBOL



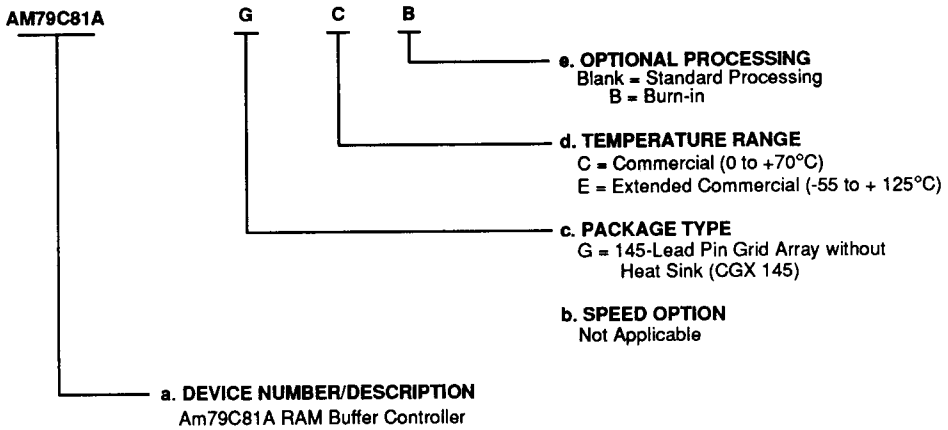
09729–003A

ORDERING INFORMATION

Standard Products

AMD standard products are available in several packages and operating ranges. The ordering number (Valid Combination) is formed by a combination of:

a. Device Number
b. Speed Option (if applicable)
c. Package Type
d. Temperature Range
e. Optional Processing



Valid Combinations	
AM79C81A	GC, GCB, GE, GEB

Valid Combinations

Valid Combinations list configurations planned to be supported in volume for this device. Consult the local AMD sales office to confirm availability of specific valid combinations, to check on newly released combinations, and to obtain additional data on AMD's standard military grade products.

PIN DESCRIPTIONS

Buffer Memory Interface

All output pins which interface to the Buffer Memory can be forced into the high-impedance state by bringing the DISRBC signal HIGH.

$ADDR_{0-15}$
Address Bus (Output; Three State)

The Address bus transmits the addresses which access the Buffer Memory. One of the six possible address pointers are selected for output on this bus; the address selected depends on the result of the request arbitration logic. Each memory transfer lasts for two BCLK cycles, and the addresses are valid for two cycles. When no memory transfer is taking place, the value on these lines is all bits HIGH (FFFF Hex).

$\overline{CSO}$
Chip Select Output (Output; Three State, Active LOW)

When $\overline{CSO}$ is LOW, a memory transfer takes place. In cases where a "buffer full" error occurs, the $\overline{CSO}$ signal becomes LOW, as in normal transfers, but no actual writing of memory occurs because the $\overline{WR}$ signal stays HIGH. For each memory access, $\overline{CSO}$ is LOW for two clock periods.

D_{0-15}
Data Bus (Inputs)

This bus is used to read pointers into the RBC from frames stored in buffer memory. The purpose of the pointers is to support the chain transmit scheme. This data is loaded into the RBC when LDRPXS (A) is HIGH.

DISRBC
Disable RBC (Input; Active HIGH)

DISRBC is used to force the Buffer Memory interface signals (ADDR-bus, $\overline{CSO}$, $\overline{RD}$, $\overline{WR}$) into the high-impedance state for initialization testing purposes only. It is not designed for use during normal FDDI operation; its misuse may cause serious network errors. Before DISRBC goes HIGH, every memory access should be complete, and no requests should be pending. The Node Processor (NP) can still load and read RBC internal pointers and registers while DISRBC is active. DISRBC is synchronous to BCLK and can stay HIGH as long as necessary.

$\overline{RD}$
Read Signal (Output; Three State, Active LOW)

When $\overline{RD}$ and $\overline{CSO}$ are LOW, data is read from the Buffer Memory. For each memory access, $\overline{RD}$ is LOW for 1.5 clock periods. Since the read operation is not destructive, the first attempt to read the memory when the receive or transmit FIFOs are empty is not prevented (as with the $\overline{WR}$ signal).

*SUPERNET is a trademark of Advanced Micro Devices, Inc.

$\overline{WR}$
Write Signal (Output; Three State, Active LOW)

When $\overline{WR}$ and $\overline{CSO}$ are LOW, data is written into the Buffer Memory, either from the DPC or from external logic on the D-bus. If the RBC attempts to service a write request while the receive or transmit FIFOs (configured in Buffer Memory by the RBC) are full, the $\overline{WR}$ signal will not switch LOW, which prevents any overwriting of data. $\overline{WR}$ is asserted for one clock period.

Node Processor (NP) Interface

The NP may run synchronous to BCLK or may be clocked at a different frequency. In the synchronous mode (BMODE pin is HIGH), the RBC expects to see signals from the NP active for an integral number of BCLK cycles. In the asynchronous mode (BMODE is LOW), the READY line is used as a handshake signal with the NP. The $INST_{0-3}$ lines (R/$\overline{W}$, $\overline{DS}$, and $\overline{CSI}$) should all be active until the RBC drives the READY line LOW.

BMODE
Bus Mode (Input)

The BMODE pin should be tied HIGH when the NP operates synchronously with the SUPERNET* chip-set. In asynchronous operation, BMODE must be tied LOW. BMODE only affects the handshake on the $\overline{DS}$, $\overline{CSI}$, and READY lines.

$\overline{CSI}$
Chip Select Input (Input; Active LOW)

The $\overline{CSI}$ pin indicates whether the NP has selected the RBC to execute an instruction. Internal execution of instructions is permitted only if $\overline{CSI}$ and $\overline{DS}$ are both LOW (these two signals are "ANDed" internally), and the instruction belongs to the RBC. When $\overline{CSI}$ is HIGH, the $INST_{0-3}$ lines are ignored and the NP-bus is set to the high-impedance state.

$\overline{DS}$
Data Strobe (Input; Active LOW)

$\overline{DS}$ is used for defining the presence of data on the NP-bus when the NP is asynchronous to the BCLK (BMODE is LOW). In this case, the $\overline{DS}$ behaves much like the data strobes on slave devices connected to standard microprocessors. When R/$\overline{W}$ and $\overline{CSI}$ are LOW, a write operation is in progress from the NP to the RBC on the NP-bus; $\overline{DS}$ should go LOW whenever the data to be written is valid. $\overline{DS}$ should stay LOW (and the data on the NP-bus should stay valid) until the RBC takes the READY line LOW. In the case of a read operation, when R/$\overline{W}$ is HIGH and $\overline{CSI}$ is LOW, the NP forces $\overline{DS}$ LOW whenever it is ready to accept the read data. Subsequently, the RBC provides valid data and also asserts READY LOW. The NP can take $\overline{DS}$ HIGH any time after the RBC drives READY LOW. Until $\overline{DS}$ is brought HIGH, the RBC continues to provide valid data on the NP-bus. When the NP is synchronous to the BCLK, $\overline{DS}$ should be LOW for only one clock cycle per instruction.

$INST_{0-3}$
Instruction Lines (Inputs)

These four instruction lines are typically connected to the NP address bus. Along with the R/$\overline{W}$ pin, they provide instructions to the RBC. Some are executed by the RBC while others are used in conjunction with the NP0-31 lines to read or load various registers internal to the RBC. These instructions are valid whenever $\overline{DS}$ and $\overline{CSI}$ are both active. When the NP runs on a clock synchronous to BCLK (as indicated by the BMODE pin being HIGH), the INST lines, as well as $\overline{DS}$ and $\overline{CSI}$, should be active for an integral number of clock cycles. When the NP runs asynchronous to the BCLK (as indicated by the BMODE pin being LOW), the INST0-3 lines, as well as $\overline{DS}$ and $\overline{CSI}$, should be active until READY becomes active (LOW).

$\overline{NMINTR}$
Non-Maskable Interrupt (Output; Open-Drain, Active LOW)

NMINTR is asserted when an error occurs during the RBC operation. This signal is used to interrupt the NP, and will stay LOW until the NP recognizes the interrupt by reading the status register.

NP_{0-15}
Node Processor Bus (Input/Output; Three State)

The NP-bus carries the data for initializing, reading, or writing the various registers and pointers in the RBC. This bus can be used by the NP for reading or writing independent of the activities of the RBC. The NP-bus is an output whenever a read instruction is provided to the RBC; i.e., when R/$\overline{W}$ is HIGH and both $\overline{CSI}$ and $\overline{DS}$ are LOW. The NP-bus is an input when a write instruction is given to the RBC (R/$\overline{W}$, $\overline{CSI}$, and $\overline{DS}$ are LOW). When either $\overline{CSI}$ or $\overline{DS}$ become HIGH, the NP-bus is in the high-impedance state.

$\overline{NPRDACK}$
NP Read Acknowledge (Output; Active LOW)

The $\overline{NPRDACK}$ signal indicates that an NP DMA request (initiated using the NPRDRQ pin) is being serviced. The NP can use $\overline{NPRDACK}$ as a signal to latch the memory data into an external register when it becomes available during the latter part of the transfer. For each request, $\overline{NPRDACK}$ is LOW for the entire two-cycle memory transfer.

NPRDRQ
NP Read Request (Input; Active HIGH)

NPRDRQ is a DMA request from the NP to the RBC. This signal selects the memory address register pointer as the source of the address with which to access the Buffer Memory. When the RBC services this request it asserts $\overline{NPRDACK}$. This operation constitutes a complete handshake. The NPRDRQ request can be either a pulse or a level signal. If it is a pulse, it should be wide enough for one BCLK transition to occur (and meet the setup times), but small enough so that two BCLK transitions do not occur. For each such pulse request, one $\overline{NPRDACK}$ is generated, lasting two BCLKs. Subsequent pulses for more requests should be generated only after the $\overline{NPRDACK}$ is generated by the RBC. If the NPRDRQ signal is level, it switches HIGH and stays at the HIGH level. The RBC constantly services this request (after arbitrating) and indicates this activity using $\overline{NPRDACK}$. If NPRDRQ is switched LOW at any time in the window of $\overline{NPRDACK}$, one more service is generated.

$\overline{NPWACKE}$
NP Write Acknowledge Enable (Output; Active LOW)

$\overline{NPWACKE}$ has the same function as NPWRACK, except that it is driven LOW only for the last half of the first clock cycle and the entire second clock cycle in a two-cycle memory transfer. The $\overline{NPWACKE}$ signal can be used by external logic to enable the memory data bus.

$\overline{NPWRACK}$
NP Write Acknowledge (Output; Active LOW)

The $\overline{NPWRACK}$ signal indicates that an NP DMA request (initiated using the NPWRRQ pin) is being serviced. The NP can use $\overline{NPWRACK}$ as a signal to enable data onto the Buffer Memory data bus D_{0-31}. For each request, $\overline{NPWRACK}$ is LOW for the entire two-cycle memory transfer.

NPWRRQ
NP Write Request (Input; Active HIGH)

The NPWRRQ is a DMA request from the NP to the RBC. NPWRRQ selects the memory address register pointer as the source of the address with which to access the Buffer Memory. When the RBC services this request, it asserts $\overline{NPWRACK}$ for two BCLK cycles.

$\overline{READY}$
Read Line (Output; Open Drain, Active LOW)

$\overline{READY}$ is a handshake signal when the RBC is in asynchronous mode. When the BMODE pin is HIGH, $\overline{READY}$ is also HIGH and cannot be used. When BMODE is LOW, the NP runs asynchronous to the BCLK, and $\overline{READY}$ indicates whether an instruction has been acted upon. In the case of a write instruction, $\overline{READY}$ goes LOW after the data is clocked into the RBC. In the case of a read instruction, $\overline{READY}$ goes LOW after the RBC supplies valid data on the NP-bus. Typically, the RBC takes between three and four BCLK cycles to assert $\overline{READY}$ after the NP asserts DS. $\overline{READY}$ stays LOW as long as $\overline{DS}$ and $\overline{CSI}$ stay LOW.

$\overline{RESET}$
Reset (Input; Active LOW)

$\overline{RESET}$ is a hardware reset which initializes the internal state machines in the RBC and places the command and status registers in a pre-defined state. This signal can be asserted asynchronously at any point in the RBC operation, but should stay LOW for at least a period of four BCLK cycles. $\overline{RESET}$ can be forced HIGH asynchronously at any time after that period. Typically, the $\overline{RESET}$ input signal is connected to a power-on-reset circuit and also to a reset switch. A hardware reset is a

necessary first step which ensures that the RBC is in a known state prior to any operations taking place on the NP-bus. Subsequent resets to the RBC can be software-driven using the reset instruction provided in the instruction set.

R/$\overline{W}$

Read/$\overline{Write}$ (Input)

When R/$\overline{W}$ is LOW, data is written into one of the RBC's registers or an instruction is executed. When R/$\overline{W}$ is HIGH, data is read from an RBC register onto the NP-bus. R/$\overline{W}$ should be controlled in the same manner that $INST_{0-3}$ are.

Miscellaneous

BCLK

Byte Clock (Input)

BCLK is the main clock which runs the RBC. All transactions between the RBC, DPC, FORMAC, and ETX are synchronous with this clock. Transactions between these chips and the Node Processor are also synchronous with this clock when BMODE is HIGH. The skew of BCLK when it is distributed to the RBC, DPC, and FORMAC should be as small as possible.

SI

Shift Scan Input (Input)

This pin is used for factory testing. During normal operation SI must be left unconnected.

SO

Shift Scan Output (Output)

This pin is used for factory testing. During normal operation SO must be left unconnected.

TEST

Test (Input; Active HIGH)

When HIGH, the TEST pin forces the RBC into the test mode, in which various internal points are monitored on the SO pins. This logic may be used for incoming testing; the TEST pin should be tied to ground for normal operation.

TESTEN

Test Enable (Input; Active HIGH)

This pin is used for factory testing. For normal operation, TESTEN should be tied to ground.

Host Processor Interface

$\overline{HSRDACK}$

$\overline{Host\ Read\ Acknowledge}$ (Output; Active LOW)

The $\overline{HSRDACK}$ signal indicates that a Host DMA request (initiated using the HSRDRQ pin) is being serviced. $\overline{HSRDACK}$ is not generated if a receive "buffer empty" error is detected and the RBC is not in the page mode (MPAGE = 0). For each Host read request, $\overline{HSRDACK}$ is asserted for two clock cycles.

HSRDRQ

Host Read Request (Input; Active HIGH)

The HSRDRQ is a DMA request from the Host to the RBC. The RBC selects the read pointer (RPR), which contains the beginning address of the receive FIFO, to be placed onto the 16-bit address bus. When the RBC services this request, it asserts $\overline{HSRDACK}$. The operation of the HSRDRQ and $\overline{HSRDACK}$ pair is exactly the same as that of the NPRDRQ and $\overline{NPRDACK}$ pair.

HSWRRQ

Host Write Request (Input; Active HIGH)

The HSWRRQ is a DMA request from the Host to the RBC. The RBC selects the write pointer (WPX), which contains the beginning address of the transmit FIFO in Buffer Memory, to be placed onto the 16-bit address bus. When the RBC services this request, it asserts $\overline{HSWRACK}$. The operation of the HSWRRQ and $\overline{HSWRACK}$ pair is exactly the same as that of the NPWRRQ and $\overline{NPWRACK}$ pair.

$\overline{HSWACKE}$

$\overline{Host\ Write\ Acknowledge\ Enable}$ (Output; Active LOW)

$\overline{HSWACKE}$ has the same function as $\overline{HSWRACK}$, and its timing is the same as $\overline{NPWACKE}$. The $\overline{HSWACKE}$ signal is used by external logic to enable the memory data bus.

$\overline{HSWRACK}$

$\overline{Host\ Write\ Acknowledge}$ (Output; Active LOW)

The $\overline{HSWRACK}$ signal indicates that a Host DMA request (initiated using the HSWRRQ pin) is being serviced. $\overline{HSWRACK}$ is not generated if a transmit "buffer full" error is detected and the RBC is not in the page mode (MPAGE = 0). For each Host write request, $\overline{HSWRACK}$ is LOW for two clock cycles.

Data Path Controller (DPC) Interface

ACKONE

Acknowledge One (Output; Active HIGH)

ACKONE indicates the completion of the first BCLK cycle of a two-cycle memory transfer for reads or writes. ACKONE is generated for all transfers, whether they involve the DPC, the NP, or the Host. In cases where the Buffer Memory becomes full, the actual transfer may not take place because the $\overline{WR}$ signal is not active. ACKONE still switches HIGH to indicate this attempted transfer.

BRCVFRM

Beginning of Received Frame (Input; Active HIGH)

Originating at the DPC, this signal indicates to the RBC that the following DWRREQ refers to the first long word of a received frame. Then, depending on the state of the MSLNSTM bit programmed in the mode register, the RBC will decide whether to skip the first location for the frame status. BRCVFRM forces the RBC into the receive state, and is asserted as soon as the DPC begins receiving a frame. BRCVFRM is synchronous with the BCLK and lasts for one clock cycle.

DISNHRQ

Disable NP and Host Request (Input; Active HIGH)

DISNHRQ prevents the RBC from servicing the pending NP and Host read or write requests. When this signal is active, only the DPC can access the Buffer Memory through the RBC, thus guaranteeing the DPC access to the total memory bandwidth. Even with DISNHRQ active, however, the NP can still communicate with the RBC using the instruction lines. If DISNHRQ becomes HIGH when the RBC is in the middle of an NP or Host service, the present service is completed and then further NP and Host requests are frozen. These requests remain pending and are serviced when DISNHRQ is released. DISNHRQ is synchronous to BCLK and lasts as long as the DPC wants to keep the RBC function for itself.

DRDACKA

DPC Read Acknowledge for A-Frame (Output; Active HIGH)

Similar to DRDACKS, DRDACKA pairs with DRDREQA to transmit one long word of an A-frame from the transmit Buffer Memory. This signal is synchronous to BCLK and lasts for two BCLK cycles for each request.

DRDACKS

DPC Read Acknowledge for S-Frame (Output; Active HIGH)

DRDACKS is a handshake signal from the RBC which advises the DPC that its DRDREQS is being serviced. Simultaneous with the output of this signal, the address bus becomes stable and CSO and RD are asserted. The latter operation enables the DPC to read one long word from the transmit queue in Buffer Memory. DRDACKS is synchronous with BCLK and lasts for two clock cycles. Together, DRDACKS and DRDREQS constitute a complete handshake between the RBC and the DPC for reading one long word of data from memory.

DRDREQA

DPC Read Request for A-Frame (Input; Active HIGH)

When the DPC is transmitting an A-frame from the asynchronous transmit queue in Buffer Memory, DRDREQA is used in place of DRDREQS. The RBC signifies that this request has been serviced by asserting DRDACKA. As with a synchronous frame, the DPC will not assert DRDREQA simultaneously with DRDREQS or DWRREQ.

DRDREQS

DPC Read Request for S-Frame (Input; Active HIGH)

When the DPC needs to read a long word (for an S-frame) from the synchronous transmit queue in Buffer Memory, it sends a DRDREQS to the RBC. DRDREQS is the second-highest priority request (DWRREQ is the highest). DRDREQS is synchronous to BCLK, and lasts until the RBC services it, which the RBC signifies by asserting DRDACKS. The DPC will not assert DWRREQ or DRDREQA at the same time as DRDREQS.

DWRACK

DPC Write Acknowledge (Output; Active HIGH)

The DWRACK signal indicates that the DWRREQ sent to the RBC is being serviced. Simultaneous with the output of this signal, the address bus becomes stable and CSO and WR are asserted. The latter operation allows the DPC to write data from its D-bus into the receive Buffer Memory. For each DWRREQ, one DWRACK is generated and lasts for two clock cycles. The operation of DWRREQ and DWRACK forms a complete handshake between the DPC and RBC, and effectively writes one long word of data into the Buffer Memory. During normal operation, the RBC expects to have its DWRACK asserted within (at most) two clock cycles after it receives DWRREQ, since this request always gets highest priority. If the DPC does not receive DWRACK two clock cycles after DWRREQ, it may generate a receive abort, assuming that something is wrong with the RBC, or that its FIFO buffer is too full to receive more data from the FORMAC.

DWRREQ

DPC Write Request (Input; Active HIGH)

DWRREQ is a signal from the DPC to the RBC which requests a memory write cycle for the receiving frame. While in the receive mode, the DPC asserts DWRREQ when it wants to write a long word to the Buffer Memory. DWRREQ is the highest priority request and will be serviced ahead of any other pending requests. DWRREQ lasts until the RBC services its request. The DPC ensures that only one of DWRREQ, DRDREQS, and DRDREQA are asserted at a time.

ERCVFRM

End of Received Frame (Input; Active HIGH)

ERCVFRM is asserted when the DPC detects the end of a received frame. The RBC uses this signal to write the frame status and length (if it is in the write mode, MSLNSTM = 1). ERCVFRM forces the RBC out of the receive mode, and should be asserted after the DPC sees a DWRACK for the last word of the received frame. ERCVFRM is synchronous with BCLK and lasts one clock cycle. Together with BRCVFRM, it marks the boundaries for one complete frame reception.

LDRPXA

Load RPX for A-Frame (Input; Active HIGH)

This signal is analogous to LDRPXS to support the chain transmit scheme for A-frames. It pairs with DRDREQA and DRDACKA, and is used to load the RPXA pointer.

LDRPXA is valid only when the RBC is programmed in the chain transmit mode (MENCHN = 1).

LDRPXS

Load RPX for S-Frame (Input; Active HIGH)

LDRPXS is a signal from the DPC which instructs the RBC to load data from the 16-bit D-bus to its RPXS pointer. This signal is given to support the chain transmit scheme of the DPC. LDRPXS is synchronous with the system clock, and is forced HIGH and inactive in the same manner as DRDREQS. When servicing DRDREQS with LDRPXS, the RBC enables its address bus (as always with DRDREQS) to load the lower 16 bits of the Buffer Memory data bus $D_{0\text{-}15}$ into the RPXS pointer. In this way, the RBC can perform a jump to the next frame location in memory (linked list scheme). LDRPXS is valid only when the RBC is programmed in the chain transmit mode (MENCHN = 1).

MDRDACK

MDR Read Acknowledge (Output; Active HIGH)

The MDRDACK signal indicates that an NP read request instruction (INPRDNI, INPRDWI) is being serviced. For each read request instruction, MDRDACK is driven HIGH for the entire two-cycle memory transfer.

MDWRACK

MDR Write Acknowledge (Output; Active HIGH)

The MDWRACK signal indicates that an NP write request instruction (INPWRWI) is being serviced. For each write request instruction, MDWRACK is driven HIGH for the entire two-cycle memory transfer.

PARERR

Parity Error (Input; Active HIGH)

The PARERR signal, originating from the DPC parity check logic, indicates that a parity error has been detected in the present read or write transaction on the D-bus. If the parity error occurs during NP or Host service, the RBC generates an interrupt to the NP and waits until the NP clears the error before further servicing requests of that type. If the parity error occurs during a DPC read or write service, the RBC also sets the appropriate status bit and interrupts the NP, but continues to service subsequent DPC requests. PARERR is asserted immediately after the ACK signal of the request becomes inactive. It is synchronous with BCLK and lasts for one clock cycle.

RBFERR

Receive Buffer Full Error (Output; Active HIGH)

The RBC asserts RBFERR to notify the DPC and the FORMAC that the receive queue in Buffer Memory is full. The receive queue is defined as "full" when the receive read pointer (RPR) and the receive write pointer (WPR) have met and the DPC still attempts further DWRREQs. Depending on the status of the page mode bit programmed in the RBC, the assertion of RBFERR can have two different results:

1) If MPAGE = 1 (RBC is in page mode) and a receive buffer full error is detected, the RBC asserts RBFERR for only two clock cycles and continues to operate as if no error had occurred.

2) If MPAGE = 0 and a receive buffer full error is detected, then:

a) the RBC generates an interrupt to the NP;
b) the RBC asserts RBFERR until the error is cleared by the NP;
c) the RBC will not generate a DWRACK while RBFERR is asserted, preventing an overwrite of old data by new data. The WPR is locked at the point where the buffer full error occurred; and
d) RBFERR forces the RBC out of the receive mode.

Fiber Optic Ring Media Access Controller (FORMAC) Interface

CLM/$\overline{\text{BEC}}$

Claim or $\overline{\text{Beacon}}$ (Input)

When INICLBN is HIGH, the state of CLM/$\overline{\text{BEC}}$ determines which mode (claim or beacon) will be initialized. The INICLBN and CLM/$\overline{\text{BEC}}$ "truth table" follows:

INICLBN	CLM/BEC	Meaning
0	0	Claim/beacon inactive
0	1	Claim/beacon inactive
1	0	Transmit beacon frame
1	1	Transmit claim frame

To initialize claim mode, the FORMAC forces CLM/$\overline{\text{BEC}}$ HIGH simultaneously with INICLBN. CLM/$\overline{\text{BEC}}$ must stay HIGH for one BCLK cycle while INICLBN is HIGH. To initialize the beacon mode, CLM/$\overline{\text{BEC}}$ must be LOW when INICLBN becomes HIGH.

FSHRCVF

Flush Received Frame (Input; Active HIGH)

The FSHRCVF instructs the RBC to "flush" a received frame; i.e., to purge the frame from Buffer Memory. If FSHRCVF becomes HIGH while the RBC is servicing a DWRREQ, the RBC will stop servicing that request and will reset the WPR pointer back to the beginning of the frame. Generally, a flush is the result of an unmatched address frame in the FORMAC address detection logic. FSHRCVF identifies the end of a flushed frame and forces the RBC out of the receive state. The FORMAC does not allow FSHRCVF to be HIGH when the RBC is not in the receive state. FSHRCVF is synchronous with BCLK and is HIGH for one clock cycle.

INICLBN

Initialize Claim/Beacon (Input; Active HIGH)

The FORMAC uses this pin to force the RBC into the claim or beacon mode. (For more information about claim/beacon, please refer to the Am79C83 FORMAC data sheet.) Together with the CLM/$\overline{\text{BEC}}$ pin, INICLBN instructs the RBC to load the claim or beacon address to

the synchronous transmit pointer (RPXS). A claim or beacon frame is subsequently read from this location and transferred to the FORMAC. INICLBN is HIGH for one clock cycle, and is used only when the RBC is in the chain transmit mode (i.e., when the programmed command bit MENCHN is HIGH).

RCVABT

Receive Abort (Input; Active HIGH)

The RCVABT signal provides a means for the FORMAC to notify the RBC that the received frame has been aborted due to an error. RCVABT is synchronous to BCLK, and is HIGH for one clock cycle. If RCVABT is HIGH for one or more cycles while a DWRREQ is being serviced, the next pending request is not serviced. Depending on the state of the MSLNSTM command bit, the RBC will set its WPR pointer back to the beginning of the frame for the aborted status, or it will wait for the next incoming frame. For an aborted frame, RCVABT denotes the end of the frame and forces the RBC out of the receive state.

Interface with External Logic

XBEERR

Transmit Buffer Empty Error (Output; Active HIGH)

The RBC generates XBEERR to notify the outside logic that the transmit buffer has become empty. XBEERR is used only when the RBC is programmed in the transmit FIFO mode (MENCHN = 0), since in the non-FIFO mode (MENCHN = 1), the RBC never checks for full or empty errors. Like RBFERR, XBEERR behaves differently depending on the status of the page mode bit programmed in the RBC:

1) If MPAGE = 1 (RBC is in page mode) and a transmit buffer empty error is detected, then the RBC asserts XBEERR for only two clock cycles and continues to operate as if nothing happened.

2) If MPAGE = 0 and a transmit buffer empty error is detected, then:

a) the RBC generates an interrupt to the NP;
b) the RBC asserts XBEERR until the error is cleared by the NP; and
c) the RBC will not generate a DRDACKS while XBEERR is asserted, preventing a reading of erroneous data. The RPXS is locked at the point where the transmit buffer empty error occurs.

Since the DPC supports the non-FIFO transmit buffer only, XBEERR is used only when the RBC is used as a stand-alone chip.

Power Supply

GND

Ground (Inputs)

There are sixteen GND pins. They must all be connected to the power return.

VCC

Power (Inputs)

There are eight VCC pins. They must all be connected to a +5-V ±5% supply.

FUNCTIONAL DESCRIPTION

Functional Overview

The RBC generates addresses within FIFOs in Buffer Memory for the frames being received and for the frames to be transmitted. It also controls the DMA transfer of data to and from the Buffer Memory. The received frames are taken from the FORMAC, converted from 8-bit to 32-bit words by the DPC, and stored in a simple receive FIFO in the Buffer Memory. The NP checks the frame to ensure that it is good; then, the data is transferred from the FIFO to the I/O buffer of the Host or NP.

The frames to be transmitted are transferred from the Host or NP I/O to one of two chain FIFOs in Buffer Memory for FDDI transmission. One chain FIFO is used for S-frames (referred to as synchronous frames in the standard) and one chain FIFO is used for A-frames (referred to as asynchronous frames in the standard). The data is transferred from the Buffer Memory to the FORMAC when transmission is permitted.

Note: The words "synchronous" and "asynchronous" as used in the FDDI standards refer to classes of transmission service and have nothing to do with transmitted signals or the synchronization of these signals. All frames are converted to electrical signals for transmission using the same format. There is only one kind of transmission over the media; therefore, S-frames and A-frames are terms used in this manual to avoid confusion.

The memory can also have byte parity (each byte has a parity bit) as an option.

The functions of the RBC can be summarized as follows:

1) NP interaction with the RBC:

a) Through the instruction lines and the NP-bus, the NP can:

1) Initialize the ten user-visible pointers of the RBC;
2) Program the mode register of the RBC to fit its particular need;
3) Read and clear the status reported by the RBC; and
4) Generate software requests to read or write to the memory.

The above functions are totally independent from those below. However, if the NP tries to initialize a pointer which is being used by the RBC, the RBC will have priority.

b) Through NP read or write request pins, the NP can have access to the Buffer Memory (either the receive or transmit buffer). One common pointer (MAR) is reserved for this function.

2) Host interaction with the RBC:

a) The Host can write to the transmit buffer via a write request to the RBC. The WPX pointer is used for the Host write.

b) The Host can read the received frames from the receive buffer via a read request to the RBC. The RPR pointer is reserved for the Host read.

3) DPC interaction with the RBC:

a) The DPC stores the received frames in the receive buffer by sending write requests to the RBC. The RBC uses the WPR pointer to serve this request.

b) The DPC transmits frames from the transmit buffer by sending read requests to the RBC. The RPXA pointer is used for asynchronous frames and the RPXS pointer is used for synchronous frames.

All memory access requests sent to the RBC are serviced on a first-come, first-served basis. However, if the NP, the Host, and the DPC attempt to access the memory at the same time, then the RBC has the responsibility of arbitrating them according to a predefined priority:

1) First priority = DPC requests
2) Second priority = NP requests
3) Third priority = Host requests

Overview of User-Accessible Resources

Programmable Resources

Instruction Set
The NP can issue software commands to the RBC through the R/$\overline{W}$, $INST_{0-3}$, and NP_{0-31} lines. $INST_{0-3}$ are normally driven by the address lines of the NP.

Mode Register (MODE)
The mode register is 8 bits wide, and can be written to and read by the NP. This register allows the NP to program the RBC to fit certain system requirements. This register determines the modes of operation of the RBC. On reset, the register is initialized so that all bits are inactive. The NP can then load the required values into the register using the ILDMOD instruction. Register contents remain unchanged until the NP loads in a new value or the chip is reset. Contents of this register and the dynamic status register can be read with the IRDMDDS instruction.

Status Registers (STAT and DSTS)
The two status registers can be read by the NP on the NP-bus. These registers store the status bits generated by the RBC. These bits, when HIGH, will generate an interrupt to the NP. The two status registers differ in the manner in which they are cleared; each type is explained below:

1) Static (autocleared) Status Register (8 Bits): this register contains RBC status bits which are automatically cleared when the NP reads them unless an external event causes a bit to be set at the same time. In such a case, the new setting overrides the autoclear so that no status is lost. This register is read with the IRDSTAT instruction.

2) Dynamic Status Register (8 Bits): this register contains fatal error status bits which disable the operation related to the error. The NP can read this register, but the read does not autoclear it. The clearing of one or more status bits in this register can be accomplished only through a specific dynamic-status-clear instruction (using the ICLDSTS instruction accompanied by appropriate bits on the NP_{8-15} lines) given by the NP. This register and the mode register can be read by using the IRDMDDS instruction.

Memory Pointers
The main purpose of the RBC is to make the random-access Buffer Memory function as a set of first-in, first-out (FIFO) buffers: one for received data and one or two for transmitted data.

The RBC stores incoming frames in the receive buffer in sequential order, automatically wrapping around to the start of the buffer when it reaches the end of the buffer.

The RBC buffers transmit data in one of two ways, depending on the state of the MENCHN bit in the mode register. When the RBC is used with the Am79C82A DPC in FDDI applications, the MENCHN bit is normally set to logic "1" (chain mode) so that the RBC manages transmit data as two linked lists, one for S-frames and one for A-frames. When the RBC is used as a stand-alone FIFO controller, the MENCHN bit is set to logic "0" (FIFO mode) so that the RBC manages transmit data as a single sequential FIFO just like the receive buffer.

To manage these buffers, the RBC has ten pointer registers that can be read or written by the NP using the RBC instruction set. Each of these registers contain a 16-bit address of a memory location (word) in external RAM. In FDDI applications using the DPC, the external memory is organized as 32-bit-wide words (plus an optional 4 bits for parity).

The ten pointer registers in the RBC visible to the user are:

1) SAR:	Start address for receive FIFO
2) EAR:	End address for receive FIFO
3) ACP:	Address to claim pointer (in chain mode) or start address for transmit FIFO (in FIFO mode)
4) RPCB:	Read pointer for claim or beacon transmit frames (in chain mode) or end address for transmit FIFO (in FIFO mode)
5) WPR:	Write pointer for receive frames
6) RPR:	Read pointer for receive frames
7) WPX:	Write pointer for transmit frames
8) RPXA:	Read pointer for asynchronous transmit frames
9) RPXS:	Read pointer for synchronous transmit frames except for claim or beacon transmit frames (in chain mode)
10) MAR:	Memory address register for NP access

The FIFO buffers can be located anywhere in the 64K memory address space. For the receive buffer, the SAR and EAR define the limits of the space allocated to the FIFO, while the RPR and WPR point to the beginning and end of the valid data in the FIFO. Since the read and write pointer registers increment automatically after they are used, the start address (SAR) should be less than the end address (EAR).

In the FIFO mode (MENCHN = 0), the transmit FIFO is defined in the same way using ACP and RPCB to mark the limits of the space allocated to the FIFO, and RPXS and WPX to mark the beginning and end of the valid data in the FIFO. The RPXA pointer can be used as a random-access pointer that is automatically incremented after each access, but does not automatically wrap around from RPCB to ACP or from EAR to SAR.

The RBC can also be compatible with larger memory requiring an address bus wider than 16 bits. In this case, the NP should program the RBC in the page mode and the 16-bit address bus from the RBC will be connected to the least significant bits of the memory address.

Hardwired Resources

Using the BMODE pin, the RBC can interface with either a synchronous or asynchronous Node Processor. When BMODE is tied HIGH, the RBC can interface with a synchronous NP. In this case, the $\overline{CSI}$ pin indicates the presence of data on the NP-bus; the $\overline{DS}$ pin indicates the presence of an instruction from the NP instruction line. When BMODE is tied LOW, the NP clock is asynchronous with the RBC clock. In this case, the $\overline{DS}$ and $\overline{CSI}$ pins can be connected together, and become LOW when the NP issues an instruction to the RBC, and the $\overline{READY}$ line signifies that the RBC received the message from the NP.

Buffer Memory Operation

The Buffer Memory is a key part of an FDDI node. It has to accept data at a high rate and transfer it immediately to make room for more data.

The NP assigns the Buffer Memory addresses for each frame of data to be transmitted. The NP also assigns the Buffer Memory space to be used for the receive FIFO and initializes the RBC pointer registers. For received data, the RBC furnishes the Buffer Memory addresses for reads and writes and increments the pointer after each service, wrapping around when the end of the FIFO is reached. For frames to be transmitted, the RBC increments the pointer after each long word in the frame is accessed. At the end of the frame, the RBC uses the pointer value in the frame to point to the address of the next frame. The RBC also arbitrates DMA requests for DMA data transfer to and from the Host, NP, and DPC. This section discusses the Buffer Memory addressing. The DMA requests from the Host, NP, and DPC to the RBC are discussed in the sections that follow.

To manage the addressing, the RBC has ten registers which are used as memory pointers. For the data being received from the media, a simple FIFO is created within the Buffer Memory by the NP. The RBC manages this FIFO after the NP has created and initialized it. Frames are stored sequentially in contiguous locations in this simple FIFO.

Because FDDI applications provide two categories of data to be transmitted (A-frames and S-frames), two separate chain queues are created by the NP in the Buffer Memory. One chain queue is used for storing A-frames and the other queue FIFO is used for storing A-frames.

Chain queues differ from simple RBC-managed FIFOs in the Buffer Memory addressing scheme used. Each frame in a chain queue contains a pointer to the address of the next frame. These frame addresses are generally assigned by the NP. The words within a frame are stored in contiguous locations. In a chain queue the frames need not be assigned to contiguous locations in Buffer Memory.

In a simple FIFO, the RBC increments a pointer register to assign the addresses sequentially to the frames. When the end of the FIFO (EAR) is reached, the RBC places the SAR value into the pointer, thus wrapping around.

After a description of the memory pointers, this section discusses the operation of simple FIFOs. This is followed by a discussion of the chain queues used for transmission.

Memory Pointers

The RBC contains ten pointer registers used as address pointers. The RBC uses four of these pointers to make a FIFO within the Buffer Memory to handle the data received from the media through the DPC. For storing frames to be transmitted, three of the remaining pointers are used to read from and write to a pair of chain queues. Each frame in a chain queue has a pointer that points to the location of the next frame. These pointers within the frames have been assigned Buffer Memory addresses by the NP.

The ten visible address pointers in the RBC are 16 bits wide and can be written into or read from by the NP using the NP-bus and instruction lines. Each pointer contains an address of a memory location of the external Buffer Memory RAM. The memory is organized as 64K long words, 32 bits wide (36 bits if parity is used). Each address accesses this complete long word of 32 bits (36 bits with parity).

The ten pointers in the RBC visible to the user are:

1) **SAR:** Start address for receive FIFO
2) **EAR:** End address for receive FIFO
3) **ACP:** Address of claim pointer (ACP + 1: Address of beacon pointer)
4) **RPCB:** Read pointer for claim or beacon frames
5) **WPR:** Write pointer for receive frames
6) **RPR:** Read pointer for receive frames
7) **WPX:** Write pointer for transmit S-frames or A-frames
8) **RPXA:** Read pointer for transmit A-frames
9) **RPXS:** Read pointer for transmit S-frames except claim or beacon frames (after INICLBN HIGH)
10) **MAR:** Memory address register for random access

Most of these pointers are self-explanatory. The others are described below:

RPCB automatically stores the current contents of ACP when INICLBN is HIGH. RPCB is the pointer used by the DPC (with the DRDREQS pin) to read the S-frame chain queue, so that claim or beacon frames will be transmitted.)

WPR is the pointer used by DPC with pin DWRREQ for writing to the receive FIFO.

RPR is the pointer used by the Host with pin HSRDRQ for reading from the receive FIFO (the FIFO for frames received from the media through DPC).

WPX is the pointer used by the Host (using the HSWRRQ pin) to write S-frames and A-frames into the respective chain FIFOs in the Buffer Memory for transmission by the media. For each frame to be written by the Host, the WPX is set by the NP to point to either the A-frame chain queue or the S-frame chain queue. Therefore, the Host must communicate to the NP the category of each frame.

RPXS is the pointer used by the DPC (used with the DRDREQS pin) to read the S-frame chain queue for frames to transmit (except claim or beacon frames).

RPXA is the pointer used by DPC (used with the DRDREQA pin) to read the A-frame chain queue for frames to transmit.

The NP controls the MAR, a random-access pointer that can point to any location in the entire memory, irrespective of whether it is within a FIFO or not. The NP can typically use this for housekeeping functions in the memory.

The RBC automatically increments the WPX, WPR, RPR, RPXA, and RPXS registers after each use except when the limits are reached in a simple FIFO, or when the DPC requests the RBC to load in a new value from the memory when operating in the chain FIFO mode. These registers cannot be simultaneously loaded or written by the NP and the RBC. If this occurs, then the RBC has priority, and the status bit SNLSLD is set, creating a non-maskable interrupt ($\overline{\text{NMINTR}}$ pin is LOW).

Receive FIFO

For received data, the NP creates a simple FIFO in Buffer Memory using pointers SAR, EAR, WPR, and RPR. The RBC operates the FIFO, advancing the pointers and wrapping around when the upper limit of the FIFO is reached (Figure 1 shows the FIFO). SAR and EAR define the limits of the receive FIFO. Since these pointers address a 32-bit-long word, the FIFO can begin or end only on a long-word boundary. Since the pointers in the RBC are designed to increment, the start address (SAR) should be smaller in magnitude than the ending address (EAR). With these requirements in mind, this FIFO can be configured anywhere in Buffer Memory within the 64K long-word address space.

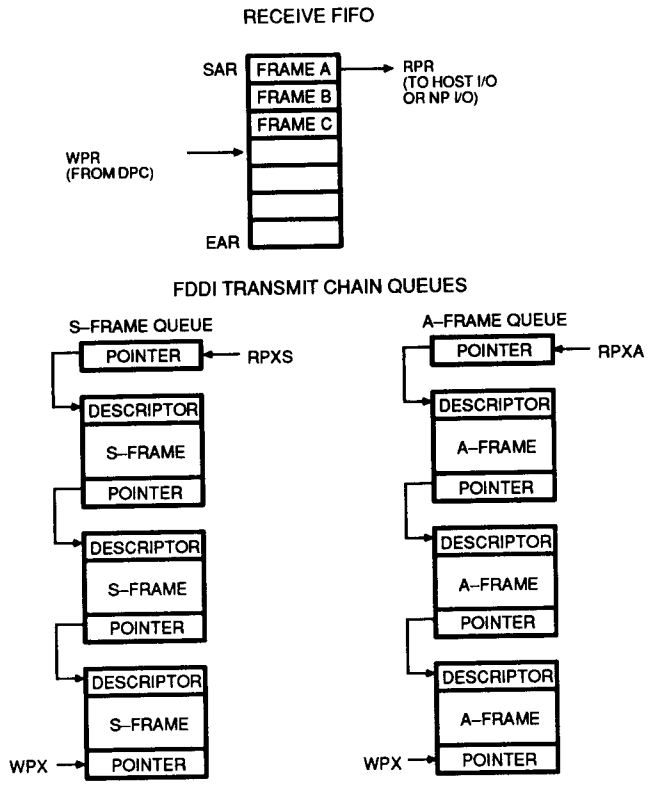


Figure 1. FIFOs and Queues in Buffer Memory

To initialize the receive FIFO, the WPR and RPR pointers are set to the same value anywhere within the receive FIFO. This indicates an empty condition of the FIFO. The NP does this initialization using the load instructions for the corresponding pointers. Once initialized, the RBC gets control of the memory buffer, and the transfer of frames between the media and the Host can begin. The frames are transferred to and from the Buffer Memory in long-word increments (32-bit words).

When data is being received from the media, the DPC generates the write request to the RBC for each word to be stored in the Buffer Memory. The WPR pointer is used in this operation. After writing a long word, the RBC increments WPR to point to the next location to be written in the receive FIFO (or sets it to SAR if EAR has been reached). Upon a read request from the Host, the RBC uses RPR similarly to read a received frame from the receive FIFO and transfer it to the Host.

The RBC also detects the full and empty conditions of the receive FIFO which may occur during the operation. In either condition, the pointer which caused the error to be generated is locked up and not moved forward until the NP recognizes the error and resets the appropriate bit in the dynamic status register, using the ICLDSTS instruction accompanied by a logic "1" on the appropriate bit on the NP-bus (see Table 4).

Transmit Chain Queues

FDDI allows the transmission of two categories of frames: S-frames and A-frames. These transmit frames are stored in two separate chain queues in the Buffer Memory, one for each category of frames. Each chain queue is accessible on a first-in, first-out basis. The NP creates and maintains the two chains (see Figure 1). Each frame in each chain ends with a pointer to the start of the next frame in that chain.

The transmit chain queues do not require initialization of boundary pointers since the address space for each S-frame and A-frame in the Buffer Memory is assigned by the NP. The RPXS and RPXA pointers are initialized by the NP to point to the first frame in each of the two chain queues. Each time that either the NP or Host wants to add frames to either the A-frame chain queue or the S-

frame chain queue, the NP must first set the WPX pointer in the RBC to point to the address assigned to the next frame in the correct chain queue.

For FDDI operation, the NP must initialize the RBC to run in chain queue mode by setting the mode bit MENCHN = 1. Chain queue mode applies to transmit only. Receive is always in simple FIFO mode.

Each time data is placed into the S-frame or A-frame chain queue (in FDDI operation), the NP must issue an IENXMTS or IENXMTA to the DPC to enable the DPC to transmit data. This enables the DPC to transmit data until it reads a frame in which the MORE bit in the descriptor has a value of zero. After a frame has been written into the Buffer Memory it cannot be altered. However, additional frames with the MORE bit = 1 in all but the last new frame may be added to either chain. The enable command may be pipelined so that two (but not more than two) enable commands are waiting to be serviced at the same time (see Figure 2).

The FORMAC determines whether to send an S-frame or A-frame on the basis of the THT and the T-late flag. The DPC uses DRDREQS to request an S-frame and issues DRDREQA to request an A-frame. Two read pointers (RPXS and RPXA) are maintained by the RBC to aid the DPC in reading frames from the two chain FIFOs.

The RBC can use the same transmit write pointer (WPX) to let the Host write to either of the two chain queues since this pointer is set by the NP. The NP knows whether the frame is an S-frame or A-frame, and can set the pointer to the correct chain.

Pointers to the beginning and end of Buffer Memory are not needed in chain mode. The NP is responsible for knowing what address space is available.

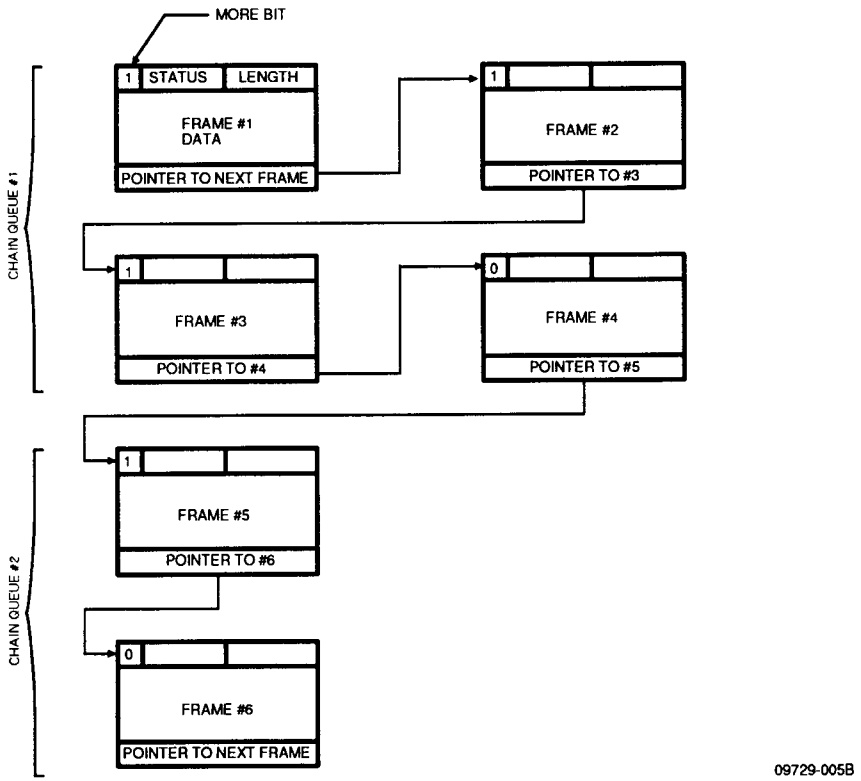


Figure 2. Chain Queues in Buffer Memory

RBC-NP Interaction

NP requests can be generated through direct hardwired signals, or by programming, using the RBC instruction set. The NP can access the Buffer Memory through the DMA (D-bus) using request pins and the MAR register. It can also access the RBC and Buffer Memory using the NP-bus and RBC instructions placed on the $INST_{0-3}$ and R/$\overline{W}$ lines.

Although NP requests can be generated in several ways, only one address pointer, MAR, is provided in the RBC for all NP services. Therefore, the user must limit the generation of NP requests to only one source at a time.

NP requests have the second highest priority in arbitration and are serviced unless a DPC request is also present. A mode bit called MENNPRQ is provided in the mode register to enable or disable the NP pin requests. If an NP pin request and instruction request are presented simultaneously, then the instruction request is serviced. If NP read and write requests are presented simultaneously, then neither one is serviced. The NP DMA requests have priority over Host DMA requests.

NP Hardwired DMA Requests

The NP has access to the Buffer Memory (including the receive and transmit FIFOs) through NP read or write DMA request pins and the D-bus. One RBC pointer (MAR) is used for this function.

DMA requests for reading or writing are made on two pins, NPRDRQ and NPWRRQ, respectively. When these are being serviced, the appropriate signal, $\overline{\text{NPRDACK}}$ or $\overline{\text{NPWRACK}}$, becomes LOW for the duration of the transfer. Also, for the write request, NPWACKE becomes LOW for 1-1/2 clock periods in the middle of the transfer. Simultaneously, the MAR value is placed on the ADDR-bus.

For a read, the NP transfers the Buffer Memory data through the D-bus to the NP memory. For a write, the NP provides the write data on the D-bus (from the NP memory). After the service, the MAR increments to point to the next higher location. The MAR is not restricted by the limits of any FIFO and always increments to the next location. When it reaches its highest value of all ones (FFFF in hex), it is cleared to a value of all zeros.

The NPRDRQ and NPWRRQ pins are typically used by the NP for block transfer of data to or from the memory using DMA. Less frequent or smaller transfers are typically made by the NP using the software instruction requests. The pin requests should be disabled during the NP instruction requests.

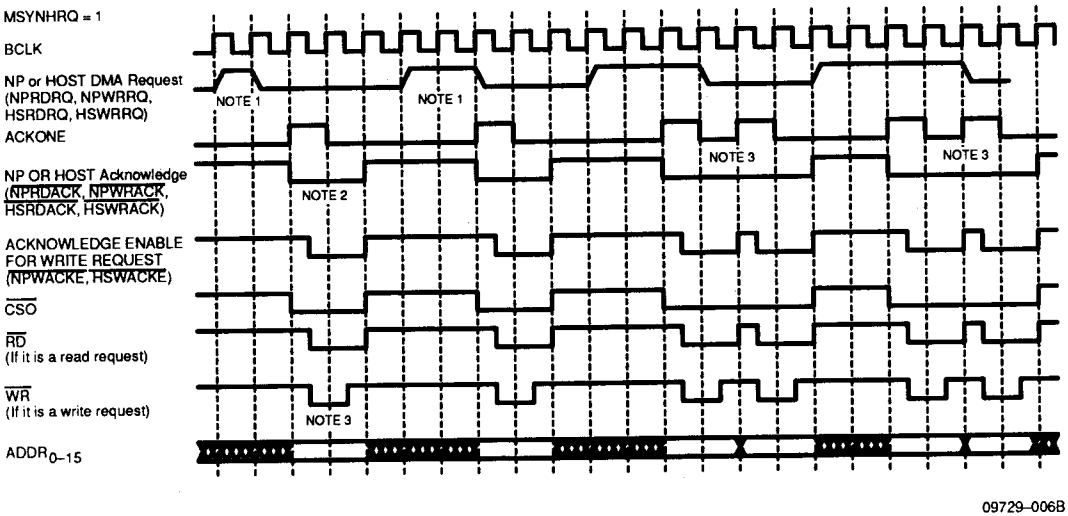


Figure 3. NP or Host DMA Request and Acknowledgement

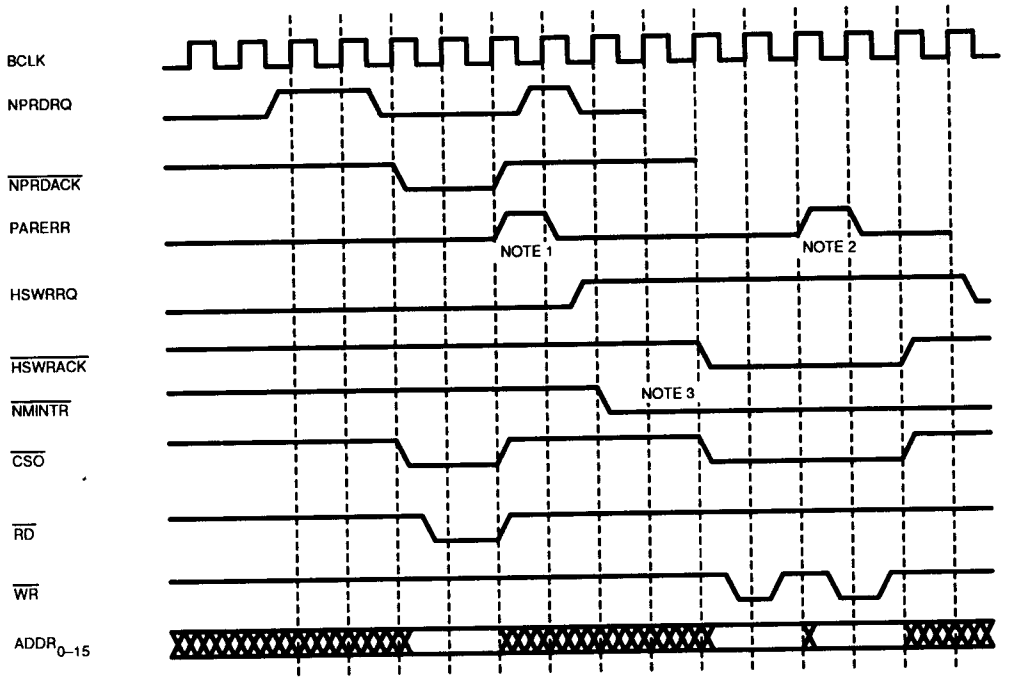
Definitions: A) Request HIGH for one or two BCLK ↑ is considered a "Pulse Request."
B) Request HIGH for three or more BCLK ↑ is considered a "Level Request."

Notes:

1) For each "Pulse Request," the RBC acknowledges one memory access.
2) Each memory access is two BCLK cycles long.

Memory Read	Memory Write
$\overline{CSO}$, $ADDR_{0-15}$, $\overline{NPRDACK}$, and $\overline{HSRDACK}$ are active for two BCLK cycles.	$\overline{CSO}$, $ADDR_{0-15}$, $\overline{NPWRACK}$, and $\overline{HSWRACK}$ are active for two BCLK cycles.
$\overline{RD}$ is active for the last half of the first BCLK and the entire second BCLK	$\overline{WR}$ is active for the last half of the first BCLK and the first half of the second BCLK cycle

3) For "Level Requests" when the RBC has no request pending from any other source, the subse quent acknowledge can be continuous as shown. Under this condition, request HIGH for (2n-1) or (2n) BCLK cycles generates (n) acknowledges (where $n \geq 2$). xxxACK and ACKONE may ANDed together to count the number of requests acknowledged.
4) If MSYNHRQ = 0, then acknowledge (ACKONE, NP, or Host Acknowledge, Acknowledge Enable) becomes active one BCLK cycle later.
5) If MSYNHRQ = 0, the request is considered a "Pulse Request" if (one BCLK + 20 ns) < pulse width < (two BCLK cycles)



Notes:

1. PARERR active right after the acknowledgement (which means that the read or write access just done has a parity error) causes further requests of the same type to be ignored.
2. However, if the request is level, even though the parity error occurs for the previous acknowledgement, it takes one extra acknowledgement before the RBC can stop servicing the request.
3. Parity error always asserts NMINTR pin. Parity error for NP or Host DMA request is a dramatic event, so the RBC locks the pointer which causes the error and waits for the Node Processor to recognize and clear the error. On the other hand, parity error in DPC read/write request or NP read/write request through instructions does not lock the pointer but only sets the status bit and causes a non-maskable interrupt.

Figure 4. Parity Error in NP or Host DMA Service

RBC Programming

Programming is performed by the NP using the instruction set and the mode bit set of the RBC. The instruction set is used not only to load and read the internal registers of the RBC, but also to change the state of the RBC.

All of the RBC programming is done by the NP using $INST_{0-3}$, $R/\overline{W}$, and the NP-bus. The NP_{0-15} bus is used for transferring data to and from the Buffer Memory by way of the MDRU and MDRL registers in the DPC and the D-bus. The NP-bus is also used to transfer pointer register and mode register values to and from the RBC, and to read the status registers. The instruction set is listed in Table 1.

Table 1. RBC Instruction Set

Instruction Mnemonic	$R/\overline{W}$	$INST_3$	$INST_2$	$INST_1$	(LSB) $INST_0$	NP_{0-15}	Function
Software Reset:							
IRESET	0	0	0	0	0	X X X X	Software Reset
Instructions to Load and Read RBC Pointer Registers:							
ILDRPXS	0	0	0	0	1	W W W W	Load RPXS
IRDRPXS	1	0	0	0	1	R R R R	Read RPXS
ILDRPXA	0	0	0	1	0	W W W W	Load RPXA
IRDRPXA	1	0	0	1	0	R R R R	Read RPXA
ILDRPR	0	0	0	1	1	W W W W	Load RPR
IRDRPR	1	0	0	1	1	R R R R	Read RPR
ILDWPX	0	0	1	0	1	W W W W	Load WPX
IRDWPX	1	0	1	0	1	R R R R	Read WPX
ILDWPR	0	0	1	1	0	W W W W	Load WPR
IRDWPR	1	0	1	1	0	R R R R	Read WPR
ILDACP	0	0	1	1	1	W W W W	Load ACP
IRDACP	1	0	1	1	1	R R R R	Read ACP
ILDSAR	0	1	0	0	0	W W W W	Load SAR
IRDSAR	1	1	0	0	0	R R R R	Read SAR
ILDRPCB	0	1	0	0	1	W W W W	Load RPCB
IRDRPCB	1	1	0	0	1	R R R R	Read RPCB
ILDEAR	0	1	0	1	0	W W W W	Load EAR
IRDEAR	1	1	0	1	0	R R R R	Read EAR
ILDMAR	0	0	1	0	0	W W W W	Load MAR
IRDMAR	1	0	1	0	0	R R R R	Read MAR
Instructions to Access Buffer Memory Using the NP-bus:							
INPRDNI	0	1	0	1	1	X X X X	Read memory without incrementing MAR
INPWRWI	0	1	1	0	1	X X X X	Write memory and increment MAR
INPRDWI	0	1	1	1	0	X X X X	Read memory and increment MAR
Instructions Using the Mode Register:							
ILDMOD	0	1	1	0	0	W W W W	Load mode
IRDMDDS	1	1	1	0	0	R R R R	Read mode and dynamic status
IRDSTAT	1	0	0	0	0	R R R R	Read static status
ICLDSTS	0	1	1	1	1	W W W W	Clear dynamic status
Reserved Instructions:							
IRSV1	1	1	1	0	1	–	Reserved
IRSV2	1	1	1	1	0	–	Reserved
IRSV3	1	1	1	1	1	–	Reserved
IRDSWPR	1	1	0	1	1	R R R R	Read SHADOW WPR

Key: W W W W = Write data to RBC register
R R R R = Read data from RBC register
X X X X = Don't Care

Instruction Set

The instruction set of the RBC consists of 29 instructions. Thirteen of these read data from the RBC internal registers onto the NP-bus and are called read instructions. Twelve are load instructions which load data into RBC internal registers from the NP-bus. The instruction bus timing is such that it can be tied to the address bus of a standard microprocessor.

Instruction Handshake. The NP can run on either a synchronous clock or on an asynchronous clock, with respect to the network clock (BCLK). The BMODE pin can be strapped HIGH to indicate a synchronous operation, and strapped LOW to indicate an asynchronous operation.

- ***Synchronous Clock (BMODE = 1).*** The synchronous case can be of two kinds. If the NP runs on a faster clock than the RBC, then there are two or more integral NP clock cycles during every RBC cycle as shown in case A of Figure 5. In this case, the NP needs to repeat the instruction an integral number of times so that the instruction is active (and glitch-free) for at least one BCLK cycle. When the NP is slower, as in case B, the instruction has to be active and glitch-free for at least one BCLK cycle. The RBC executes the instruction only once, provided the $\overline{DS}$ and $\overline{CSI}$ are LOW for only one BCLK cycle when the instructions are valid. If $\overline{DS}$ and $\overline{CSI}$ stay LOW for several cycles, then the instruction will be executed several times.

- ***Asynchronous Clock (BMODE = 0).*** In the asynchronous case, a handshake convention is used as shown in Figure 6. The NP gives the instruction by making $\overline{CSI}$ and $\overline{DS}$ LOW. This is synchronized inside the RBC. This can take from three to four cycles. The RBC executes this instruction once, and at the end of the execution, $\overline{READY}$ is driven LOW. For a load instruction, $\overline{READY}$ b goes LOW after the data is loaded into an internal register or the instruction executed. For a read instruction, the read data is latched and provided continuously on the NP-bus while $\overline{CSI}$ and $\overline{DS}$ are LOW. When the NP has completed the instruction, it brings $\overline{CSI}$ and/or $\overline{DS}$ HIGH. This causes READY to go HIGH and completes one instruction. If the handshake is violated, the results are undefined.

Software Reset. Software reset performs the same function as the RESET pin. Software reset places the RBC state machines into a known state, and clears the mode and status registers.

Instructions to Load and Read RBC Pointer Registers. The NP can change the contents of the RBC's pointer registers by using the appropriate load instruction from the instruction set. The instruction is placed on the R/W and $INST_{0-3}$ lines, and the new value to be entered into the register is placed on the NP_{0-15} lines. The NP initializes the RBC by loading these registers with the transmit and receive FIFO parameters, thereby creating the FIFOs.

In a similar manner, the read instructions in the instruction set permit the NP to read the value of any of these pointer registers by the use of the $INST_{0-3}$, R/$\overline{W}$, and the NP-bus.

Instructions to Access Buffer Memory Using NP-bus. Besides accessing the Buffer Memory via DMA using the NPRDRQ or NPWRRQ pin, the NP can access the Buffer Memory by the three instructions described below. These three instructions are designed to use the MDR register in the DPC chip as an external 32-bit holding register. The MDWRACK and MDRDACK outputs from the RBC are handshake signals that cause the DPC to transfer data between the Buffer Memory and the MDR register.

Before issuing a write instruction, the Node Processor should load the MDR with the data to be written by issuing two 16-bit writes to the DPC.

The SNPPND status bit is provided for use with the NP instruction requests. SNPPND indicates if there are any requests waiting to be serviced. New NP requests should not be given to the RBC as long as SNPPND is HIGH. The NP should read SNPPND status to see that it is clear before issuing a new instruction request.

- ***INPWRWI.*** This instruction is a write access to Buffer Memory followed by an increment of MAR at the end of the write cycle. The write request is arbitrated in the same way as the NP pin request. The SNPPND status bit becomes HIGH when the instruction is given and stays HIGH until the request is serviced.

- ***INPRDWI.*** This instruction is a read access to Buffer Memory followed by an increment of MAR at the end of read cycle. The read request is arbitrated in the same way as the NP pin request. The SNPPND status bit becomes HIGH when the instruction is given and stays HIGH until the request is serviced.

- ***INPRDNI.*** This instruction is a read access to Buffer Memory without an increment of MAR at the end of read cycle. This instruction is serviced the same way as INPRDWI, except that at the end of read cycle, the value of MAR stays intact.

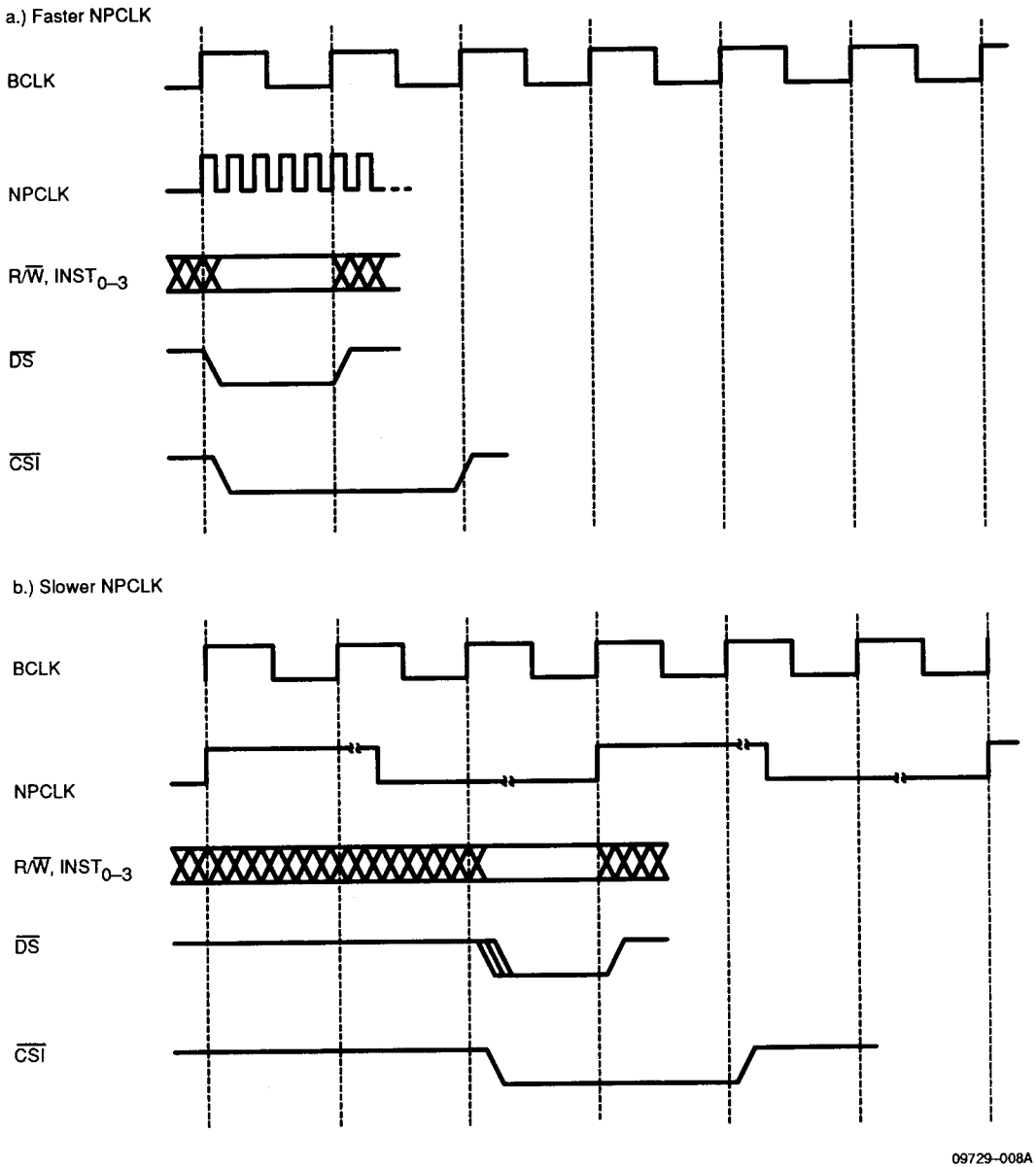
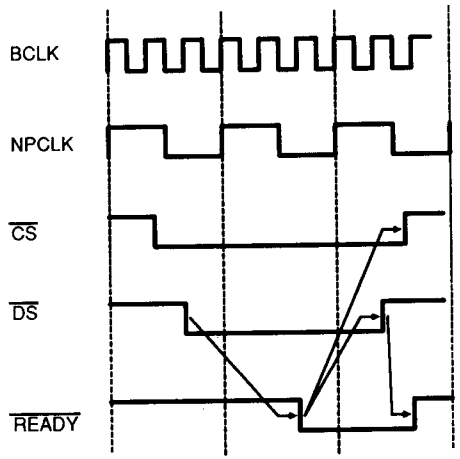
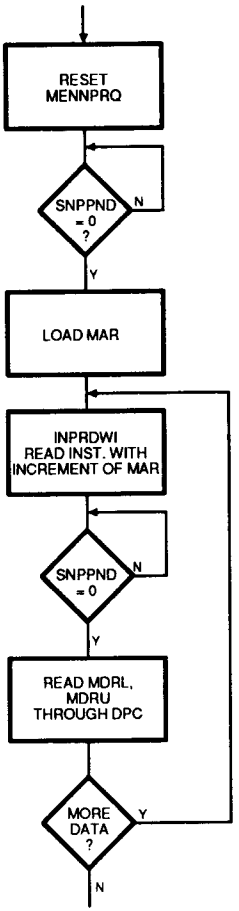


Figure 5. Instruction Timing for Synchronous NP (BMODE = 1)



09729-009A

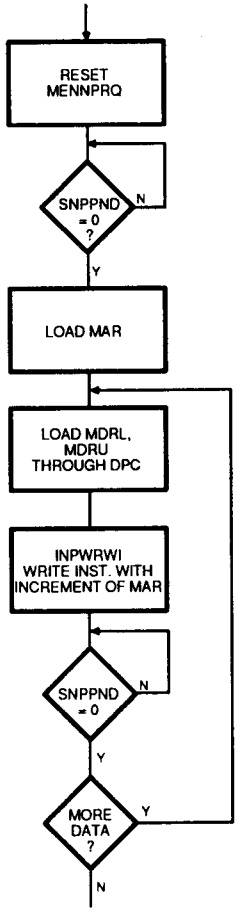
Figure 6. Asynchronous Handshake Between NP and RBC (BMODE = 0)



09729–010A

Note: The NP DMA channel (pulse request) cannot be used when reading buffer memory with the method illustrated above.

Figure 7. Read From Buffer Memory Using Instruction Set



09729-011A

Note: The NP DMA channel (pulse request) cannot be used when writing buffer memory with the method illustrated above.

Figure 8. Write to Buffer Memory Using Instruction Set

Instructions Using the Mode Register. The ILDMOD instruction is used to load 8-bit values from the NP-bus into the mode register. IRDMDDS is used to read the contents of the mode and dynamic status registers onto the NP-bus. The eight least significant bits of the NP-bus are used to load or read the mode register (NP_{0-7}). The IRDMDDS instruction uses the eight most significant bits of the NP-bus for dynamic status and the least significant bits for the mode register.

Reserved Instructions. These are three reserved instructions that will cause unpredictable results if they are executed.

Table 2. Mode Register Bit Assignment

Bit	Mnemonic	Function
7	UNUSED	
6	MPAGE	Set RBC to page mode
5	MENHRRQ	Enable Host read requests
4	MENHWRQ	Enable Host write requests
3	MENNPRQ	Enable NP requests
2	MENCHN	Enable chain-FIFO transmit mode
1	MSLNSTM	Store length and status in memory
0 (LSB)	MSYNHRQ	Synchronize NP and Host requests

Mode Register (MODE)

When reset (either through the RESET pin or through software IRESET), MODE is initialized so that all the bits are logic "0". The NP then loads the required value into the mode register. The mode register bit assignments are listed in Table 2.

MPAGE: (Bit 6) Set the RBC to Page Mode. This bit is set when the RBC is used with a larger Buffer Memory. Since the RBC has a 16-bit address bus, it can only accommodate 64K RAM. If a larger RAM is used, then the user should:

1) Set the MPAGE bit;
2) Use the 16-bit address bus of RBC as least significant bits of the RAM address; and
3) Load start address of the receive FIFO (SAR) to all zeros (0000 Hex), and load the end address of the receive FIFO (EAR) to all ones (FFFF Hex).

In the page mode, the RBC does not lock the pointer and interrupt the NP when it detects a buffer-full condition. Instead it puts out a two-clock-cycle pulse on the RBFERR or XBFERR pin that can be used by external logic to switch pages. In this mode, external address detection logic must be used to distinguish between the end of a page and the end of a buffer.

Differences between page mode and non-page mode are as follows:

MPAGE = 0

1) Set SAR to begin receive buffer, EAR to end of receive buffer
2) If (WPR = RPR) then
   a) RBFERR, $\overline{\text{NMINTR}}$ pins active until NP clears
   b) Status registers set
   c) RCV stopped until error is cleared

MPAGE = 1

1) Set SAR to 0000 Hex, EAR to FFFF Hex;
2) If (WPR = RPR) then
   a) RBFERR pin active for two clocks
   b) Status registers intact
   c) RCV not stopped

MENHRRQ: (Bit 5) Enable Host Read Requests. After MENHRRQ is set, Host DMA read requests pass through and are served by the RBC. If MENHRRQ is reset with a request pending, then the pending request is served but the RBC ignores further DMA requests. If NP and the RBC are using the complete memory bandwidth, it may take an arbitrarily long time to service this last pending request.

MENHWRQ: (Bit 4) Enable Host Write Requests. Operation of this bit is similar to the MENHRRQ.Setting it enables the Host write requests. The above two bits can be used by the NP to control bandwidth allowed to the Host read and write operations.

MENNPRQ: (Bit 3) Enable NP Requests. When this bit is reset, future NP requests through pins are ignored but the last pending request is serviced. While these requests are pending, SNPPND is also HIGH.

MENCHN: (Bit 2) Enable Chain-FIFO Transmit Mode This mode bit places the RBC in the chain mode required for FDDI. In this mode, the transmit buffer is a chain queue managed by the NP. The receive FIFO is not changed. The RBC checks the receive FIFO but not the transmit chain queue for full or empty errors. In this mode, transmit frames are not restricted to contiguous locations in the memory.

MSLNSTM: (Bit 1) Store Length and Status In-Memory. When this bit is set, the status and length of the receiving frame are stored in the Buffer Memory at the beginning of the frame. The status and length are generated by the DPC, which does the housekeeping of the frame it is receiving. This bit should be set if the RBC is used with the DPC.

MSYNHRQ: (Bit 0) Synchronize NP and Host Requests. When this bit is set to logic "1", the RBC assumes that NP and HOST request pulses are externally synchronized to BCLK, and removes an FF from the path of the request signal, thereby reducing acknowledge delay by one BCLK period.

When this bit is reset, the RBC assumes that the NP and HOST requests are not synchronous to BCLK, and latches the request signal in an FF using BCLK before presenting it to the internal logic.

MSYNHRQ = 1
– takes at least one clock to acknowledge

MSYNHRQ = 0
– takes at least two clocks to acknowledge

Static Status Register (STAT)
Status registers are read-only registers. During its operation, the RBC accumulates and reports the status to the NP through its status registers: the static status register and the dynamic status register. Both registers accumulate system-level and frame-related status.

The static status register is autocleared when the NP reads it, unlike the dynamic status register, which needs a dynamic command to clear it. This register is 8 bits wide and contains frame-related and other status generated by the RBC.

A read (using the IRDSTAT instruction) clears all the bits in the static status register unless a bit is being set at the same time. In this case, the setting overrides the autoclear so that no status is lost. Any bit being set in this register causes a non-maskable interrupt to the NP (by asserting the NMINTR pin).

The static status register bit assignments are listed in Table 3.

Table 3. Static Status Register (Stat) Bit Assignments

Bit	Mnemonic	Function
7	UNUSED	
6	SDRAPAR	A-frame parity error during DPC read
5	SDRSPAR	S-frame parity error during DPC read
4	SDWRPAR	Parity error during DPC write
3	SNLSLD	NP and RBC simultaneous load (of RBC pointer)
2	SNHPAR	NP or Host parity error
1	SBFME	Buffer full or empty condition
0 (LSB)	SRABT	DPC receive write abort

Most of the static status register bit assignments are self-explanatory; the exceptions are described below:

SNHPAR: (Bit 2) NP or Host Parity Error. This is set whenever the parity checking logic in the DPC detects a parity error for the NP or Host. Additional parity error bits which localize the specific error are also set in the dynamic status register.

SBFME: (Bit 1) Buffer Full or Empty Condition. This bit is set when an empty or full condition is detected in the receive FIFO (or the transmit FIFO if chain mode is inhibited). SRBMT or SRBFL of dynamic status bits being set also sets the SBFME. The NP can find out what caused the interrupt by examining the SXBMT, SXBFL, SRBMT, and SRBFL bits in the dynamic status register.

SRABT: (Bit 0) DPC Receive Write Abort. SRABT is set when the RBC sees RCVABT asserted while it is servicing the DPC write request.

Dynamic Status Register (DSTS)

The dynamic status register is different from the static status register. A status condition sets a corresponding bit, and this is read onto NP7-15 using an IRDMDDS instruction. However, this read doesn't cause an autoclear. Except for bit #7, each of the status bits disables the operation which has caused it to be set. When the error has been corrected, the NP can re-enable the operation by clearing the status bit. This is done by providing an ICLDSTS instruction accompanied by one of the bits described in Table 4. Table 5 describes the bit assignments in the dynamic status register.

Table 4. Instructions To Clear Dynamic Status Register
(Used with ICLDSTS Instruction)

Bit	Mnemonic	Function
15 (MSB)	CXMTBEE	Clear transmit buffer empty error (SXBMT) (not used with FDDI chain FIFOs)
14	CXMTBFE	Clear transmit buffer full error (SXBFL) (not used with FDDI chain FIFOs)
13	CRCVBEE	Clear receive buffer empty error (SRBMT)
12	CRCVBFE	Clear receive buffer full error (SRBFL)
11	CPENWR	Clear parity error on NP write (SPENWR)
10	CPENRD	Clear parity error on NP read (SPENRD)
9	CPEHWR	Clear parity error on Host write (SPEHWR)
8	CPEHRD	Clear parity error on Host read (SPEHRD)

Table 5. Dynamic Status Register Fields

Bit	Mnemonic	Function
15 (MSB)	SXBMT	Transmit FIFO empty (not applicable to FDDI)
14	SXBFL	Transmit FIFO full (not applicable to FDDI)
13	SRBMT	Receive FIFO empty
12	SRBFL	Receive FIFO full
11	SPENWR	Parity error for NP write
10	SPENRD	Parity error for NP read
9	SPEHWR	Parity error for Host write
8	SPEHRD	Parity error for Host read
7	SNPPND	NP request pending

SRBMT: (Bit 13) Receive FIFO Empty. This bit is set when a read is attempted and an empty condition is detected in the receive FIFO. If a read operation causes a buffer-empty condition, the error is flagged by not asserting the $\overline{\text{HSRDACK}}$ signal.

SRBFL: (Bit 12) Receive FIFO Full. This bit is set when a write operation is attempted and a full condition is detected in the receive FIFO. Setting this bit or SRBMT also sets the SBFME bit in the static status register. When the RBC is in Page mode (MPAGE = 1), these full and empty status bits can never be set.

The SRBFL condition causes a receive abort on the DPC side. Similarly, SXBMT causes a transmit abort. If a potential write operation causes a buffer-full condition, then WR is HIGH for that operation, so the memory contents are unaffected. Additional write attempts would not be served since the write pointer is locked at this location.

When both pointers (read and write) of a FIFO are initialized, no error is generated even though the pointers are made equal. Another likely pointer equal condition could arise when the WPR goes back to the beginning of the frame to write status and length, and the RPR is waiting there to begin reading this frame. This situation is noticed and no error is generated. The transmit full and empty conditions are never set when the RBC is in the chain transmit mode (MENCHN = 1). None of the full and empty conditions are set when the RBC is in the Page mode (MPAGE = 1).

SPENWR, SPENRD, SPEHWR, SPEHRD: (Status for Parity Errors for NP). Write, NP read, Host write and Host read, and the D-bus. These are set on parity errors on the D-bus during NP or Host write or read, respectively. The parity is checked or generated by the DPC when necessary, and the RBC is notified of errors to prevent further requests from being serviced. The status bit disables the corresponding operation that set this bit (e.g., SPEHWD disables further Host write requests on the pin). If any of these bits get set, the SNHPAR bit in the static status register also gets set.

Note that even though SPENWD or SPENRD disable further NP write requests or NP read requests, the NP can still give a read or write request using an instruction. In fact, the NP instruction request is always serviced even if a parity error occurs.

SNPPND: (Bit 7) NP Request Pending. If SNPPND is set HIGH, there is one NP read or write request pending arbitration and service. If SNPPND is LOW, no NP requests are pending, and the NP can make further requests. Since only one pointer is provided in the RBC (MAR) for the NP hardware and software DMA request, SNPPND is a useful signal provided to the NP to indicate if any request is yet unserviced.

SNPPND is also intended to be used with software requests through instructions. Whenever the NP wants to execute a software request, it should first disable the hardware DMA requests by clearing the MENNPRQ bit in the mode register. This prevents any further hardware requests from going through. However, a previous request may still be pending, waiting to be serviced. After this service is completed, SNPPND is reset and the NP can give a software request. Subsequent software requests should only be given after SNPPND is reset.

RBC-DPC Interaction

The DPC interacts with the RBC by transferring frames received from the media into the Buffer Memory receive FIFO, and by transferring frames to be transmitted from the Buffer Memory S-frame chain FIFO or A-frame chain FIFO to the media. During the transfer, the DPC converts the received data from 8-bit bytes to 32-bit words, and the data to be transmitted from 32-bit words to 8-bit bytes.

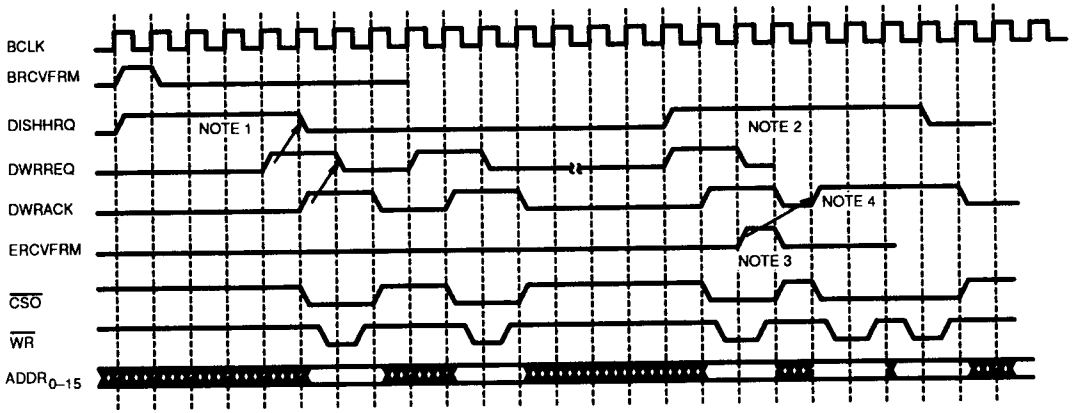
Received Data

The DPC transfers received data by issuing write requests to the RBC. The RBC uses the WPR pointer to serve this request.

At any time the DPC can receive or transmit, but not simultaneously. During frame reception, the DPC generates write requests to the RBC to write the frame in the Buffer Memory. When serviced, the RBC enables the Buffer Memory to write a long word of data present on the D-bus using the address contained in the RBC pointer called WPR. After writing the location, the RBC increments WPR, setting up the pointer for the next word to be written. When WPR equals EAR, it then wraps around to the SAR value.

For received frames (DPC write requests), the DPC does the housekeeping for each frame. It accumulates status information and frame length, which it sends to the Buffer Memory at the end of each normal frame transfer (if the mode register bit MSLNSTM is set HIGH). The RBC adds this data to the beginning of the frame in Buffer Memory.

The timing relationships of the various signal lines involved in receiving one normal frame (MSLNSTM bit is HIGH) is shown in Figure 9.



Notes:

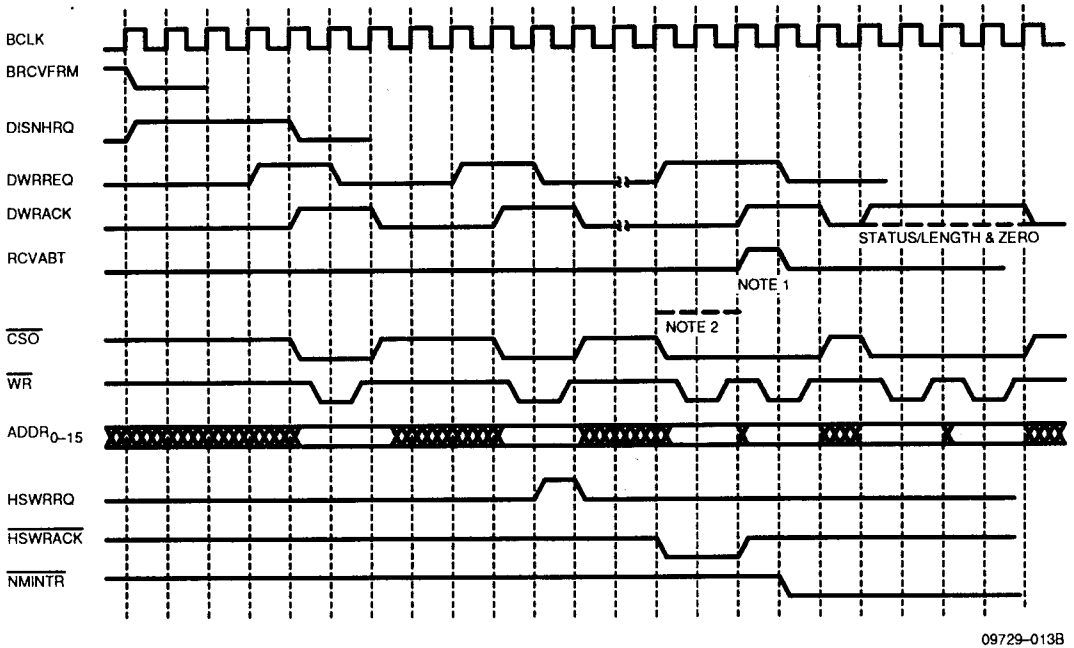
1. When the DPC begins to receive a packet, it asserts the DISNHRQ pin until the first DWRREQ.
2. DISNHRQ is asserted one clock cycle after RECEIVE is driven LOW by the FORMAC. DISNHRQ remains HIGH until the zero field is written.
3. The earliest that ERCVFRM can be asserted by the DPC is at the second cycle of the DWRACK for the last word of the receiving frame. One cycle after ERCVFRM, the RBC sets WPR to point to the start of the frame (the skipped location) for writing the status and length of the frame.
4. One clock cycle after ERCVFRM goes inactive, DWRACK is active for four clock cycles for writing the status and length at the beginning of the frame and zero at the end of the frame.

Figure 9. Receive One Normal Frame

Aborted Receive Frame

When a received frame gets aborted (RCVABT is HIGH), the RBC stores the status and length of the aborted frame and writes zero at the next memory location.

The timing of a receive frame that gets aborted at the middle is shown in Figure 10. The MSLNSTM bit in RBC is HIGH.



Notes:

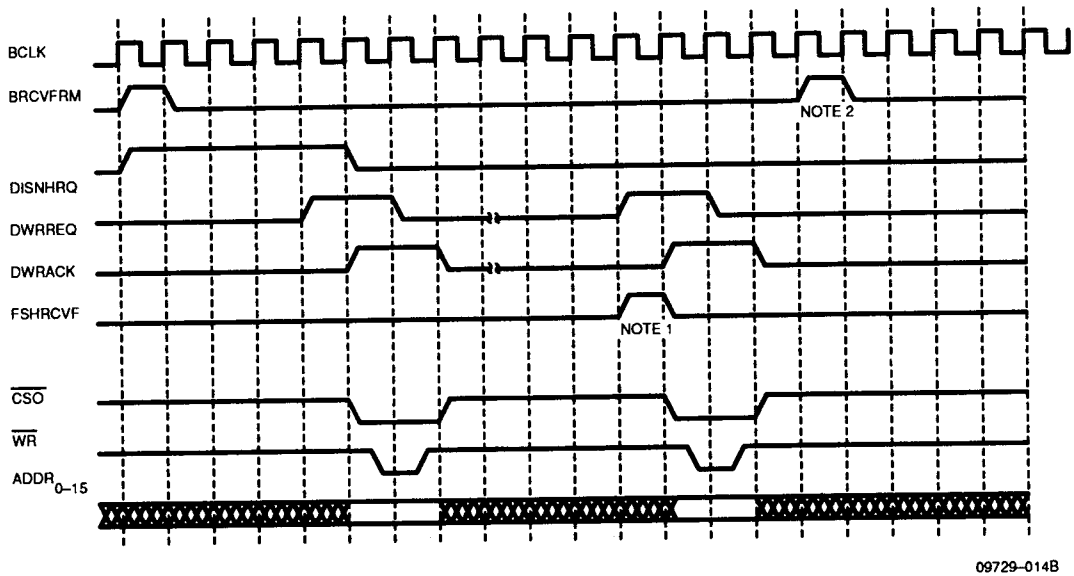
1. RCVABT can become active anytime after BRCVFRM becomes active. In this example, RCVABT occurs when DWRACK is active.
2. In this example, DWRACK takes two clocks after DWRREQ to be active because the RBC is servicing a previous Host write request. At the second clock of the service, the RBC arbitrates again and sees DWRREQ. Therefore, it takes the RBC at most two clock cycles to acknowledge the DPC read or write request.

Figure 10. Aborted Receive Frame

Flushed Receive Frame

The timing of a receive frame that is flushed in the middle of the receive is shown in Figure 11. The flush function is usually used when the frame has a mismatched destination address. The RBC has the MSLNSTM bit HIGH in this example.

When a frame is being received and the FORMAC asserts FSHRCVP, the whole receiving frame is ignored and no status is written for that frame.



Notes:

1. When the RBC receives FSHRCVF, it resets its WPR to the beginning of the packet after it finishes the present DWRACK (if there is one) and waits for a new frame (waits for BRCVFRM).
2. At the earliest, BRCVFRM for the next frame could be asserted one clock after DWRACK of the previous frame is de–asserted.

Figure 11. Flushed Receive Frame

Receive Frame Without Status and Length

When the RBC is programmed in MSLNSTM = 0 mode, the frames are stored next to each other without frame status/length at the start of each frame.

Receive FIFO Full Error

When the RBFERR becomes HIGH, the RBC detects a full error in the receive buffer. When RBFERR becomes HIGH, $\overline{WR}$ is not asserted for that DWRREQ even though $\overline{CSO}$ and DWRACK are active as usual. After that point, further DWRREQs are ignored by the RBC, and RBFERR stays HIGH.

RBFERR sets the NMINTR pin, which stays set until the Node Processor reads the status register (IRDSTAT).

An example of the timing for a receive FIFO Full Error is shown in Figure 12. The RBC has the MPAGE bit set LOW.

The receive buffer full or empty errors behave the same:

1) No acknowledgement is given;

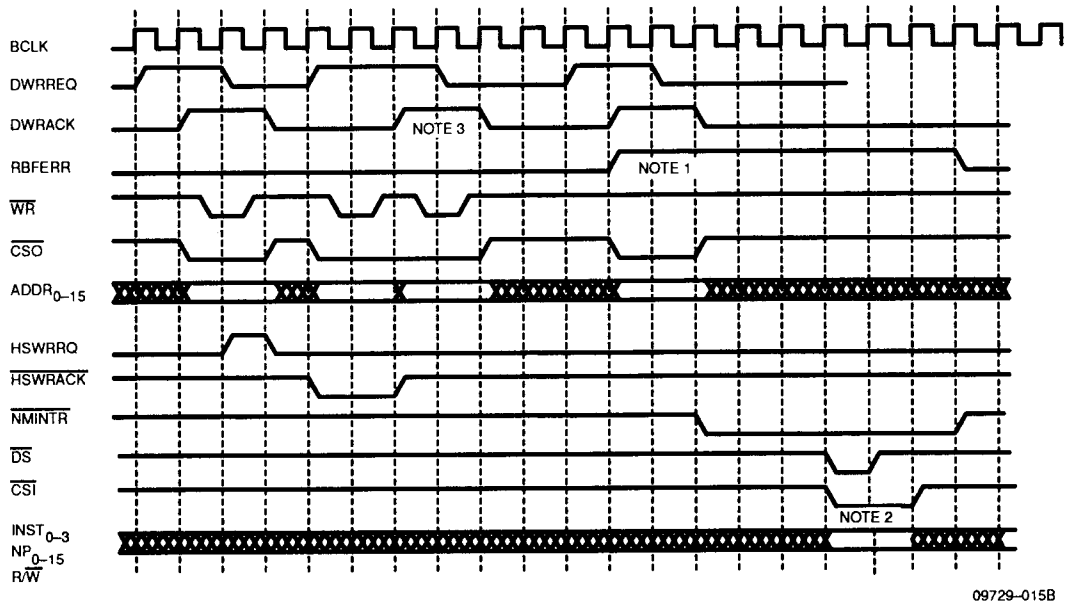
2) The appropriate status bit is set to interrupt the NP;

3) Further requests related to the error are ignored, so the pointer stays as it was when the error occurred;

4) The error is cleared only by the NP through the INST lines and NP- bus; and

5) The RBC still receives and services other requests not related to the error.

After the full or empty error is cleared, the Node Processor should reload a new value to the pointer that caused the problem to ensure that the frame is received properly.



Notes:

1. When the RBFERR becomes active, the RBC detects a full error in the receive buffer. When RBFERR becomes active, $\overline{WR}$ is not asserted for that DWRREQ even though $\overline{CSO}$ and DWRACK are active as usual. After that point, futher DWRREQ are ignored by the RBC, and RBFERR stays active.RBFERR sets the $\overline{NMINTR}$ pin which stays set until the Node Processor reads the status register (IRDSTAT).
2. The Node Processor recognizes the full error and clears it by asserting the appropriate value for the INST bus and the NP bus (ICLDSTS). One cycle after that, RBFERR and $\overline{NMINTR}$ become inactive,and the DPC may start writing a new frame (BRCVFRM should precede a new frame).
3. In this example, it takes two clocks for the DWRREQ to be acknowledged because the RBC is busy servicing a Host write request.

Figure 12. Receive FIFO Full Error

Data To Be Transmitted

Data to be transmitted using FDDI standards is divided into two categories: S-frames and A-frames. Each category is stored in the Buffer Memory in a separate chain FIFO. Through the use of two timers, THT and T-late, the FORMAC can determine whether to access the S-frame chain or the A-frame chain for the next transmission.

When the S-frame FIFO has data, the DPC issues a request to the FORMAC to transmit S-frames. When the A-frame FIFO has data, the DPC issues a request to the FORMAC to transmit A-frames. When the FORMAC receives a token, the FORMAC timers determine the type of frame to send, and the FORMAC acknowledges the DPC request for that type of data if a DPC request is pending. The DPC then issues a DRDREQS or DRDREQA to the RBC to transfer the type of data the FORMAC authorized. The RBC uses two pointers, RPXA and RPXS, for the A-frames and S-frames respectively.

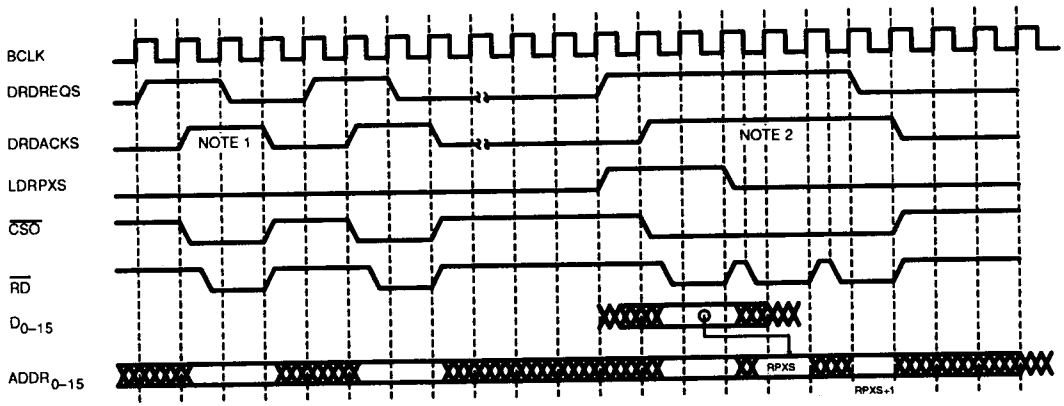
All Buffer Memory access requests from the NP, DPC, and Host arriving at the RBC are serviced on a first-come, first-served basis. However, if the NP or Host and the DPC try to access the memory at the same time, then the DPC is serviced and the other request is placed in a pending queue.

S-frame Transmission

For S-frame transmission, the DPC will issue a DRDREQS request for each long word in each frame. RPXS is incremented automatically by the RBC after each long word is transferred until an LDRPXS is issued with the last request.

The DPC must know the length of a frame to determine when the end of a frame is reached. This information is in the descriptor at the start of the frame. When the end of a frame is reached, the DPC will issue an LDRPXS along with the DRDREQS. This lets the RBC know that the RPXS is pointing to the last location in the frame. This location contains the address of the start of the next frame. The RBC transfers this value to the RPXS pointer (RPXS is loaded from the D_0-D_{15} bus). If the LDRPXS is not issued with the DRDREQS, the current value in RPXS is used.

The timing diagram to transmit an FDDI S-frame in chain FIFO transmit mode (MENCHN = 1) is shown in Figure 13.



09729-016B

Notes:

1. As with DWRREQ, the RBC takes one or two clocks to acknowledge DRDREQS.
2. To get the RBC to load a new value into its RPXS, the DPC asserts and then de-asserts DRDREQS and LDRPXS simultaneously. After that cycle, RPXS gets its new value from the D-bus to start reading a new frame. The RPXS is incremented after each word is transferred until the RBC again receives a DRDREQS and LDRPXS simultaneously.

Figure 13. Transmit an S-frame

A-frame Transmission

To send A-frames, the DPC will issue a DRDREQA request for each long word in each frame. RPXA is incremented automatically by the RBC after each long word is transferred until an LDRPXA is issued with the last request.

The DPC must know when the end of a frame is reached so that it can issue an LDRPXA along with the DRDREQA. This lets the RBC know that the RPXA is pointing to the last location in the frame. This location contains the address of the start of the next frame. The RBC transfers this value to the RPXA pointer (RPXA is loaded from the D_0-D_{15} bus). If the LDRPXA is not issued with the DRDREQA, the current value in RPXA is used.

As with DWRREQ, the RBC takes one or two clocks to acknowledge DRDREQS (A).

Claim/Beacon Mode

Claim or beacon mode can be initiated only when the RBC is in the FDDI (chain queue mode (MENCHN = 1). If MENCHN = 0, the two pins INICLBN and CLM/BEC are ignored.

INICLBN can be HIGH any time during the normal operation of the RBC. If INICLBN is HIGH during a receive window, it will be treated as a receive abort. If INICLBN is HIGH simultaneously with ERCVFRM, then the RBC will finish storing that frame before switching to claim or beacon mode.

Figure 14 illustrates the claim/beacon mode of the RBC. The MENCHN bit in the RBC is HIGH.

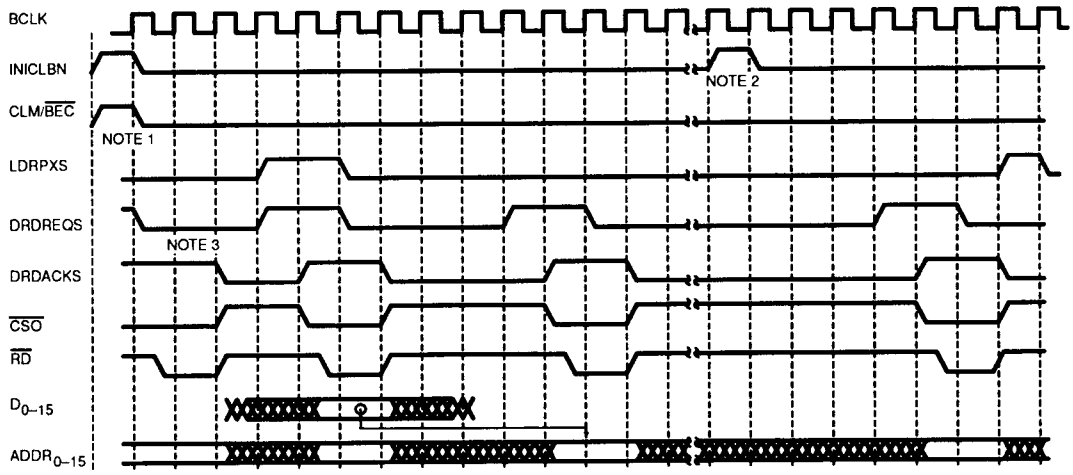
As part of the initialization, the NP must load the ACP register with the address of a two long-word block of Buffer Memory that contains a pointer to the claim frame followed by a pointer to the beacon frame.

When INICLBN and CLM/$\overline{\text{BEC}}$ are both HIGH, the RBC is put into claim mode. When INICLBN is HIGH and CLM/$\overline{\text{BEC}}$ is LOW, the RBC is put into beacon mode. In claim mode, the RBC loads the RPCB register with the contents of the ACP register. In beacon mode, the RBC loads the RPCB register with the contents of the ACP+1.

When INICLBN is HIGH, the DPC will bring DRDREQS LOW. The DPC has to wait one clock after DRDACKS becomes LOW before giving a read request for transmitting a claim or beacon frame. The claim or beacon mode is not entered until the RBC finishes all DRDREQSs that are pending. The maximum latency between the DRDREQS for a normal transmit and DRDREQS for a claim or beacon transmit is three clock cycles.

The DPC will drive DRDREQS and LDRPXS HIGH to initiate the claim or beacon process. The RBC will drive the ADDR bus with the contents of the RPCB register and will load the resulting value from the $D_{0\text{-}D15}$ bus into the RPCB register.

The DPC will drive the LDRPXS HIGH with DRDREQS LOW one clock cycle after XMTABTO is received from the FORMAC to signal the end of a claim or beacon process.



09729-017B

Notes:

1. INICLBN and CLM/$\overline{\text{BEC}}$ both HIGH simultaneously drive the RBC to the Claim mode.
2. INICLBN HIGH and CLM/$\overline{\text{BEC}}$ LOW drive the RBC to the Beacon mode.
3. When INICLBN is active, the DPC should de–assert DRDREQS. Then it has to wait one clock after DRDACKS is inactive to give a read request for transmitting claim or beacon frame. The Claim or Beacon mode is not entered until the RBC finishes all the DREDQS pending. The maximum latency between the DRDREQS for normal transmit and DRDREQS for Claim or Beacon is three clock cycles.

Figure 14. Claim or Beacon Mode

RBC-Host Interaction

The Host is the link between the Buffer Memory and the upper levels of the station. It removes received frames from the receive FIFO by issuing read requests to the RBC. The RPR pointer is used for the Host reads. The Host also issues write requests to load the S-frame FIFO and the A-frame FIFO in Buffer Memory with frames to be transmitted. The WPX pointer is used for this purpose. The NP maintains the correct addresses in the RPR and the WPX registers based on information from the Host.

Host Requests

Host requests for access to the Buffer Memory use the HSRDRQ and HSWRRQ pins. When HSRDRQ is serviced, $\overline{\text{HSRDACK}}$ becomes LOW. The Host can then read the data on the D-bus. When HSWRRQ is serviced, $\overline{\text{HSWRACK}}$ and $\overline{\text{HSWACKE}}$ become LOW and the Host can write data to the D-bus.
The HSRDRQ pin is typically used for emptying out the data from the receive FIFO. The RPR value is selected on the ADDR-bus when HSRDRQ is serviced.

The HSWRRQ pin is used for loading S-frames and A-frames into the chain FIFOs for transmission. The WPX value is selected on the ADDR-bus when HSWRRQ is serviced. HSWRRQ and HSRDRQ have equal priority and are serviced alternatively.

Two mode bits are provided to independently enable/disable the HSRDRQ and HSWRRQ pins. If MENHRRQ is HIGH, HSRDRQ is enabled; if LOW, disabled. In the same way, if MENHWRQ is HIGH, HSWRRQ is enabled; if LOW, disabled.

The acknowledge pins for the NP and Host are also used by the DPC to generate or check parity for those requests when necessary.

The NP and Host pin requests can be either synchronous or asynchronous with the RBC depending on the MSYNHRQ mode bit. In that way, the RBC can service both synchronous and asynchronous DMA requests.

ABSOLUTE MAXIMUM RATINGS

Storage Temperature	-65 to +150°C
Ambient Operating Temperature	-55 to +125°C
Maximum V_{CC}	-0.3 to +7.0 V
DC Voltage Applied to Any Pin	-0.5 to V_{CC} +0.3 V

Stresses above those listed under ABSOLUTE MAXIMUM RATINGS may cause permanent device failure. Functionality at or above these limits is not implied. Exposure to absolute maximum ratings for extended periods may affect device reliability.

OPERATING RANGES

Commercial (C) Devices

Ambient Temperature (T_A)	0 to +70°C
Supply Voltage (V_{CC})	+4.75 to +5.25 V

Extended Commercial (E) Devices

Ambient Temperature (T_A)	-55 to +125°C
Supply Voltage (V_{CC})	+4.5 to +5.5 V

Operating ranges define those limits between which the functionality of the device is guaranteed.

DC CHARACTERISTICS over COMMERCIAL operating range

Parameter Symbol	Parameter Description	Test Conditions	Min.	Max.	Unit
V_{IL}	Input LOW Voltage			0.8	V
V_{IH}	Input HIGH Voltage		2.0		V
V_{OL}	Output LOW Voltage	I_{OL} = 4.0 mA		0.4	V
V_{OH}	Output HIGH Voltage (Note 1)	I_{OH} = -4.0 mA	2.4		V
I_{IX}	Input Leakage Current (Note 2)	0V< V_{IN} < V_{CC}	-10	10	µA
I_{OZ}	Output Leakage Current (Note 3)	0.4 V<V_{OUT} < V_{CC}	-10	10	µA
I_{CC}	Power Supply Current	V_{CC} = Max. f(BCLK) =12.5 MHz		120	mA

Notes:
1. V_{OH} does not apply to open-drain output pins.
2. I_{IX} applies to all input-only pins.
3. I_{OZ} applies to all three-state output pins and bidirectional pins.

CAPACITANCE*

Parameter Symbol	Parameter Description	Typ.	Unit
C_{IN}	Input Pins	15	pF
C_{IO}	Bidirectional Pins	15	pF

* Pin capacitance is characterized at a frequency of 1MHz, but is not 100% tested.

SWITCHING CHARACTERISTICS over COMMERCIAL operating range (Notes 1 and 3)

Parameter Number	Parameter Symbol	Parameter Description	Min. (Note 2)	Max. (Note 2)	Unit
1	Clock Period	BCLK	80		ns
2	HIGH Pulse Width	BCLK	35		ns
3	LOW Pulse Width	BCLK	35		ns
4	Setup Time Before BCLK ↑	$\overline{CSI}$,$\overline{DS}$, $INST_{0-3}$, R/$\overline{W}$	45		ns
5	Hold Time After BCLK ↑	$\overline{CSI}$, R/$\overline{W}$	2		ns
6	Hold Time After BCLK ↑	$\overline{DS}$	6		ns
7	Hold Time After BCLK ↑	$INST_{0-3}$	10		ns
8	Hold Time After BCLK ↑	NP_{0-15}	15		ns
9	Setup Time Before BCLK ↑	NP_{0-15}	12		ns
10	R/$\overline{W}$, ↑ $\overline{CSI}$,$\overline{DS}$ (Whichever Occurs Last) Until NP-bus is Enabled (Synchronous Mode)	NP_{0-15}	3		ns
11	Signal Invalid After BCLK ↑	NP_{0-15}		51	ns
12	Signal Invalid After BCLK ↑	NP_{0-15}	3		ns
13	Signal Invalid After CSI ↑ or R/$\overline{W}$ ↓ (Whichever Occurs First at the End of a Synchronous Read Cycle)	NP_{0-15}	3		ns
14	R/$\overline{W}$ ↓ or $\overline{CSI}$ ↑ (Whichever Occurs First at the End of a Synchronous Read Cycle) to Bus Inactive	NP_{0-15}		35	ns
15	Hold Time After $\overline{DS}$ ↑ or $\overline{CSI}$ ↑ (Whichever Occurs First in Asynchronous Write)	NP_{0-15}	0		ns
16	Signal LOW After NP-bus Valid	READY	(T1 - 20)		ns
17-19	Unused				
20	Setup Time Before $\overline{CSI}$ ↓ or $\overline{DS}$ ↓ (Whichever Occurs Last in Asynchronous Read/Write)	R/$\overline{W}$ ↓ (Write) or R/$\overline{W}$ ↑ (Read), $INST_{0-3}$	0		ns
21	Unused				
22	Pulse Width HIGH (Asynchronous Read or Write)	$\overline{DS}$,$\overline{CSI}$	1.5 x T1		ns
23	Hold Time After $\overline{DS}$ ↓ or $\overline{CSI}$ ↑ (Whichever Occurs First at the End of Asynchronous Read/Write)	R/$\overline{W}$ ↑ (Write) or R/$\overline{W}$ ↓ (Read), $INST_{0-3}$	0		ns
24	Signal LOW After DS ↓ or $\overline{CSI}$ ↓	READY	3 x T1	(4 x T1 + 70)	ns
25	Signal Disabled After DS ↑ or $\overline{CSI}$ ↑	READY		40	ns
26	Bus Active After $\overline{DS}$ ↓ or $\overline{CSI}$ ↓ (Whichever Occurs Last in Asynchronous Read)	NP_{0-15}	0		ns
27	Bus Valid After DS ↓ or CSI ↓ (Whichever Occurs Last in Asynchronous Read)	NP_{0-15}		(3 x T1 + 48)	ns

SWITCHING CHARACTERISTICS, (Continued)

No.	Parameter	Signal Name	Min. (Note 2)	Max. (Note 2)	Unit
28	Signal Invalid After DS ↑ or CSI ↑ (Whichever Occurs First in Asynchronous Read)	NP_{0-15}	0		ns
29	Bus Disabled After DS ↑ or CSI ↑ (Whichever Occurs First in Asynchronous Read)	NP_{0-15}		35	ns
30	Setup Time Before DS ↓ or CSI ↓ (Whichever Occurs First in Asynchronous Write)	NP_{0-15}	-(T1 x 2 20)		ns
31	Setup Time Before BCLK ↑	BRCVFRM, ERCVFRM, FSHRCVF, INICLBN, CLM/$\overline{BEC}$, DISNHRQ, RCVABT, PARERR	20		ns
32	Hold Time After BCLK ↑	BRCVFRM, ERCVFRM, FSHRCVF, INICLBN, CLM/$\overline{BEC}$, DISNHRQ	8		ns
33	Setup Time Before BCLK ↑	DRDREQS, DRDREQA, DWRREQ	20		ns
34	Hold Time After BCLK ↑	DRDREQS, DRDREQA, DWRREQ	8		ns
35	BCLK ↑ to Signal Invalid Delay	DWRACK, MDWRACK		36	
36	BCLK ↑ to Signal Valid Delay	DWRACK, DRDACKA, DRDACKS, MDWRACK, MDRDACK		47	ns
37	BCLK ↑ to Signal Invalid Delay	DWRACK, DRDACKS, DRDACKA, MDWRACK, MDRDACK	8		ns
38	Hold Time After BCLK ↑	RCVABT, PARERR	7		ns
39	BCLK ↑ to Signal Valid Delay	RBFERR, XBEERR		50	ns
40	BCLK ↑ to Signal Disabled Delay (Note 1h)	$\overline{NMINTR}$	0		ns
41	BCLK ↑ to Signal LOW Delay	$\overline{NMINTR}$		45	ns
42-44	Unused				
45	Setup Time Before BCLK ↑	DISRBC	30		ns
46	Hold Time After BCLK ↑	DISRBC	5		ns
47	Unused				
48	Setup Time Before BCLK ↑	LDRPXS, LDRPXA	20		ns
49	Hold Time After BCLK ↑	LDRPXS, LDRPXA	8		ns
50	Setup Time Before BCLK ↑	HSRDRQ, HSWRRQ, NPRDRQ, NPWRRQ	20		ns
51	Hold Time After BCLK ↑	HSRDRQ, HSWRRQ, NPRDRQ, NPWRRQ	5		ns
52	Pulse Width HIGH for Single DMA Transfer (Asynchronous Mode)	NPRDRQ, NPWRRQ, HSRDRQ, HSWRRQ	(T1 + T51)	(2 x T1) - T50	ns
53	BCLK ↑ to Signal Invalid Delay	$\overline{HSRDACK}$, $\overline{NPRDACK}$, $\overline{HSWRACK}$, $\overline{NPWRACK}$	5		ns
54	BCLK ↑ to Signal Valid Delay	$\overline{HSRDACK}$, $\overline{NPRDACK}$ $\overline{HSWRACK}$, $\overline{NPWRACK}$		50	ns
55	BCLK ↑ to Signal Valid Delay	$\overline{CSO}$		45	ns
56	BCLK ↑ to Signal Invalid Delay	$\overline{CSO}$	5		ns

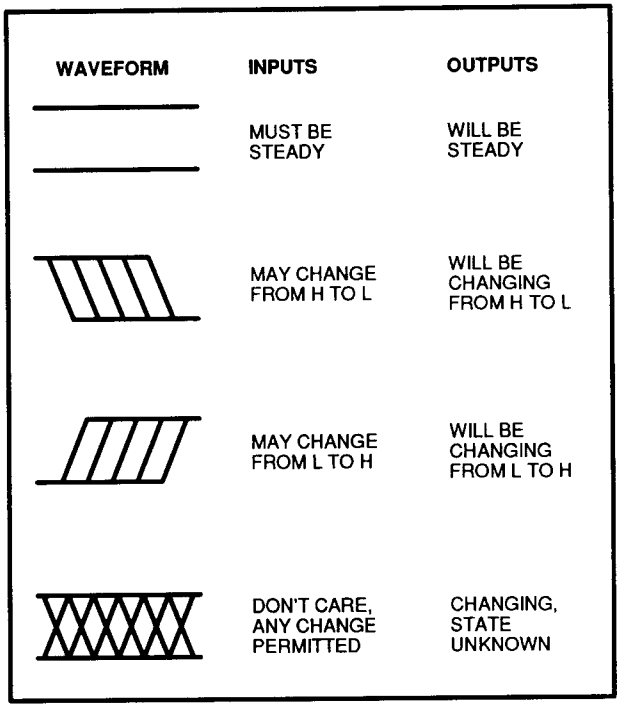
SWITCHING CHARACTERISTICS, (Continued)

No.	Parameter	Signal Name	Min. (Note 2)	Max. (Note 2)	Unit
57	BCLK ↑ to Signal Valid Delay	$ADDR_{0-15}$		70	ns
58	BCLK ↑ to Signal Invalid Delay	$ADDR_{0-15}$	6		ns
59	BCLK ↑ to Signal Valid Delay	ACKONE		35	ns
60	BCLK ↑ to Signal Invalid Delay	ACKONE	8		ns
61	Bus Valid Before $\overline{RD}$ ↓	$ADDR_{0-15}$	10		ns
62	Request Hold Time After Acknowledge	NPRDRQ, HSRDRQ, NPWRRQ, HSWRRQ	0		ns
63-70	Unused				
71	$\overline{RD}$ ↑ to Signal Invalid Delay	$ADDR_{0-15}$	-15		ns
72	Unused				
73	BCLK ↑ or BCLK ↓ to Output Invalid Delay	$\overline{RD}$	5		ns
74	Setup Time Before BCLK ↑	D_{0-15}	10		ns
75	Hold Time After BCLK ↑	D_{0-15}	10		ns
76	BCLK ↑ or BCLK ↓ to Output Valid Delay	$\overline{RD}$	6	50	ns
77	BCLK ↓ to Signal Invalid Delay (Note 4)	$\overline{WR}$	90		ns
78	MDWRACK ↑ or DWRACK ↑ to Signal Invalid Delay (Note 4)	$\overline{WR}$	93		ns
79	$ADDR_{0-15}$ Valid to Signal Invalid Delay (Note 4)	$\overline{WR}$	72		ns
80	Bus Valid Before $\overline{WR}$ ↓	$ADDR_{0-15}$	10		ns
81	$\overline{WR}$ ↑ to Signal Invalid	$ADDR_{0-15}$	35		ns
82	Unused				ns
83	BCLK ↑ to Signal Invalid Delay	$\overline{HSWACKE}$,$\overline{NPWACKE}$		50	ns
84	Signal Invalid After $\overline{WR}$ ↑	$\overline{HSWACKE}$,$\overline{NPWACKE}$	T3		
85	BCLK ↓ to Signal Valid Delay	$\overline{HSWACKE}$,$\overline{NPWACKE}$		40	ns
86	BCLK ↓ to Signal Valid Delay	$\overline{WR}$	6	42	ns

Notes:

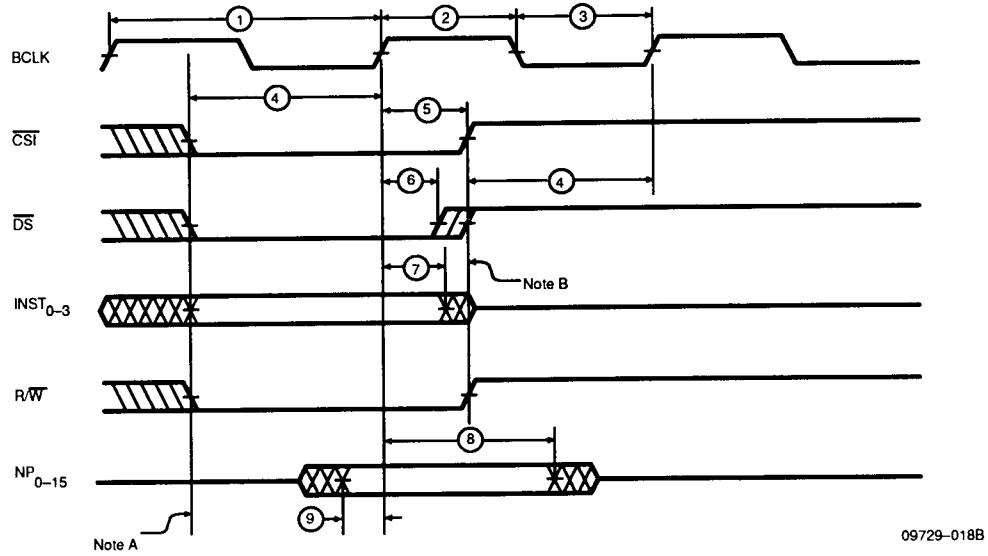
1. Measurement points for timing parameters are the following:
   a) Input waveforms: +1.4 V
   b) Output HIGH threshold: > 2.0 V
   c) Output LOW threshold: < 0.8 V
   d) Output valid: < 0.8 V or > 2.0 V
   e) Output invalid minimum: output is verified to be still valid at the minimum spec time.
   f) Output enabled: inflection point at which the output driver turns on.
   g) Output disabled: inflection point at which the output driver turns off.
   h) Guaranteed by design
2. Numbered parameters such as T1 or T51 refer to the switching characteristic number listed in the far left-hand column.
3. Maximum "BCLK to Signal Valid" delay and minimum "BCLK to Signal Invalid" delay specifications apply to both rising and falling edges of the signal listed, even though the measurement is only shown once in the Switching Waveforms section that follows. Valid and Invalid do not mean HIGH or LOW; see notes 1d and 1e and Switching Waveforms for the definition of Valid and Invalid.
4. a) Data setup time for buffer memory is T78 (RBC) - T34 (DPC), or T77 (RBC) - T35 (DPC) if MDWRACK or DWRACK ↑ before BCLK ↓.
   b) Address setup time to $\overline{WR}$ ↑ for buffer memory is always T79 (RBC).

Key to Switching Waveforms



09730–040A

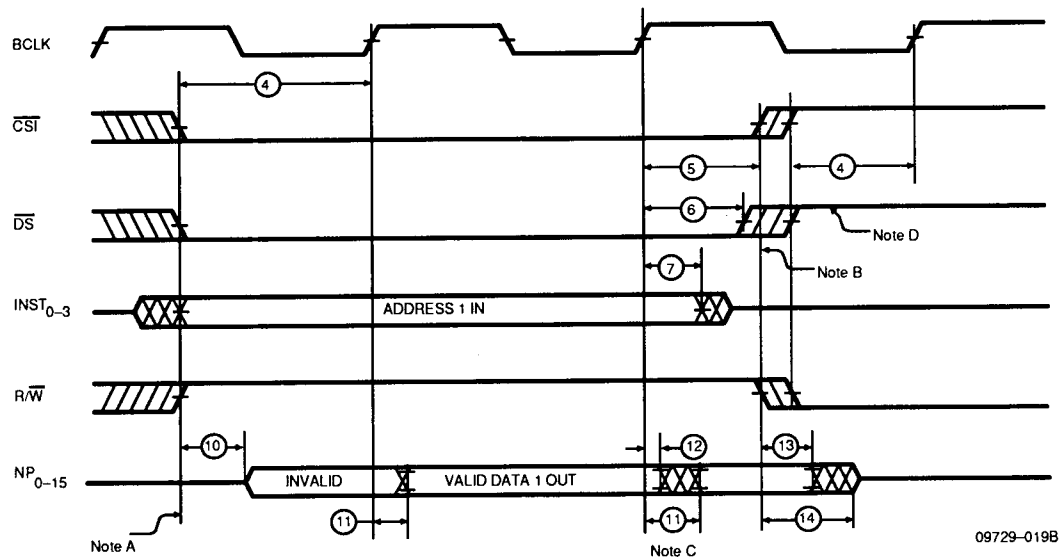
SWITCHING WAVEFORMS



Notes:

A. Timing measured from falling edge of CSI, DS, or R/W or the assertion of INST 0-3, whichever occurs last.
B. Parameter #5 is measured from the rising edge of CSI, or R/W, whichever occurs first;
Parameter #4 is measured from the rising edge of CSI, DS, or R/W, or the assertion of INST 0-3, whichever occurs first.

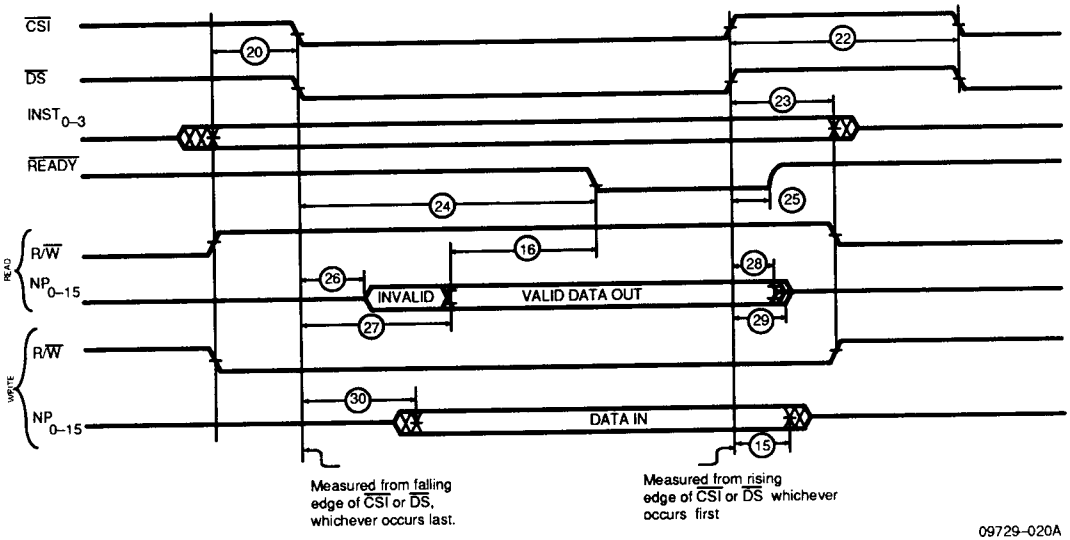
NP-bus Synchronous Write Timing



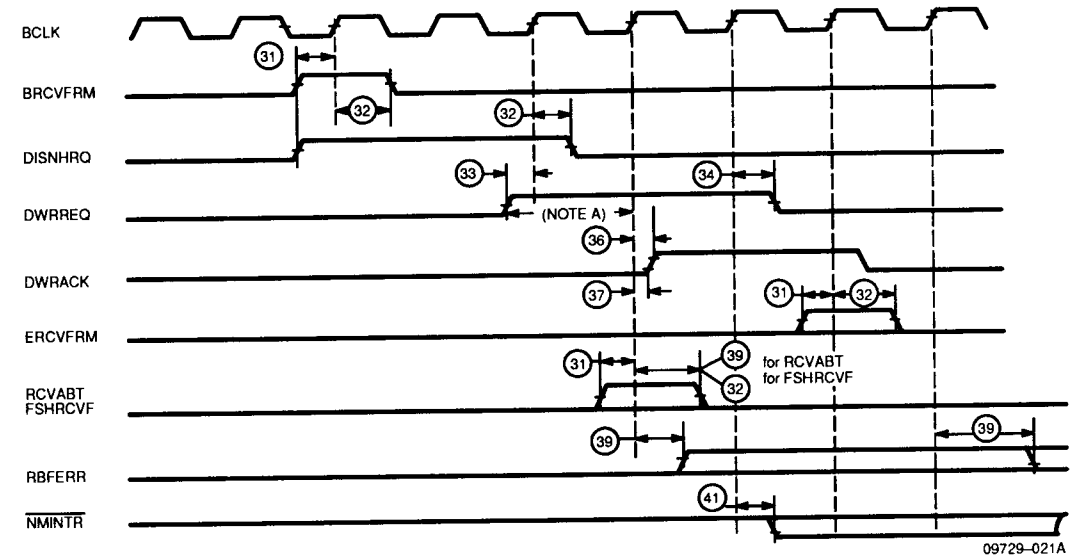
Notes:

A. Timing measured from falling edge of CSI, DS, or the rising edge of R/W (or the assertion of INST 0-3, for parameter #4 only) whichever occurs last.
B. Timing is measured from the rising edge of CSI or the falling edge of R/W, whichever occurs first.
C. If an address register iln the FORMAC is being addressed, then the regilster being read onto the NP-bus will be incremented with each rising edge of BCLK beginning with the third BCLK ↑ .
D. DS must be driven HIGH at the completion of each read cycle.

NP-bus Synchronous Read Timing



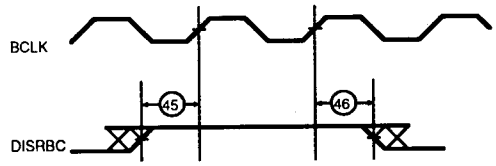
NP-bus Asynchronous Read/Write Timing



Notes:

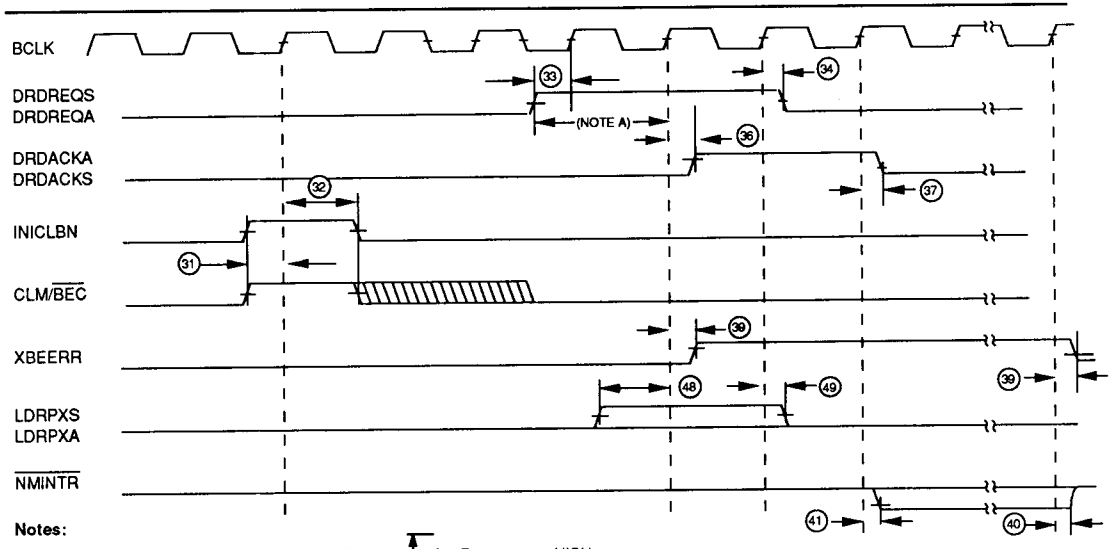
A. DWRACK will go HIGH after the first or second BCLK ↑ after DWRREQ goes HIGH.

RBC Frame Receive Timing



09729–022A

DISRBC Timing

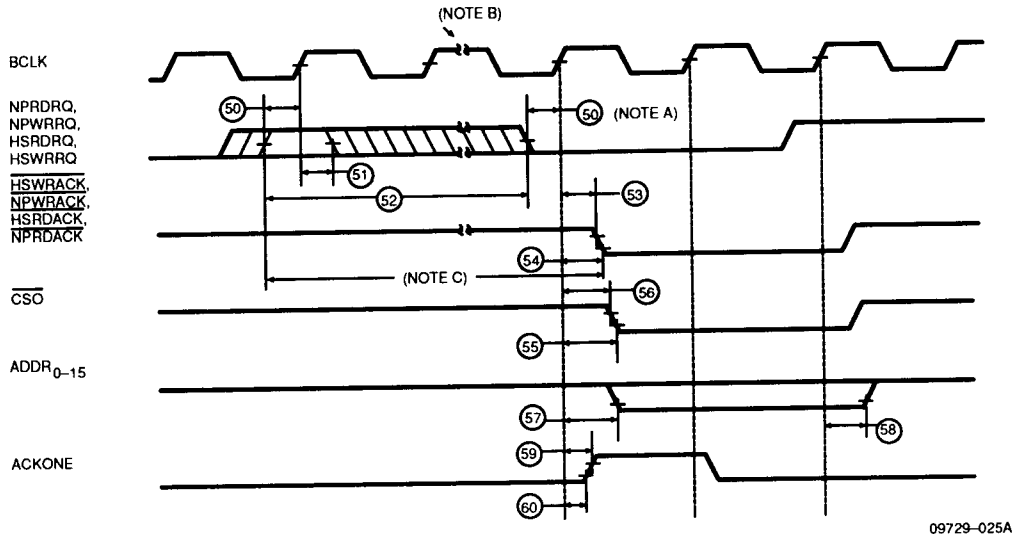


Notes:

A. Acknowledge goes HIGH on the first or second BCLK ↑ after Request goes HIGH.

09729B-24

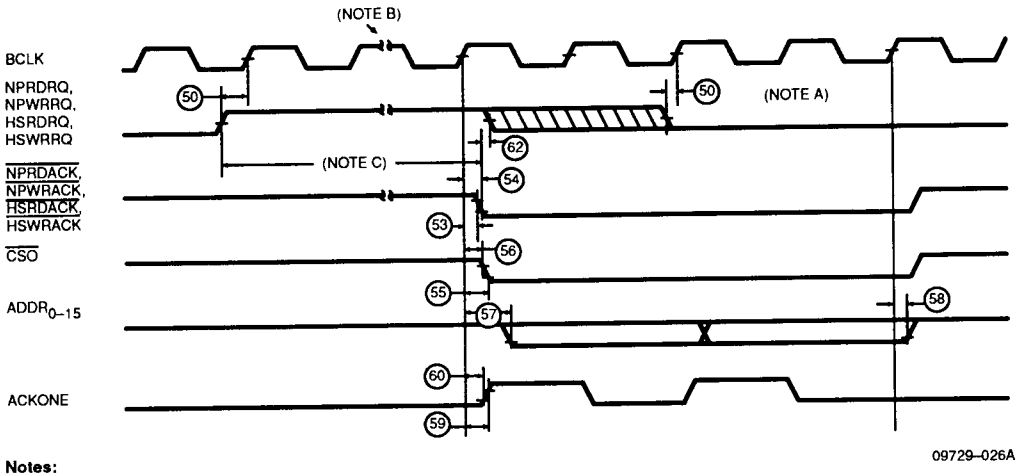
RBC Frame Transmit Timing



Notes:

A. If the request line is held HIGH longer than this, another transfer cycle will be activated by the RBC. If the acknowledge line is active, then the request line is sampled on every second rising edge of BCLK; if the acknowledge line is inactive, then the request line is sampled on each rising edge of BCLK.
B. If MSYNHRQ = 1, then subtract one BCLK cycle here.
C. Acknowledge (for $\overline{\text{NPRDACK}}$, $\overline{\text{HSRDACK}}$, $\overline{\text{NPWRACK}}$, or $\overline{\text{HSWRACK}}$ signals) goes LOW no sooner than the second LOW–to–HIGH transition of BCLKafter the corresponding request line goes HIGH if MSYNHRQ = 1, and no sooner than the third LOW–to–HIGH transition of BCLK if

NP-Host DM Request /Acknowledge Timing
(Single Transfer)

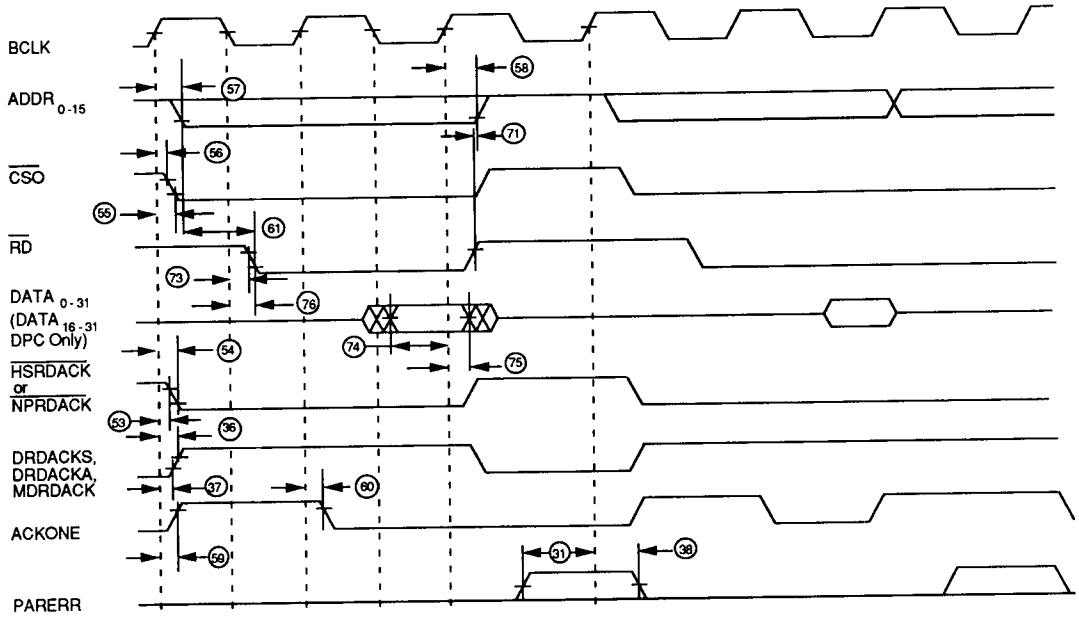


Notes:

A. the request line is held HIGH longer than this, another transfer cycle will be activated by the RBC. If the acknowledge line is active, then the request line is sampled on every second rising edge of BCLK; if the acknowledge line is inactive, then the request line is sampled on each rising edge of BCLK.
B. MSYNHRQ = 1, then subtract one BCLK cycle here.
C. Acknowledge (for $\overline{\text{NPRDACK}}$, $\overline{\text{HSRDACK}}$, $\overline{\text{NPWRACK}}$, or $\overline{\text{HSWRACK}}$ signals) goes LOW no sooner than the second LOW–to–HIGH transition of BCLK after the corresponding request line goes HIGH if MSYNHRQ = 1, and no sooner than the third LOW–to–HIGH transition of BCLK if MSYNHRQ = 0.

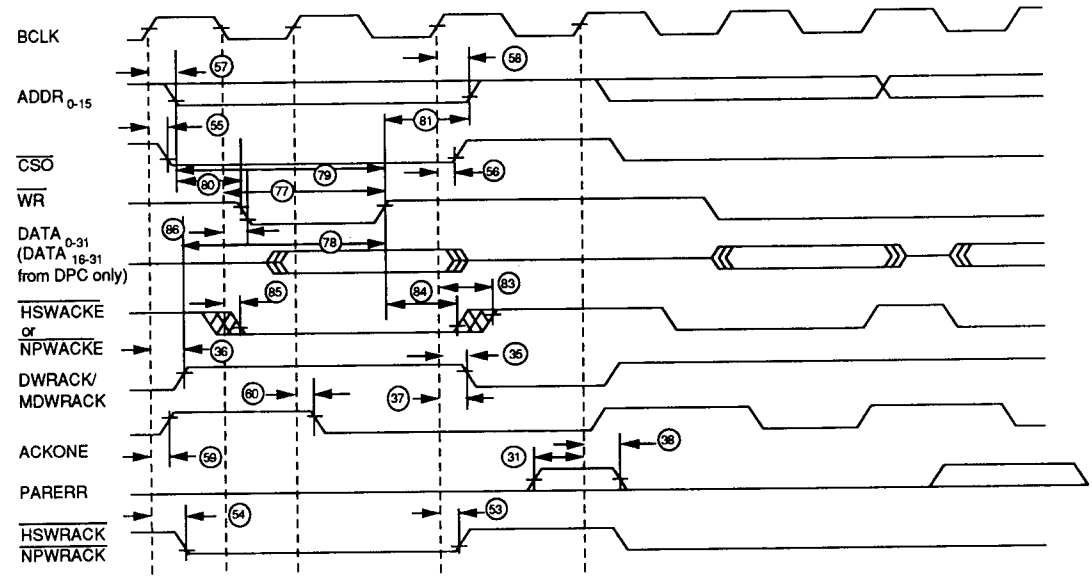
NP/Host Request/Acknowledge Timing
(Multiple Transfers)

SWITCHING WAVEFORMS, (Continued)



09729-026B

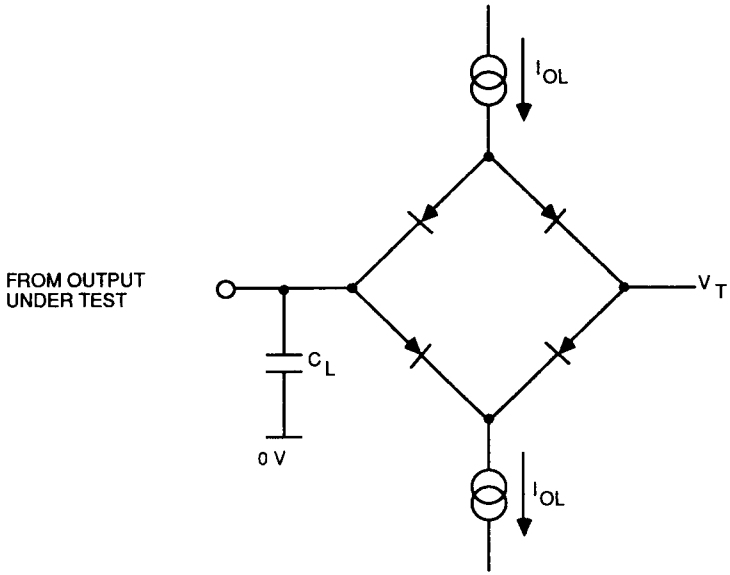
Buffer Memory Write Timing



09729-027A

Buffer Memory Read Timing

SWITCHING TEST CIRCUIT

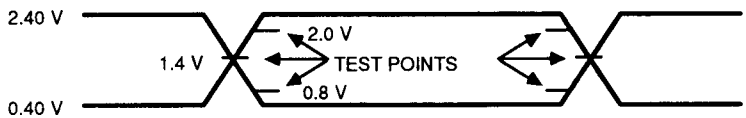


Note: C_L = 100 pF for pins NP_{0-15}, $\overline{READY}$, $\overline{MINTR}$, $\overline{NMINTR}$, D_{0-31}, and DP_{0-3}. C_L = 50 pF for all other output pins.

09730-038B

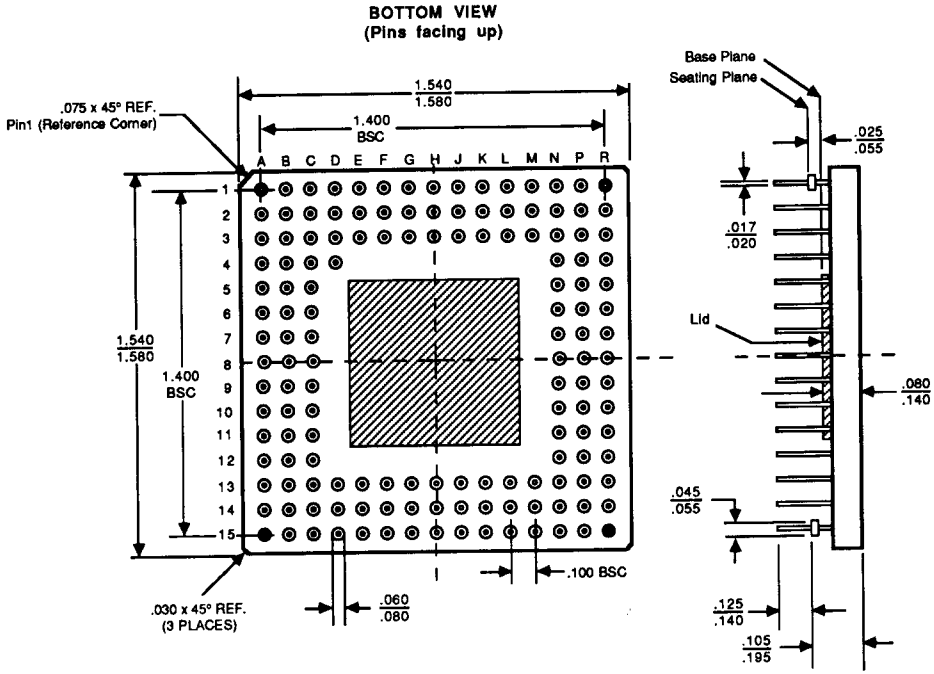
Standard Test Load

SWITCHING TEST WAVEFORM



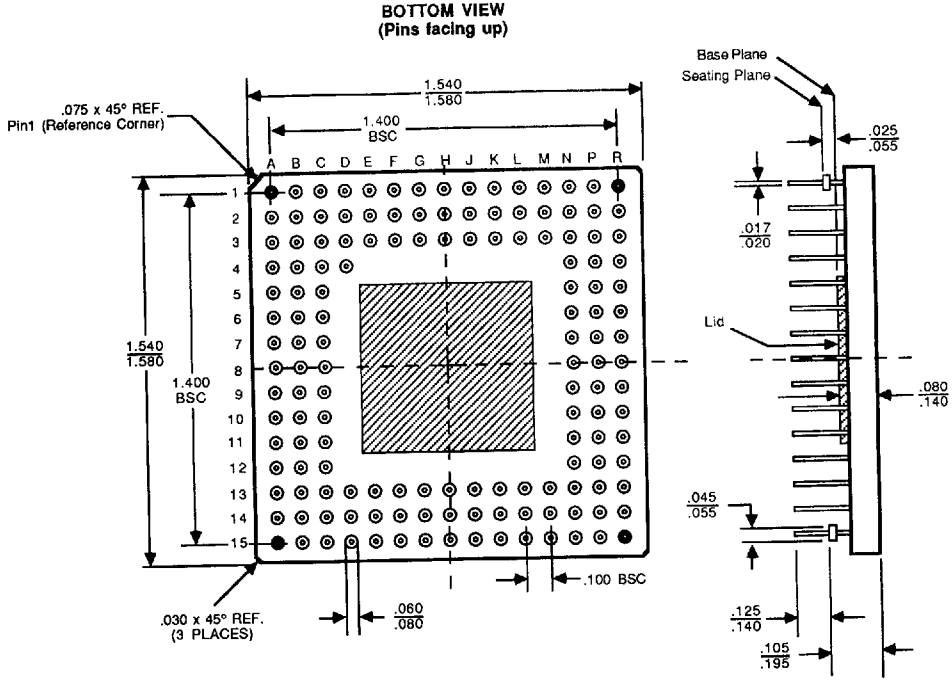
09730-039B

Input/Output Waveform


BOTTOM VIEW
(Pins facing up)
.075 x 45° REF.
Pin1 (Reference Corner)
1.540
1.580
1.400
BSC
A B C D E F G H J K L M N P R
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
1.540
1.580
1.400
BSC
.030 x 45° REF.
(3 PLACES)
.060
.080
.100 BSC
Base Plane
Seating Plane
.025
.055
.017
.020
Lid
.080
.140
.045
.055
.125
.140
.105
.195

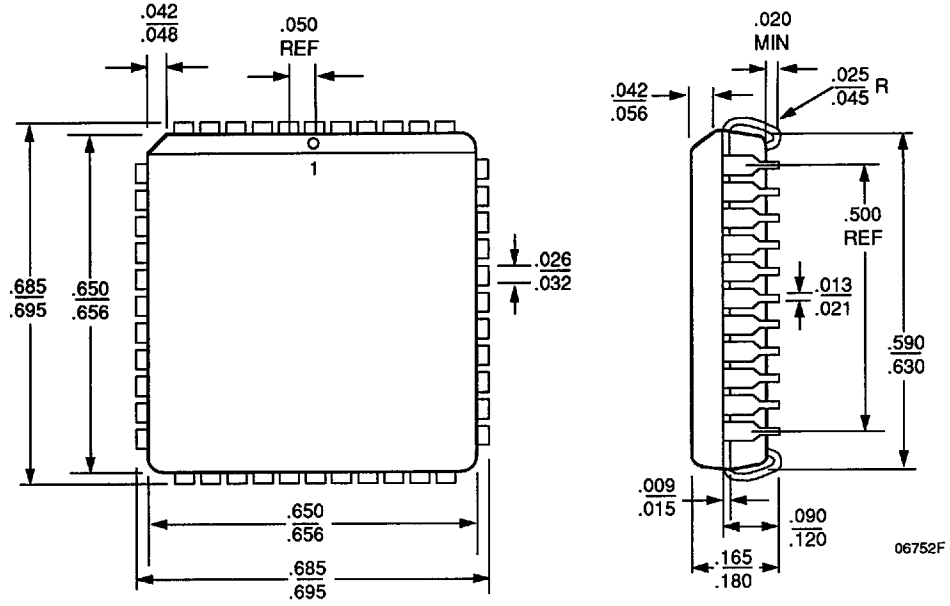
T-90-20

CGX145
145-Lead Pin Grid Array without Heat Sink

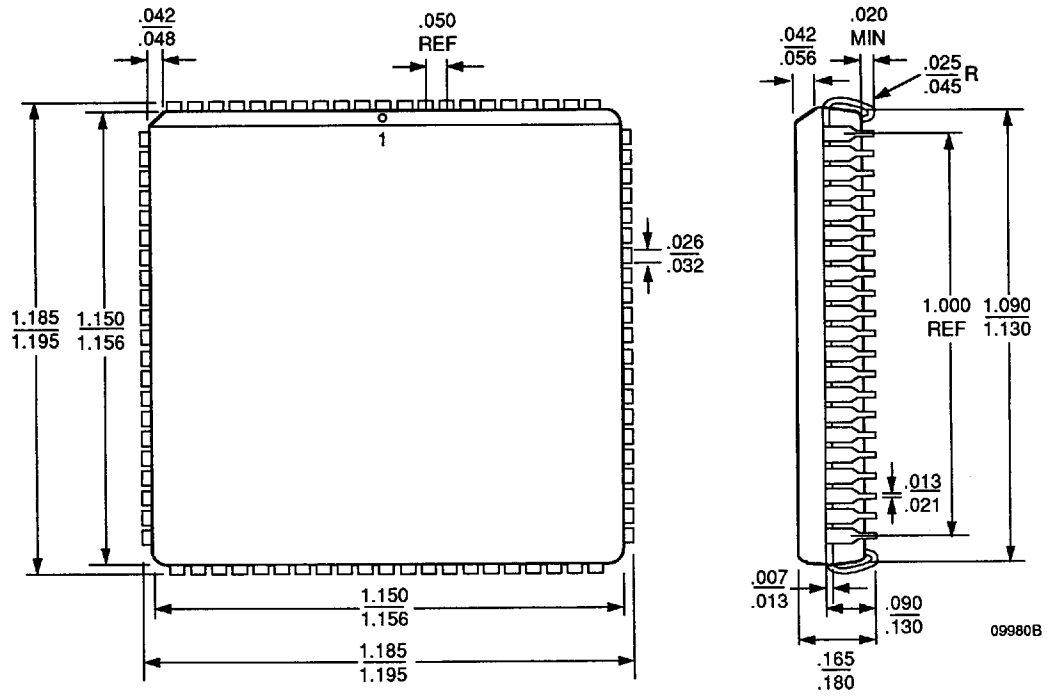


PID # 09691B

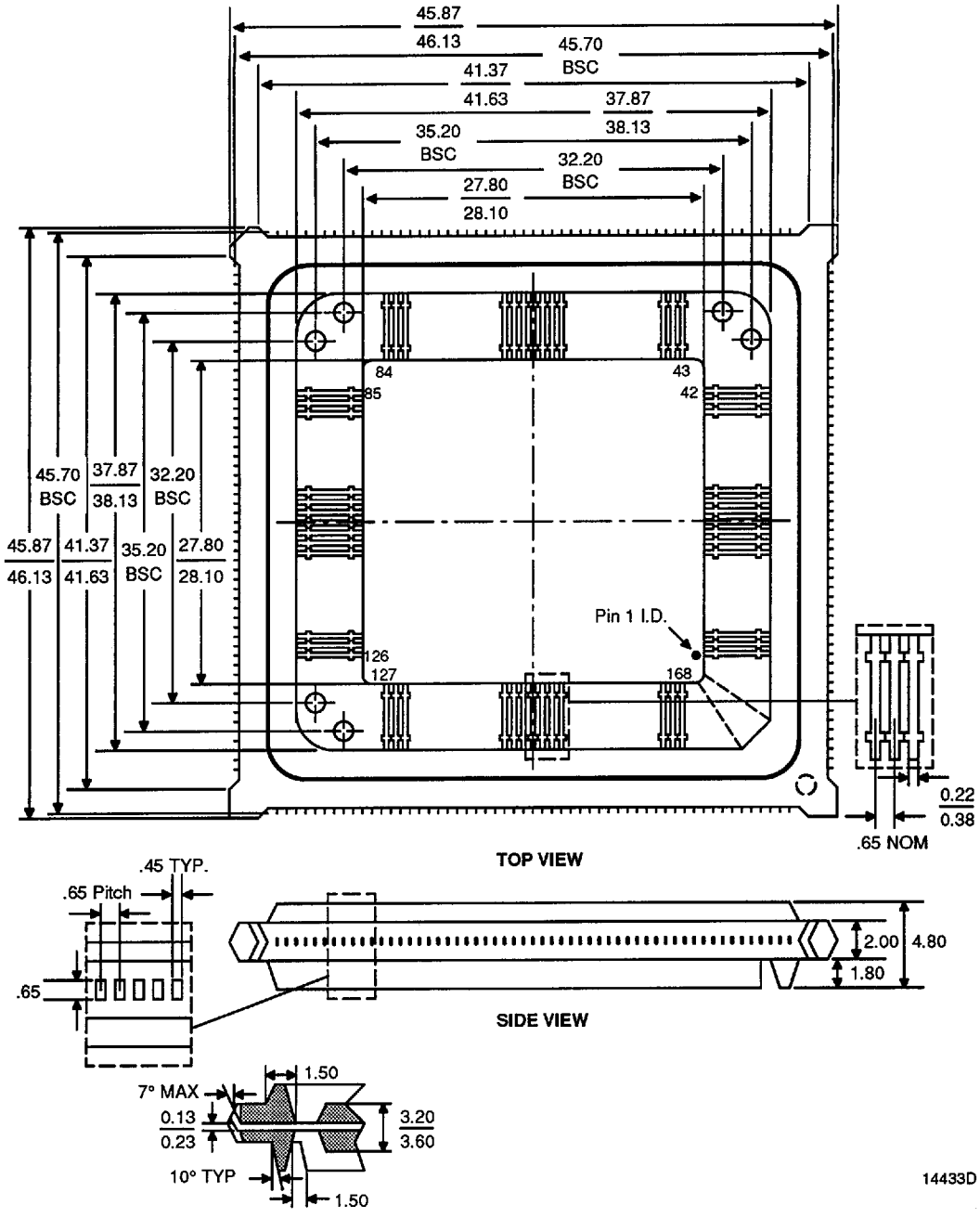
PL 044
44-Pin Plastic Leaded Chip Carrier



PL 084
84-Pin Plastic Leaded Chip Carrier

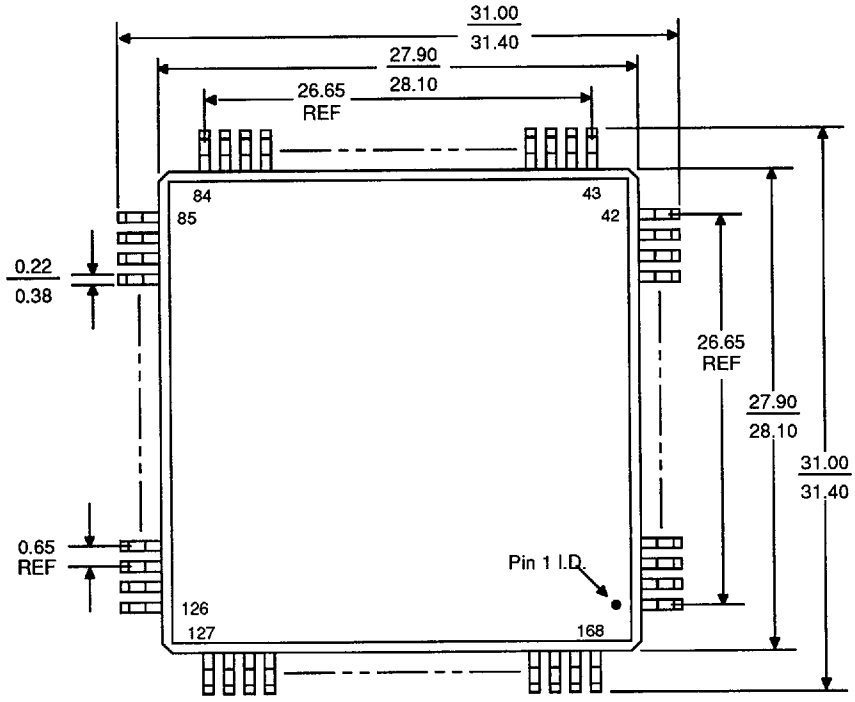


PQR168**
168-Pin Plastic Quad Flat Pack (Tape Pak)

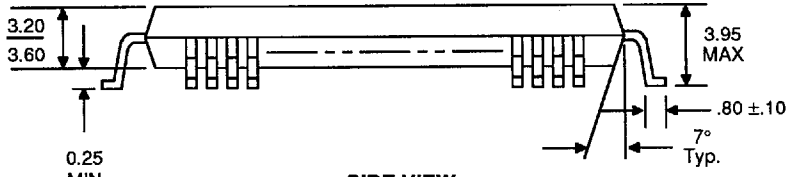


**Measured in Millimeters

PQJ168**
168-Pin Plastic Quad Flat Pack (Trimmed and Formed)



TOP VIEW



SIDE VIEW

**Measured in Millimeters