Am79C965A

PCnet™-32 Single-Chip 32-Bit Ethernet Controller

DISTINCTIVE CHARACTERISTICS

- **Single-chip Ethernet controller for 486 and Video Electronics Standards Association (VESA) local buses**
- **Supports ISO 8802-3 (IEEE/ANSI 802.3) and Ethernet standards**
- **Direct interface to the 486 local bus or VESA VL-Bus**
- **Enhanced burst mode with support for Am486™ burst read/write operations**
- **Software-compatible with AMD's Am7990 LANCE, Am79C90 C-LANCE, Am79C960 PCnet-ISA, Am79C961 PCnet-ISA+, Am79C961A PCnet-ISA II, Am79C970A PCnet-PCI II, and Am79C900 ILACC™ register and descriptor architecture**
- **Compatible with Am2100/Am1500T and Novell NE2100/NE1500 driver software**
- **High-performance Bus Master architecture with integrated DMA buffer management unit for low CPU and bus utilization**
- **Built-in byte-swap logic supports both big and little endian byte alignment**
- **Microwire EEPROM interface supports jumperless design**
- **Single +5 V power supply operation**
- **Low-power, CMOS design with sleep modes allows reduced power consumption for critical battery-powered applications and Green PCs**
- **Look-Ahead Packet Processing (LAPP) allows protocol analysis to begin before end of receive frame**
- **Integrated Manchester encoder/decoder**
- **Individual 136-byte transmit and 128-byte receive FIFOs provide frame buffering for increased system latency tolerance and support the following features:**
  - —Automatic retransmission with no FIFO reload
  - —Automatic receive stripping and transmit padding (individually programmable)
  - —Automatic runt packet rejection
  - —Automatic deletion of received collision frames
- **JTAG Boundary Scan (IEEE 1149.1) test access port interface for board-level production test**
- **Provides integrated attachment unit interface (AUI) and 10BASE-T transceiver with automatic port selection**
- **Automatic twisted-pair receive polarity detection and automatic correction of the receive polarity**
- **Optional byte padding to long-word boundary on receive**
- **Dynamic transmit FCS generation programmable on a frame-by-frame basis**
- **Internal/external loopback capabilities**
- **Supports the following types of network interfaces:**
  - —AUI to external 10BASE-2, 10BASE-5, 10BASE-T or 10BASE-F MAU
  - —Internal 10BASE-T transceiver with Smart Squelch to twisted-pair medium
- **Supports LANCE/C-LANCE/PCnet-ISA general purpose serial interface (GPSI)**
- **160-pin PQFP package**

GENERAL DESCRIPTION

The PCnet-32 single-chip 32-bit Ethernet controller is a highly integrated Ethernet system solution designed to address high-performance system application requirements. It is a flexible bus-mastering device that can be used in any networking application, including network-ready PCs, printers, fax modems, and bridge/router designs. The bus-master architecture provides high data throughput in the system and low CPU and bus utilization. The PCnet-32 controller is fabricated with AMD's advanced low-power CMOS process to provide low operating and standby current for power-sensitive applications.

Publication# **18219** Rev: **D** Amendment/**0**
Issue Date: **August 2000**

The PCnet-32 controller is a complete Ethernet node integrated into a single VLSI device. It contains a bus interface unit, a DMA buffer management unit, an ISO 8802-3 (IEEE/ANSI 802.3)-defined media access control (MAC) function, individual 136-byte transmit and 128-byte receive FIFOs, an ISO 8802-3 (IEEE/ANSI 802.3)-defined attachment unit interface (AUI) and twisted-pair transceiver medium attachment unit (10BASE-T MAU), and a microwire EEPROM interface. The PCnet-32 controller is also register-compatible with the LANCE (Am7990) Ethernet controller, the C-LANCE (Am79C90) Ethernet controller, the ILACC (Am79C900) Ethernet controller, and all Ethernet controllers in the PCnet family, including the PCnet-ISA controller (Am79C960), the PCnet-ISA+ controller (Am79C961), PCnet-ISA II (Am79C961A), and the PCnet-PCI II controller (Am79C970A). The buffer management unit supports the LANCE, ILACC (Am79C900), and PCnet descriptor software models. The PCnet-32 controller is software-compatible with Novell NE2100 and NE1500 Ethernet adapter card architectures. In addition, a Sleep function has been incorporated to provide low standby current, which is essential for notebooks and Green PCs.

The 32-bit demultiplexed bus interface unit provides a direct interface to the VESA VL-Bus and 486 series microprocessors, simplifying the design of an Ethernet node in a PC system. With its built-in support for both little and big endian byte alignment, this controller also addresses proprietary non-PC applications.

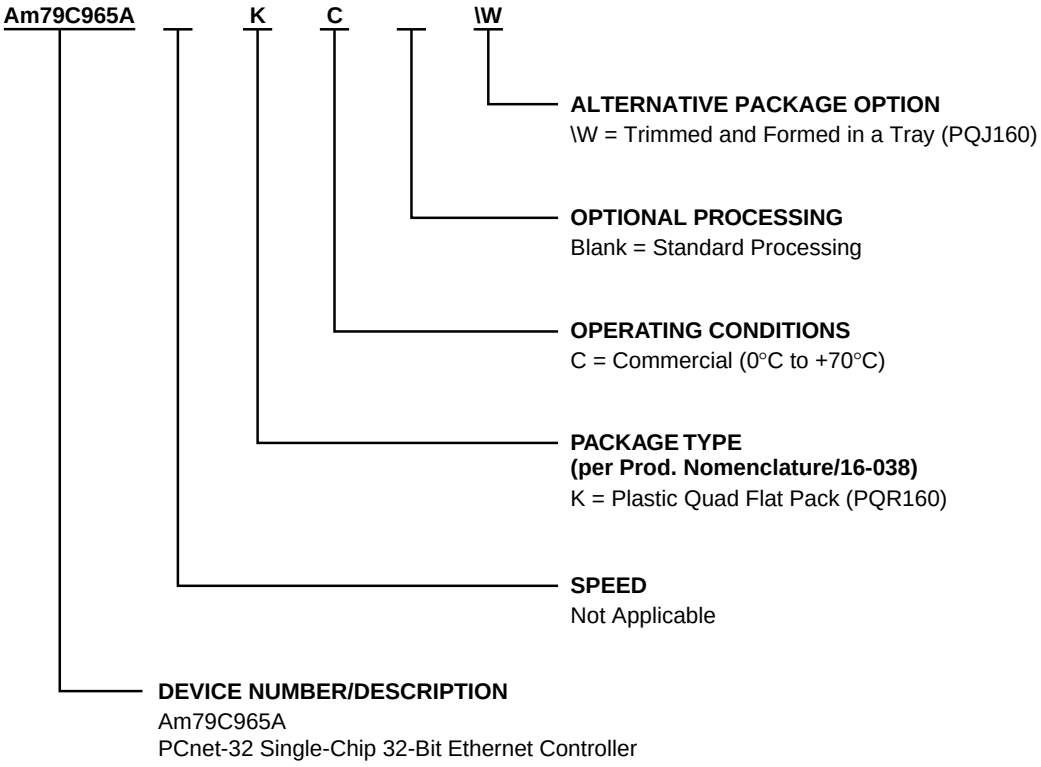
Key PCnet-32 configuration parameters, including the unique IEEE physical address, can be read from an external non-volatile memory (serial EEPROM) immediately following system reset. In addition, the I/O location at which the internal registers are accessed may be stored in the EEPROM, allowing the software model of the device to be located appropriately in system I/O space during system initialization.

The controller has the capability to select automatically either the AUI port or the twisted-pair transceiver. Only one interface is active at any one time. The individual transmit and receive FIFOs reduce system overhead, providing sufficient latency during frame transmission and reception, and minimizing intervention during normal network error recovery. The integrated Manchester encoder/decoder (MENDEC) eliminates the need for an external serial interface adapter (SIA) in the node system. The built-in general purpose serial interface (GPSI) allows the MENDEC to be bypassed. In addition, the device provides programmable on-chip LED drivers for transmit, receive, collision, receive polarity, link integrity, or jabber status. The PCnet-32 controller also provides an external address detection interface (EADI) to allow external hardware address filtering in inter-networking applications.

ORDERING INFORMATION

Standard Products:

AMD standard products are available in several packages and operating ranges. The order number (valid combination) is formed by a combination of the elements below.



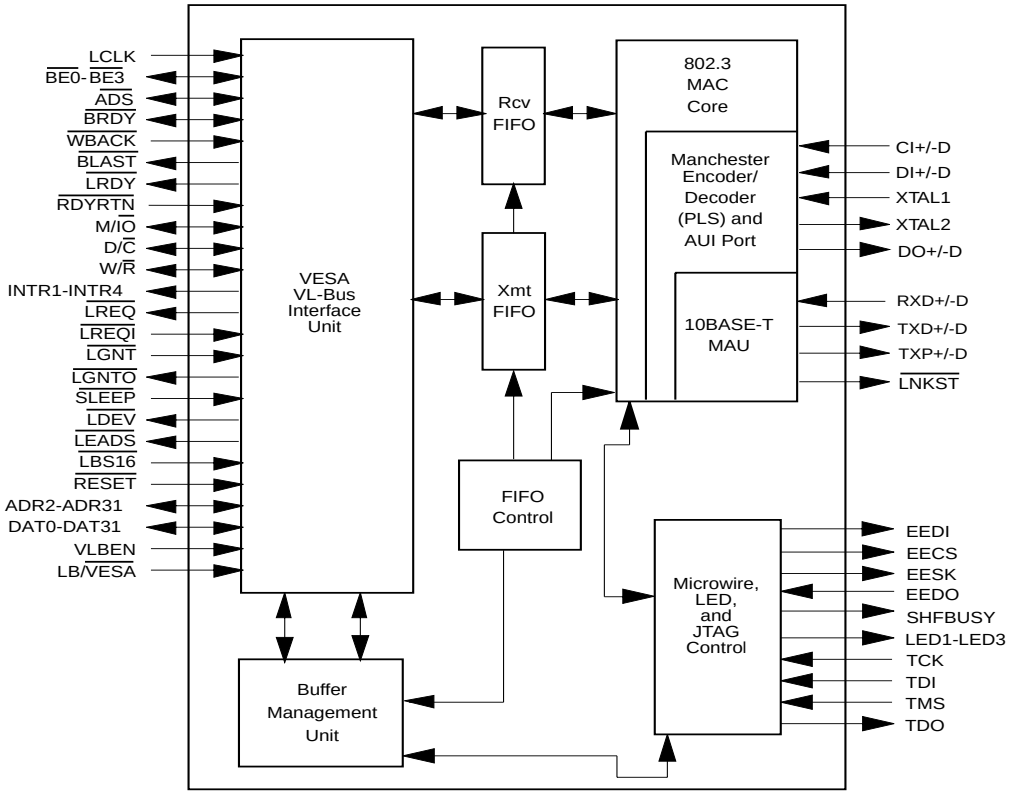
Valid Combinations	
Am79C965A	KC, KC\W

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 and to check on newly released combinations.

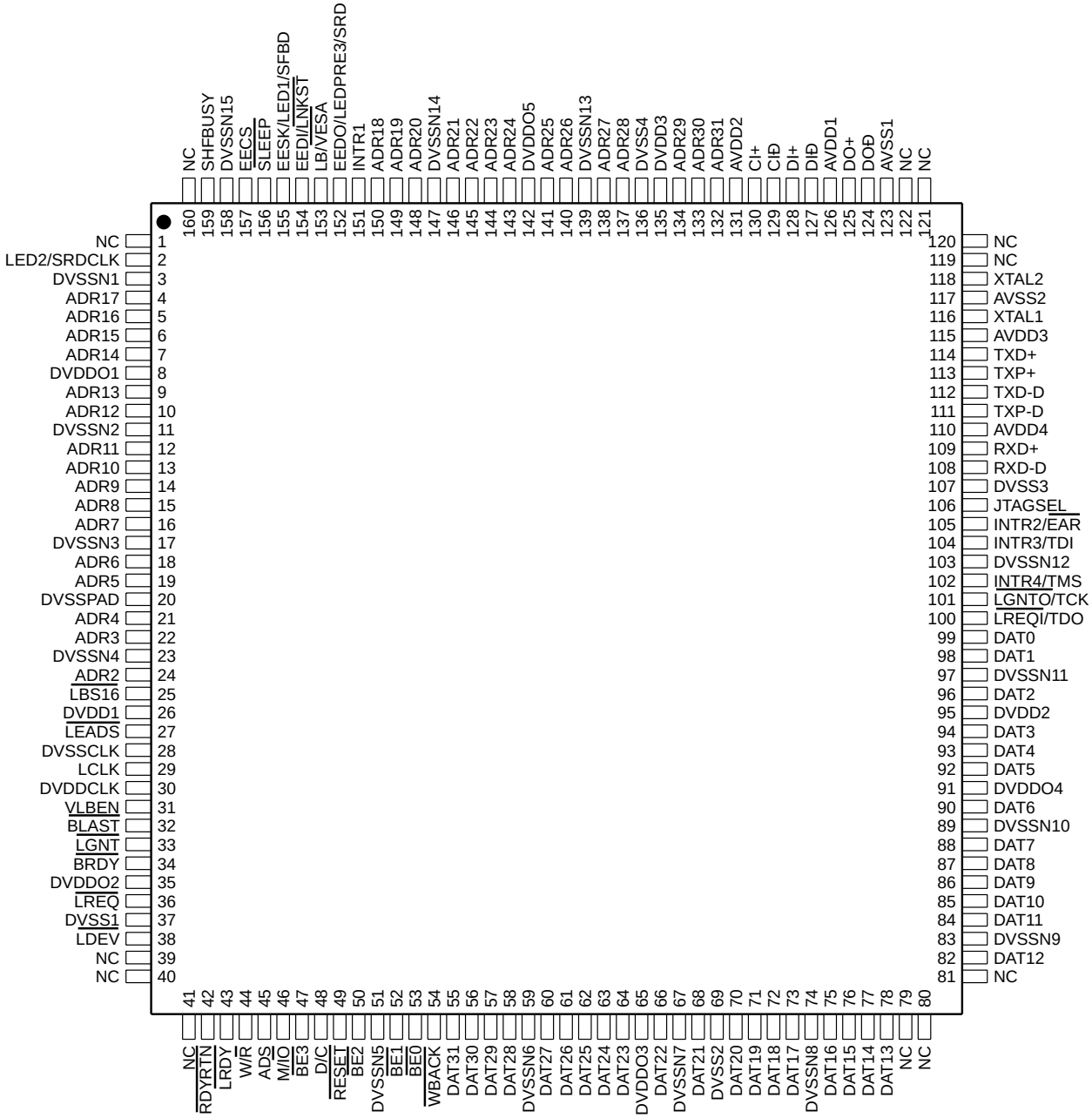
TABLE OF CONTENTS

BLOCK DIAGRAM: VESA VL-BUS MODE



18219-1

CONNECTION DIAGRAM: VESA VL-BUS MODE



18219-3

PIN DESIGNATIONS: VESA VL-BUS MODE

Listed by Pin Number

Pin No.	Name	Pin No.	Name	Pin No.	Name	Pin No.	Name
1	NC	41	NC	81	NC	121	NC
2	LED2/SRDCLK	42	$\overline{\text{RDYRTN}}$	82	DAT12	122	NC
3	DVSSN1	43	LRDY	83	DVSSN9	123	AVSS1
4	ADR17	44	W/R	84	DAT11	124	DO–
5	ADR16	45	ADS	85	DAT10	125	DO+
6	ADR15	46	M/IO	86	DAT9	126	AVDD1
7	ADR14	47	BE3	87	DAT8	127	DI–
8	DVDDO1	48	D/$\overline{\text{C}}$	88	DAT7	128	DI+
9	ADR13	49	$\overline{\text{RESET}}$	89	DVSSN10	129	CI–
10	ADR12	50	$\overline{\text{BE2}}$	90	DAT6	130	CI+
11	DVSSN2	51	DVSSN5	91	DVDDO4	131	AVDD2
12	ADR11	52	$\overline{\text{BE1}}$	92	DAT5	132	ADR31
13	ADR10	53	$\overline{\text{BE0}}$	93	DAT4	133	ADR30
14	ADR9	54	$\overline{\text{WBACK}}$	94	DAT3	134	ADR29
15	ADR8	55	DAT31	95	DVDD2	135	DVDD3
16	ADR7	56	DAT30	96	DAT2	136	DVSS4
17	DVSSN3	57	DAT29	97	DVSSN11	137	ADR28
18	ADR6	58	DAT28	98	DAT1	138	ADR27
19	ADR5	59	DVSSN6	99	DAT0	139	DVSSN13
20	DVSSPAD	60	DAT27	100	$\overline{\text{LREQI}}$/TDO	140	ADR26
21	ADR4	61	DAT26	101	$\overline{\text{LGNTO}}$/TCK	141	ADR25
22	ADR3	62	DAT25	102	INTR4/TMS	142	DVDDO5
23	DVSSN4	63	DAT24	103	DVSSN12	143	ADR24
24	ADR2	64	DAT23	104	INTR3/TDI	144	ADR23
25	$\overline{\text{LBS16}}$	65	DVDDO3	105	INTR2/$\overline{\text{EAR}}$	145	ADR22
26	DVDD1	66	DAT22	106	JTAGSEL	146	ADR21
27	$\overline{\text{LEADS}}$	67	DVSSN7	107	DVSS3	147	DVSSN14
28	DVSSCLK	68	DAT21	108	RXD–	148	ADR20
29	LCLK	69	DVSS2	109	RXD+	149	ADR19
30	DVDDCLK	70	DAT20	110	AVDD4	150	ADR18
31	VLBEN	71	DAT19	111	TXP–	151	INTR1
32	$\overline{\text{BLAST}}$	72	DAT18	112	TXD–	152	EEDO/LEDPRE3/SRD
33	$\overline{\text{LGNT}}$	73	DAT17	113	TXP+	153	LB/$\overline{\text{VESA}}$
34	$\overline{\text{BRDY}}$	74	DVSSN8	114	TXD+	154	EEDI/$\overline{\text{LNKST}}$
35	DVDDO2	75	DAT16	115	AVDD3	155	EESK/LED1/SFBD
36	$\overline{\text{LREQ}}$	76	DAT15	116	XTAL1	156	$\overline{\text{SLEEP}}$
37	DVSS1	77	DAT14	117	AVSS2	157	EECS
38	$\overline{\text{LDEV}}$	78	DAT13	118	XTAL2	158	DVSSN15
39	NC	79	NC	119	NC	159	SHFBUSY
40	NC	80	NC	120	NC	160	NC

PIN DESIGNATIONS: VESA VL-BUS MODE

Listed by Pin Name

Name	Pin No.	Name	Pin No.	Name	Pin No.	Name	Pin No.
ADR2	24	$\overline{BE3}$	47	DO+	125	JTAGSEL	106
ADR3	22	$\overline{BLAST}$	32	DO–	124	LB/$\overline{VESA}$	153
ADR4	21	$\overline{BRDY}$	34	DVDD1	26	$\overline{LBS16}$	25
ADR5	19	CI+	130	DVDD2	95	LCLK	29
ADR6	18	CI–	129	DVDD3	135	$\overline{LDEV}$	38
ADR7	16	D/$\overline{C}$	48	DVDDCLK	30	$\overline{LEADS}$	27
ADR8	15	DAT0	99	DVDDO1	8	LED2/SRDCLK	2
ADR9	14	DAT1	98	DVDDO2	35	$\overline{LGNT}$	33
ADR10	13	DAT2	96	DVDDO3	65	$\overline{LGNTO}$/TCK	101
ADR11	12	DAT3	94	DVDDO4	91	$\overline{LRDY}$	43
ADR12	10	DAT4	93	DVDDO5	142	$\overline{LREQ}$	36
ADR13	9	DAT5	92	DVSS1	37	$\overline{LREQI}$/TDO	100
ADR14	7	DAT6	90	DVSS2	69	M/$\overline{IO}$	46
ADR15	6	DAT7	88	DVSS3	107	NC	1
ADR16	5	DAT8	87	DVSS4	136	NC	39
ADR17	4	DAT9	86	DVSSCLK	28	NC	40
ADR18	150	DAT10	85	DVSSN1	3	NC	41
ADR19	149	DAT11	84	DVSSN2	11	NC	79
ADR20	148	DAT12	82	DVSSN3	17	NC	80
ADR21	146	DAT13	78	DVSSN4	23	NC	81
ADR22	145	DAT14	77	DVSSN5	51	NC	119
ADR23	144	DAT15	76	DVSSN6	59	NC	120
ADR24	143	DAT16	75	DVSSN7	67	NC	121
ADR25	141	DAT17	73	DVSSN8	74	NC	122
ADR26	140	DAT18	72	DVSSN9	83	NC	160
ADR27	138	DAT19	71	DVSSN10	89	$\overline{RDYRTN}$	42
ADR28	137	DAT20	70	DVSSN11	97	$\overline{RESET}$	49
ADR29	134	DAT21	68	DVSSN12	103	RXD+	109
ADR30	133	DAT22	66	DVSSN13	139	RXD–	108
ADR31	132	DAT23	64	DVSSN14	147	SHFBUSY	159
$\overline{ADS}$	45	DAT24	63	DVSSN15	158	$\overline{SLEEP}$	156
AVDD1	126	DAT25	62	DVSSPAD	20	TXD+	114
AVDD2	131	DAT26	61	EECS	157	TXD–	112
AVDD3	115	DAT27	60	EEDI/$\overline{LNKST}$	154	TXP+	113
AVDD4	110	DAT28	58	EEDO/LEDPRE3/SRD	152	TXP–	111
AVSS1	123	DAT29	57	EESK/LED1/SFBD	155	VLBEN	31
AVSS2	117	DAT30	56	INTR1	151	W/$\overline{R}$	44
$\overline{BE0}$	53	DAT31	55	INTR2/$\overline{EAR}$	105	$\overline{WBACK}$	54
$\overline{BE1}$	52	DI+	128	INTR3/TDI	104	XTAL1	116
$\overline{BE2}$	50	DI–	127	INTR4/TMS	102	XTAL2	118

PIN DESIGNATIONS: VESA VL-BUS MODE

Listed by Group

Pin Name	Pin Function	Type	Driver	No. of Pins
VESA VL-Bus Interface				
ADR2–ADR31	Address Bus	IO	TS	30
$\overline{\text{ADS}}$	Address Status	I/O	TS	1
$\overline{\text{BE0}}$–$\overline{\text{BE3}}$	Byte Enable	I/O	TS	4
$\overline{\text{BLAST}}$	Burst Last	O	TS	1
$\overline{\text{BRDY}}$	Burst Ready	I/O	TS	1
D/$\overline{\text{C}}$	Data/Control Select	I/O	TS	1
DAT0–DAT31	Data Bus	I/O	TS	32
INTR1	Interrupt Number 1	O	TS	1
INTR2	Interrupt Number 2	I/O	TS	1
INTR3	Interrupt Number 3	I/O	TS	1
INTR4	Interrupt Number 4	I/O	TS	1
JTAGSEL	JTAG Select	I		1
LB/$\overline{\text{VESA}}$	Local Bus/VESA VL-Bus Select pin	I		1
$\overline{\text{LBS16}}$	Local Bus Size 16	I		1
LCLK	Local Clock	I		1
$\overline{\text{LDEV}}$	Local Device	O	O4	1
$\overline{\text{LEADS}}$	Local External Address Strobe	O	TS	1
$\overline{\text{LGNT}}$	Local Bus Grant	I		1
$\overline{\text{LGNTO}}$	Local Grant Out	I/O	TS	1
$\overline{\text{LRDY}}$	Local Ready	O	TS	1
$\overline{\text{LREQ}}$	Local Bus Request	O	O8	1
$\overline{\text{LREQI}}$	Local Bus Request In	I/O	TS	1
M/$\overline{\text{IO}}$	Memory/I/O Select	I/O	TS	1
$\overline{\text{RDYRTN}}$	Ready Return	I		1
$\overline{\text{RESET}}$	Reset	I		1
VLBEN	Burst Enable	I		1
W/$\overline{\text{R}}$	Write/Read Select	I/O	TS	1
$\overline{\text{WBACK}}$	Write Back	I		1
Board Interface				
EECS	Microwire Serial EEPROM Chip Select	O	O8	1
EEDI/$\overline{\text{LNKST}}$	Microwire Serial EEPROM Data In/Link Status	O	LED	1
EEDO/LEDPRE3	Microwire Address EEPROM Data Out/LED3 predriver	I/O	LED	1
EESK/LED1	Microwire Serial EEPROM Clock/LED1	O	LED	1
LED2	LED Output Number 2	O	LED	1
SHFBUSY	Shift Busy (for external EEPROM-programmable logic)	O	O8	1
$\overline{\text{SLEEP}}$	Sleep Mode	I		1
XTAL1	Crystal Input	I		1
XTAL2	Crystal Output	O		1

PIN DESIGNATIONS: VESA VL-BUS MODE (continued)

Listed by Group

Pin Name	Pin Function	Type	Driver	No. of Pins
Attachment Unit Interface (AUI)				
CI+/CI–	AUI Collision Differential Pair	I		2
DI+/DI–	AUI Data In Differential Pair	I		2
DO+/DO–	AUI Data Out Differential Pair	O	DO	2
Twisted-Pair Transceiver Interface (10BASE-T)				
RXD+/RXD–	Receive Differential Pair	I		2
TXD+/TXD–	Transmit Differential Pair	O	TDO	2
TXP+/TXP–	Transmit Predistortion Differential Pair	O	TPO	2
LNKST/EEDI	Link Status/Microwire Serial EEPROM Data In	O	LED	1
IEEE 1149.1 Test Access Port Interface (JTAG)				
TCK	Test Clock	I/O	TS	1
TDI	Test Data In	I/O	TS	1
TDO	Test Data Out	I/O	TS	1
TMS	Test Mode Select	I/O	TS	1
External Address Detection Interface (EADI)				
$\overline{\text{EAR}}$	External Address Reject Low	I/O	TS	1
SRD	Serial Receive Data	I/O	TS	1
SRDCLK	Serial Receive Data Clock	I/O	TS	1
SFBD	Start Frame—Byte Delimiter	O	LED	1
Power Supplies				
AVDD	Analog Power	P		4
AVSS	Analog Ground	P		2
DVDD	Digital Power	P		3
DVDDCLK	Digital Power Clock	P		1
DVDDO	I/O Buffer Digital Power	P		5
DVSS	Digital Ground	P		4
DVSSCLK	Digital Ground Clock	P		1
DVSSN	I/O Buffer Digital Ground	P		15
DVSSPAD	Digital Ground Pad	P		1

PIN DESIGNATIONS: VESA VL-BUS MODE

Driver Type

Table 1. Output Drive Type

Name	Type	I_{OL} (mA)	I_{OH} (mA)	pF
TS	Tri-State	8	-0.4	50
O8	Totem Pole	8	-0.4	50
O4	Totem Pole	4	-0.4	50
OD	Open Drain	8		50
LED	LED	12	-0.4	50

Table 2. Pins with Pullup

Signal	Pullups
TDI	≥10 kΩ
TMS	≥10 kΩ
TCK	≥10 kΩ

PIN DESCRIPTION: VESA VL-BUS MODE

Configuration Pins

JTAGSEL

JTAG Function Select ***Input***

The value of this pin will asynchronously select between JTAG Mode and Multi-Interrupt Mode.

The value of this pin will asynchronously affect the function of the JTAG–INTR–Daisy chain arbitration pins, regardless of the state of the RESET pin and regardless of the state of the LCLK pin. If the value is a "1", then the PCnet-32 controller will be programmed for JTAG mode. If the value is a "0", then the PCnet-32 controller will be programmed for Multi-Interrupt Mode.

When programmed for JTAG mode, four pins of the PCnet-32 controller will be configured as a JTAG (IEEE 1149.1) Test Access Port. When programmed for Multi-Interrupt Mode, two of the JTAG pins will become interrupts and two JTAG pins will be used for daisy chain arbitration support. Table 3 below outlines the pin changes that will occur by programming the JTAGSEL pin.

Table 3. JTAG Pin Changes

Pin	JTAGSEL=1 JTAG Mode	JTAGSEL=0 Multi-Interrupt Mode
$\overline{\text{LGNT0}}$/TCK	TCK	$\overline{\text{LGNTO}}$
$\overline{\text{LGNT1}}$/TDO	TDO	$\overline{\text{LREQI}}$
$\overline{\text{LGNT2}}$/TDI	TDI	INTR3
$\overline{\text{LGNT3}}$/TMS	TMS	INTR4

The JTAGSEL pin may be tied directly to V_{DD} or V_{SS}. A series resistor may be used but is not necessary.

LB/$\overline{\text{VESA}}$

Local Bus/VESA VL-Bus Select **Input**

The value of this pin will asynchronously determine the operating mode of the PCnet-32 controller, regardless of the state of the RESET pin and regardless of the state of the LCLK pin. If the LB/$\overline{\text{VESA}}$ pin is a tied to V_{DD}, then the PCnet-32 controller will be programmed for Local Bus Mode. If the LB/$\overline{\text{VESA}}$ pin is tied to V_{SS}, then the PCnet-32 controller will be programmed for VESA-VL Bus Mode.

Note that the setting of LB/$\overline{\text{VESA}}$ determines the functionality of the following pins (names in parentheses are pins in 486 local bus mode): VLBEN ($\overline{\text{Am486}}$), RESET (RESET), $\overline{\text{LBS16}}$ (AHOLD), $\overline{\text{LREQ}}$ (HOLD), $\overline{\text{LGNT}}$ (HLDA), $\overline{\text{LREQI}}$ (HOLDI) and $\overline{\text{LGNTO}}$ (HLDAO).

VLBEN

Burst Enable ***Input***

This pin is used to determine whether or not bursting is supported by the PCnet-32 device in VESA VL-Bus mode. The VLBEN pin is sampled at every rising edge of LCLK while the RESET pin is asserted.

In VESA-VL mode (the LB/$\overline{\text{VESA}}$ pin is tied to V_{SS}), if the sampled value of VLBEN is low, then the BREADE and BWRITE bits in BCR18 will be forced low, and the PCnet-32 controller will never attempt to perform linear burst reads or writes. If the sampled value of VLBEN is high, linear burst accesses are permitted, consistent with the values programmed into BREADE and BWRITE.

Because of byte-duplication conventions within a 32-bit Am386 system, the PCnet-32 controller will always produce the correct bytes in the correct byte lanes in accordance with the Am386DX data sheet. This byte duplication will automatically occur, regardless of the operating mode selected by the LB/$\overline{\text{VESA}}$ pin.

The VLBEN pin may be tied directly to V_{DD} or V_{SS}. A series resistor may be used but is not necessary.

The VLBEN pin need only be valid when the RESET pin is active (regardless of the connection of the LB/$\overline{\text{VESA}}$ pin) and may be tied to ID(3) in a VESA VL-Bus version 1.0 system, or to the logical AND of ID(4), ID(3), ID(1), and ID(0) in a VESA-VL-Bus version 1.1 or 2.0 system.

Note: *This pin needs to be tied low when the LB/$\overline{\text{VESA}}$ pin has been tied to V_{DD}. See the pin description for the Am486 pin in the Local Bus Mode section.*

Configuration Pin Settings Summary

Table 4 shows the possible pin configurations that may be invoked with the PCnet-32 controller configuration pins.

Table 4. Configuration Pin Settings

LB/VESA	VLBEN	JTAGSEL	Mode Selected
0	X*	0	VL Bus mode with 4 interrupts and daisy chain arbitration
0	X*	1	VL Bus mode with 2 interrupts and JTAG
1	0	0	VL Bus mode with 4 interrupts and daisy chain arbitration
1	0	1	VL Bus mode with 2 interrupts and JTAG
1	1	X	Reserved

**X = Don't care*

Pin Connections to V_{DD} or V_{SS}

Several pins may be connected to V_{DD} or V_{SS} for various application options. Some pins are required to be connected to V_{DD} or V_{SS} in order to set the controller into a particular mode of operation, while other pins might be connected to V_{DD} or V_{SS} if that pin's function is not implemented in a specific application. Table 5 shows which pins require a connection to V_{DD} or V_{SS}, and which pins may optionally be connected to V_{DD} or V_{SS} because the application does not support that pin's function. The table also shows whether or not the connections need to be resistive.

VESA VL-Bus Interface

ADR2–ADR31

Address Bus ***Input/Output***

Address information which is stable during a bus operation, regardless of the source. When the PCnet-32 controller is Current Master, A2–A31 will be driven. When the PCnet-32 controller is not Current Master, the A2–A31 lines are continuously monitored to determine if an address match exists for I/O slave transfers.

$\overline{\text{ADS}}$

Address Status ***Input/Output***

When driven LOW, this signal indicates that a valid bus cycle definition and address are available on the M/$\overline{\text{IO}}$, D/$\overline{\text{C}}$, W/$\overline{\text{R}}$ and A2–A31 pins of the local bus interface. At that time, the PCnet-32 controller will examine the combination of M/$\overline{\text{IO}}$, D/$\overline{\text{C}}$, W/$\overline{\text{R}}$, and the A2–A31 pins to determine if the current access is directed toward the PCnet-32 controller.

$\overline{\text{ADS}}$ will be driven LOW when the PCnet-32 controller performs a bus master access on the local bus.

$\overline{\text{BE0}}$–$\overline{\text{BE3}}$

Byte Enable ***Input/Output***

These signals indicate which bytes on the data bus are active during read and write cycles. When $\overline{\text{BE3}}$ is active, the byte on DAT31–DAT24 is valid. $\overline{\text{BE2}}$–$\overline{\text{BE0}}$ active indicate valid data on pins DAT23–DAT16, DAT15– DAT8, DAT7–DAT0, respectively. The byte enable signals are outputs for bus master and inputs for bus slave operations.

$\overline{\text{BLAST}}$

Burst Last ***Output***

When the $\overline{\text{BLAST}}$ signal is asserted, then the next time that $\overline{\text{BRDY}}$ or $\overline{\text{RDYRTN}}$ is asserted, the burst cycle is complete.

$\overline{\text{BRDY}}$

Burst Ready ***Input/Output***

$\overline{\text{BRDY}}$ functions as an input to the PCnet-32 controller during bus master cycles. When $\overline{\text{BRDY}}$ is asserted during a master cycle, it indicates to the PCnet-32 controller that the target device is accepting burst transfers. It also serves the same function as $\overline{\text{RDYRTN}}$ does for non-burst accesses. That is, it indicates that the target device has accepted the data on a master write cycle, or that the target device has presented valid data onto the bus during master read cycles.

Table 5. Pin Connections to Power/Ground

Pin Name	Pin No	Supply Strapping	Resistive Connection to Supply	Recommended Resistor Size
LED2/SRDCLK	2	Required	Required	324 Ω in series with LED, or 10 KΩ without LED
$\overline{\text{LBS16}}$	25	Optional	Required	10 KΩ
VLBEN	31	Required	Optional	NA
$\overline{\text{WBACK}}$	54	Optional	Required	10 KΩ
$\overline{\text{LREQI}}$/TDO	100	Optional	Required	10 KΩ
JTAGSEL	106	Required	Optional	NA
EEDO/LEDPRE3/SRD	152	Optional	Required	10 KΩ
LB/$\overline{\text{VESA}}$	153	Required	Optional	NA
EEDI/$\overline{\text{LNKST}}$	154	Optional	Required	324 Ω in series with LED, or 10 KΩ without LED
EESK/LED1SFBD	155	Required	Required	324 Ω in series with LED, or 10 KΩ without LED
$\overline{\text{SLEEP}}$	156	Optional	Required	10 KΩ
All Other Pins	—	Optional	Required	10 KΩ

If $\overline{\text{BRDY}}$ and $\overline{\text{RDYRTN}}$ are sampled active in the same cycle, then $\overline{\text{RDYRTN}}$ takes precedence, causing the next transfer cycle to begin with a T1 cycle.

$\overline{\text{BRDY}}$ functions as an output during PCnet-32 controller slave cycles and is always driven inactive (HIGH).

$\overline{\text{BRDY}}$ is floated if the PCnet-32 controller is not being accessed as the current slave device on the local bus.

D/$\overline{\text{C}}$

Data/Control Select ***Input/Output***

During slave accesses to the PCnet-32 controller, the D/$\overline{\text{C}}$ pin, along with M/$\overline{\text{IO}}$ and W/$\overline{\text{R}}$, indicates the type of cycle that is being performed. PCnet-32 controller will only respond to local bus accesses in which D/$\overline{\text{C}}$ is driven HIGH by the local bus master.

During PCnet-32 controller bus master accesses, the D/$\overline{\text{C}}$ pin is an output and will always be driven HIGH.

D/$\overline{\text{C}}$ is floated if the PCnet-32 controller is not the current master on the local bus.

DAT0–DAT31

Data Bus ***Input/Output***

Used to transfer data to and from the PCnet-32 controller to system resources via the local bus. DAT31–DAT0 are driven by the PCnet-32 controller when performing bus master writes and slave read operations. Data on DAT31–DAT0 is latched by the PCnet-32 controller when performing bus master reads and slave write operations.

The PCnet-32 controller will always follow Am386DX byte lane conventions. This means that for word and byte accesses in which PCnet-32 controller drives the data bus (i.e. master write operations and slave read operations), the PCnet-32 controller will produce duplicates of the active bytes on the unused half of the 32-bit data bus. Table 6 illustrates the cases in which duplicate bytes are created.

Table 6. Bye Duplication on Data Bus

$\overline{\text{BE3}}$-$\overline{\text{BE0}}$	DAT [31:24]	DAT [23:16]	DAT [15:8]	DAT [7:0]
1110	Undef	Undef	Undef	A
1101	Undef	Undef	A	Undef
1011	Undef	A	Undef	Copy A
0111	A	Undef	Copy A	Undef
1100	Undef	Undef	B	A
1001	Undef	C	B	Undef
0011*	D	C	Copy D	Copy C
1000	Undef	C	B	A
0001	D	C	B	Undef
0000	D	C	B	A

***Note:** Byte duplication does not apply during an $\overline{\text{LBS16}}$ access, see Table 8.*

INTR1–INTR4

Interrupt Request ***Output***

An attention signal which indicates that one or more of the following status flags is set: BABL, MISS, MERR,RINT, IDON, MFCO, RCVCCO, TXSTRT, or JAB. Each of these status flags has a mask bit which allows for suppression of INTR assertion. These flags have the meaning shown in Table 7.

Table 7. Status Flags

BABL	Babble (CSR0, bit 14)
MISS	Missed Frame (CSR0, bit 12)
MERR	Memory Error (CSR0, bit 11)
RINT	Receive Interrupt (CSR0, bit 10)
IDON	Initialization Done (CSR0, bit 8)
MFCO	Missed Packet Count Overflow (CSR4, bit 9)
RCVCCO	Receive Collision Count Overflow (CSR4, bit 5)
TXSTRT	Transmit Start (CSR4, bit 3)
JAB	Jabber (CSR4, bit 1)

Note that there are four possible interrupt pins, depending upon the mode that has been selected with the JTAGSEL pin. Only one interrupt pin may be used at one time. The active interrupt pin is selected by programming the interrupt select register (BCR21). The default setting of BCR121will select interrupt INTR1 as the active interrupt. Note that BCR21 is EEPROM-programmable. Inactive interrupt pins are floated. The polarity of the interrupt signal is determined by the INTLEVEL bit of BCR2. The interrupt pins may be programmed for level-sensitive or edge-sensitive operation. PCnet-32 controller interrupt pins will be floated at H_RESET and will remain floated until either the EEPROM has been successfully read, or, following an EEPROM read failure, a Software Relocatable Mode sequence has been successfully executed.

$\overline{\text{LBS16}}$

Local Bus Size 16 ***Input***

$\overline{\text{BS16}}$ is sampled during PCnet-32 controller bus mastering activity to determine if the target device on the VL-Bus is 32 or 16 bits in width. If the $\overline{\text{LBS16}}$ signal is sampled active at least one clock period before the assertion of $\overline{\text{LRDY}}$ during a PCnet-32 controller bus master transfer, then the PCnet-32 controller will convert a single 32-bit transfer into two 16-bit transfers. Not all 32-bit transfers need to be split into two 16-bit transfers. Table 8 shows the sequence of transfers that will be executed for each possible 32-bit bus transfer that encounters a proper assertion of the $\overline{\text{LBS16}}$ signal.

Table 8. Data Transfer Sequence from 32-Bit Wide to 16-Bit Wide

Current Access			Next with $\overline{\text{LBS16}}$				
$\overline{\text{BE3}}$	$\overline{\text{BE2}}$	$\overline{\text{BE1}}$	$\overline{\text{BE0}}$	$\overline{\text{BE3}}$	$\overline{\text{BE2}}$	$\overline{\text{BE1}}$	$\overline{\text{BE0}}$
1	1	1	0	NR			
1	1	0	0	NR			
1	0	0	0	1	0	1	1
0	0	0	0	0	0	1	1
1	1	0	1	NR			
1	0	0	1	1	0	1	1
0	0	0	1	0	0	1	1
1	0	1	1	NR			
0	0	1	1	NR			
0	1	1	1	NR			

NR = No second access Required for these cases.

During accesses in which PCnet-32 controller is acting the VL-Bus target device, the $\overline{\text{LBS16}}$ signal will not be driven. In this case, it is expected that the VL-Bus required pull-up device will bring the $\overline{\text{LBS16}}$ signal to an inactive level and the PCnet-32 controller will be seen by the VL-Bus master as a 32-bit peripheral.

LCLK

Local Clock ***Input***

LCLK is a 1x clock that follows the same phase as a 486-type CPU clock. LCLK is always driven by the system logic or the VL-Bus controller to all VL-Bus masters and targets. The rising edge of the clock signifies the change of CPU states, and hence, the change of PCnet-32 controller states.

LDEV

Local Device ***Output***

LDEV is driven by the PCnet-32 controller when it recognizes an access to PCnet-32 controller I/O space. Such recognition is dependent upon a valid sampled ADS strobe plus valid M/IO, D/C and ADR31–ADR5 values.

LEADS

Local External Address Strobe ***Output***

During VL-Bus master write and read accesses the LEADS pin will be asserted on every T1 cycle as is specified in the VESA VL-Bus specification, regardless the settings of the GCIC bit of BCR18 and the CLL bits BCR18.

LGNT

Local Bus Grant ***Input***

When LGNT is asserted and LREQ is being asserted by the PCnet-32 controller, the PCnet-32 controller assumes ownership of the VL bus.

Note that this pin changes polarity when Local Bus mode has been selected (see pin description of HLDA in 486 Local Bus Interface section).

LGNTO

Local Grant Out ***Output***

This signal is multiplexed with the TCK pin, and is available only when the Multi-Interrupt mode has been selected with the JTAGSEL pin.

An additional local bus master may daisy-chain its LGNT signal through the PCnet-32 controller LGNTO pin. The PCnet-32 controller will deliver a LGNTO signal the additional local bus master whenever the PCnet-32 controller receives a LGNT from the arbitration logic, but is not simultaneously requesting the bus internally. The second local bus master must connect its LREQ output to the LREQI input of the PCnet-32 controller in order to complete the local bus daisy-chain arbitration control.

When SLEEP is not asserted, daisy chain arbitration signals that pass through the PCnet-32 controller will experience a one-clock delay from input to output (i.e. LREQI to LREQ and LGNT to LGNTO).

While SLEEP is asserted (either in **snooze** mode or **coma** mode), if the PCnet-32 controller is configured for daisy chain (LREQI and LGNTO signals have been selected with the JTAGSEL pin), then the system arbitration signal LGNT will be passed directly to the daisy-chain signal LGNTO without experiencing a one-clock delay. However, some combinatorial delay will be introduced in this path.

Note that this pin changes polarity when Local Bus mode has been selected (see pin description of HLDAO 486 Local Bus Interface section).

LRDY

Local Ready ***Output***

LRDY functions as an output from the PCnet-32 controller during PCnet-32 controller slave cycles. During PCnet-32 controller slave read cycles, LRDY is asserted to indicate that valid data has been presented on the data bus. During PCnet-32 controller slave write cycles, LRDY is asserted to indicate that the data on the data bus has been internally latched. LRDY will be asserted low for one clock period when the PCnet-32 controller wishes to terminate the cycle. LRDY is then driven high for one-half of one clock period before being released.

LRDY is floated if the PCnet-32 controller is not the current slave on the local bus.

LREQ

Local Bus Request ***Output***

LREQ is used by the PCnet-32 controller to gain control of the VL-Bus and become the active VL Bus Master. LREQ is active low. Once asserted, LREQ remains active until LGNT has become active, independent of subsequent assertion of SLEEP or setting of the STOP bit or access to the S_RESET port (offset 14h).

Note that this pin changes polarity when Local Bus mode has been selected (see pin description of HOLD in 486 Local Bus Interface section).

LREQI

Local Bus Request In ***Input***

This signal is multiplexed with the TDO pin, and is available only when the Multi-Interrupt mode has been selected with the JTAGSEL pin.

An additional local bus master may daisy-chain its bus hold request signal through the PCnet-32 controller LREQI pin. The PCnet-32 controller will convey the LREQI request to the arbitration logic via the PCnet-32 controller LREQ output. The second local bus master must connect its LGNT input to the LGNTO output of the PCnet-32 controller in order to complete the local bus daisy-chain arbitration control.

When SLEEP is not asserted, daisy chain arbitration signals that pass through the PCnet-32 controller will experience a one-clock delay from input to output (i.e. LREQI to LREQ and LGNT to LGNTO).

While SLEEP is asserted (either in **snooze** mode or **coma** mode), if the PCnet-32 controller is configured for daisy chain (LREQI and LGNTO signals have been selected with the JTAGSEL pin), then the daisy-chain signal LREQI will be passed directly to the system arbi-

tration signal $\overline{LREQ}$ without experiencing a one-clock delay. However, some combinatorial delay will be introduced in this path.

Multi-Interrupt mode has been selected and the daisy-chain arbitration feature is not used, then the $\overline{LREQI}$ input should be tied to V_{DD}.

Note that this pin changes polarity when Local Busmode has been selected (see pin description of HOLDI in 486 Local Bus Interface section).

M/$\overline{IO}$

Memory I/O Select ***Input/Output***

During slave accesses to the PCnet-32 controller, the M/IO pin, along with D/$\overline{C}$ and W/$\overline{R}$, indicates the type cycle that is being performed. PCnet-32 controller will only respond to local bus accesses in which M/$\overline{IO}$ sampled as a zero by the PCnet-32 controller.

During PCnet-32 controller bus master accesses, the M/$\overline{IO}$ pin is an output and will always be driven high.

M/$\overline{IO}$ is floated if the PCnet-32 controller is not the current master on the local bus.

$\overline{RDYRTN}$

Ready Return ***Input***

$\overline{RDYRTN}$ functions as an input to the PCnet-32 controller. $\overline{RDYRTN}$ is used to terminate all master accesses performed by the PCnet-32 controller, except that linear burst transfers may also be terminated with the $\overline{BRDY}$ signal. $\overline{RDYRTN}$ is used to terminate slave read accesses to PCnet-32 controller I/O space.

When asserted during slave read accesses to PCnet-32 controller I/O space, $\overline{RDYRTN}$ indicates that the bus mastering device has seen the $\overline{LRDY}$ that was generated by the PCnet-32 controller and has accepted the PCnet-32 controller slave read data. Therefore, PCnet-32 controller will hold slave read data on the bus until it synchronously samples the $\overline{RDYRTN}$ input as active low. The PCnet-32 controller will not hold $\overline{LRDY}$ valid asserted during this time. The duration of the $\overline{LRDY}$ pulse generated by the PCnet-32 controller will always be a single LCLK cycle.

$\overline{RDYRTN}$ is ignored during slave write accesses PCnet-32 controller I/O space. Slave write accesses PCnet-32 controller I/O space are considered terminated by the PCnet-32 controller at the end of the cycle during which the PCnet-32 controller issues an active RDY.

In systems where both a $\overline{LRDY}$ and $\overline{RDYRTN}$ (or equivalent) signals are provided, then $\overline{LRDY}$ must not be tied to $\overline{RDYRTN}$. Most systems now provide for local device ready input to the memory controller that separate from the CPU $\overline{READY}$ signal. This second $\overline{READY}$ signal is usually labeled as $\overline{READYIN}$. This signal should be connected to the PCnet-32 controller LRDY signal. The CPU $\overline{READY}$ signal should be connected to the PCnet-32 controller $\overline{RDYRTN}$ pin.

In systems where only one $\overline{READY}$ signal is provided, then the PCnet-32 controller $\overline{LRDY}$ output may be tied the PCnet-32 controller $\overline{RDYRTN}$ input.

$\overline{RESET}$

System Reset ***Input***

When $\overline{RESET}$ is asserted low and the LB/$\overline{VESA}$ pin has been tied to VSS, then the PCnet-32 controller performs an internal system reset of the H_RESET type (HARDWARE_ RESET). The $\overline{RESET}$ pin must be held for a minimum of 30 LCLK periods when VL mode has been selected. While in the H_RESET state, the PCnet-32 controller will float or de-assert all outputs.

W/$\overline{R}$

Write/Read Select ***Input/Output***

During slave accesses to the PCnet-32 controller, the W/$\overline{R}$ pin, along with D/$\overline{C}$ and M/$\overline{IO}$, indicates the type of cycle that is being performed.

During PCnet-32 controller bus master accesses, the W/$\overline{R}$ pin is an output.

W/$\overline{R}$ is floated if the PCnet-32 controller is not the current master on the local bus.

$\overline{WBACK}$

Write Back ***Input***

$\overline{WBACK}$ is monitored as in input during VL-Bus Master Accesses. When PCnet-32 controller is current VL-Bus master, the PCnet-32 controller will float all appropriate bus mastering signals within 1 clock period of the assertion of $\overline{WBACK}$. When $\overline{WBACK}$ is de-asserted, PCnet-32 controller will re-execute any accesses that were suspended due to the assertion of $\overline{WBACK}$ and then will proceed with other scheduled accesses, if any.

Register access cannot be performed to the PCnet-32 device while $\overline{WBACK}$ is asserted.

Board Interface

LED1

LED1 ***Output***

This pin is shared with the EESK function. When operating as LED1, the function and polarity on this pin are programmable through BCR5. The LED1 output from the PCnet-32 controller is capable of sinking the necessary 12 mA of current to drive an LED directly.

The LED1 pin is also used during EEPROM Auto-detection to determine whether or not an EEPROM is present at the PCnet-32 controller microwire interface. At the trailing edge of $\overline{RESET}$, this pin is sampled to determine the value of the EEDET bit in BCR19. A sampled HIGH value means that an EEPROM is present, and EEDET will be set to ONE. A sampled

LOW value means that an EEPROM is not present, and EEDET will be set to ZERO. See the “EEPROM Auto-detection” section for more details.

If no LED circuit is to be attached to this pin, then a pull-up or pull-down resistor must be attached instead, in order to resolve the EEDET setting.

LED2

LED2 ***Output***

This pin is shared with the SRDCLK function. When operating as LED2, the function and polarity on this pin are programmable through BCR6. The LED2 output from the PCnet-32 controller is capable of sinking the necessary 12 mA of current to drive an LED directly.

This pin also selects address width for Software Relocatable Mode. When this pin is HIGH during Software Relocatable Mode, then the device will be programmed to use 32 bits of addressing while snooping accesses on the bus during Software Relocatable Mode. When this pin is LOW during Software Relocatable Mode, then the device will be programmed to use 24 bits of addressing while snooping accesses on the bus during Software Relocatable Mode. The upper 8 bits of address will be assumed to match during the snooping operation when LED2 is LOW. The 24-bit addressing mode is intended for use in systems that employ the GPSI signals. For more information on the GPSI function see section *General Purpose Serial Interface*.

If no LED circuit is to be attached to this pin, then a pull up or pull down resistor must be attached instead, in order to resolve the Software Relocatable Mode address width setting.

LEDPRE3

LEDPRE3 ***Output***

This pin is shared with the EEDO function. When operating as LEDPRE3, the function and polarity on this pin are programmable through BCR7. This signal is labeled as LED “*PRE*” 3 because of the multi-function nature of this pin. If an LED circuit were directly attached to this pin, it would create an IOL requirement that could not be met by the serial EEPROM that would also be attached to this pin. Therefore, if this pin is to be used as an additional LED output while an EEPROM is used in the system, then buffering is required between the LEDPRE3 pin and the LED circuit. If no EEPROM is included in the system design, then the LEDPRE3 signal may be directly connected to an LED without buffering. The LEDPRE3 output from the PCnet-32 controller is capable of sinking the necessary 12 mA of current to drive an LED in this case. For more details regarding LED connection, see the section on LEDs.

$\overline{\text{LNKST}}$

Link Status ***Output***

This pin provides 12 mA for driving an LED. It indicates an active link connection on the 10BASE-T interface. The function and polarity are programmable through BCR4. Note that this pin is multiplexed with the EEDI function.

This pin remains active in snooze mode.

SHFBUSY

Shift Busy ***Output***

The function of the SHFBUSY signal is to indicate when the last byte of the EEPROM contents has been shifted out of the EEPROM on the EEDO signal line. This information is useful for *external EEPROM-programmable registers* that do not use the microwire protocol, as is described herein: When the PCnet-32 controller is performing a serial read of the EEPROM through the microwire interface, the SHFBUSY signal will be driven HIGH. SHFBUSY can serve as a serial shift enable to allow the EEPROM data to be serially shifted into an external device or series of devices. The SHFBUSY signal will remain actively driven HIGH until the end of the EEPROM read operation. If the EEPROM checksum was verified, then the SHFBUSY signal will be driven LOW at the end of the EEPROM read operation. If the EEPROM checksum verification failed, then the SHFBUSY signal will remain HIGH. This function effectively demarcates the end of a successful EEPROM read operation and therefore is useful as a programmable-logic *low-active output enable* signal. For more details on external EEPROM-programmable registers, see the EEPROM *Microwire Access* section under Hardware Access.

This pin can be controlled by the host system by writing to BCR19, bit 3 (EBUSY).

$\overline{\text{SLEEP}}$

Sleep ***Input***

When $\overline{\text{SLEEP}}$ input is asserted (active LOW), the PCnet-32 controller performs an internal system reset of the S_RESET type and then proceeds into a power savings mode. (The reset operation caused by $\overline{\text{SLEEP}}$ assertion will not affect BCR registers.) All outputs will be placed in their normal S_RESET condition. During sleep mode, all PCnet-32 controller inputs will be ignored except for the $\overline{\text{SLEEP}}$ pin itself. De-assertion of $\overline{\text{SLEEP}}$ results in wake-up. The system must refrain from starting the network operations of the PCnet-32 controller for 0.5 seconds following the de-assertion of the $\overline{\text{SLEEP}}$ signal in order to allow internal analog circuits to stabilize.

Both LCLK and XTAL1 inputs must have valid clock signals present in order for the $\overline{\text{SLEEP}}$ command to take effect.

If $\overline{\text{SLEEP}}$ is asserted while $\overline{\text{LREQ}}$ is asserted, then the PCnet-32 controller will perform an internal system S_RESET and then wait for the assertion of $\overline{\text{LGNT}}$. When $\overline{\text{LGNT}}$ is asserted, the $\overline{\text{LREQ}}$ signal will be de-asserted and then the PCnet-32 controller will proceed to the power savings mode. Note that the internal system S_RESET will not cause the $\overline{\text{LREQ}}$ signal to be de-asserted.

The $\overline{\text{SLEEP}}$ pin should not be asserted during power supply ramp-up. If it is desired that $\overline{\text{SLEEP}}$ be asserted at power-up time, then the system must delay the assertion of $\overline{\text{SLEEP}}$ until three LCLK cycles after the completion of a valid pin $\overline{\text{RESET}}$ operation.

XTAL1–XTAL2

Crystal Oscillator Inputs ***Input/Output***

The crystal frequency determines the network data rate. The PCnet-32 controller supports the use of quartz crystals to generate a 20 MHz frequency compatible with the ISO 8802-3 (IEEE/ANSI 802.3) network frequency tolerance and jitter specifications. See the section *External Crystal Characteristics* (in section Manchester Encoder/Decoder) for more detail.

The network data rate is one-half of the crystal frequency. XTAL1 may alternatively be driven using an external CMOS level source, in which case XTAL2 must be left unconnected. Note that when the PCnet-32 controller is in coma mode, there is an internal 22 KW resistor from XTAL1 to ground. If an external source drives XTAL1, some power will be consumed driving this resistor. If XTAL1 is driven LOW at this time power consumption will be minimized. In this case, XTAL1 must remain active for at least 30 cycles after the assertion of $\overline{\text{SLEEP}}$ and de-assertion of $\overline{\text{LREQ}}$.

Microwire EEPROM Interface

EESK

EEPROM Serial Clock ***Output***

The EESK signal is used to access the external ISO 8802-3 (IEEE/ANSI 802.3) address PROM. This pin is designed to directly interface to a serial EEPROM that uses the microwire interface protocol. EESK is connected to the microwire EEPROM's Clock pin. It is controlled by either the PCnet-32 controller directly during a read of the entire EEPROM, or indirectly by the host system by writing to BCR19, bit 1. EESK can be used during programming of external *EEPROM-programmable registers* that do not use the microwire protocol as follows:

When the PCnet-32 controller is performing a serial read of the IEEE Address EEPROM through the microwire interface, the SHFBUSY signal will serve as a serial shift enable to allow the EEPROM data to be serially shifted into an external device or series of devices. This same signal can be used to gate the *output* of the programmed logic to avoid the problem of releasing intermediate values to the rest of the system board logic. The EESK signal can serve as the clock, and EEDO will serve as the input data stream to the programmable shift register.

EEDO

EEPROM Data Out ***Input***

The EEDO signal is used to access the external ISO 8802-3 (IEEE/ANSI 802.3) address PROM. This pin is designed to directly interface to a serial EEPROM that uses the microwire interface protocol. EEDO is connected to the microwire EEPROM's Data Output pin. It is controlled by the EEPROM during reads. It may be read by the host system by reading BCR19, bit 0.

EEDO can be used during programming of *external EEPROM-programmable registers* that do not use the microwire protocol as follows:

When the PCnet-32 controller is performing a serial read of the IEEE Address EEPROM through the microwire interface, the SHFBUSY signal will serve as a serial shift enable to allow the EEPROM data to be serially shifted into an external device or series of devices. This same signal can be used to gate the *output* of the programmed logic to avoid the problem of releasing intermediate values to the rest of the system board logic. The EESK signal can serve as the clock, and EEDO will serve as the input data stream to the programmable shift register.

EECS

EEPROM Chip Select ***Output***

The function of the EECS signal is to indicate to the microwire EEPROM device that it is being accessed. The EECS signal is active high. It is controlled by either the PCnet-32 controller during a read of the entire EEPROM, or indirectly by the host system by writing to BCR19, bit 2.

EEDI

EEPROM Data In ***Output***

The EEDI signal is used to access the external ISO 8802-3 (IEEE/ANSI 802.3) address PROM. EEDI functions as an output. This pin is designed to directly interface to a serial EEPROM that uses the microwire interface protocol. EEDI is connected to the microwire EEPROM's Data Input pin. It is controlled by either the PCnet-32 controller during command portions of a read of the entire EEPROM, or indirectly by the host system by writing to BCR19, bit 0.

Attachment Unit Interface

CI±

Collision In ***Input***

A differential input pair signaling the PCnet-32 controller that a collision has been detected on the network media, indicated by the CI± inputs being driven with a 10 MHz pattern of sufficient amplitude and pulse width to meet ISO 8802-3 (IEEE/ANSI 802.3) standards. Operates at pseudo ECL levels.

DI±

Data In ***Input***

A differential input pair to the PCnet-32 controller carrying Manchester encoded data from the network. Operates at pseudo ECL levels.

DO±

Data Out ***Output***

A differential output pair from the PCnet-32 controller for transmitting Manchester encoded data to the network. Operates at pseudo ECL levels.

Twisted Pair Interface

RXD±

10BASE-T Receive Data ***Input***

10BASE-T port differential receivers.

TXD±

10BASE-T Transmit Data ***Output***

10BASE-T port differential drivers.

TXP±

10BASE-T Pre-distortion Control ***Output***

These outputs provide transmit predistortion control in conjunction with the 10BASE-T port differential drivers.

External Address Detection Interface

The EADI interface is enabled through bit 3 of BCR2 (EADISEL).

$\overline{\text{EAR}}$

External Address Reject Low ***Input***

An EADI input signal. The incoming frame will be checked against the internally active address detection mechanisms and the result of this check will be OR'd with the value on the $\overline{\text{EAR}}$ pin. The $\overline{\text{EAR}}$ pin is defined as $\overline{\text{REJECT}}$.

See the EADI section for details regarding the function and timing of this signal.

Note that this pin is multiplexed with the INTR2 pin.

SFBD

Start Frame–Byte Delimiter ***Output***

Start Frame–Byte Delimiter Enable. EADI output signal. An initial rising edge on this signal indicates that a start of frame delimiter has been detected. The serial bit stream will follow on the SRD signal, commencing with the destination address field. SFBD will go high for 4 bit times (400 ns) after detecting the second "1" in the SFD (Start of Frame Delimiter) of a received frame. SFBD will subsequently toggle every 400 ns (1.25 MHz frequency) with each rising edge indicating the first bit of each subsequent byte of the received serial bit stream. SFBD will be inactive during frame transmission.

Note that this pin is multiplexed with the LED1 pin.

SRD

Serial Receive Data ***Output***

An EADI output signal. SRD is the decoded NRZ data from the network. This signal can be used for external address detection. Note that when the 10BASE-T port is selected, transitions on SRD will only occur during receive activity. When the AUI port is selected, transitions on SRD will occur during both transmit and receive activity.

Note that this pin is multiplexed with the LEDPRE3 pin.

SRDCLK

Serial Receive Data Clock ***Output***

An EADI output signal. Serial Receive Data is synchronous with reference to SRDCLK. Note that when the 10BASE-T port is selected, transitions on SRDCLK will only occur during receive activity. When the AUI port is selected, transitions on SRDCLK will occur during both transmit and receive activity.

Note that this pin is multiplexed with the LED2 pin.

General Purpose Serial Interface

The GPSI interface is selected through the PORTSEL bits of the Mode register (CSR15) and enabled through the TSTSHDW[1] bit (BCR18) or the CORETEST bit (CSR124).

Note that when GPSI test mode is invoked, slave address decoding must be restricted to the lower 24 bits of the address bus by setting the IOAW24 bit in BCR2 and by pulling LED2 LOW during Software Relocatable Mode. The upper 8 bits of the address bus will always be considered matched when examining incoming I/O addresses. During master accesses while in GPSI mode, the PCnet-32 controller will not drive the upper 8 bits of the address bus with address information. See the GPSI section for more detail.

TXDAT

Transmit Data ***Input/Output***

TXDAT is an output, providing the serial bit stream for transmission, including preamble, SFD data and FCS field, if applicable.

Note that the TxDAT pin is multiplexed with the A31 pin.

TXEN

Transmit Enable ***Input/Output***

TXEN is an output, providing an enable signal for transmission. Data on the TXDAT pin is not valid unless the TXEN signal is HIGH.

Note that the TXEN pin is multiplexed with the A30 pin.

STDCLK

Serial Transmit Data Clock ***Input***

STDCLK is an input, providing a clock signal for MAC activity, both transmit and receive. Rising edges of the STDCLK can be used to validate TXDAT output data.

The STDCLK pin is multiplexed with the A29 pin.

Note that this signal must meet the frequency stability requirement of the ISO 8802-3 (IEEE/ANSI 802.3) specification for the crystal.

CLSN

Collision ***Input/Output***

CLSN is an input, indicating to the core logic that a collision has occurred on the network.

Note that the CLSN pin is multiplexed with the A28 pin.

RXCRS

Receive Carrier Sense ***Input/Output***

RXCRS is an input. When this signal is HIGH, it indicates to the core logic that the data on the RXDAT input pin is valid.

Note that the RXCRS pin is multiplexed with the A27 pin.

SRDCLK

Serial Receive Data Clock ***Input/Output***

SRDCLK is an input. Rising edges of the SRDCLK signal are used to sample the data on the RXDAT input whenever the RXCRS input is HIGH.

Note that the SRDCLK pin is multiplexed with the A26 pin.

RXDAT

Receive Data ***Input/Output***

RXDAT is an input. Rising edges of the SRDCLK signal are used to sample the data on the RXDAT input whenever the RXCRS input is HIGH.

Note that the RXDAT pin is multiplexed with the A25 pin.

IEEE 1149.1 Test Access Port Interface

TCK

Test Clock ***Input***

The clock input for the boundary scan test mode operation. TCK can operate up to 10 MHz. If left unconnected, this pin has a default value of HIGH.

TDI

Test Data Input ***Input***

The test data input path to the PCnet-32 controller. If left unconnected, this pin has a default value of HIGH.

TDO

Test Data Output ***Output***

The test data output path from the PCnet-32 controller. TDO is floated when the JTAG port is inactive.

TMS

Test Mode Select ***Input***

A serial input bit stream is used to define the specific boundary scan test to be executed. If left unconnected, this pin has a default value of HIGH.

Power Supply Pins

AVDD

Analog Power (4 Pins) ***Power***

There are four analog +5 Volt supply pins. Special attention should be paid to the printed circuit board layout to avoid excessive noise on these lines. Refer to Appendix B and the *PCnet Family Technical Manual (PID# 18216A)* for details.

AVSS

Analog Ground (2 Pins) ***Power***

There are two analog ground pins. Special attention should be paid to the printed circuit board layout to avoid excessive noise on these lines. Refer to Appendix B and the *PCnet Family Technical Manual* for details.

DVDD

Digital Power (3 Pins) ***Power***

There are 3 digital power supply pins (DVDD1, DVDD 2, and DVDD3) used by the internal digital circuitry.

DVDDCLK

Digital Power Clock (1 Pin) ***Power***

This pin is used to supply power to the clock buffering circuitry.

DVDDO

I/O Buffer Digital Power (5 Pins) ***Power***

There are 5 digital power supply pins (DVDDO1–DVDDO5) used by Input/Output buffer drivers.

DVSS

Digital Ground (4 Pins) ***Ground***

There are 4 digital ground pins (DVSS1–DVSS4) used by the internal digital circuitry.

DVSSCLK

Digital Ground Clock (1 Pin) ***Ground***

This pin is used to supply a ground to the clock buffering circuitry.

DVSSN

I/O Buffer Digital Ground (15 Pins) ***Ground***

These 15 ground pins (DVSSN1–DVSSN15) are used by the Input/Output buffer drivers.

DVSSPAD

Digital Ground Pad (1 Pin) ***Ground***

This pin is used by the Input/Output logic circuits.

VESA VL-BUS / LOCAL BUS PIN CROSS-REFERENCE

Listed by Pin Number

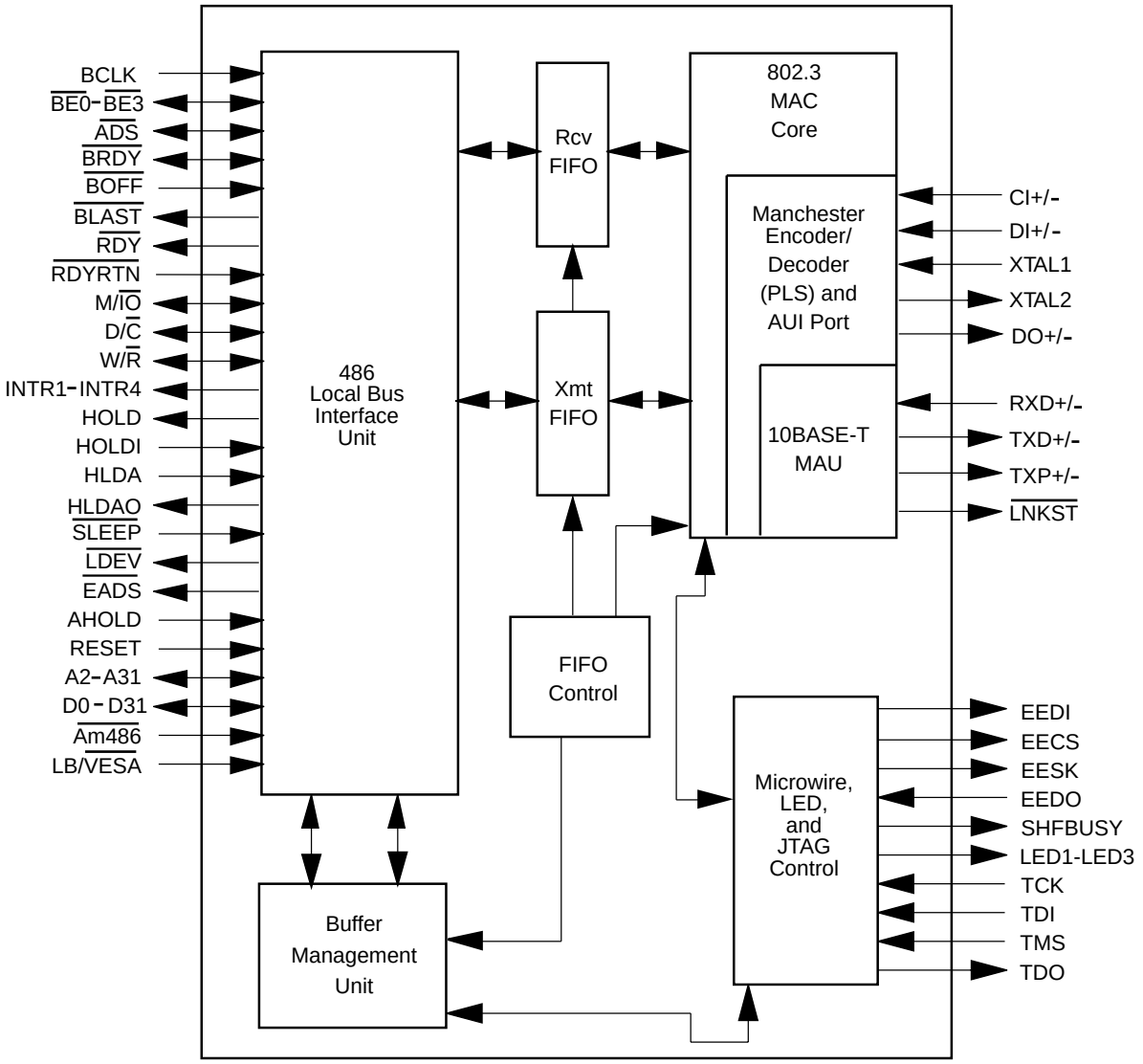
Pin #	VESA VL-Bus Mode Pin Name	Local Bus Mode Pin Name	Pin #	VESA VL-Bus Mode Pin Name	Local Bus Mode Pin Name
1	NC	NC	**41**	NC	NC
2	LED2/SRDCLK	LED2/SRDCLK	**42**	$\overline{\text{RDYRTN}}$	$\overline{\text{RDYRTN}}$
3	DVSSN1	DVSSN1	**43**	$\overline{\text{LRDY}}$	$\overline{\text{RDY}}$
4	ADR17	A17	**44**	W/$\overline{\text{R}}$	W/$\overline{\text{R}}$
5	ADR16	A16	**45**	$\overline{\text{ADS}}$	$\overline{\text{ADS}}$
6	ADR15	A15	**46**	M/$\overline{\text{IO}}$	M/$\overline{\text{IO}}$
7	ADR14	A14	**47**	$\overline{\text{BE3}}$	$\overline{\text{BE3}}$
8	DVDDO1	DVDDO1	**48**	D/$\overline{\text{C}}$	D/$\overline{\text{C}}$
9	ADR13	A13	**49**	$\overline{\text{RESET}}$	RESET
10	ADR12	A12	**50**	$\overline{\text{BE2}}$	$\overline{\text{BE2}}$
11	DVSSN2	DVSSN2	**51**	DVSSN5	DVSSN5
12	ADR11	A11	**52**	$\overline{\text{BE1}}$	$\overline{\text{BE1}}$
13	ADR10	A10	**53**	$\overline{\text{BE0}}$	$\overline{\text{BE0}}$
14	ADR9	A9	**54**	$\overline{\text{WBACK}}$	$\overline{\text{BOFF}}$
15	ADR8	A8	**55**	DAT31	D31
16	ADR7	A7	**56**	DAT30	D30
17	DVSSN3	DVSSN3	**57**	DAT29	D29
18	ADR6	A6	**58**	DAT28	D28
19	ADR5	A5	**59**	DVSSN6	DVSSN6
20	DVSSPAD	DVSSPAD	**60**	DAT27	D27
21	ADR4	A4	**61**	DAT26	D26
22	ADR3	A3	**62**	DAT25	D25
23	DVSSN4	DVSSN4	**63**	DAT24	D24
24	ADR2	A2	**64**	DAT23	D23
25	$\overline{\text{LBS16}}$	AHOLD	**65**	DVDDO3	DVDDO3
26	DVDD1	DVDD1	**66**	DAT22	D22
27	$\overline{\text{LEADS}}$	$\overline{\text{EADS}}$	**67**	DVSSN7	DVSSN7
28	DVSSCLK	DVSSCLK	**68**	DAT21	D21
29	LCLK	BCLK	**69**	DVSS2	DVSS2
30	DVDDCLK	DVDDCLK	**70**	DAT20	D20
31	VLBEN	$\overline{\text{Am486}}$	**71**	DAT19	D19
32	$\overline{\text{BLAST}}$	$\overline{\text{BLAST}}$	**72**	DAT18	D18
33	$\overline{\text{LGNT}}$	HLDA	**73**	DAT17	D17
34	$\overline{\text{BRDY}}$	$\overline{\text{BRDY}}$	**74**	DVSSN8	DVSSN8
35	DVDDO2	DVDDO2	**75**	DAT16	D16
36	$\overline{\text{LREQ}}$	HOLD	**76**	DAT15	D15
37	DVSS1	DVSS1	**77**	DAT14	D14
38	$\overline{\text{LDEV}}$	$\overline{\text{LDEV}}$	**78**	DAT13	D13
39	NC	NC	**79**	NC	NC
40	NC	NC	**80**	NC	NC

VESA VL-BUL / LOCAL BUS PIN CROSS-REFERENCE (continued)

Listed by Pin Number

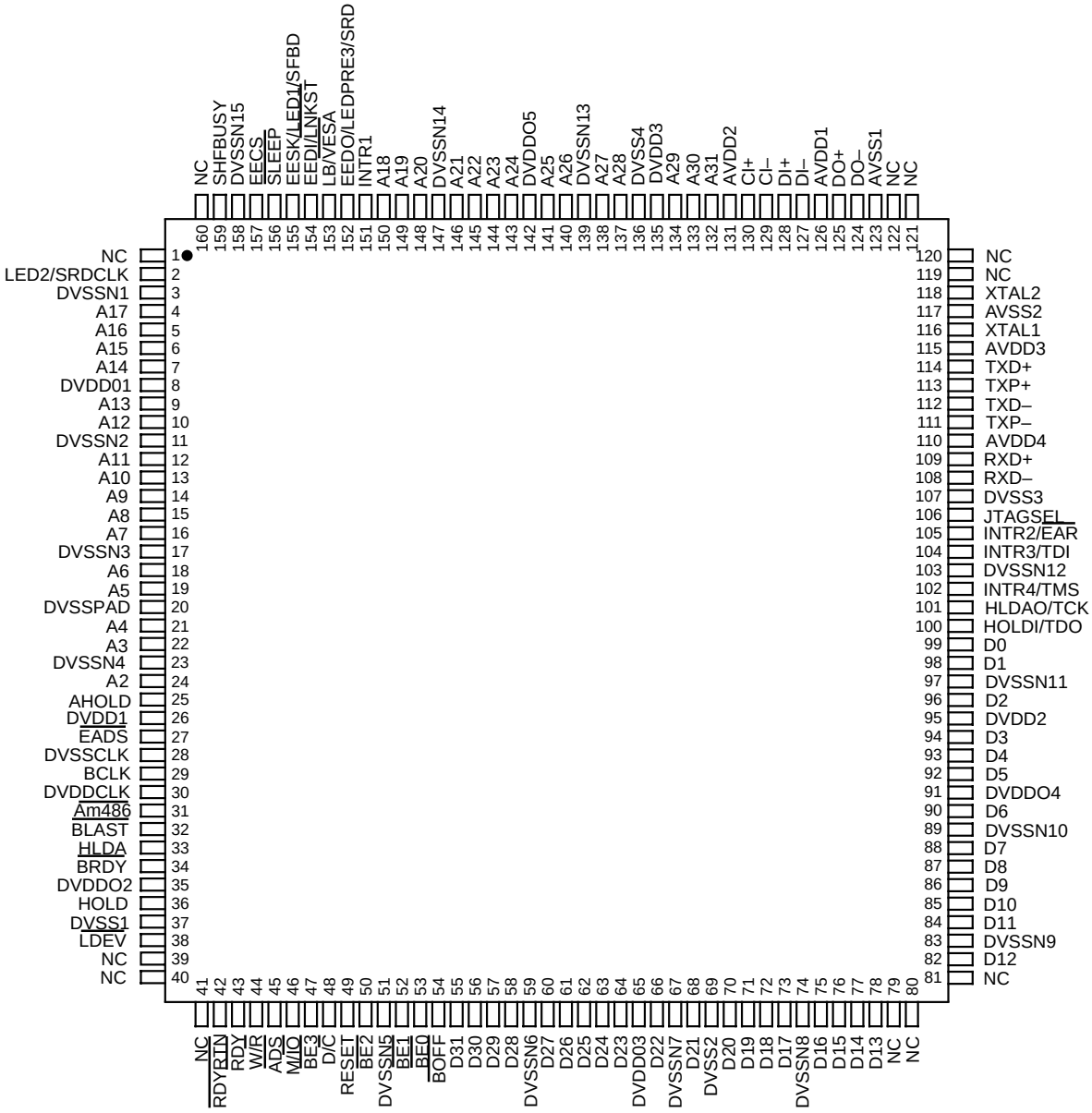
Pin #	VESA VL-Bus Mode Pin Name	Local Bus Mode Pin Name	Pin #	VESA VL-Bus Mode Pin Name	Local Bus Mode Pin Name
81	NC	NC	**121**	NC	NC
82	DAT12	D12	**122**	NC	NC
83	DVSSN9	DVSSN9	**123**	AVSS1	AVSS1
84	DAT11	D11	**124**	DO-	DO-
85	DAT10	D10	**125**	DO+	DO+
86	DAT9	D9	**126**	AVDD1	AVDD1
87	DAT8	D8	**127**	DI-	DI-
88	DAT7	D7	**128**	DI+	DI+
89	DVSSN10	DVSSN10	**129**	CI-	CI-
90	DAT6	D6	**130**	CI+	CI+
91	DVDDO4	DVDDO4	**131**	AVDD2	AVDD2
92	DAT5	D5	**132**	ADR31	A31
93	DAT4	D4	**133**	ADR30	A30
94	DAT3	D3	**134**	ADR29	A29
95	DVDD2	DVDD2	**135**	DVDD3	DVDD3
96	DAT2	D2	**136**	DVSS4	DVSS4
97	DVSSN11	DVSSN11	**137**	ADR28	A28
98	DAT1	D1	**138**	ADR27	A27
99	DAT0	D0	**139**	DVSSN13	DVSSN13
100	$\overline{\text{LREQI}}$/TDO	HOLDI/TDO	**140**	ADR26	A26
101	$\overline{\text{LGNTO}}$/TCK	$\overline{\text{HLDAO}}$/TCK	**141**	ADR25	A25
102	INTR4/TMS	INTR4/TMS	**142**	DVDDO5	DVDDO5
103	DVSSN12	DVSSN12	**143**	ADR24	A24
104	INTR3/TDI	INTR3/TDI	**144**	ADR23	A23
105	INTR2/$\overline{\text{EAR}}$	INTR2/$\overline{\text{EAR}}$	**145**	ADR22	A22
106	JTAGSEL	JTAGSEL	**146**	ADR21	A21
107	DVSS3	DVSS3	**147**	DVSSN14	DVSSN14
108	RXD-	RXD-	**148**	ADR20	A20
109	RXD+	RXD+	**149**	ADR19	A19
110	AVDD4	AVDD4	**150**	ADR18	A18
111	TXP-	TXP-	**151**	INTR1	INTR1
112	TXD-	TXD-	**152**	EEDO/LEDPRE3/SRD	EEDO/LEDPRE3/SRD
113	TXP+	TXP+	**153**	LB/$\overline{\text{VESA}}$	LB/$\overline{\text{VESA}}$
114	TXD+	TXD+	**154**	EEDI/$\overline{\text{LNKST}}$	EEDI/$\overline{\text{LNKST}}$
115	AVDD3	AVDD3	**155**	EESK/LED1/SFBD	EESK/LED1/SFBD
116	XTAL1	XTAL1	**156**	$\overline{\text{SLEEP}}$	$\overline{\text{SLEEP}}$
117	AVSS2	AVSS2	**157**	EECS	EECS
118	XTAL2	XTAL2	**158**	DVSSN15	DVSSN15
119	NC	NC	**159**	SHFBUSY	SHFBUSY
120	NC	NC	**160**	NC	NC

BLOCK DIAGRAM: 486 LOCAL BUS MODE



18219-2

CONNECTION DIAGRAM: 486 LOCAL BUS MODE



18219-4

PIN DESIGNATIONS: 486 LOCAL BUS MODE

Listed by Pin Number

Pin No.	Name	Pin No.	Name	Pin No.	Name	Pin No.	Name
1	NC	41	NC	81	NC	121	NC
2	LED2/SRDCLK	42	$\overline{\text{RDYRTN}}$	82	D12	122	NC
3	DVSSN1	43	$\overline{\text{RDY}}$	83	DVSSN9	123	AVSS1
4	A17	44	W/$\overline{\text{R}}$	84	D11	124	DO–
5	A16	45	$\overline{\text{ADS}}$	85	D10	125	DO+
6	A15	46	M/$\overline{\text{IO}}$	86	D9	126	AVDD1
7	A14	47	$\overline{\text{BE3}}$	87	D8	127	DI–
8	DVDDO1	48	D/$\overline{\text{C}}$	88	D7	128	DI+
9	A13	49	RESET	89	DVSSN10	129	CI–
10	A12	50	$\overline{\text{BE2}}$	90	D6	130	CI+
11	DVSSN2	51	DVSSN5	91	DVDDO4	131	AVDD2
12	A11	52	$\overline{\text{BE1}}$	92	D5	132	A31
13	A10	53	$\overline{\text{BE0}}$	93	D4	133	A30
14	A9	54	$\overline{\text{BOFF}}$	94	D3	134	A29
15	A8	55	D31	95	DVDD2	135	DVDD3
16	A7	56	D30	96	D2	136	DVSS4
17	DVSSN3	57	D29	97	DVSSN11	137	A28
18	A6	58	D28	98	D1	138	A27
19	A5	59	DVSSN6	99	D0	139	DVSSN13
20	DVSSPAD	60	D27	100	HOLDI/TDO	140	A26
21	A4	61	D26	101	HLDAO/TCK	141	A25
22	A3	62	D25	102	INTR4/TMS	142	DVDDO5
23	DVSSN4	63	D24	103	DVSSN12	143	A24
24	A2	64	D23	104	INTR3/TDI	144	A23
25	AHOLD	65	DVDDO3	105	INTR2/$\overline{\text{EAR}}$	145	A22
26	DVDD1	66	D22	106	JTAGSEL	146	A21
27	$\overline{\text{EADS}}$	67	DVSSN7	107	DVSS3	147	DVSSN14
28	DVSSCLK	68	D21	108	RXD–	148	A20
29	BCLK	69	DVSS2	109	RXD+	149	A19
30	DVDDCLK	70	D20	110	AVDD4	150	A18
31	$\overline{\text{Am486}}$	71	D19	111	TXP–	151	INTR1
32	$\overline{\text{BLAST}}$	72	D18	112	TXD–	152	EEDO/LEDPRE3/SRD
33	HLDA	73	D17	113	TXP+	153	LB/$\overline{\text{VESA}}$
34	$\overline{\text{BRDY}}$	74	DVSSN8	114	TXD+	154	EEDI/$\overline{\text{LNKST}}$
35	DVDDO2	75	D16	115	AVDD3	155	EESK/LED1/SFBD
36	HOLD	76	D15	116	XTAL1	156	$\overline{\text{SLEEP}}$
37	DVSS1	77	D14	117	AVSS2	157	EECS
38	$\overline{\text{LDEV}}$	78	D13	118	XTAL2	158	DVSSN15
39	NC	79	NC	119	NC	159	SHFBUSY
40	NC	80	NC	120	NC	160	NC

PIN DESIGNATIONS: 486 LOCAL BUS MODE

Listed by Pin Name

Name	Pin No.	Name	Pin No.	Name	Pin No.	Name	Pin No.
A2	24	$\overline{BE0}$	53	D30	56	EESK/LED1/SFBD	155
A3	22	$\overline{BE1}$	52	D31	55	HLDA	33
A4	21	$\overline{BE2}$	50	DI+	128	HLDAO/TCK	101
A5	19	$\overline{BE3}$	47	DI–	127	HOLD	36
A6	18	$\overline{BLAST}$	32	DO+	125	HOLDI/TDO	100
A7	16	$\overline{BOFF}$	54	DO–	124	INTR1	151
A8	15	$\overline{BRDY}$	34	DVDD1	26	INTR2/$\overline{EAR}$	105
A9	14	CI+	130	DVDD2	95	INTR3/TDI	104
A10	13	CI–	129	DVDD3	135	INTR4/TMS	102
A11	12	D/$\overline{C}$	48	DVDDCLK	30	JTAGSEL	106
A12	10	D0	99	DVDDO1	8	LB/$\overline{VESA}$	153
A13	9	D1	98	DVDDO2	35	LDEV	38
A14	7	D2	96	DVDDO3	65	LED2/SRDCLK	2
A15	6	D3	94	DVDDO4	91	M/$\overline{IO}$	46
A16	5	D4	93	DVDDO5	142	NC	1
A17	4	D5	92	DVSS1	37	NC	39
A18	150	D6	90	DVSS2	69	NC	40
A19	149	D7	88	DVSS3	107	NC	41
A20	148	D8	87	DVSS4	136	NC	79
A21	146	D9	86	DVSSCLK	28	NC	80
A22	145	D10	85	DVSSN1	3	NC	81
A23	144	D11	84	DVSSN2	11	NC	119
A24	143	D12	82	DVSSN3	17	NC	120
A25	141	D13	78	DVSSN4	23	NC	121
A26	140	D14	77	DVSSN5	51	NC	122
A27	138	D15	76	DVSSN6	59	NC	160
A28	137	D16	75	DVSSN7	67	$\overline{RDY}$	43
A29	134	D17	73	DVSSN8	74	$\overline{RDYRTN}$	42
A30	133	D18	72	DVSSN9	83	RESET	49
A31	132	D19	71	DVSSN10	89	RXD+	109
ADS	45	D20	70	DVSSN11	97	RXD–	108
AHOLD	25	D21	68	DVSSN12	103	SHFBUSY	159
$\overline{Am486}$	31	D22	66	DVSSN13	139	$\overline{SLEEP}$	156
AVDD1	126	D23	64	DVSSN14	147	TXD+	114
AVDD2	131	D24	63	DVSSN15	158	TXD–	112
AVDD3	115	D25	62	DVSSPAD	20	TXP+	113
AVDD4	110	D26	61	$\overline{EADS}$	27	TXP–	111
AVSS1	123	D27	60	EECS	157	W/$\overline{R}$	44
AVSS2	117	D28	58	EEDI/$\overline{LNKST}$	154	XTAL1	116
BCLK	29	D29	57	EEDO/LEDPRE3/SRD	152	XTAL2	118

PIN DESIGNATIONS: 486 LOCAL BUS MODE

Listed by Group

Pin Name	Pin Function	Type	Driver	No. of Pins
486/386DX Local Bus Interface				
A2–A31	Address Bus	IO	TS	30
$\overline{ADS}$	Address Status	I/O	TS	1
AHOLD	Address Hold	I		1
$\overline{Am486}$	Am486 Mode Select	I		1
BCLK	Bus Clock	I		1
$\overline{BE0}$-$\overline{BE3}$	Byte Enable	I/O	TS	4
$\overline{BLAST}$	Burst Last	O	TS	1
$\overline{BOFF}$	Backoff	I		1
$\overline{BRDY}$	Burst Ready	I/O	TS	1
D/$\overline{C}$	Data/Control Select	I/O	TS	1
D0–D31	Data Bus	I/O	TS	32
$\overline{EADS}$	External Address Strobe	O	TS	1
HLDA	Hold Acknowledge	I		1
HLDAO	Hold Acknowledge Out	I/O	TS	1
HOLD	Hold Request	O	O8	1
HOLDI	Hold Request In	I/O	TS	1
INTR1	Interrupt Number 1	O	TS	1
INTR2	Interrupt Number 2	I/O	TS	1
INTR3	Interrupt Number 3	I/O	TS	1
INTR4	Interrupt Number 4	I/O	TS	1
JTAGSEL	JTAG Select	I		1
LB/$\overline{VESA}$	Local Bus/VESA VL-Bus Select pin	I		1
$\overline{LDEV}$	Local Device	O	O4	1
M/$\overline{IO}$	Memory/I/O Select	I/O	TS	1
$\overline{RDY}$	Ready	O	TS	1
$\overline{RDYRTN}$	Ready Return	I		1
W/$\overline{R}$	Write/Read Select	I/O	TS	1
Board Interface				
EECS	Microwire Serial PROM Chip Select	O	O8	1
EEDI/$\overline{LNKST}$	Microwire Serial EEPROM Data In/Link Status	O	LED	1
EEDO/LEDPRE3	Microwire Address PROM Data Out/LED3 predriver	I/O	LED	1
EESK/LED1	Microwire Serial PROM Clock/LED1	O	LED	1
LED2	LED Output Number 2	O	LED	1
RESET	Reset	I		1
SHFBUSY	Shift Busy (for external EEPROM-programmable logic)	O	O8	1
$\overline{SLEEP}$	Sleep Mode	I		1
XTAL1	Crystal Input	I		1
XTAL2	Crystal Output	O		1

PIN DESIGNATIONS: 486 LOCAL BUS MODE (continued)

Listed by Group

Pin Name	Pin Function	Type	Driver	No. of Pins
Attachment Unit Interface (AUI)				
CI+/CI–	AUI Collision Differential Pair	I		2
DI+/DI–	AUI Data In Differential Pair	I		2
DO+/DO–	AUI Data Out Differential Pair	O	DO	2
Twisted-Pair Transceiver Interface (10BASE-T)				
RXD+/RXD–	Receive Differential Pair	I		2
TXD+/TXD–	Transmit Differential Pair	O	TDO	2
TXP+/TXP–	Transmit Predistortion Differential Pair	O	TPO	2
$\overline{\text{LNKST}}$/EEDI	Link Status/Microwire Serial EEPROM Data In	O	LED	1
IEEE 1149.1 Test Access Port Interface (JTAG)				
TCK	Test Clock	I/O	TS	1
TDI	Test Data In	I/O	TS	1
TDO	Test Data Out	I/O	TS	1
TMS	Test Mode Select	I/O	TS	1
External Address Detection Interface (EADI)				
$\overline{\text{EAR}}$	External Address Reject Low	I/O	TS	1
SRD	Serial Receive Data	I/O	TS	1
SRDCLK	Serial Receive Data Clock	I/O	TS	1
SFBD	Start Frame—Byte Delimiter	O	LED	1
Power Supplies				
AVDD	Analog Power	P		4
AVSS	Analog Ground	P		2
DVDD	Digital Power	P		3
DVDDCLK	Digital Power Clock	P		1
DVDDO	I/O Buffer Digital Power	P		5
DVSS	Digital Ground	P		4
DVSSCLK	Digital Ground Clock	P		1
DVSSN	I/O Buffer Digital Ground	P		15
DVSSPAD	Digital Ground Pad	P		1

PIN DESIGNATIONS: 486 LOCAL BUS MODE

Driver Type

Table 9. Output Driver Types

Name	Type	I_{OL}(mA)	I_{OL}(mA)	pF
TS	Tri-State	8	-0.4	50
O8	Totem Pole	8	-0.4	50
O4	Totem Pole	4	-0.4	50
OD	Open Drain	8		50
LED	LED	12	-0.4	50

Table 10. Pins with Pullups

Signal	Pullup
TDI	≥ 10 KΩ
TMS	≥ 10 KΩ
TCK	≥ 10 KΩ

PIN DESCRIPTION: 486 LOCAL BUS MODE

Configuration Pins

JTAGSEL

JTAG Function Select ***Input***

The value of this pin will asynchronously select between JTAG Mode and Multi-Interrupt Mode.

The value of this pin will asynchronously affect the function of the JTAG–INTR–Daisy chain arbitration pins, regardless of the state of the RESET pin and regardless of the state of the BCLK pin. If the value is a “1”, then the PCnet-32 controller will be programmed for JTAG mode. If the value is a “0”, then the PCnet-32 controller will be programmed for Multi-Interrupt Mode.

When programmed for JTAG mode, four pins of the PCnet-32 controller will be configured as a JTAG (IEEE 1149.1) Test Access Port. When programmed for Multi-Interrupt Mode, two of the JTAG pins will become interrupts and two JTAG pins will be used for daisy chain arbitration support. Table 11 below outlines the pin changes that will occur by programming the JTAGSEL pin.

Table 11. JTAG Pin Changes

Pin	JTAGSEL=1 JTAG Mode	JTAGSEL=0 Multi-Interrupt Mode
HLDAO/TCK	TCK	HLDAO
HOLDI/TDO	TDO	HOLDI
INTR3/TDI	TDI	INTR3
INTR4/TMS	TMS	INTR4

The JTAGSEL pin may be tied directly to V_{DD} or V_{SS}. A series resistor may be used but is not necessary.

LB/$\overline{\text{VESA}}$

Local Bus/VESA VL-Bus Select ***Input***

The value of this pin will asynchronously determine the operating mode of the PCnet-32 controller, regardless of the state of RESET and regardless of the state of the BCLK pin. If the LB/$\overline{\text{VESA}}$ pin is tied to V_{DD}, then the PCnet-32 controller will be programmed for Local Bus Mode. If the LB/$\overline{\text{VESA}}$ pin is tied to V_{SS}, then the PCnet-32 controller will be programmed for VESA VL-Bus Mode.

Note that the setting of LB/$\overline{\text{VESA}}$ determines the functionality of the following pins (names in parentheses are pins in the VESA VL-Bus Mode): Am486 (VLBEN), RESET ($\overline{\text{RESET}}$), AHOLD ($\overline{\text{LBS16}}$), HOLD ($\overline{\text{LREQ}}$), HLDA($\overline{\text{LGNT}}$), HOLDI ($\overline{\text{LREQI}}$), and HLDAO ($\overline{\text{LGNTO}}$).

$\overline{\text{Am486}}$

Am486 Mode Select ***Input***

The $\overline{\text{Am486}}$ pin should be tied directly to V_{SS}. A series resistor may be used but is not necessary.

Note: *This pin is used to enable bursing in the VESA VL-Bus mode when the LB/$\overline{\text{VESA}}$ pin has been tied to VSS. See the pin description for the VLBEN pin in the VESA VL-Bus Mode section.*

Configuration Pin Settings Summary

Table 12 shows the possible pin configurations that may be invoked with the PCnet-32 controller configuration pins.

Table 12. Configuration Pin Settings

LB/VESA	Am486/Am386	JTAGSEL	Mode Selected
0	X*	0	VL Bus mode with 4 interrupts and daisy chain arbitration
0	X*	1	VL Bus mode with 2 interrupts and JTAG
1	0	0	Am486 mode with 4 interrupts and daisy chain arbitration
1	0	1	Am486 mode with 2 interrupts and JTAG
1	1	X	Reserved

**X = Don’t care*

Table 13. Pin Connections to Power/Ground

Pin Name	Pin No	Supply Strapping	Resistive Connection to Supply	Recommended Resistor Size
LED2/SRDCLK	2	Required	Required	324 Ω in series with LED, or 10 KΩ without LED
AHOLD	25	Optional	Required	10 KΩ
$\overline{\text{Am486}}$	31	Required	Optional	NA
$\overline{\text{BOFF}}$	54	Optional	Required	10 KΩ
HOLDI/TDO	100	Optional	Required	10 KΩ
JTAGSEL	106	Required	Optional	NA
EEDO/LEDPRE3/SRD	152	Optional	Required	10 KΩ
LB/$\overline{\text{VESA}}$	153	Required	Optional	NA
EEDI/$\overline{\text{LNKST}}$	154	Optional	Required	324 Ω in series with LED, or 10 KΩ without LED
EESK/LED1/SFBD	155	Required	Required	324 Ω in series with LED, or 10 KΩ without LED
$\overline{\text{SLEEP}}$	156	Optional	Required	10 KΩ
All Other Pins	—	Optional	Required	10 KΩ

Pin Connections to V_{DD} or V_{SS}

Several pins may be connected to V_{DD} or V_{SS} for various application options. Some pins are required to be connected to V_{DD} or V_{SS} in order to set the controller into a particular mode of operation, while other pins might be connected to V_{DD} or V_{SS} if that pin's function is not implemented in a specific application. Table 13 shows which pins require a connection to V_{DD} or V_{SS}, and which pins may optionally be connected to V_{DD} or V_{SS} because the application does not support that pin's function. The table also shows whether or not the connections need to be resistive.

Local Bus Interface

A2–A31

Address Bus ***Input/Output***

Address information which is stable during a bus operation, regardless of the source. When the PCnet-32 controller is Current Master, A1–A31 will be driven. When the PCnet-32 controller is not Current Master, the A2–A31 lines are continuously monitored to determine if an address match exists for I/O slave transfers.

Some portion of the Address Bus will be floated at the time of an address hold operation, which is signaled with the AHOLD pin. The number of Address Bus pins to be floated will be determined by the value of the Cache Line Length register (BCR18, bits 15-11).

$\overline{\text{ADS}}$

Address Status ***Input/Output***

When driven LOW, this signal indicates that a valid bus cycle definition and address are available on the M/$\overline{\text{IO}}$, D/$\overline{\text{C}}$, W/$\overline{\text{R}}$ and A2–A31 pins of the local bus interface. At that time, the PCnet-32 controller will examine the combination of M/$\overline{\text{IO}}$, D/$\overline{\text{C}}$, W/$\overline{\text{R}}$, and the A2–A31 pins to determine if the current access is directed toward the PCnet-32 controller.

$\overline{\text{ADS}}$ will be driven LOW when the PCnet-32 controller performs a bus master access on the local bus.

AHOLD

Address Hold **Input**

This pin is always an input. The PCnet-32 controller will put some portion of the address bus into a high impedance state whenever this signal is asserted. AHOLD may be asserted by an external cache controller when a cache invalidation cycle is being performed. AHOLD may be asserted at any time, including times when the PCnet-32 controller is the active bus master. Note that this pin is multiplexed with a VESA VL function: $\overline{\text{LBS16}}$.

Some portion of the Address Bus will be floated at the time of an address hold operation, which is signaled with the AHOLD pin. The number of Address Bus pins to be floated will be determined by the value of the Cache Line Length (CLL) register (BCR18, bits 15-11) as shown in Table 14.

Table 14. CLL Value and Floating Address Pins

CLL Value	Floated Portion of Address Bus During AHOLD
00000	None
00001	A31–A2
00010	A31–A3
00011	Reserved CLL Value
00100	A31–A4
00101–00111	Reserved CLL Values
01000	A31–A5
01001–01111	Reserved CLL Values
10000	A31–A6
10001–11111	Reserved CLL Values

BCLK

Bus Clock ***Input***

Clock input that provides timing edges for all interface signals. This clock is used to drive the system bus interface and the internal buffer management unit. This clock is NOT used to drive the network functions.

$\overline{BE0}$–$\overline{BE3}$

Byte Enable ***Input/Output***

These signals indicate which bytes on the data bus are active during read and write cycles. When $\overline{BE3}$ is active, the byte on D31–D24 is valid. $\overline{BE2}$–$\overline{BE0}$ active indicate valid data on pins D23–D16, D15–D8, D7–D0, respectively. The byte enable signals are outputs for bus master and inputs for bus slave operations.

$\overline{BLAST}$

Burst Last Output

When the $\overline{BLAST}$ signal is asserted, then the next time that $\overline{BRDY}$ or $\overline{RDYRTN}$ is asserted, the burst cycle is complete.

$\overline{BOFF}$

Backoff ***Input***

$\overline{BOFF}$ is monitored as an input during Bus Master accesses. When PCnet-32 controller is current local bus master, it will float all appropriate bus mastering signals within 1 clock period of the assertion of $\overline{BOFF}$. When $\overline{BOFF}$ is de-asserted, PCnet-32 controller will restart any accesses that were suspended due to the assertion of $\overline{BOFF}$ and then will proceed with other scheduled accesses, if any. Register access cannot be performed to the PCnet-32 device while $\overline{BOFF}$ is asserted.

$\overline{BRDY}$

Burst Ready ***Input/Output***

$\overline{BRDY}$ functions as an input to the PCnet-32 controller during bus master cycles. When $\overline{BRDY}$ is asserted during a master cycle, it indicates to the PCnet-32 controller that the target device is accepting burst transfers. It also serves the same function as $\overline{RDYRTN}$ does for non-burst accesses. That is, it indicates that the target device has accepted the data on a master write cycle, or that the target device has presented valid data onto the bus during master read cycles.

If $\overline{BRDY}$ and $\overline{RDYRTN}$ are sampled active in the same cycle, then $\overline{RDYRTN}$ takes precedence, causing the next transfer cycle to begin with a T1 cycle.

$\overline{BRDY}$ functions as an output during PCnet-32 controller slave cycles and is always driven inactive (HIGH).

$\overline{BRDY}$ is floated if the PCnet-32 controller is not being accessed as the current slave device on the local bus.

D/$\overline{C}$

Data/Control Select ***Input/Output***

During slave accesses to the PCnet-32 controller, the D/$\overline{C}$ pin, along with M/$\overline{IO}$ and W/$\overline{R}$, indicates the type of cycle that is being performed. PCnet-32 controller will only respond to local bus accesses in which D/$\overline{C}$ is driven HIGH by the local bus master.

During PCnet-32 controller bus master accesses, the D/$\overline{C}$ pin is an output and will always be driven HIGH.

D/$\overline{C}$ is floated if the PCnet-32 controller is not the current master on the local bus.

D0–D31

Data Bus ***Input/Output***

Used to transfer data to and from the PCnet-32 controller to system resources via the local bus. D31–D0 are driven by the PCnet-32 controller when performing bus master writes and slave read operations. Data on D31–D0 is latched by the PCnet-32 controller when performing bus master reads and slave write operations.

The PCnet-32 controller will always follow Am386DX byte lane conventions. This means that for word and byte accesses in which PCnet-32 controller drives the data bus (i.e. master write operations and slave read operations) the PCnet-32 controller will produce duplicates of the active bytes on the unused half of the 32-bit data bus. Table 15 illustrates the cases in which duplicate bytes are created.

Table 15. Byte Duplication on Data Bus

$\overline{BE3}$-$\overline{BE0}$	DAT [31:24]	DAT [23:16]	DAT [15:8]	DAT [7:0]
1110	Undef	Undef	Undef	A
1101	Undef	Undef	A	Undef
1011	Undef	A	Undef	Copy A
0111	A	Undef	Copy A	Undef
1100	Undef	Undef	B	A
1001	Undef	C	B	Undef
0011*	D	C	Copy D	Copy C
1000	Undef	C	B	A
0001	D	C	B	Undef
0000	D	C	B	A

* ***Note:*** *Byte duplication does not apply during a $\overline{LBS16}$ access. See Table 8.*

$\overline{EADS}$

External Address Strobe ***Output***

During master write accesses in which Generate Cache Invalidation Cycles mode has been selected, the $\overline{EADS}$ pin will be asserted as part of the PCnet-32 controller cache invalidation cycle. Cache invalidation cycles will occur as often as a new cache line is reached. The cache line size can be set with the cache line length bits of BCR18 (bits [15:11]).

HLDA

Bus Hold Acknowledge ***Input***

PCnet-32 controller examines the HLDA signal to determine when it has been granted ownership of the bus. HLDA is active HIGH.

When HLDA is asserted and HOLD is being asserted by the PCnet-32 controller, the PCnet-32 controller assumes ownership of the local bus. However, if the PCnet-32 controller is asserting HOLD because HOLDI is asserted (as in a daisy chain arbitration), then PCnet-32 controller will assert HLDAO and will not assume ownership of the local bus.

Note that it changes polarity when the VL mode is selected (see pin description of $\overline{LGNT}$ in VESA VL-Bus Interface section).

HLDAO

Bus Hold Acknowledge Out ***Output***

This signal is multiplexed with the TCK pin, and is available only when the Multi-Interrupt mode has been selected with the JTAGSEL pin.

An additional local bus master may daisy-chain its HLDA signal through the PCnet-32 controller HLDAO pin. The PCnet-32 controller will deliver a HLDAO signal to the additional local bus master whenever the PCnet-32 controller receives a HLDA from the CPU, but is not simultaneously requesting the bus internally. The second local bus master must connect its HOLD output to the HOLDI input of the PCnet-32 controller in order to complete the local bus daisy-chain arbitration control.

When $\overline{SLEEP}$ is not asserted, daisy chain arbitration signals that pass through the PCnet-32 controller will experience a one-clock delay from input to output (i.e. HOLDI to HOLD and HLDA to HLDAO).

While $\overline{SLEEP}$ is asserted (either in **snooze** mode or **coma** mode), if the PCnet-32 controller is configured for a daisy chain (HOLDI and HLDAO signals have been selected with the JTAGSEL pin), then the system arbitration signal HLDA will be passed directly to the daisy-chain signal HLDAO without experiencing a one-clock delay. However, some combinatorial delay will be introduced in this path.

Note that this pin changes polarity when VL mode has been selected (see pin description of $\overline{LGNTO}$ in VESA VL-Bus Interface section).

HOLD

Bus Hold Request ***Output***

PCnet-32 controller asserts the HOLD pin as a signal that it wishes to become the local bus master. HOLD is active high. Once asserted, HOLD remains active until HLDA has become active, independent of subsequent assertion of $\overline{SLEEP}$ or setting of the STOP bit or access to the S_RESET port (offset 14h).

Note t hat this pin changes polarity when the VL mode is selected (see pin description of $\overline{LREQ}$ in VESA VL-Bus Interface section).

HOLDI

Bus Hold Request In ***Input***

This signal is multiplexed with the TDO pin, and is available only when the Multi-Interrupt mode has been selected with the JTAGSEL pin.

An additional local bus master may daisy-chain its bus hold request signal through the PCnet-32 controller HOLDI pin. The PCnet-32 controller will convey the HOLDI request to the CPU via the PCnet-32 controller HOLD output. The second local bus master must connect its HLDA input to the HLDAO output of the PCnet-32 controller in order to complete the local bus daisy-chain arbitration control.

When $\overline{SLEEP}$ is not asserted, daisy chain arbitration signals that pass through the PCnet-32 controller will

experience a one-clock delay from input to output (i.e. HOLDI to HOLD and HLDA to HLDAO).

While SLEEP is asserted (either in **snooze** mode or **coma** mode), if the PCnet-32 controller is configured for a daisy chain (HOLDI and HLDAO signals have been selected with the JTAGSEL pin), then the daisy-chain signal HOLDI will be passed directly to the system arbitration signal HOLD without experiencing a one-clock delay. However, some combinatorial delay will be introduced in this path.

If Multi-Interrupt mode has been selected and the daisy-chain arbitration feature is not used, then the HOLDI input should be tied to VSS through a resistor.

Note that this pin changes polarity when VL mode has been selected (see pin description of LREQI in VESA VL-Bus Interface section).

INTR1–INTR4

Interrupt Request ***Output***

An attention signal which indicates that one or more of the following status flags is set: BABL, MISS, MERR, RINT, IDON, MFCO, RCVCCO, TXSTRT, or JAB. Each of these status flags has a mask bit which allows for suppression of INTR assertion. These flags have the meaning shown in Table 16.

Table 16. Status Flags

BABL	Babble (CSR0, bit 14)
MISS	Missed Frame (CSR0, bit 12)
MERR	Memory Error (CSR0, bit 11)
RINT	Receive Interrupt (CSR0, bit 10)
IDON	Initialization Done (CSR0, bit 8)
MFCO	Missed Packet Count Overflow (CSR4, bit 9)
RCVCCO	Receive Collision Count Overflow (CSR4, bit 5)
TXSTRT	Transmit Start (CSR4, bit 3)
JAB	Jabber (CSR4, bit 1)

Note that there are four possible interrupt pins, depending upon the mode that has been selected with the JTAGSEL pin. Only one interrupt pin may be used at one time. The active interrupt pin is selected by programming the interrupt select register (BCR21). The default setting of BCR121will select interrupt INTR1 as the active interrupt. Note that BCR21 is EEPROM-programmable. Inactive interrupt pins are floated.

The polarity of the interrupt signal is determined by the INTLEVEL bit of BCR2. The interrupt pins may be programmed for level-sensitive or edge-sensitive operation.

PCnet-32 controller interrupt pins will be floated at H_RESET and will remain floated until either the EEPROM has been successfully read, or, following an EEPROM read failure, a Software Relocatable Mode sequence has been successfully executed.

LDEV

Local Device ***Output***

LDEV is driven by the PCnet-32 controller when it recognizes an access to PCnet-32 controller I/O space. Such recognition is dependent upon a valid sampled ADS strobe plus valid M/IO, D/C and A31–A5 values.

M/IO

Memory I/O Select ***Input/Output***

During slave accesses to the PCnet-32 controller, the M/IO pin, along with D/C and W/R, indicates the type of cycle that is being performed. PCnet-32 controller will only respond to local bus accesses in which M/IO is sampled as a zero by the PCnet-32 controller.

During PCnet-32 controller bus master accesses, the M/IO pin is an output and will always be driven high.

M/IO is floated if the PCnet-32 controller is not the current master on the local bus.

RESET

System Reset ***Input***

When RESET is asserted high and the LB/VESA pin has been tied to VDD, then the PCnet-32 controller performs an internal system reset of the type H_RESET (HARDWARE_ RESET). The RESET pin must be held for a minimum of 30 BCLK periods. While in the H_RESET state, the PCnet-32 controller will float or de-assert all outputs.

Note that this pin changes polarity when VL mode has been selected (see pin description of RESET in VESA VL-Bus Interface section).

RDY

Ready ***Output***

RDY functions as an output from the PCnet-32 controller during PCnet-32 controller slave cycles. During PCnet-32 controller slave read cycles, RDY is asserted to indicate that valid data has been presented on the data bus. During PCnet-32 controller slave write cycles, RDY is asserted to indicate that the data on the data bus has been internally latched. RDY is asserted for one BCLK period. RDY is then driven high for one-half of one clock period before being released.

RDY is floated if the PCnet-32 controller is not the current slave on the local bus.

In systems where both RDY and RDYRTN (or equivalent) signals are provided, RDY must NOT be tied to RDYRTN. Most systems now provide for a local device ready input to the memory controller that is separate from the CPU READY signal. This second READY signal is usually labeled as READYIN. This signal should be connected to the PCnet-32 controller RDY signal. The CPU READY signal should be connected to the PCnet-32 controller RDYRTN pin.

In systems where only one READY signal is provided, then RDY may be tied to RDYRTN.

RDYRTN

Ready Return ***Input***

RDYRTN functions as an input to the PCnet-32 controller. RDYRTN is used to terminate all master accesses performed by the PCnet-32 controller, except that linear burst transfers may also be terminated with the BRDY signal. RDYRTN is used to terminate slave read accesses to PCnet-32 controller I/O space.

When asserted during slave read accesses to PCnet-32 controller I/O space, RDYRTN indicates that the bus mastering device has seen the RDY that was generated by the PCnet-32 controller and has accepted the PCnet-32 controller slave read data. Therefore, PCnet-32 controller will hold slave read data on the bus until it synchronously samples the RDYRTN input as active low. The PCnet-32 controller will not hold RDY valid asserted during this time. The duration of the RDY pulse generated by the PCnet-32 controller will always be a single BCLK cycle.

RDYRTN is ignored during slave write accesses to PCnet-32 controller I/O space. Slave write accesses to PCnet-32 controller I/O space are considered terminated by the PCnet-32 controller at the end of the cycle during which the PCnet-32 controller issues an active RDY.

In systems where both a RDY and RDYRTN (or equivalent) signals are provided, then RDY must not be tied to RDYRTN. Most systems now provide for a local device ready input to the memory controller that is separate from the CPU READY signal. This second READY signal is usually labeled as READYIN. This signal should be connected to the PCnet-32 controller RDY signal. The CPU READY signal should be connected to the PCnet-32 controller RDYRTN pin.

In systems where only one READY signal is provided, then the PCnet-32 controller RDY output may be tied to the PCnet-32 controller RDYRTN input.

W/R

Write/Read Select Input/Output

During slave accesses to the PCnet-32 controller, the W/R pin, along with D/C and M/IO, indicates the type of cycle that is being performed.

During PCnet-32 controller bus master accesses, the W/R pin is an output.

W/R is floated if the PCnet-32 controller is not the current master on the local bus.

Board Interface

LED1

LED1 ***Output***

This pin is shared with the EESK function. When operating as LED1, the function and polarity on this pin are programmable through BCR5. The LED1 output from the PCnet-32 controller is capable of sinking the necessary 12 mA of current to drive an LED directly.

The LED1 pin is also used during EEPROM Autodetection to determine whether or not an EEPROM is present at the PCnet-32 controller microwire interface. At the trailing edge of RESET, this pin is sampled to determine the value of the EEDET bit in BCR19. A sampled HIGH value means that an EEPROM is present, and EEDET will be set to ONE. A sampled LOW value means that an EEPROM is not present, and EEDET will be set to ZERO. See the “EEPROM Autodetection” section for more details.

If no LED circuit is to be attached to this pin, then a pull-up or pull-down resistor must be attached instead, in order to resolve the EEDET setting.

LED2

LED2 ***Output***

This pin is shared with the SRDCLK function. When operating as LED2, the function and polarity on this pin are programmable through BCR6. The LED2 output from the PCnet-32 controller is capable of sinking the necessary 12 mA of current to drive an LED directly.

This pin also selects address width for Software Relocatable Mode. When this pin is HIGH during Software Relocatable Mode, then the device will be programmed to use 32 bits of addressing while snooping accesses on the bus during Software Relocatable Mode. When this pin is LOW during Software Relocatable Mode, then the device will be programmed to use 24 bits of addressing while snooping accesses on the bus during Software Relocatable Mode. The upper 8 bits of address will be assumed to match during the snooping operation when LED2 is LOW. The 24-bit addressing mode is intended for use in systems that employ the GPSI signals. For more information on the GPSI function see section General Purpose Serial Interface.

If no LED circuit is to be attached to this pin, then a pull up or pull down resistor must be attached instead, in order to resolve the Software Relocatable Mode address setting.

LEDPRE3

LEDPRE3 ***Output***

This pin is shared with the EEDO function. When operating as LEDPRE3, the function and polarity on this pin are programmable through BCR7. This signal is labeled as LED " PRE" 3 because of the multi-function nature of this pin. If an LED circuit were directly attached to this pin, it would create an IOL requirement that could not be met by the serial EEPROM that would also be attached to this pin. Therefore, if this pin is to be used as an additional LED output while an EEPROM is used in the system, then buffering is required between the LEDPRE3 pin and the LED circuit. If no EEPROM is included in the system design, then the LEDPRE3 signal may be directly connected to an LED without buffering. The LEDPRE3 output from the PCnet-32 controller is capable of sinking the necessary 12 mA of current to drive an LED in this case. For more details regarding LED connection, see the section on LEDs.

$\overline{\text{LNKST}}$

LINK Status ***Output***

This pin provides 12 mA for driving an LED. It indicates an active link connection on the 10BASE-T interface. The signal is programmable through BCR4. Note that this pin is multiplexed with the EEDI function.

This pin remains active in snooze mode.

SHFBUSY

Shift Busy ***Output***

The function of the SHFBUSY signal is to indicate when the last byte of the EEPROM contents has been shifted out of the EEPROM on the EEDO signal line. This information is useful for *external EEPROM-programmable registers* that do not use the microwire protocol, as is described herein: When the PCnet-32 controller is performing a serial read of the EEPROM through the microwire interface, the SHFBUSY signal will be driven HIGH. SHFBUSY can serve as a serial shift enable to allow the EEPROM data to be serially shifted into an external device or series of devices. The SHFBUSY signal will remain actively driven HIGH until the end of the EEPROM read operation. If the EEPROM checksum was verified, then the SHFBUSY signal will be driven LOW at the end of the EEPROM read operation. If the EEPROM checksum verification failed, then the SHFBUSY signal will remain HIGH. This function effectively demarcates the end of a successful EEPROM read operation and therefore is useful as a programmable-logic *low-active output enable* signal. For more details on external EEPROM-programmable registers, see the *EEPROM Microwire Access* section under *Hardware Access*.

This pin can be controlled by the host system by writing to BCR19, bit 3 (EBUSY).

$\overline{\text{SLEEP}}$

Sleep ***Input***

When $\overline{\text{SLEEP}}$ input is asserted (active LOW), the PCnet-32 controller performs an internal system reset and then proceeds into a power savings mode. (The reset operation caused by $\overline{\text{SLEEP}}$ assertion will not affect BCR registers.) All outputs will be placed in their normal reset condition. During sleep mode, all PCnet-32 controller inputs will be ignored except for the $\overline{\text{SLEEP}}$ pin itself. De-assertion of $\overline{\text{SLEEP}}$ results in wake-up. The system must refrain from starting the network operations of the PCnet-32 controller for 0.5 seconds following the de-assertion of the $\overline{\text{SLEEP}}$ signal in order to allow internal analog circuits to stabilize.

Both BCLK and XTAL1 inputs must have valid clock signals present in order for the $\overline{\text{SLEEP}}$ command to take effect.

If $\overline{\text{SLEEP}}$ is asserted while $\overline{\text{LREQ}}$/HOLD is asserted, then the PCnet-32 controller will perform an internal system reset and then wait for the assertion of $\overline{\text{LGNT}}$/HLDA. When $\overline{\text{LGNT}}$/HLDA is asserted, the $\overline{\text{LREQ}}$/HOLD signal will be de-asserted and then the PCnet-32 controller will proceed to the power savings mode. Note that the internal system reset will not cause the HOLD/$\overline{\text{LREQ}}$ signal to be de-asserted.

The $\overline{\text{SLEEP}}$ pin should not be asserted during power supply ramp-up. If it is desired that $\overline{\text{SLEEP}}$ be asserted at power up time, then the system must delay the assertion of $\overline{\text{SLEEP}}$ until three BCLK cycles after the completion of a valid pin $\overline{\text{RESET}}$ operation.

XTAL1–XTAL2

Crystal Oscillator Inputs ***Input/Output***

The crystal frequency determines the network data rate. The PCnet-32 controller supports the use of quartz crystals to generate a 20 MHz frequency compatible with the ISO 8802-3 (IEEE/ANSI 802.3) network frequency tolerance and jitter specifications. See the section *External Crystal Characteristics* (in section Manchester Encoder/ Decoder) for more detail.

The network data rate is one-half of the crystal frequency. XTAL1 may alternatively be driven using an external CMOS level source, in which case XTAL2 must be left unconnected. Note that when the PCnet-32 controller is in coma mode, there is an internal 22 KW resistor from XTAL1 to ground. If an external source drives XTAL1, some power will be consumed driving this resistor. If XTAL1 is driven LOW at this time power consumption will be minimized. In this case, XTAL1 must remain active for at least 30 cycles after the assertion of $\overline{\text{SLEEP}}$ and de-assertion of HOLD.

Microwire EEPROM Interface

EESK

EEPROM Serial Clock ***Output***

The EESK signal is used to access the external ISO 8802-3 (IEEE/ANSI 802.3) address PROM. This pin is designed to directly interface to a serial EEPROM that uses the microwire interface protocol. EESK is connected to the microwire EEPROM's Clock pin. It is controlled by either the PCnet-32 controller directly during a read of the entire EEPROM, or indirectly by the host system by writing to BCR19, bit 1.

EESK can be used during programming of *external EEPROM-programmable registers* that do not use the microwire protocol as follows:

When the PCnet-32 controller is performing a serial read of the IEEE Address EEPROM through the microwire interface, the SHFBUSY signal will serve as a serial shift enable to allow the EEPROM data to be serially shifted into an external device or series of devices. This same signal can be used to gate the *output* of the programmed logic to avoid the problem of releasing intermediate values to the rest of the system board logic. The EESK signal can serve as the clock, and EEDO will serve as the input data stream to the programmable shift register.

EEDO

EEPROM Data Out ***Input***

The EEDO signal is used to access the external ISO 8802-3 (IEEE/ANSI 802.3) address PROM. This pin is designed to directly interface to a serial EEPROM that uses the microwire interface protocol. EEDO is connected to the microwire EEPROM's Data Output pin. It is controlled by the EEPROM during reads. It may be read by the host system by reading BCR19, bit 0.

EEDO can be used during programming of *external EEPROM-programmable registers* that do not use the microwire protocol as follows:

When the PCnet-32 controller is performing a serial read of the IEEE Address EEPROM through the microwire interface, the SHFBUSY signal will serve as a serial shift enable to allow the EEPROM data to be serially shifted into an external device or series of devices. This same signal can be used to gate the *output* of the programmed logic to avoid the problem of releasing intermediate values to the rest of the system board logic. The EESK signal can serve as the clock, and EEDO will serve as the input data stream to the programmable shift register.

EECS

EEPROM Chip Select ***Output***

The function of the EECS signal is to indicate to the microwire EEPROM device that it is being accessed. The EECS signal is active high. It is controlled by either the PCnet-32 controller during a read of the entire EEPROM, or indirectly by the host system by writing to BCR19, bit 2.

EEDI

EEPROM Data In ***Output***

The EEDI signal is used to access the external ISO 8802-3 (IEEE/ANSI 802.3) address PROM. EEDI functions as an output. This pin is designed to directly interface to a serial EEPROM that uses the microwire interface protocol. EEDI is connected to the microwire EEPROM's Data Input pin. It is controlled by either the PCnet-32 controller during command portions of a read of the entire EEPROM, or indirectly by the host system by writing to BCR19, bit 0.

Attachment Unit Interface

CI±

Collision In ***Input***

A differential input pair signaling the PCnet-32 controller that a collision has been detected on the network media, indicated by the CI± inputs being driven with a 10 MHz pattern of sufficient amplitude and pulse width to meet ISO 8802-3 (IEEE/ANSI 802.3) standards. Operates at pseudo ECL levels.

DI±

Data In ***Input***

A differential input pair to the PCnet-32 controller carrying Manchester encoded data from the network. Operates at pseudo ECL levels.

DO±

Data Out Output

A differential output pair from the PCnet-32 controller for transmitting Manchester encoded data to the network. Operates at pseudo ECL levels.

Twisted Pair Interface

RXD±

10BASE-T Receive Data ***Input***

10BASE-T port differential receivers.

TXD±

10BASE-T Transmit Data ***Output***

10BASE-T port differential drivers.

TXP±

10BASE-T Pre-distortion Control ***Output***

These outputs provide transmit predistortion control in conjunction with the 10BASE-T port differential drivers.

External Address Detection Interface

The EADI interface is enabled through BCR2, bit 3 (EADISEL).

$\overline{\text{EAR}}$

External Address Reject Low Input

An EADI input signal. The incoming frame will be checked against the internally active address detection mechanisms and the result of this check will be OR'd with the value on the $\overline{\text{EAR}}$ pin. The $\overline{\text{EAR}}$ pin is defined as $\overline{\text{REJECT}}$.

See the EADI section for details regarding the function and timing of this signal.

Note that this pin is multiplexed with the INTR2 pin.

SFBD

Start Frame–Byte Delimiter ***Output***

Start Frame–Byte Delimiter Enable. EADI output signal. An initial rising edge on this signal indicates that a start of frame delimiter has been detected. The serial bit stream will follow on the SRD signal, commencing with the destination address field. SFBD will go high for 4 bit times (400 ns) after detecting the second "1" in the SFD (Start of Frame Delimiter) of a received frame. SFBD will subsequently toggle every 400 ns (1.25 MHz frequency) with each rising edge indicating the first bit of each subsequent byte of the received serial bit stream. SFBD will be inactive during frame transmission.

Note that this pin is multiplexed with the LED1 pin.

SRD

Serial Receive Data ***Output***

An EADI output signal. SRD is the decoded NRZ data from the network. This signal can be used for external address detection. Note that when the 10BASE-T port is selected, transitions on SRD will only occur during receive activity. When the AUI port is selected, transitions on SRD will occur during both transmit and receive activity.

Note that this pin is multiplexed with the LEDPRE3 pin.

SRDCLK

Serial Receive Data Clock ***Output***

An EADI output signal. Serial Receive Data is synchronous with reference to SRDCLK. Note that when the 10BASE-T port is selected, transitions on SRDCLK will only occur during receive activity. When the AUI port is selected, transitions on SRDCLK will occur during both transmit and receive activity.

Note that this pin is multiplexed with the LED2 pin.

General Purpose Serial Interface

The GPSI interface is selected through the PORTSEL bits of the Mode register (CSR15) and enabled through the TSTSHDW[1] bit (BCR18) or the GPSIEN bit (CSR124).

Note that when GPSI test mode is invoked, slave address decoding must be restricted to the lower 24 bits of the address bus by setting the IOAW24 bit in BCR2 and by pulling LED2 LOW during Software Reloactable Mode. The upper 8 bits of the address bus will always be considered matched when examining incoming I/O addresses. During master accesses while in GPSI mode, the PCnet-32 controller will not drive the upper 8 bits of the address bus with address information. See the GPSI section for more detail.

TXDAT

Transmit Data ***Input/Output***

TXDAT is an output, providing the serial bit stream for transmission, including preamble, SFD data and FCS field, if applicable.

Note that the TxDAT pin is multiplexed with the A31 pin.

TXEN

Transmit Enable ***Input/Output***

TXEN is an output, providing an enable signal for transmission. Data on the TXDAT pin is not valid unless the TXEN signal is HIGH.

Note that the TXEN pin is multiplexed with the A30 pin.

STDCLK

Serial Transmit Data Clock ***Input***

STDCLK is an input, providing a clock signal for MAC activity, both transmit and receive. Rising edges of the STDCLK can be used to validate TXDAT output data.

The STDCLK pin is multiplexed with the A29 pin.

Note that this signal must meet the frequency stability requirement of the ISO 8802-3 (IEEE/ANSI 802.3) specification for the crystal.

CLSN

Collision ***Input/Output***

CLSN is an input, indicating to the core logic that a collision has occurred on the network.

Note that the CLSN pin is multiplexed with the A28 pin.

RXCRS

Receive Carrier Sense ***Input/Output***

RXCRS is an input. When this signal is HIGH, it indicates to the core logic that the data on the RXDAT input pin is valid.

Note that the RXCRS pin is multiplexed with the A27 pin.

SRDCLK

Serial Receive Data Clock ***Input/Output***

SRDCLK is an input. Rising edges of the SRDCLK signal are used to sample the data on the RXDAT input whenever the RXCRS input is HIGH.

Note that the SRDCLK pin is multiplexed with the A26 pin.

RXDAT

Receive Data ***Input/Output***

RXDAT is an input. Rising edges of the SRDCLK signal are used to sample the data on the RXDAT input whenever the RXCRS input is HIGH.

Note that the RXDAT pin is multiplexed with the A25 pin.

IEEE 1149.1 Test Access Port Interface

TCK

Test Clock ***Input***

The clock input for the boundary scan test mode operation. TCK can operate up to 10 MHz. If left unconnected this pin has a default value of HIGH.

TDI

Test Data Input ***Input***

The test data input path to the PCnet-32 controller. If left unconnected, this pin has a default value of HIGH.

TDO

Test Data Output ***Output***

The test data output path from the PCnet-32 controller. TDO is floated when the JTAG port is inactive.

TMS

Test Mode Select ***Input***

A serial input bit stream is used to define the specific boundary scan test to be executed. If left unconnected, this pin has a default value of HIGH.

Power Supply Pins

AVDD

Analog Power (4 Pins) ***Power***

There are four analog +5 V supply pins. Special attention should be paid to the printed circuit board layout to avoid excessive noise on these lines. Refer to Appendix B and the *PCnet Family Technical Manual (PID# 18216A)* for details.

AVSS

Analog Ground (2 Pins) ***Power***

There are two analog ground pins. Special attention should be paid to the printed circuit board layout to avoid excessive noise on these lines. Refer to Appendix B and to the *PCnet Family Technical Manual* for details.

DVDD

Digital Power (3 Pins) ***Power***

There are 3 digital power supply pins. (DVDD1, DVDD2, and DVDD3) used by the internal circuitry.

DVDDCLK

Digital Power Clock (1 Pin) ***Power***

This pin is used to supply power to the clock buffering circuitry.

DVDDO

I/O Buffer Digital Power (5 Pins) ***Power***

There are 5 digital power supply pins (DVDD01–DVDD05) used by Input/Output buffer drivers.

DVSS

Digital Ground (4 Pins) ***Ground***

There are 4 digital ground pins (DVSS1–DVSS4) used by the internal digital circuitry.

DVSSCLK

Digital Ground Clock (1 Pin) ***Ground***

This pin is used to supply a ground to the clock buffering circuitry.

DVSSN

I/O Buffer Digital Ground (15 Pins) ***Ground***

These 15 ground pins (DVSSN1–DVSSN15) are used by the Input/Output buffer drivers.

DVSSPAD

Digital Ground Pad (1 Pin) ***Ground***

This pin is used by the Input/Output logic circuits.

BASIC FUNCTIONS

System Bus Interface Function

The PCnet-32 controller is designed to operate as a Bus Master during normal operations. Some slave accesses to the PCnet-32 controller are required in normal operations as well. Initialization of the PCnet-32 controller is achieved through a combination of Bus Slave accesses, Bus Master accesses and an optional read of a serial EEPROM that is performed by the PCnet-32 controller. The EEPROM read operation is performed through the microwire interface. The ISO 8802-3 (IEEE/ANSI 802.3) Ethernet Address may reside within the serial EEPROM. Some PCnet-32 controller configuration registers may also be programmed by the EEPROM read operation.

The address PROM, on-chip board-configuration registers, and the Ethernet controller registers occupy 32 bytes of I/O space which can be located by modifying the I/O base address register. The default base address can be written to the EEPROM. The PCnet-32 controller will automatically read the I/O Base Address from the EEPROM after H_RESET, or at any time that the software requests that the EEPROM should be read. When no EEPROM is attached to the serial microwire interface, the PCnet-32 controller detects the condition, and enters Software Relocatable Mode. While in Software Relocatable Mode, the PCnet-32 controller will not respond to any bus accesses, but will snoop the bus for accesses to I/O address 378h. When a successfully executed and uninterrupted sequence of write operations is seen at this location, the PCnet-32 controller will accept the next sequence of accesses as carrying I/O Base Address relocation and interrupt pin programming information. After this point, the PCnet-32 controller will begin to respond directly to accesses directed toward offsets from the newly loaded I/O Base Address. This scheme allows for jumperless relocatable I/O implementations.

Software Interface

The software interface to the PCnet-32 controller is divided into two parts. One part is the direct access to the I/O resources of the PCnet-32 controller. The PCnet-32 controller occupies 32 bytes of I/O space that must begin on a 32-byte block boundary. The I/O Base Address can be changed to any 32-bit quantity that begins on a 32-byte block boundary through the function of the Software Relocatable Mode. It can also be changed to any 32-bit value that begins on a 32-byte block boundary through the automatic EEPROM read operation that occurs immediately after the H_RESET function has completed. This read operation automatically alters the I/O Base Address of the PCnet-32 controller.

The 32-byte I/O space is used by the software to program the PCnet-32 controller operating mode, to enable and disable various features, to monitor operating status and to request particular functions to be executed by the PCnet-32 controller.

The other portion of the software interface is the descriptor and buffer areas that are shared between the software and the PCnet-32 controller during normal network operations. The descriptor area boundaries are set by the software and do not change during normal network operations. There is one descriptor area for receive activity and there is a separate area for transmit activity. The descriptor space contains relocatable pointers to the network packet data and it is used to transfer packet status from the PCnet-32 controller to the software. The buffer areas are locations that hold packet data for transmission or that accept packet data that has been received.

Network Interfaces

The PCnet-32 controller can be connected to an 802.3 network via one of two network interfaces. The Attachment Unit Interface (AUI) provides an ISO 8802-3 (IEEE/ANSI 802.3) compliant differential interface to a remote MAU or an on-board transceiver. The 10BASE-T interface provides a twisted-pair Ethernet port. While in auto-selection mode, the interface in use is determined by an auto-sensing mechanism which checks the link status on the 10BASE-T port. If there is no active link status, then the device assumes an AUI connection.

DETAILED FUNCTIONS

Bus Interface Unit

The bus interface unit is built of several state machines that run synchronously to BCLK. One bus interface unit state machine handles accesses where the PCnet-32 controller is the bus slave, and another handles accesses where the PCnet-32 controller is the bus master. All inputs are synchronously sampled *except* $\overline{ADS}$, M/$\overline{IO}$, D/$\overline{C}$, W/$\overline{R}$ and the A[31:5] bus when this bus is an input to the PCnet-32 controller. All outputs are synchronously generated on the rising edge of BCLK *with the following exceptions*:

$\overline{LDEV}$ is generated asynchronously.

RDY is *driven/floated* on falling edges of BCLK, but will change state on rising edges of BCLK.

The following sections describe the various bus master and bus slave operations that will be performed by the PCnet-32 controller. The timing diagrams that are included in these sections (*Bus Acquisition* section through *Slave Timing* section) show the signals and timings of the Am486 32-bit mode of operation. The sections from Bus Acquisition through Linear Burst DMA Transfers show bus master operations. The *Slave Timing* section shows bus slave operations. Note that the PCnet-32 controller operation in Am486 32-bit mode represents a merger of the requirements of the VESA VL-Bus specification and Am486 bus specification, whichever is more stringent. The concepts discussed in the following sections and the basic nature of the timings shown is applicable in a general sense to PCnet-32 controller operational modes. For specific differences in timing between modes and for examples of timing diagrams showing basic transfers in each of the modes, please consult the following sections:

Am486 32-bit mode:	*Bus Acquisition* section through *Slave Timing* section
VESA VL-Bus mode:	VESA VL-Bus Mode Timing section

For selection of each mode, consult the section on Configuration Pins in the *Pin Description* section.

Note that all timing diagrams in this document have been drawn showing reference to two groups of address pins: namely, A4–A31 and A2–A3, $\overline{BE0}$–$\overline{BE3}$. In the AHOLD timing diagrams, the two groups are shown separately, because the upper address pins become floated, while the lower address pins do not. The point of division between the two groups of address pins will depend upon the value of CLL in BCR18. In the case of Linear Burst operations, the upper address pins are shown separately because that group does not change its value through a single linear burst, while the lower address pins do change value. In this case, the point of division between the two groups of address pins will depend upon the value of LINBC in BCR18. In all other timing diagrams, the two groups are shown separately just to maintain consistency with the AHOLD and Linear Burst timing diagrams. For more details, see the AHOLD and Linear Burst Count sections.

Bus Acquisition

The PCnet-32 controller microcode (in the buffer management section) will determine when a DMA transfer should be initiated. The first step in any PCnet-32 controller bus master transfer is to acquire ownership of the bus. This task is handled by synchronous logic within the BIU. Bus ownership is requested with the HOLD signal and ownership is granted by the CPU (or an arbiter) through the HLDA signal. The PCnet-32 controller additionally supplies HOLDI and HLDAO signals to allow daisy chaining of devices through the PCnet-32 controller. Priority of the HOLDI input versus the PCnet-32 controller's own internal request for bus mastership can be set using the PRPCNET bit of BCR17. Simple bus arbitration (HOLD, HLDA only) is shown in Figure 1.

Note that assertion of the STOP bit will **not** cause a de-assertion of the HOLD signal. Note also that a read of the S_RESET register (I/O resource at offset 14h from the PCnet-32 controller base I/O address) will **not** cause a de-assertion of the HOLD signal. Either of these actions will cause the internal master state machine logic to cease operations, but the HOLD signal will remain active until the HLDA signal is synchronously sampled as asserted. Following either of the above actions, on the next clock cycle after the HLDA signal is synchronously sampled as asserted, the PCnet-32 controller will de-assert the HOLD signal. No bus master accesses will have been performed during this brief bus ownership period.

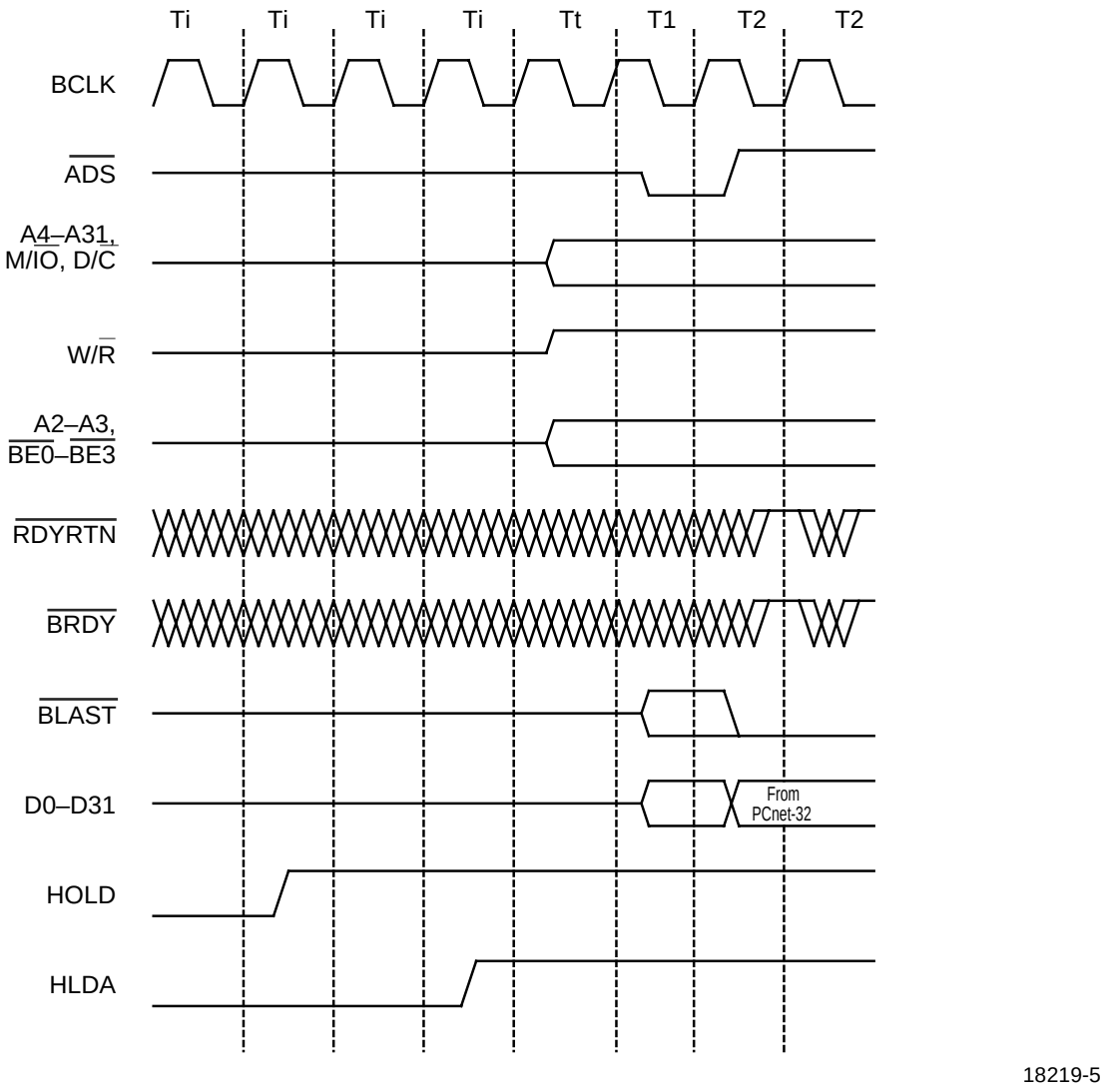


Figure 1. Bus Acquisition

Assertion of a minimum-width pulse on the RESET pin will cause the HOLD signal to de-assert within six clock cycles following the assertion of the RESET pin. In this case, the PCnet-32 controller will not wait for the assertion of the HLDA signal before de-asserting the HOLD signal.

This description of behavior is identical for the VESA VL-Bus mode of operation, except that the HOLD, HLDA and RESET signals are replaced by the inverted sense signals LREQ, LGNT, and RESET, respectively; the BCLK, BOFF, EADS, and RDY signals are replaced by LCLK, WBACK, LEADS, and LRDY, respectively.

Bus Master DMA Transfers

There are three primary types of DMA transfers. All DMA transfers will have wait states inserted until either RDYRTN or BRDY is asserted. Figure 2 and Figure 3 show the basic bus transfers of read and write without ready wait states and then with ready wait states.

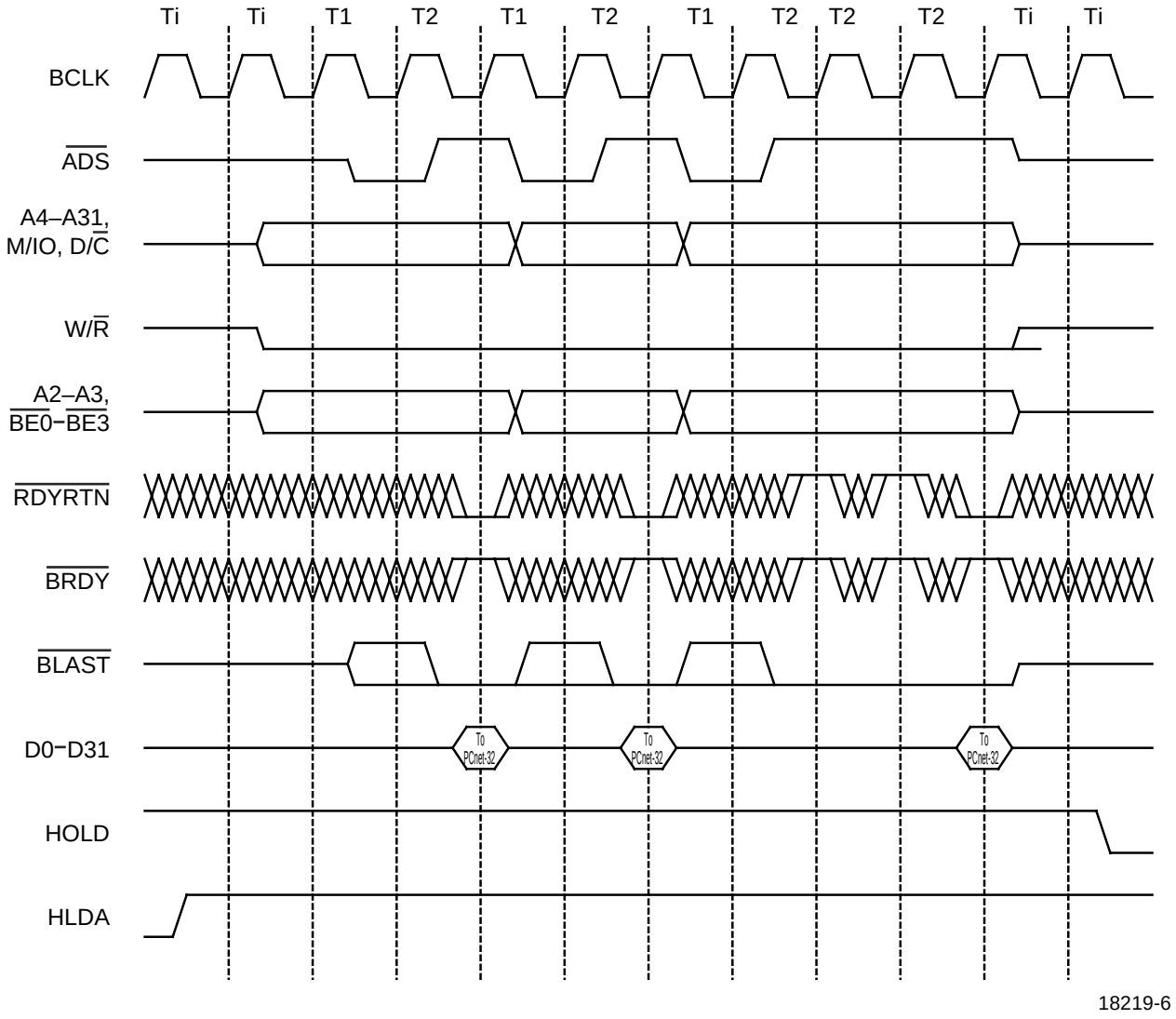


Figure 2. Basic Read Cycles without and with Wait States

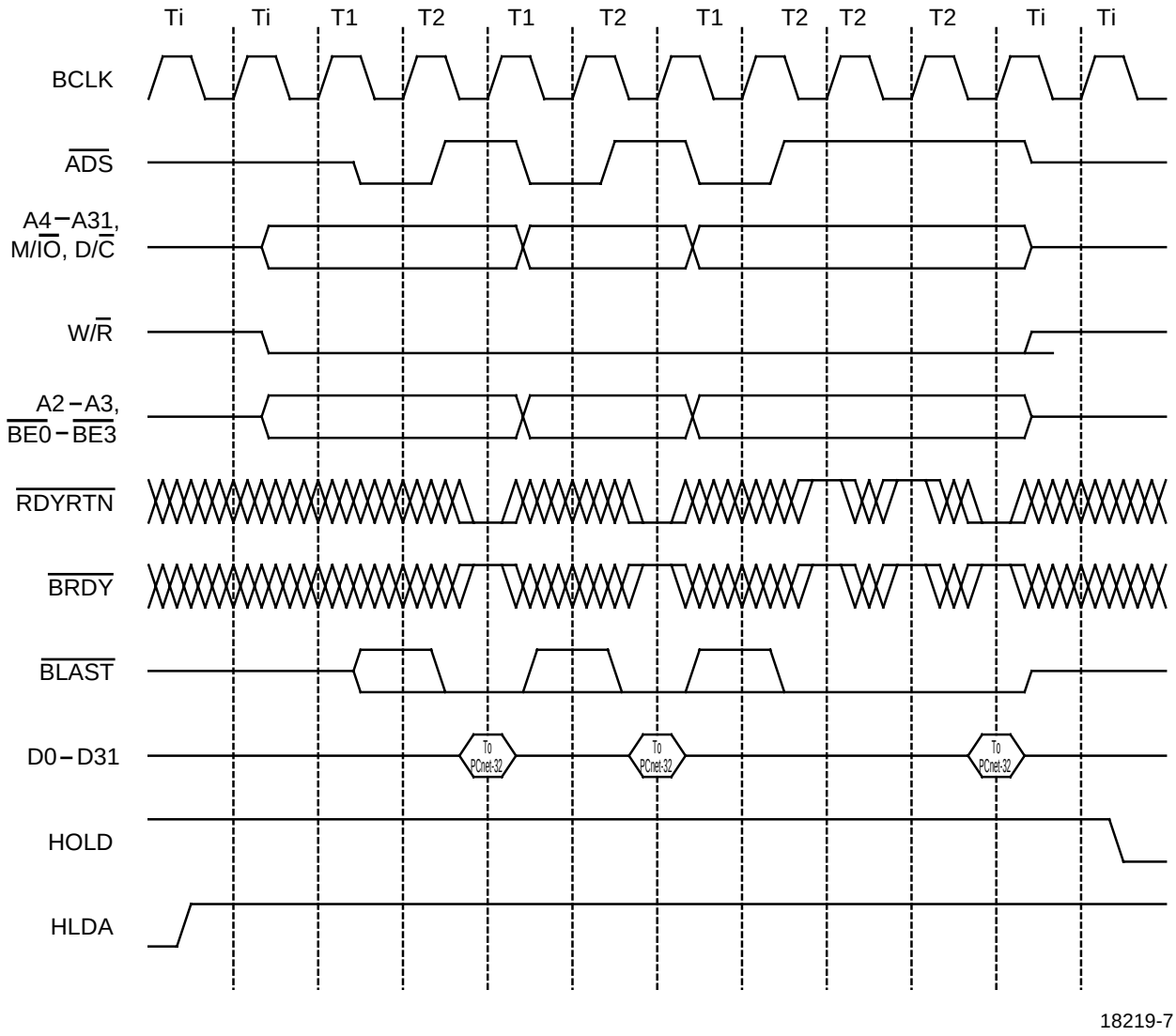


Figure 3. Basic Write Cycles without and with Wait States

Effect of $\overline{BOFF}$

Assertion of $\overline{BOFF}$ during bus master transfers will cause the PCnet-32 controller to float all of its bus signals beginning at the next clock cycle. Any access which has been interrupted by $\overline{BOFF}$ will be restarted when the $\overline{BOFF}$ signal is de-asserted. Simultaneous assertion of $\overline{RDYRTN}$ (or $\overline{BRDY}$) and $\overline{BOFF}$ is resolved in favor of $\overline{BOFF}$. The $\overline{RDYRTN}$ (or $\overline{BRDY}$) is ignored for such a cycle, and when $\overline{BOFF}$ is de-asserted, the cycle is restarted. See Figure 4 for detail.

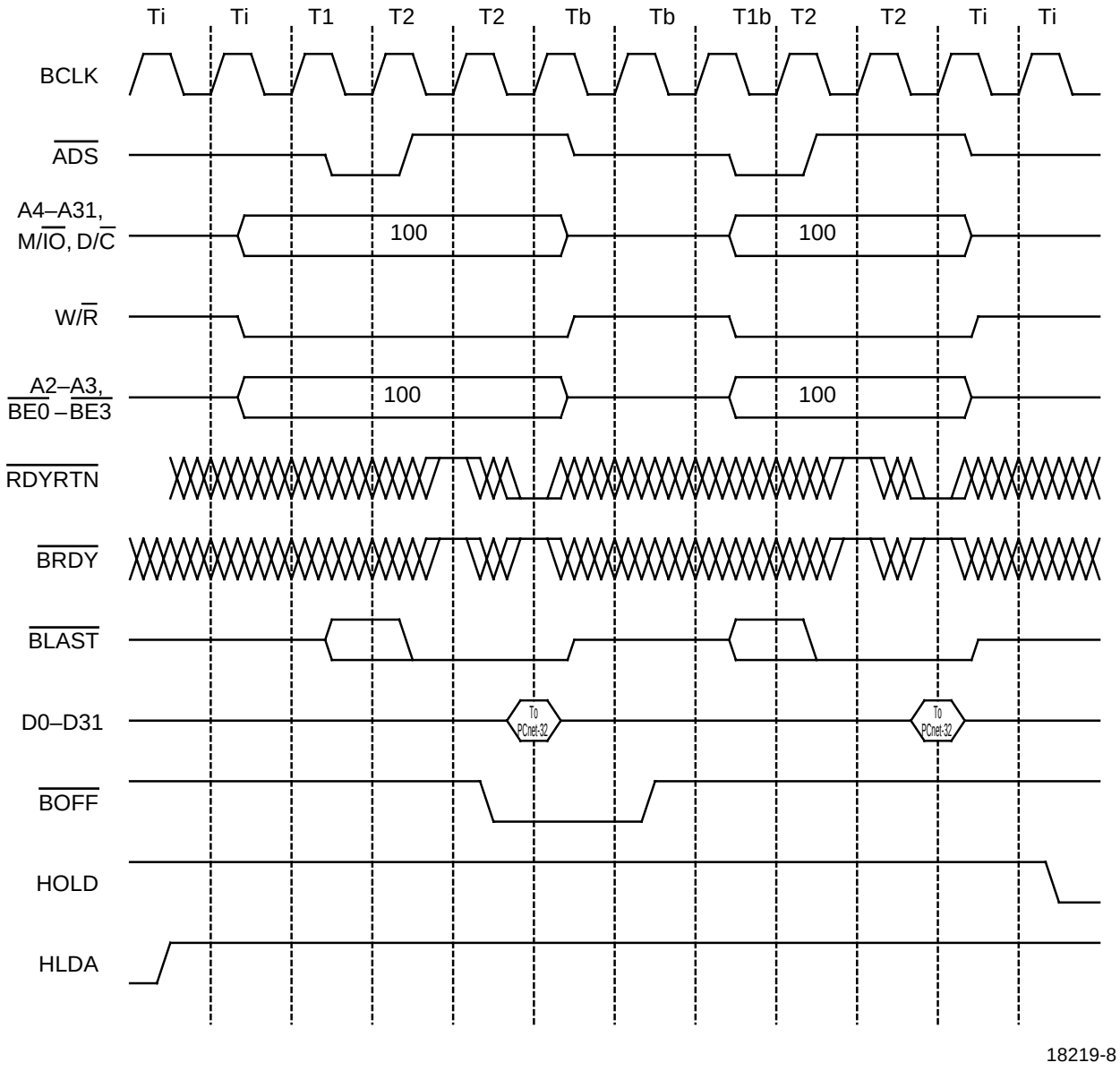


Figure 4. Restarted Read Cycle

Effect of AHOLD

Assertion of AHOLD during bus master transfers will cause the PCnet-32 controller to float some portion of the address bus beginning at the next clock cycle. If $\overline{\text{RDYRTN}}$ is returned while AHOLD is active, then the cycle completes, since the data bus may remain active during AHOLD. However, a new cycle will not be started while AHOLD is active.

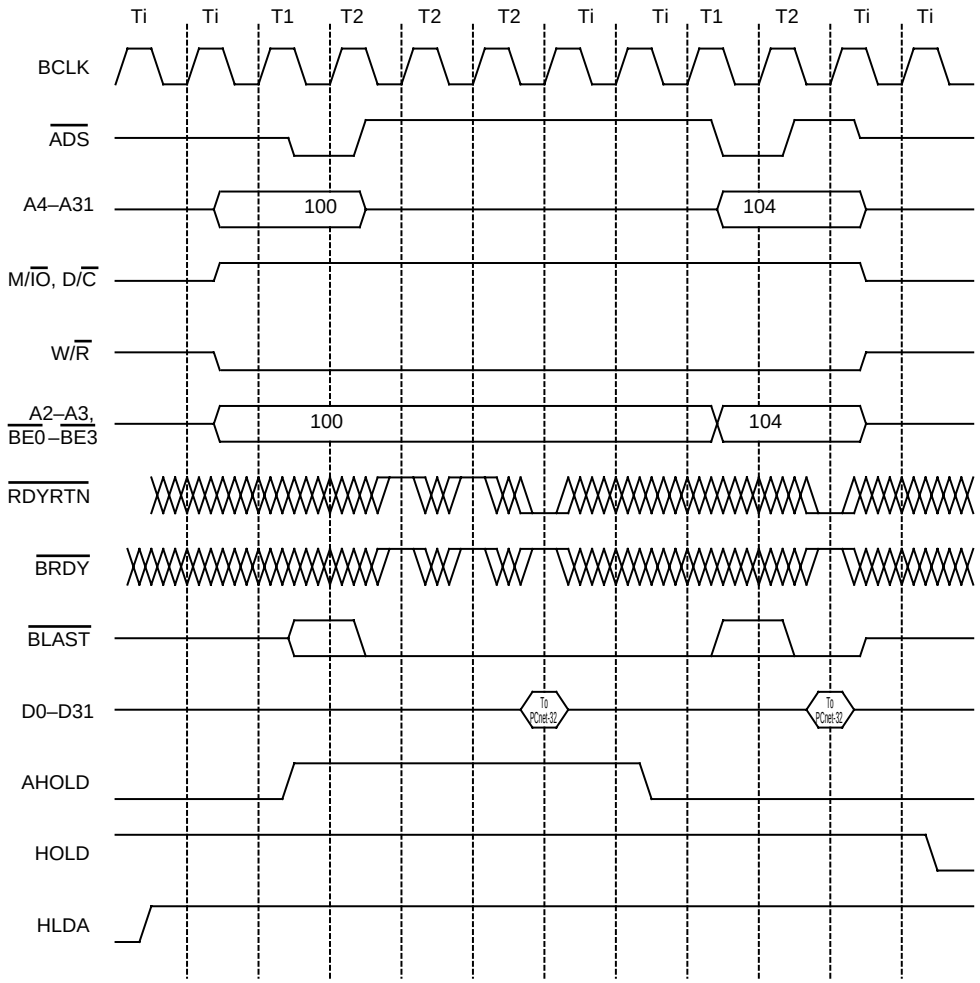
The portion of the Address Bus that will be floated at the time of an address hold operation will be determined by the value of the Cache Line Length register (BCR18, bits 15-11). Table 17 lists all of the legal values of CLL showing the portion of the Address Bus that will become floated during an address hold operation.

If the $\overline{\text{RDYRTN}}$ signal is not returned while AHOLD is active, then the PCnet-32 controller will resume driving the same address onto the address bus when AHOLD is released. The PCnet-32 controller will not reissue the $\overline{\text{ADS}}$ signal at this time. See Figure 5 and Figure 6 for details.

Table 17. CLL Value and Floating Address Pins

CLL Value	Floated Portion of Address Bus During AHOLD
00000	None
00001	A31–A2
00010	A31–A3
00011	Reserved CLL Value
00100	A31–A4
00101–00111	Reserved CLL Values
01000	A31–A5
01001–01111	Reserved CLL Values
10000	A31–A6
10001–11111	Reserved CLL Values

Note: *The default value of CLL after H_RESET is 00100. All timing diagrams in this document are drawn with the assumption that this is the value of CLL.*



18218-9

Figure 5. Read Cycle with AHOLD

Note that initial access is allowed to complete in spite of AHOLD, but next access is prevented from beginning until AHOLD is de-asserted.

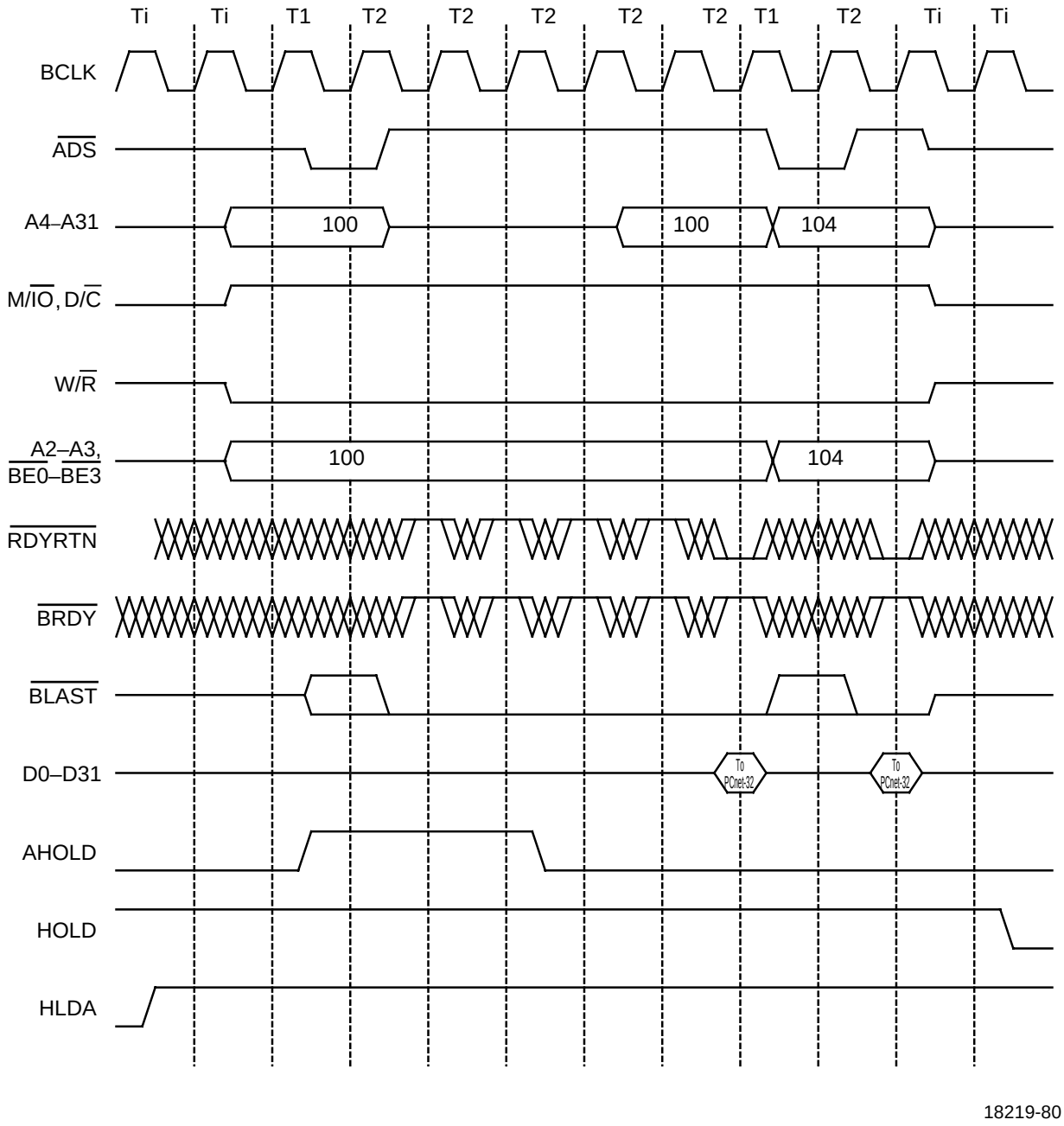


Figure 6. Read Cycle with AHOLD

Address is re-driven when AHOLD is de-asserted, since RDYRTN had not yet arrived.

Effect of Bus Preemption

If a bus preemption event occurs during a basic transfer cycle, then the behavior of the PCnet-32 controller will depend upon which specific type of access is being performed. The general response of the PCnet-32 controller is that the current operation will complete before the PCnet-32 controller relinquishes the bus in response to the preemption. “Current operation” in this sense refers to the general PCnet-32 controller operation, such as “descriptor access”. Note that a “descriptor access” consists of one or two basic transfers. Therefore, both transfers of a descriptor access must be completed before the bus will be released in response to a preemption. See each of the sections for Initialization Block DMA transfers, Descriptor DMA transfers, FIFO DMA transfers and Linear Burst Transfers for more specific information.

Effect of $\overline{\text{LBS16}}$ (VL-Bus mode only)

Dynamic bus sizing is recognized by the PCnet-32 controller while operating in the VL-Bus mode. The $\overline{\text{LBS16}}$ signal is used to indicate to the PCnet-32 controller whether the VL-Bus target is a 16-bit or 32-bit peripheral. When the target device indicates that it is 16 bits in width by asserting the $\overline{\text{LBS16}}$ signal at least one LCLK period before asserting the $\overline{\text{RDYRTN}}$ signal, then the PCnet-32 controller will dynamically respond to the size constraints of the peripheral by performing additional accesses. Table 18 shows the sequence of accesses that will be performed by the PCnet-32 controller in response to the assertion of $\overline{\text{LBS16}}$.

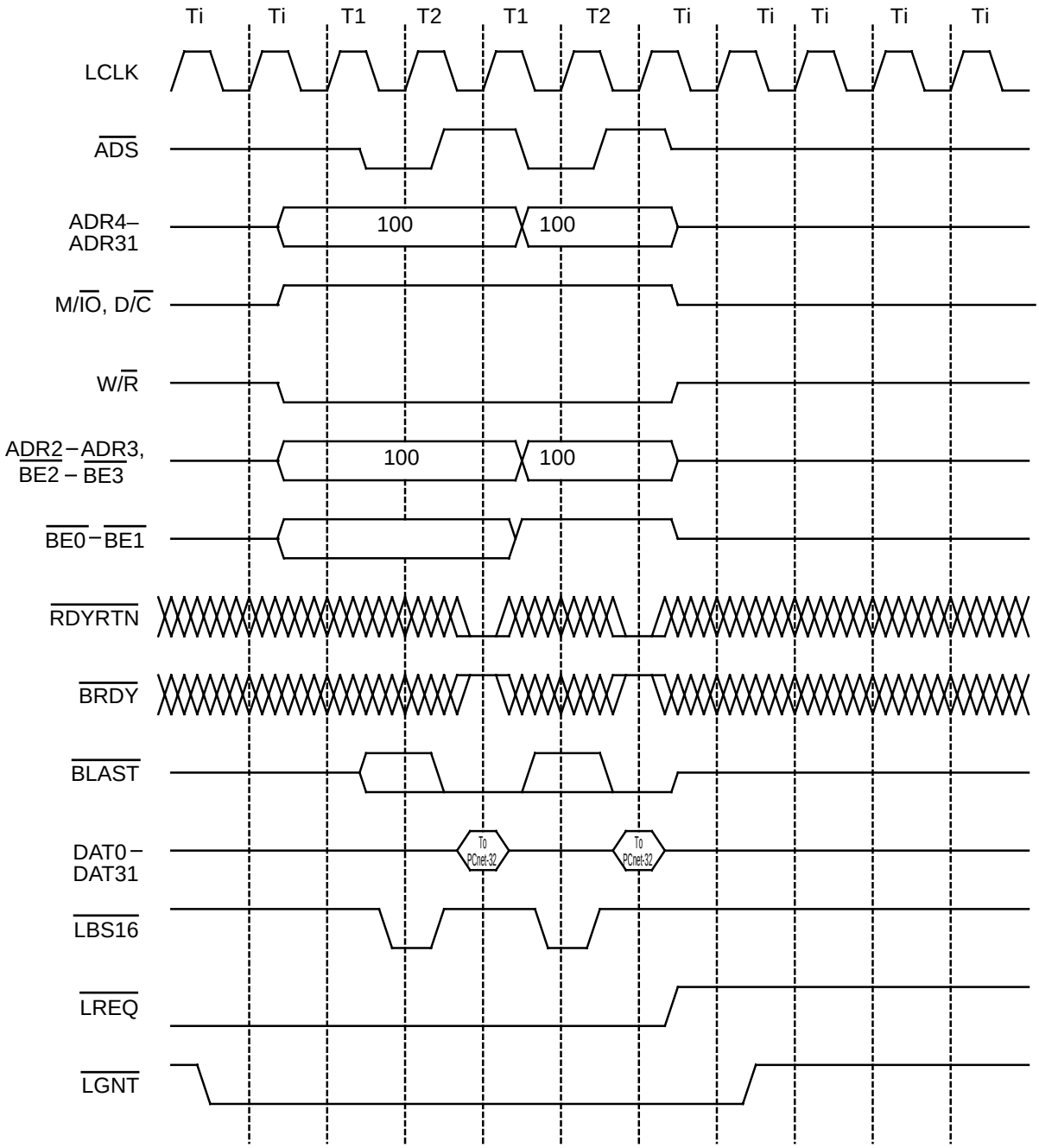
Figure 7 shows an example of an exchange between a 16-bit VL-Bus peripheral and the PCnet-32 controller. Note that the $\overline{\text{LBS16}}$ signal is asserted during the LCLK that precedes the assertion of $\overline{\text{RDYRTN}}$. In this particular case, in order to maintain zero-wait state accesses, the 16-bit target must generate $\overline{\text{LBS16}}$ in a very short time in order to meet the required setup time of $\overline{\text{LBS16}}$ into the PCnet-32 controller. If the peripheral were incapable of meeting the required setup time, then a wait state would be needed in order to insure that $\overline{\text{LBS16}}$ is asserted at least one LCLK prior to the assertion of the $\overline{\text{RDYRTN}}$ signal.

When the assertion of $\overline{\text{LBS16}}$ during a PCnet-32 controller master access has created the need for a second access as specified in the table above, **and** the $\overline{\text{WBACK}}$ signal becomes active during the second access, then, when $\overline{\text{WBACK}}$ is de-asserted, the PCnet-32 controller will repeat both accesses of the pair.

Table 18. Data Transfer Sequence from 32-Bit Wide to 16-Bit Wide

Current Access				Next with $\overline{\text{LBS16}}$			
$\overline{\text{BE3}}$	$\overline{\text{BE2}}$	$\overline{\text{BE1}}$	$\overline{\text{BE0}}$	$\overline{\text{BE3}}$	$\overline{\text{BE2}}$	$\overline{\text{BE1}}$	$\overline{\text{BE0}}$
1	1	1	0	NR*			
1	1	0	0	NR*			
1	0	0	0	1	0	1	1
0	0	0	0	0	0	1	1
1	1	0	1	NR*			
1	0	0	1	1	0	1	1
0	0	0	1	0	0	1	1
1	0	1	1	NR*			
0	0	1	1	NR*			
0	1	1	1	NR*			

**NR = No second access Required for these cases*



18219-10

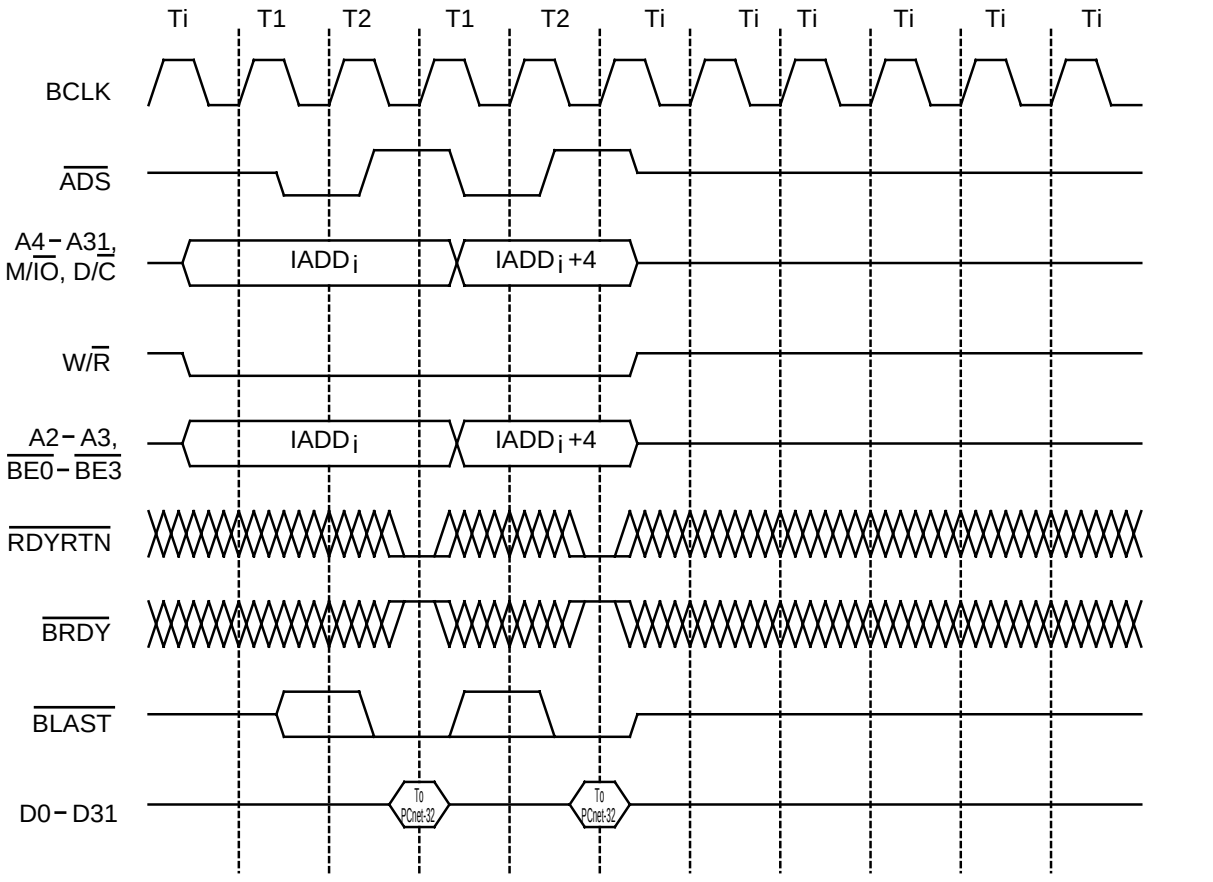
Figure 7. VL-BUS Ready Cycle with $\overline{\text{LBS16}}$ Asserted
Four-byte single cycle access is converted to two two-byte accesses.

Initialization Block DMA Transfers

During execution of the PCnet-32 controller bus master initialization procedure, the PCnet-32 controller microcode will repeatedly request DMA transfers from the BIU. During each of these initialization block DMA transfers, the BIU will perform two data transfer cycles (eight bytes) and then it will relinquish the bus (see Figure 8). The two transfers within the mastership period will always be read cycles to ascending contiguous addresses. The two transfers in each initialization block DMA transfer will never be executed using linear burst mode. In 32-bit software mode, the number of bus mastership periods needed to complete the initialization procedure is 4. There are 7 double-words to transfer during the bus master initialization procedure, so four bus mastership periods are needed in order to complete the initialization sequence. Note that the last double-word transfer of the last bus mastership period of the initialization sequence accesses an unneeded location. Data from this transfer is discarded internally.

If a bus preemption event occurs during an initialization block DMA transfer, then the PCnet-32 controller will complete both of the two data transfer cycles of the initialization block DMA transfer before releasing the HOLD signal and relinquishing the bus.

When SSIZE32 = 0 (CSR58[8]/BCR20[8]), then the number of bus mastership periods needed to complete the initialization procedure is 3 or 4.



18219-11

Figure 8. Initialization DMA Transfer

Descriptor DMA Transfers

PCnet-32 controller will determine when a descriptor access is required. A descriptor DMA read will consist of two double-word transfers. A descriptor DMA write will consist of one **or** two double word transfer. The transfers within a descriptor DMA transfer mastership period will always be of the same type (either all read or all write). The transfers will be to addresses in the order as specified in Table 19 and Table 20 (note that MD indicates TMD or RMD).

If buffer chaining is used (see Transmit and Receive Descriptor Table Entry sections), writes to the descriptors that do **not** contain the End of Packet bit will consist of only one double-word. This write will be to the same location as the second of the two writes performed when the End of Frame has been processed (i.e. to the location that contains the descriptor OWNership bit, MD1[31]).

Descriptor DMA transfers will never be executed using linear burst mode. During read accesses, the byte enable signals will indicate that all byte lanes are active. Should some of the bytes not be needed, then the PCnet-32 controller will internally discard the extraneous information that was gathered during such a read. During write accesses, only the bytes which need to be written are enabled, by activating the corresponding byte enable pins. See Figure 9 and Figure 10.

If a bus preemption event occurs during a descriptor DMA transfer, then the PCnet-32 controller will complete both of the two data transfer cycles of the descriptor DMA transfer, before releasing the HOLD signal and relinquishing the bus.

The only significant differences between descriptor DMA transfers and initialization DMA transfers are that the addresses of the accesses follow different ordering.

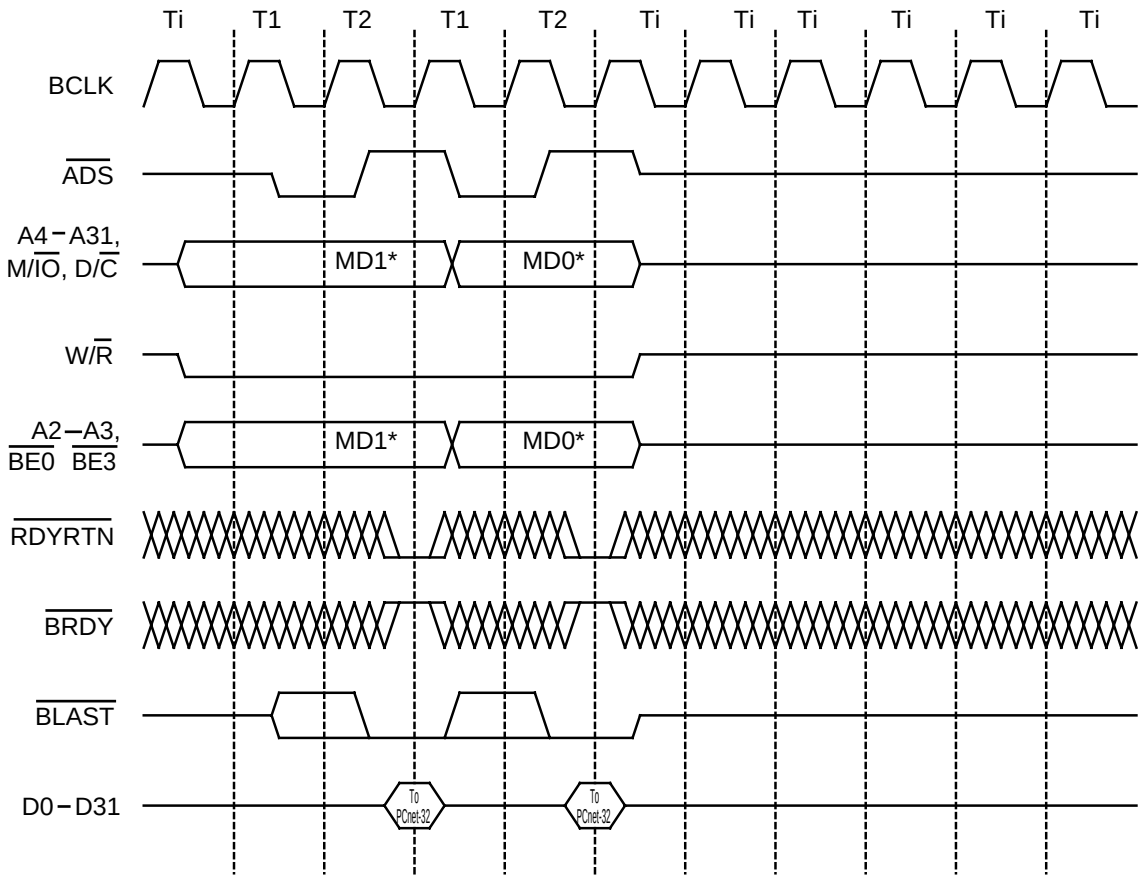
Table 19. Bus Master Reads of Descriptors

16-Bit Software Mode			32-Bit Software Mode		
Address SequenceA[7:0]	**LANCE Item Accessed**	**PCnet-32 Item Accessed**	**Address SequenceA[7:0]***	**LANCE Item Accessed**	**PCnet-32 Item Accessed**
00	MD1[15:0], MD0[15:0]	MD1[31:24], MD0[23:0]	04	MD1[15:8], MD2[15:0]	MD1[31:0]
04	MD3[15:0], MD2[15:0]	MD2[15:0], MD1[15:0]	00	MD1[7:0], MD0[15:0]	MD0[31:0]
Bus Break			Bus Break		

Table 20. Bus Master Writes of Descriptors

16-Bit Software Mode			32-Bit Software Mode		
Address SequenceA[7:0]	**LANCE Item Accessed**	**PCnet-32 Item Accessed**	**Address SequenceA[7:0]***	**LANCE Item Accessed**	**PCnet-32 Item Accessed**
04	MD3[15:0], MD2[15:0]	MD2[15:0], MD1[15:0]	08	MD3[15:0], MD2[15:0]	MD2[31:0]
00	MD1[15:0], MD0[15:0]	MD1[31:24], MD0[23:0]	04	MD1[15:8], MD2[15:0]	MD1[31:0]
Bus Break			Bus Break		

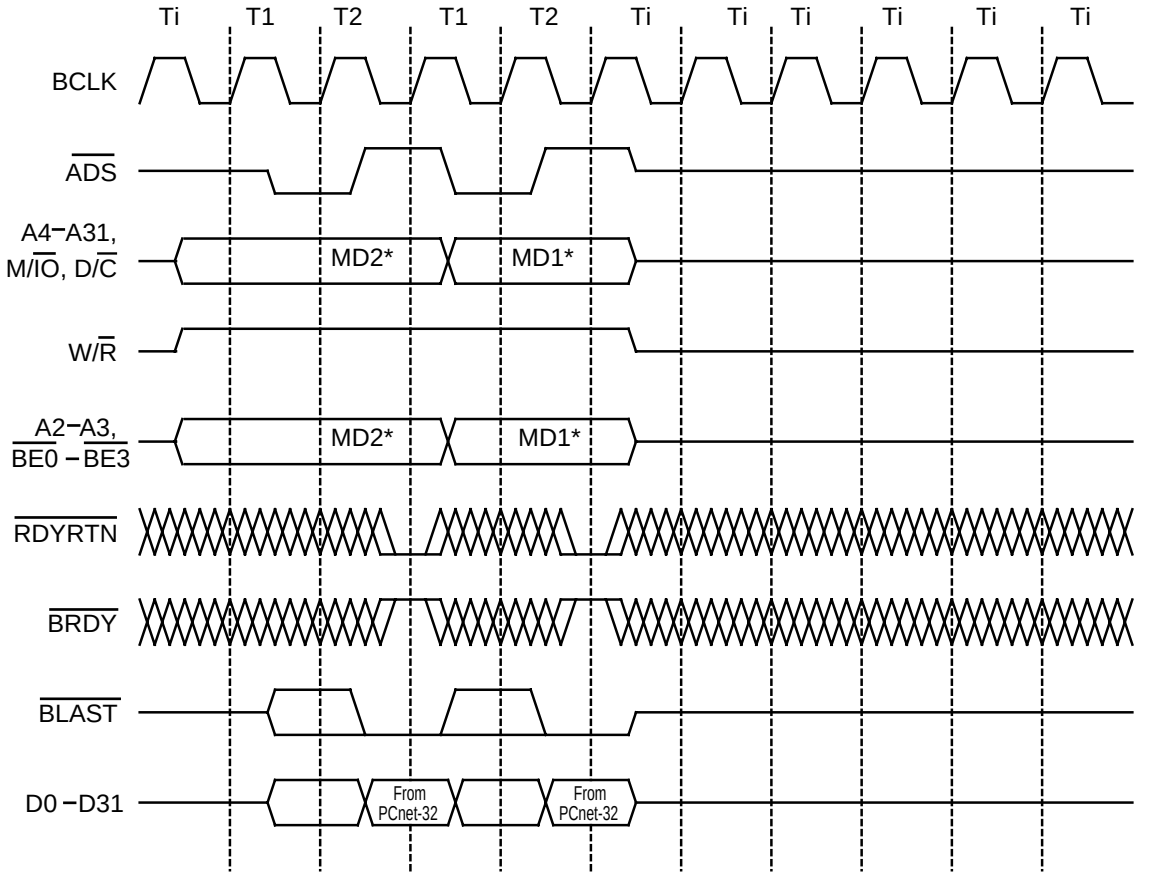
** Address values for A[31:8] are constant throughout any single descriptor DMA transfer. Note that even though bits A{1:0] do not physically exist in the system, these bits must be set to ZERO in the descriptor base address.*



**Note that Message Descriptor addresses 1 and 0 are in descending order.*

18219-12

Figure 9. Descriptor DMA Read



**Note that Message Descriptor addresses 2 and 1 are in descending order.*

18219-13

Figure 10. Descriptor DMA Write

FIFO DMA Transfers

PCnet-32 controller microcode will determine when a FIFO DMA transfer is required. This transfer mode will be used for transfers of data to and from the PCnet-32 controller FIFOs. Once the PCnet-32 controller BIU has been granted bus mastership, it will perform a series of consecutive transfer cycles before relinquishing the bus. Each transfer will be performed sequentially, with the issue of an address, and the transfer of the corresponding data with appropriate output signals to indicate selection of the active data bytes during the transfer. All transfers within the master cycle will be either read or write cycles, and all transfers will be to contiguous, ascending addresses. The number of data transfer cycles contained within a single bus cycle is in general, dependent on the programming of the DMAPLUS option (CSR4, bit 14). Several other factors will also affect the length of the bus cycle period. The possibilities are as follows:

If DMAPLUS = 0, a maximum of 16 transfers will be performed by default. This default value may be changed by writing to the burst register (CSR80). Note that DMAPLUS = 0 merely sets a maximum value. The minimum number of transfers in the bus cycle will be determined by all of the following variables: the settings of the FIFO watermarks, the particular conditions existing within the FIFOs, receive and transmit status conditions, the value of the DMA Burst Cycle (CSR80), the value of the DMA Bus Activity Timer (CSR82), and the timing of any occurrence of preemption that takes place during the FIFO DMA transfer.

If DMAPLUS = 1, the bus cycle will continue until the transmit FIFO is filled to its high threshold (read transfers) or the receive FIFO is emptied to its low threshold (write transfers), or until the DMA Bus Activity Timer value (CSR82) has expired. Other variables may also affect the end point of the burst in this mode. Among those variables are: the particular conditions existing within the FIFOs, receive and transmit status conditions, and bus preemption events.

The FIFO thresholds are programmable (see description of CSR80), as are the Burst Cycle and Bus Activity Timer values. The exact number of transfer cycles in the case of DMAPLUS = 1 will be dependent on the latency of the system bus to the PCnet-32 controller's mastership request and the speed of bus operation, but

will be limited by the value in the Bus Activity Timer register, the FIFO condition, receive and transmit status, and by preemption events, if any. Barring a time-out by either of these registers, or a bus preemption by another mastering device, or exceptional receive and transmit events, or an end of packet signal from the FIFO, the FIFO watermark settings and the extent of Bus Acknowledge latency will be the major factors determining the number of accesses performed during any given arbitration cycle when DMAPLUS = 1.

The READY response of the memory device will also affect the number of transfers when DMAPLUS = 1, since the speed of the accesses will affect the state of the FIFO. (During accesses, the FIFO may be filling or emptying on the network end. A slower memory response will allow additional data to accumulate inside of the FIFO (during write transfers from the receive FIFO). If the accesses are slow enough, a complete double word may become available before the end of the arbitration cycle and thereby increase the number of transfers in that cycle.) The general rule is that the longer the bus grant latency or the slower the bus transfer operations (or clock speed) or the higher the transmit watermark or the lower the receive watermark or any combination thereof the longer will be the average burst length.

If a bus preemption event occurs during a FIFO DMA transfer, then the PCnet-32 controller will complete the current transfer **and** it will complete a maximum of four additional data transfer cycles before releasing the HOLD signal and relinquishing the bus.

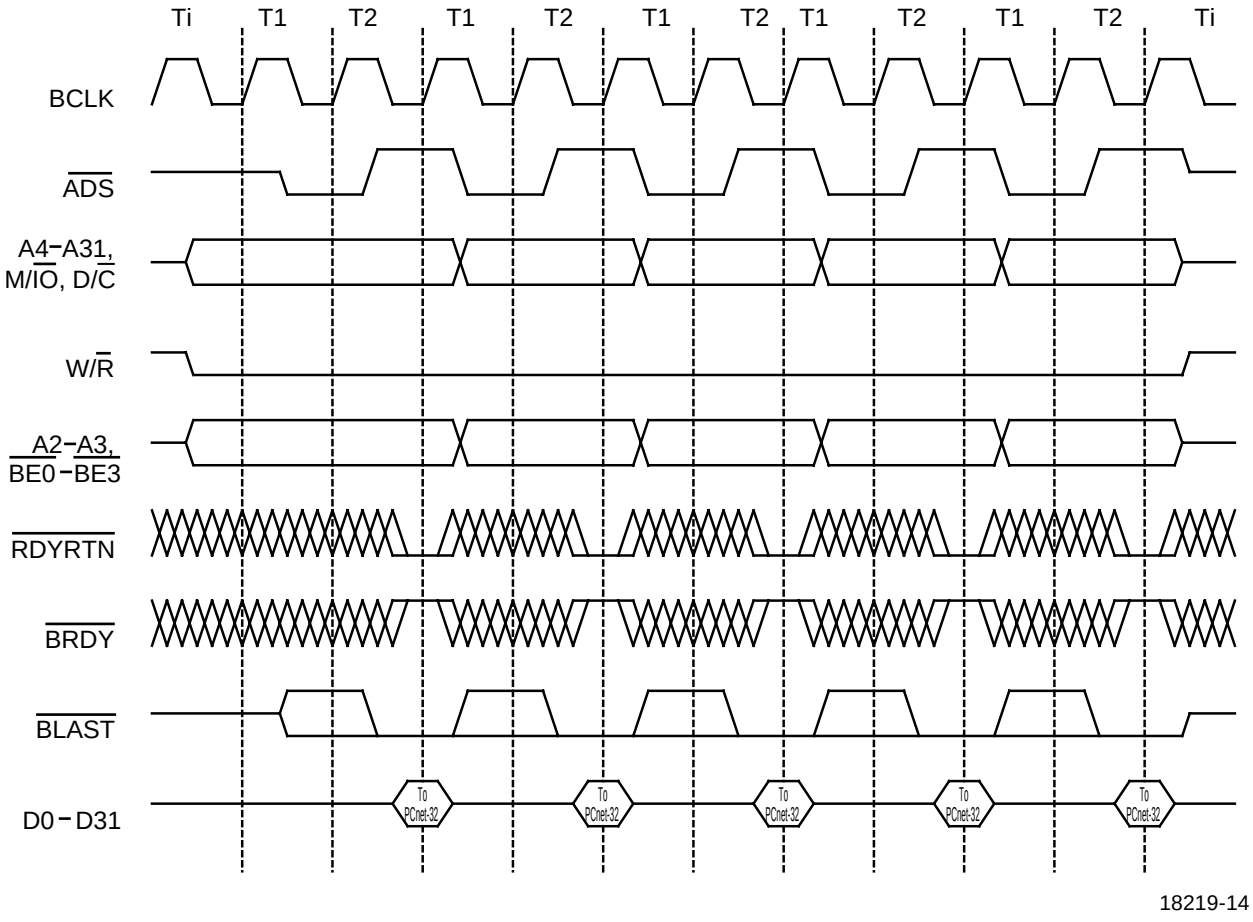


Figure 11. FIFO DMA Read

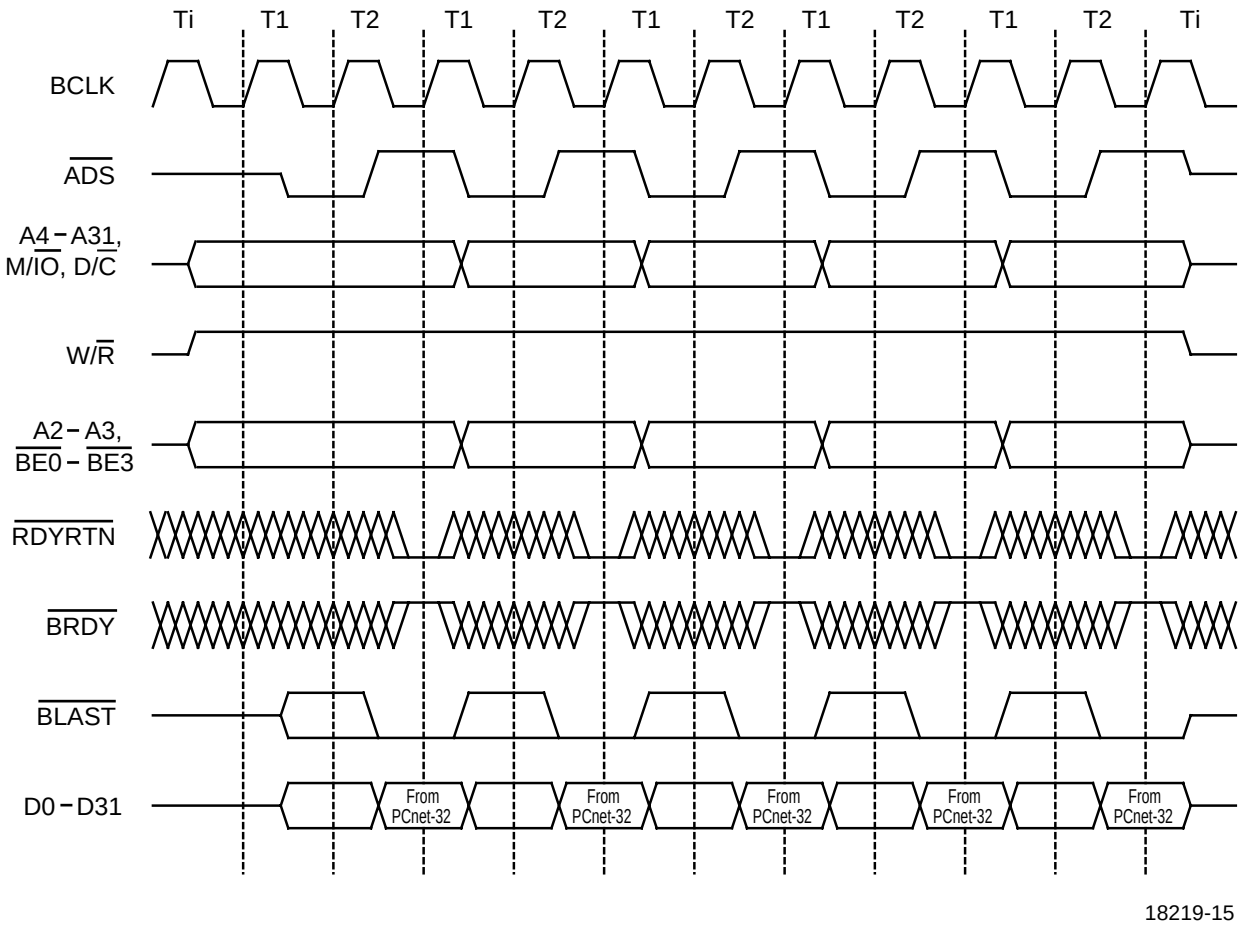


Figure 12. FIFO DMA Write

Note that A[1:0] do not exist in a 32-bit system, but both of these bits do exist in the buffer pointers that are passed to the PCnet-32 controller in the descriptor. A[1:0] values will be decoded and presented on the bus as byte enable ($\overline{BE0}$-$\overline{BE3}$) values during FIFO DMA transfers.

Linear Burst DMA Transfers

Once the PCnet-32 controller has been granted bus mastership, the PCnet-32 controller may request to perform linear burst cycles by de-asserting the $\overline{BLAST}$ signal. If the device being accessed wishes to support linear bursting, then it must assert $\overline{BRDY}$ and de-assert $\overline{RDYRTN}$, with the same timing that $\overline{RDYRTN}$ would normally be provided. Linear bursting is only performed by the PCnet-32 controller if the BREADE and/or BWRITE bits of BCR18 are set. These bits individually enable/disable the ability of the PCnet-32 controller to perform linear burst accesses during master read operations and master write operations, respectively. Only FIFO data transfers will make use of the linear burst mode.

The first transfer in the linear burst will consist of both an address and a data cycle, but subsequent transfers will contain data only, until the LINBC upper limit of transfers have been executed. LINBC is a value from the BCR18 register. The linear burst “upper limit” is created by taking the BCR18 LINBC[2:0] value and multiplying by 4. The result is the number of transfers that will be performed within a single linear burst sequence.

The entire address bus will still be driven with appropriate values during the data cycles. When the LINBC upper limit of data transfers have been performed, a new $\overline{ADS}$ may be asserted (if there is more data to be transferred), with a new address on the A2–A31 pins. Following the new $\overline{ADS}$ cycle, the linear bursting of data will resume. Ownership of the bus will be maintained until other variables cause the PCnet-32 controller to relinquish the bus. These variables have been discussed in the FIFO DMA transfers section above. They will be reviewed again within this section of the document.

Transfers within a linear burst cycle will either be all read or all write cycles, and will always be to contiguous ascending addresses. Linear Bursting of Read and Write operations can be individually enabled or disabled through the BREADE and BWRITE bits of BCR18 (bits 6 and 5).

Linear Burst DMA transfers should be considered as a superset of the FIFO DMA transfers. Linear burst DMA transfers will only be used for data transfers to and from the PCnet-32 controller FIFOs and they will only be allowed when the burst enable bits of BCR18 have been set. Linear Read bursting and Linear Write bursting have individual enable bits in BCR18. Any combination of linear burst enable bit settings is permissible.

Linear bursting is not allowed in systems that have BCLK frequencies above 33 MHz. Linear bursting is automatically disabled in VL-Bus systems that operate above this frequency by connecting the VLBEN pin to either ID(3) (for VL-Bus version 1.0 systems) or ID(4) AND ID(3) AND ID(1) AND $\overline{ID}(\overline{0})$ (for VL-Bus version 1.1 or 2.0 systems). In Am486-style systems that have BCLK frequencies above 33 MHz, disabling the linear burst capability is ideally carried out through EEPROM bit programming, since the EEPROM programming can be setup for a particular machine's architecture.

All byte lanes are always considered to be active during all linear burst transfers. The $\overline{BE3}$–$\overline{BE0}$ signals will reflect this fact.

Linear Burst DMA Starting Address Restrictions

A PCnet-32 controller linear burst will begin only when the address of the current transfer meets the following condition:

$$A[31:0] \text{ MOD } (LINBC \times 16) = 0,$$

The following table illustrates all possible starting address values for all legal LINBC values. Note that A[31:6] are don't care values for all addresses. Also note that while A[1:0] do not physically exist within a 32 bit system, they are valid bits within the buffer pointer field of descriptor word 0. Thus, where A[1:0] are listed, they refer to the lowest two bits of the descriptor's buffer pointer field. These bits will have an affect on determining when a PCnet-32 controller linear burst operation may legally begin and they will affect the output values of the $\overline{BE3}$–$\overline{BE0}$ pins, therefore they have been included in Table 21 as A[1:0].

It is not necessary for the software to insure that the buffer address pointer contained in descriptor word 0 matches the address restrictions given in the table. *If the buffer pointer does not meet the conditions set forth in the table, then the PCnet-32 controller will simply postpone the start of linear bursting until enough ordinary FIFO DMA transfers have been performed to bring the current working buffer pointer value to a valid linear burst starting address.* This operation is referred to as "aligning" the buffer address to a valid linear burst starting address. Once this has been done, the PCnet-32 controller will recognize that the address for the current access is a valid linear burst starting address, and it will automatically begin to perform linear burst accesses at that time, provided of course that the software has enabled the linear burst mode.

Note that if the software *would* provide only valid linear burst starting addresses in the buffer pointer, then the PCnet-32 controller could avoid performing the alignment operation. It would begin linear burst accesses on the very first of the buffer transfers thereby allowing a slight gain in bus bandwidth efficiency.

Because of the linear burst starting address restrictions given in the table above, the PCnet-32 controller linear burst mode is completely compatible with the Am486-style burst cycle when the LINBC[2:0] bits have been programmed with the value of 001.

Table 21. Linear Burst Addess

LINBC[2:0]	LBS = Linear Burst Size (No. of Transfers)	Linear Burst Addresses Beginning A[5:0] = (A[31:6] = Don't Care)
0	0 (no linear bursting)	Not Applicable
1	4	00, 10, 20, 30
2	8	00, 20
4	16	00
3,5,6,7	Reserved	Not Applicable

Linear Burst DMA Timing Diagram Explanatory Note

Note that in all of the following timing diagrams for linear burst operations, a LINBC[2:0] value of 001 has been assumed. This translates to a linear burst length of four transfers. When the linear burst size is four transfers, then A[31:4] are stable within a single linear burst sequence, while A[3:2] and $\overline{BE3}$–$\overline{BE0}$ will change to reflect the address of the current transfer. Note that for larger values of LINBC[2:0] which correspond to longer linear burst lengths, the range of address pins that is stable during each burst sequence is smaller. For example, if LINBC[2:0] has the value of 010, then the linear burst length is eight double word transfers or 32 bytes of data. With this value of LINBC, it takes five address bits to track the changing addresses through the burst. This means that only A[31:5] are stable during each linear burst sequence, while A[4:2] and $\overline{BE3}$–$\overline{BE0}$ will change to reflect the address of the current transfer. For LINBC[2:0] = 100, only A[31:6] are stable during each linear burst sequence, while A[5:2] and $\overline{BE3}$–$\overline{BE0}$ will change to reflect the address of the current transfer, and so on. Table 22 summarizes this information.

Table 22. Stable Address Lines During Linear Burst

LINBC Value	Portion of Address Bus Stable During Linear Burst
000	Linear Bursting Disabled
001	A[31:4]
010	A[31:5]
100	A[31:6]

Values of LINBC not shown in the table are not allowed. See the LINBC section of BCR18 for more details.

Since all of the timing diagrams assume a LINBC[2:0] value of 001, then A[31:4] are shown to be stable within each linear burst sequence, and A[3:2] and $\overline{BE3}$–$\overline{BE0}$ are shown as changing in order to reflect the address of the current transfer.

Linear Burst DMA Address Alignment

Linear bursting may begin during a bus mastership period which was initially performing only ordinary DMA operations. (i.e. the value of $\overline{BLAST}$ is not restricted to ZERO for an entire bus mastership period if ZERO was the value of $\overline{BLAST}$ on the first access of a bus mastership period.) A change from non-linear bursting to linear bursting will normally occur during linear burst DMA address alignment operations.

If the PCnet-32 controller is programmed for LINEAR burst mode (i.e. BREADE and/or BWRITE bits of BCR18 are set to ONE), and the PCnet-32 controller requests the bus, but the starting address of the first transaction does not meet the conditions as specified in the table above, then the PCnet-32 controller will perform burst-cycle accesses (i.e. it will provide an $\overline{ADS}$ for each transfer) until it arrives at an address that does meet the conditions described in the table. At that time, and without releasing the bus, the PCnet-32 controller will invoke the linear burst mode. The simple external manifestation of this event is that the value of the $\overline{BLAST}$ signal will change to de-asserted (driven high) on the next T2 cycle, thereby indicating a willingness of the PCnet-32 controller to perform linear bursting. (Note that burst-cycle accesses are performed with $\overline{BLAST}$ = 0.)

Figure 13 shows an example of a linear burst DMA alignment operation being performed:

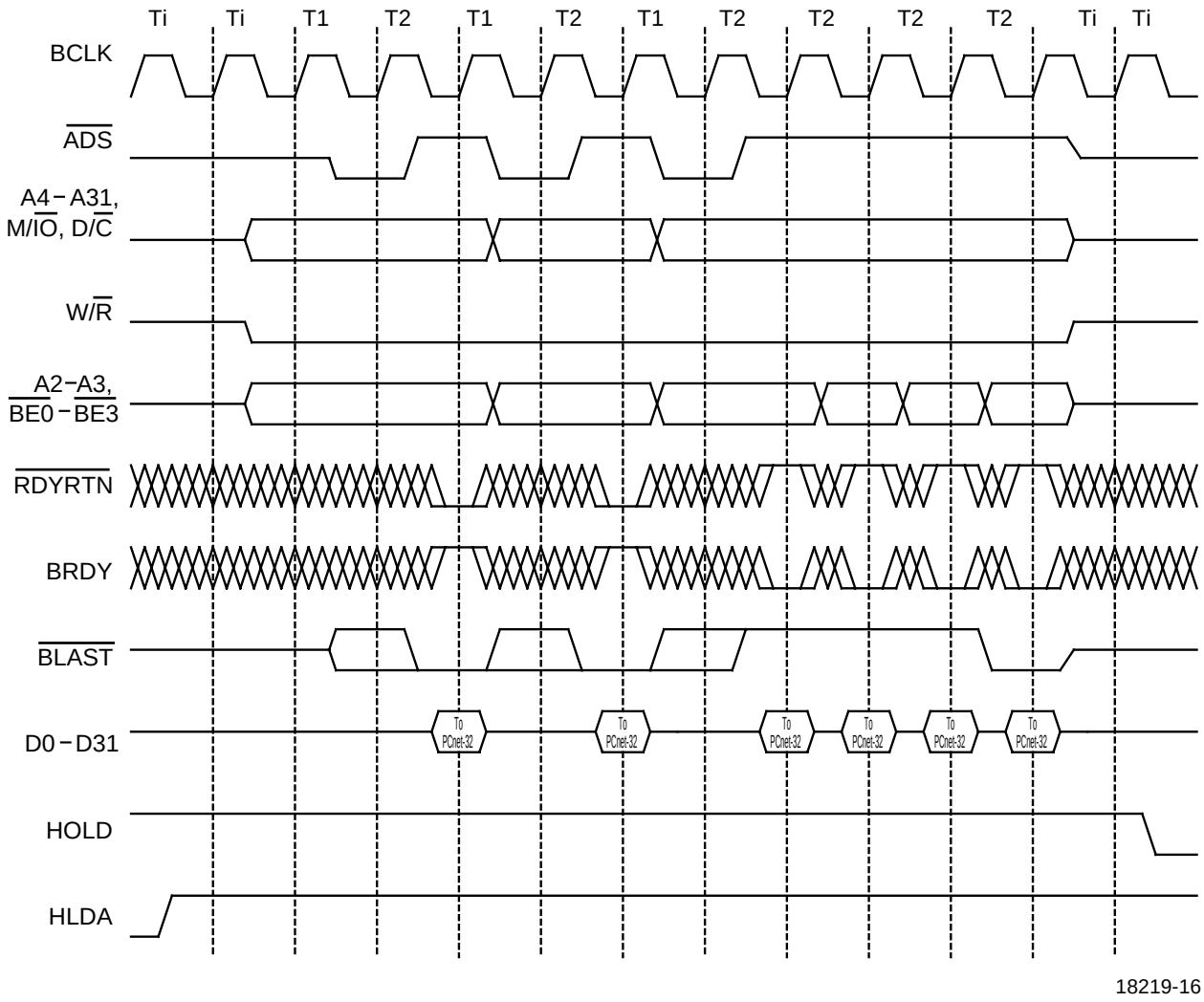


Figure 13. FIFO DMA Read Followed by Linear Burst Read During a Single Bus Mastership Period

Linear Burst DMA BLAST Signal Timing

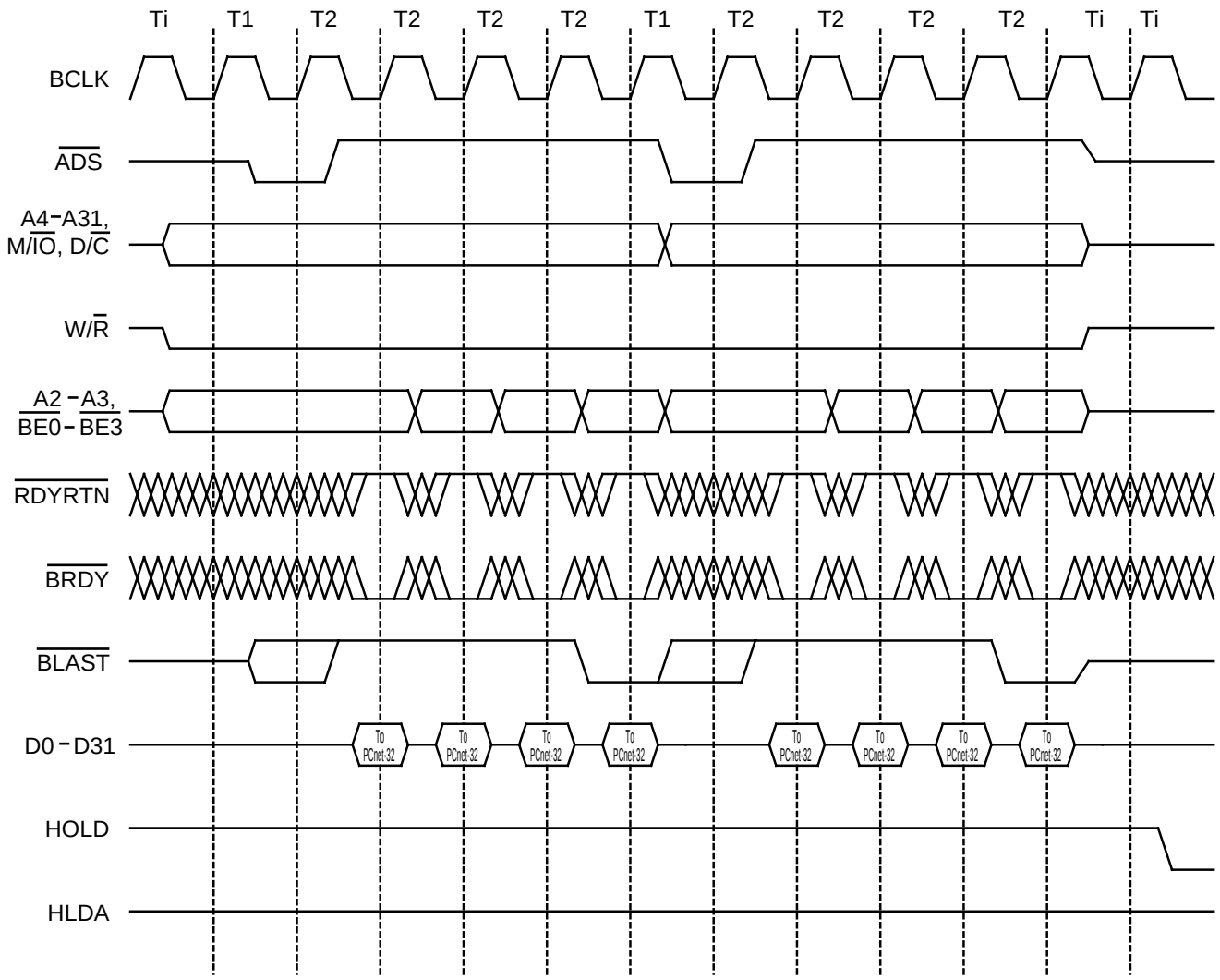
Linear burst cycles are requested by the PCnet-32 controller by de-asserting the BLAST signal (i.e. BLAST = 1). When BLAST is de-asserted by the PCnet-32 controller, the slave device is under no obligation to provide BRDY. Instead, it may provide RDYRTN in response to each of the PCnet-32 controller transfers. If RDYRTN is asserted during accesses in which the PCnet-32 controller has de-asserted BLAST, then the current transfer reverts to ordinary burst-cycle (see FIFO DMA Transfer section).

When BLAST is asserted, it signals the end of the current linear burst sequence.

In a cycle in which BLAST is asserted and following the assertion of either RDYRTN and/or BRDY by the slave device, the PCnet-32 controller may either relinquish the bus or it may initiate a new sequence of linear burst transfers without relinquishing the bus. If the PCnet-32 controller continues with a new sequence of linear burst transfers, the address asserted during the next T1 cycle will always be the next address in sequence from the previous T2 cycle. In other words, the PCnet-32 controller will never execute cycles within a single bus master-ship period that are not both ascending and contiguous, with the exception of the descriptor DMA accesses described above.

The decision to continue bus ownership will depend upon several variables, including the state of the receive or transmit FIFO. All factors related to this decision are discussed later in this section.

Figure 14 illustrates a typical case of multiple linear burst sequences being performed during a single bus mastership period.



18219-17

Figure 14. Linear Burst Read

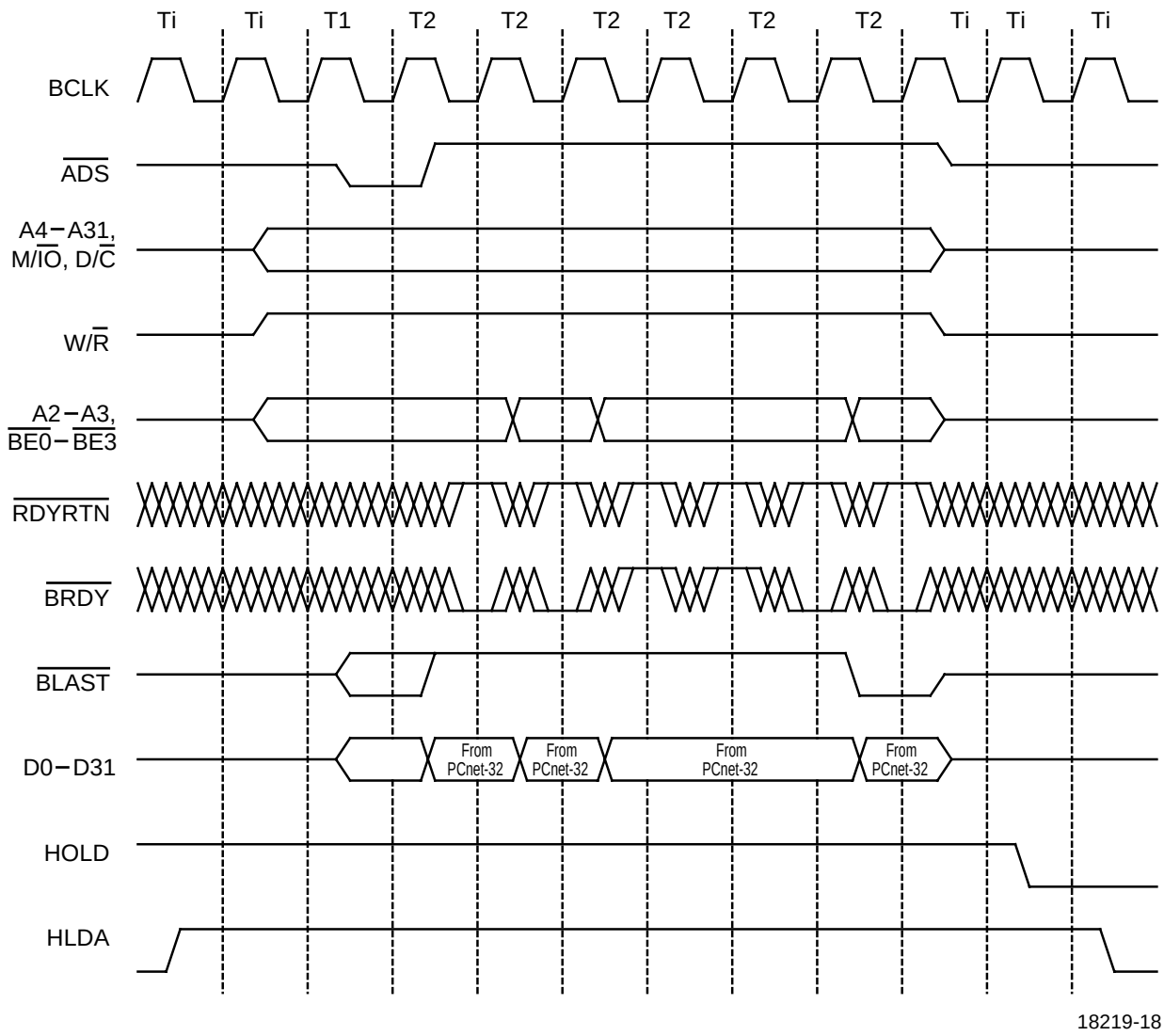


Figure 15. Linear Burst Write with Wait States Added by the Slave Device on the Third Transfer

Linear Burst DMA Ready Wait States

The PCnet-32 controller will insert wait states into linear burst DMA cycles if neither RDYRTN nor BRDY are sampled asserted at the end of each T2 cycle.

Interrupted Linear Burst DMA Cycles

The assertion of RDYRTN in the place of BRDY within a linear burst cycle will cause the linear burst to be "interrupted."

In that case, the PCnet-32 controller will revert to ordinary two-cycle transfers that contain both a T1 and a T2 cycle, except that BLAST will remain de-asserted to show that linear bursting is still being requested by the PCnet-32 controller. This situation is defined as an interrupted linear burst cycle. If BRDY is sampled asserted (without also sampling RDYRTN asserted during the same access) during an interrupted linear burst cycle in which BLAST is de-asserted, then linear bursting will resume. (Note that BLAST will become asserted during an interrupted linear burst cycle during the transfer that would have been the last transfer of the linear burst sequence, had the sequence not been interrupted.)

When an interrupted linear burst cycle is resumed, then the next assertion of ADS will depend upon the initial starting point of the linear burst, rather than on the resumption point.

For example, if the linear burst length = 4 (LINBC = 1), and BRDY is asserted during the first transfer, but RDYRTN is asserted on the second, then the PCnet-32 controller will revert to ordinary DMA transfers on the third transfer. If the responding device again asserts

BRDY on the third access (while RDYRTN is de-asserted), the PCnet-32 controller will resume linear bursting from the current address. It will produce 1 more data cycle before asserting the next ADS, i.e. PCnet-32 controller will keep track of the initial linear burst end point and will continue with the original linear burst after the RDYRTN interruption has occurred.

Figure 16 illustrates an example of an interrupted linear burst. Note that BLAST is asserted in the fourth transfer after the initial linear burst began, even though the linear burst was interrupted with the assertion of RDYRTN and a new ADS was driven for the third transfer. The external effect of the RDYRTN interruption is completely manifested in the insertion of the T1 cycle that contains the asserted ADS. The linear burst cycle is not affected in any other way.

Partial Linear Burst

Certain factors may cause the PCnet-32 controller to linearly burst fewer than the LINBC limit during a single linear burst sequence. BLAST will be asserted during the last data transfer. A linear burst that is terminated by BLAST before the LINBC limit is reached is called a partial linear burst.

A partial linear burst is recognizable in that the BLAST signal is asserted while fewer than the LINBC limit of transfers has been performed in the current linear burst sequence.

Factors that could generate a partial linear burst include:

- No more data available for transfers from the current transmit buffer
- No more data available for transfer from the receive FIFO for this packet
- No more space available for transfers to the current receive buffer

If any of these conditions occurs, then the PCnet-32 controller will end the Linear Burst by asserting BLAST. Typically, during the case of a master read operation (for transmit buffer transfers), the last transfer in the linear burst sequence will be the last transfer executed before the PCnet-32 controller releases the bus. This is true of both partial and completed linear burst sequences.

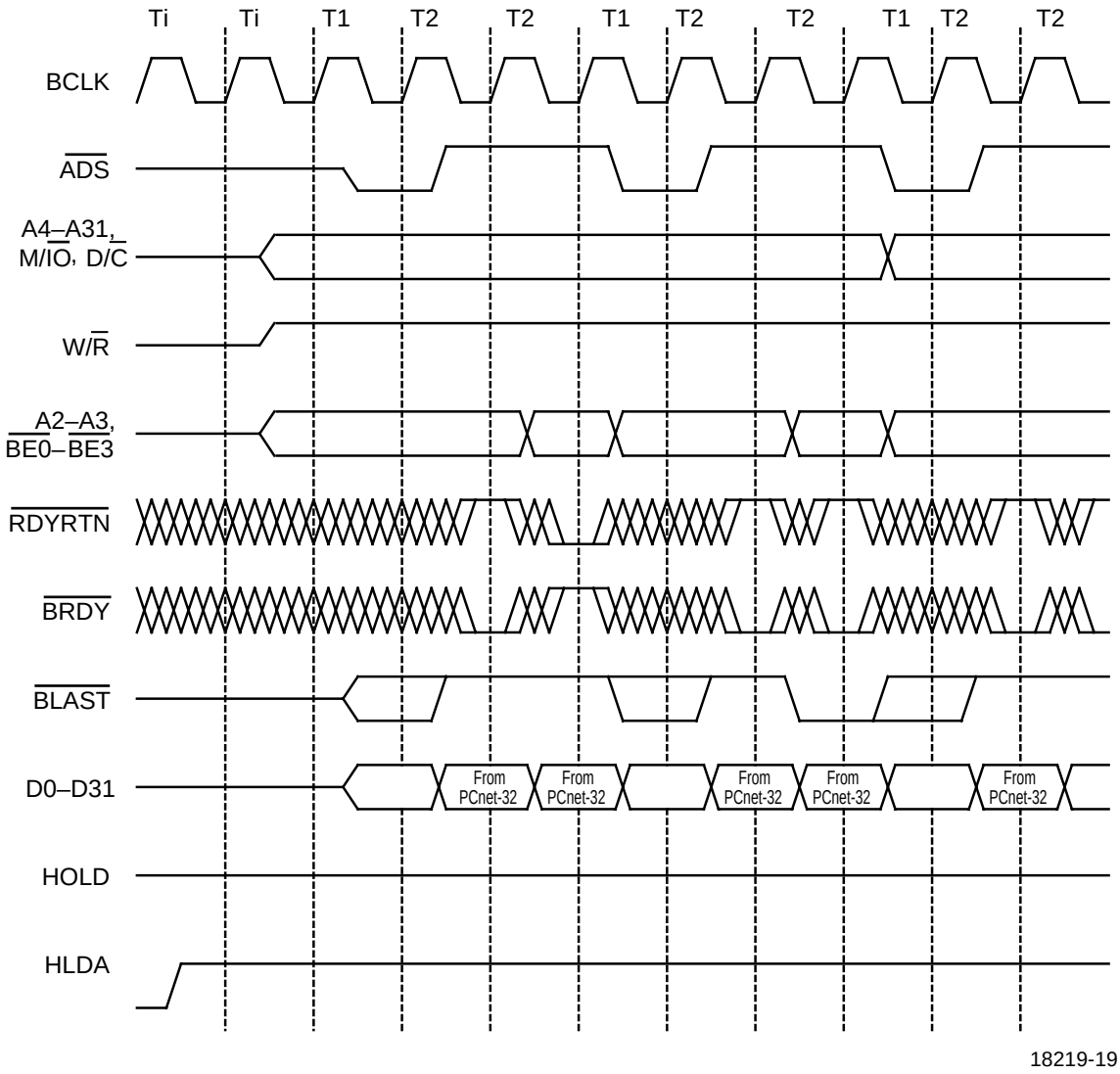


Figure 16. "Interrupted" Linear Burst Write

Typically, during the case of a master write operation (for receive buffer transfers) when receive packet data has ended, the last transfer in the linear burst sequence will be the last transfer executed before the PCnet-32 controller releases the bus. This is true of both partial and completed linear burst sequences.

However, if the next transfer that the PCnet-32 controller is scheduled to execute will be to the last available location of a receive or transmit buffer, then the PCnet-32 controller may assert BLAST on the current transfer and then use an ordinary cycle to make the last transfer to the buffer. This event occurs because of the restrictions placed upon the byte enable signals during the linear burst operation. As mentioned in the initial description of linear burst accesses (section Linear Burst DMA Transfers), all byte lanes of the data bus are always enabled during linear burst operations. Note, however, that in the case of the last buffer location, the PCnet-32 controller may own only a portion of the double-word location. In such cases, it is necessary to discontinue linear burst accesses on the second from last buffer location so that an basic transfer with some byte lanes disabled can be used for the final transfer.

Figure 17 shows a partial linear burst that occurred while approaching the transfer of the last bytes of data to a receive buffer. The linear burst begins when 10 bytes of space still remain in the receive buffer. (The number of spaces remaining for the figure as drawn could be anywhere from 9 to 12 spaces. The value of 10 spaces has been chosen just for purposes of illustration.) After the first linear burst transfer, the PCnet-32 controller sees that between 6 bytes of space remain, and knowing that the second transfer will use another 4 bytes of space, the PCnet-32 controller is

able to predict that the third transfer will be the last. Therefore, it asserts $\overline{BLAST}$ on the second transfer to terminate the linear burst operation. However, the PCnet-32 controller retains ownership of the bus so that it may, immediately thereon, make an basic transfer to the last two spaces in the buffer.

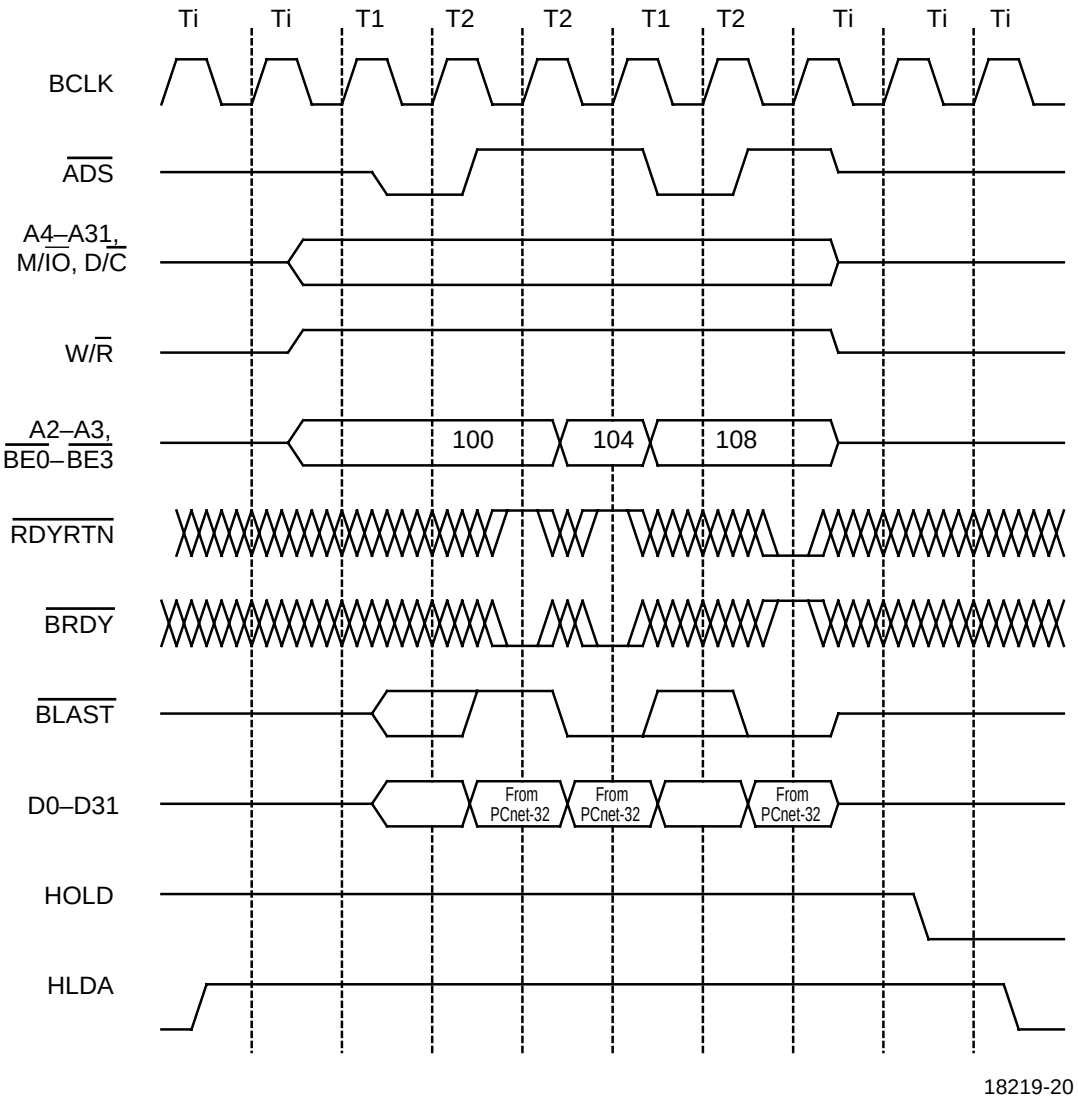


Figure 17. Typical Partial Linear Burst Write, Followed by a Basic Transfer During the Same Bus Mastership Period

Length of Bus Mastership Period

The number of data transfer cycles within the total bus mastership period is dependent on the programming of the DMAPLUS option (CSR4, bit 14). The possibilities are as follows:

If DMAPLUS = 0, a maximum of 16 transfers will be performed by default. This default value may be changed by writing to the burst register (CSR80). Note that DMAPLUS = 0 merely sets a maximum value. The minimum number of transfers in the burst will be determined by all of the following variables: the settings of the FIFO watermarks and the conditions of the FIFOs, the value of the DMA Burst Cycle (CSR80), the value of the DMA Bus Activity Timer (CSR82), and any occurrence of preemption that takes place during the burst.

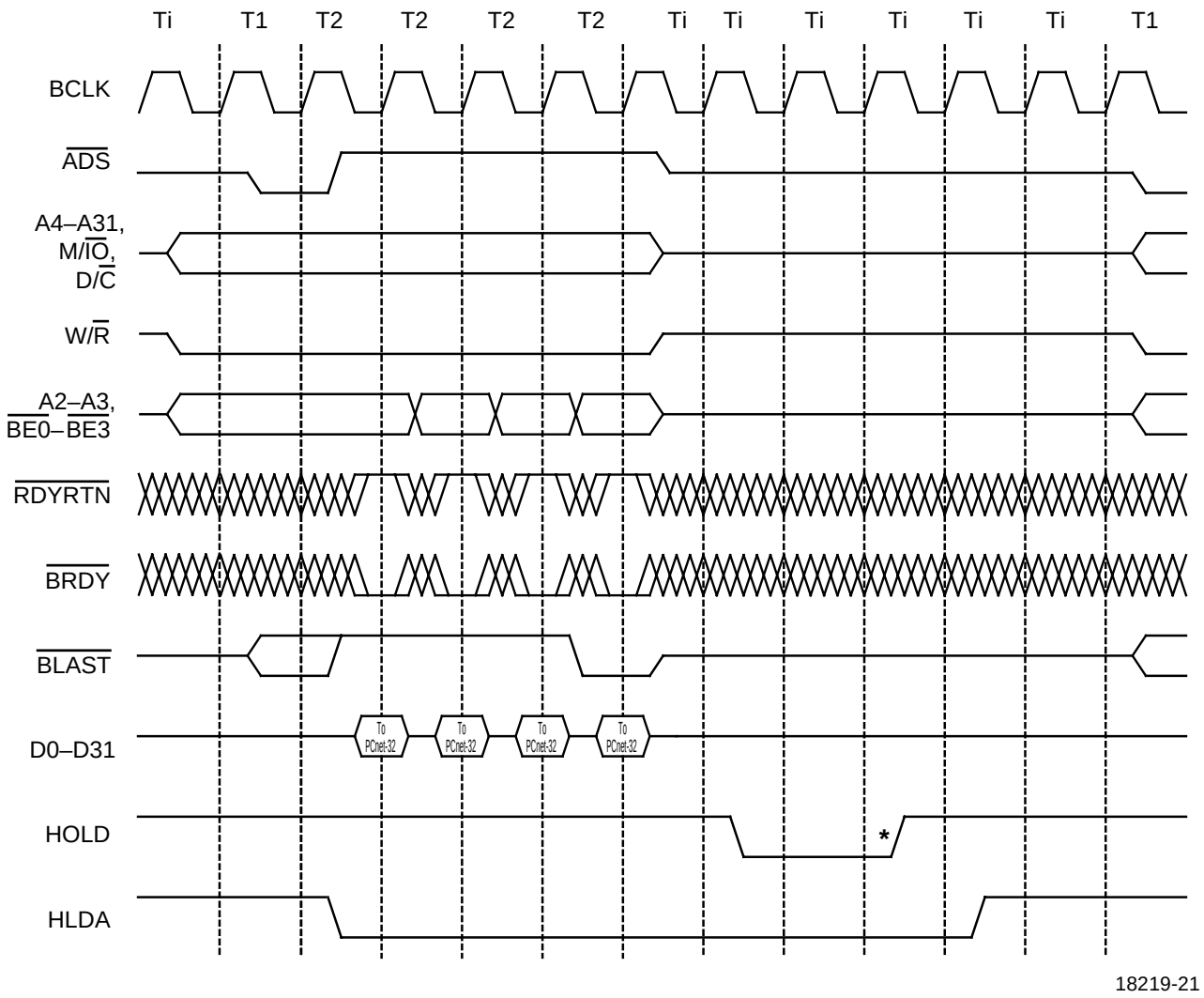
If DMAPLUS = 1, linear bursting will continue until the transmit FIFO is filled to its high threshold or the receive FIFO is emptied to its low threshold, or until the DMA Bus Activity Timer value (CSR82) has expired. A bus preemption event is another cause of termination of cycles. The FIFO thresholds are programmable (see description of CSR80), as are the Burst Cycle and Bus Activity Timer values. The exact number of total transfer cycles in the case of DMAPLUS = 1 will be dependent on the latency of the system bus to the PCnet-32 controller's mastership request and the speed of bus operation, but will be limited by the value in the Bus Activity Timer Register, the FIFO condition and by preemption occurrences, if any.

The exact response of the PCnet-32 controller to any of the conditions mentioned above can be complicated. For detail of the response to any particular stimulus, see each of the sections that describes PCnet-32 controller response.

Note that the number of transfer cycles between each $\overline{\text{ADS}}$ assertion will always only be controlled by LINBC, $\overline{\text{RDYRTN}}$, $\overline{\text{BOFF}}$, HLDA and FIFO conditions. The number of transfer cycles separating $\overline{\text{ADS}}$ assertions will not be affected by DMAPLUS or by the values in the Burst Cycle and Bus Activity Timer Register. However, these factors can influence the number of transfers that is performed during any given arbitration cycle.

Barring a time-out by the Burst Cycle or the Bus Activity Timer Register, or a bus preemption by another mastering device, the FIFO watermark settings and the extent of Bus Acknowledge latency will be the major factors in determining the number of accesses performed during any given arbitration cycle. The $\overline{\text{BRDY}}$ response time of the memory device will also affect the number of transfers, since the speed of the accesses will affect the state of the FIFO. (During accesses, the FIFO may be filling or emptying on the network end. For example, on a Receive operation, a slower device will allow additional data to accumulate inside of the FIFO. If the accesses are slow enough, a complete double word may become available before the end of the arbitration cycle and thereby increase the number of transfers in that cycle.) The general rule is that the longer the bus grant latency or the slower the bus transfer operations or the slower the clock speed or the higher the transmit watermark or the lower the receive watermark or any combination thereof, will produce longer total burst lengths.

If a bus preemption event occurs after the execution of the first T2 cycle of the fourth from the last transfer cycle within a linear burst DMA sequence, then the PCnet-32 controller will complete the current linear burst sequence and will execute a new linear burst sequence before releasing the HOLD signal and relinquishing the bus. If a bus preemption event occurs before or concurrent with the execution of the first T2 cycle of the fourth from the last transfer cycle within a linear burst DMA sequence, then the PCnet-32 controller will complete the current linear burst sequence and then will release the HOLD signal and will relinquish the bus. Within the context of this explanation, a single transfer cycle refers to the execution of a data transfer, regardless of the number of clock cycles taken, i.e. wait states are included in this definition of a transfer cycle. See Figure 18.



***Note:** Hold will always go inactive for 2 BLCK cycles before being reasserted.*

Figure 18. Linear Burst Read with Preemption During T2

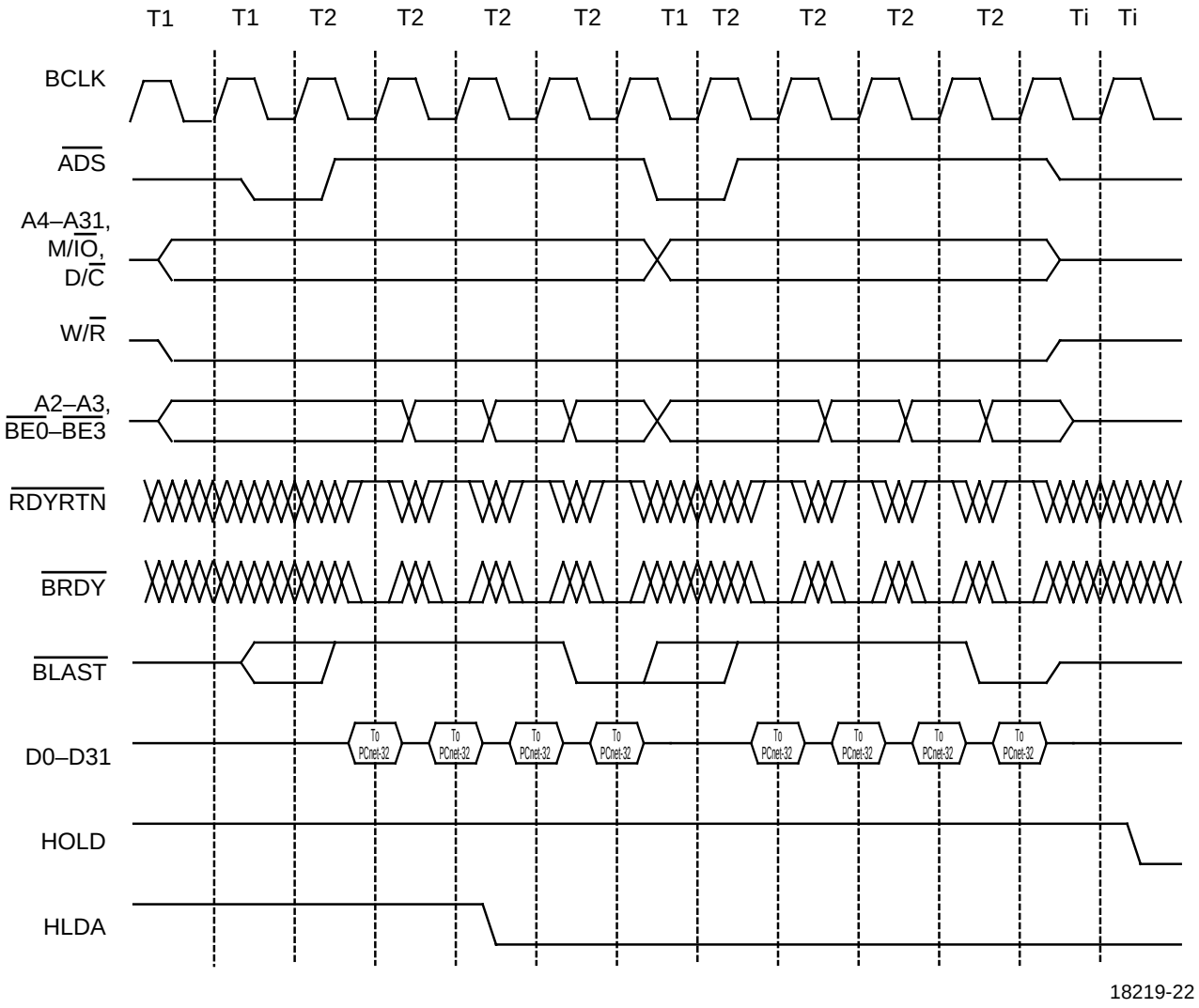
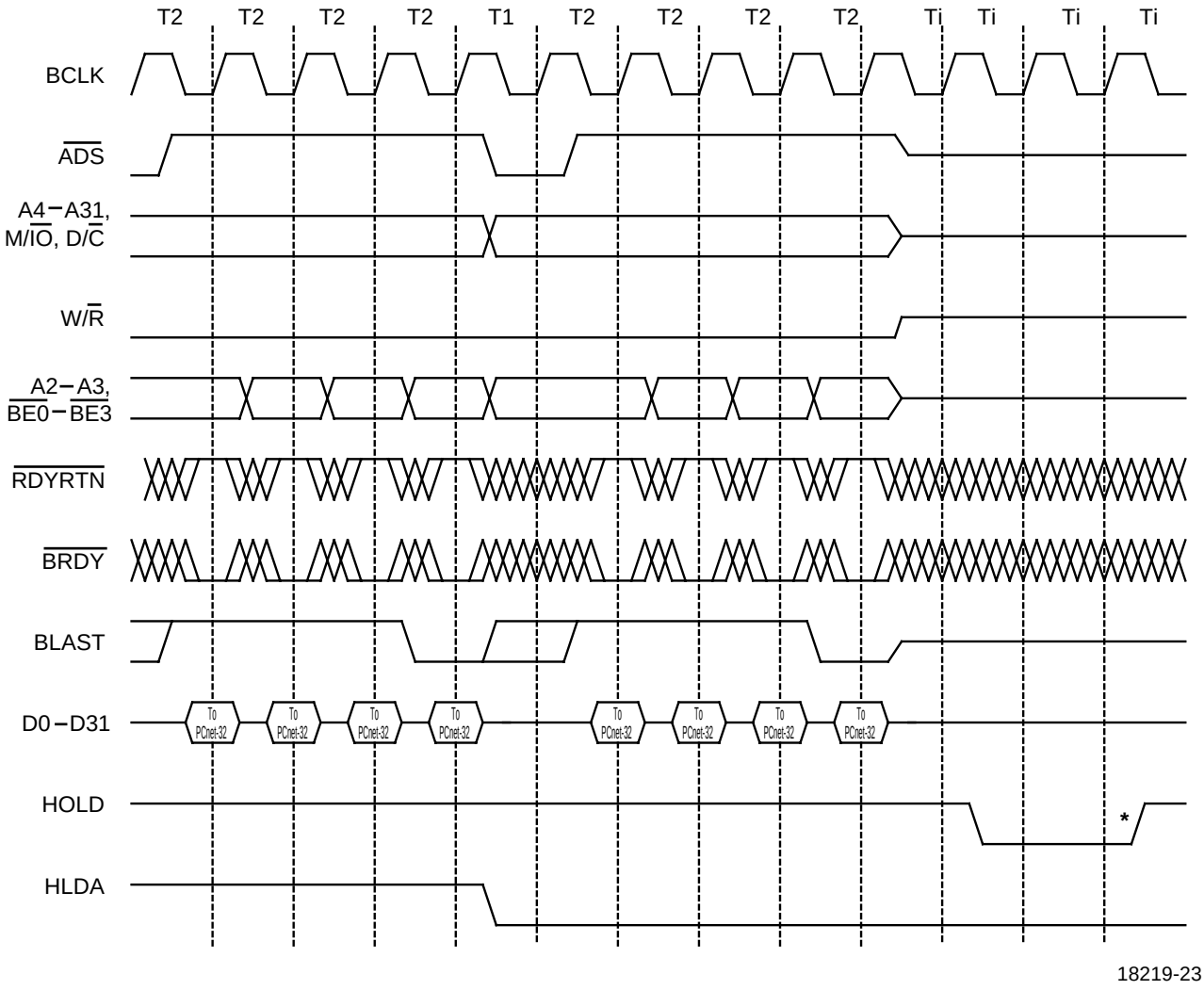


Figure 19. Linear Burst Read with Preemption During One of the Last Three T2 Cycles of the Sequence

If a bus preemption event occurs on a T1 cycle (specifically, a T1 cycle in which the ADS signal for a new linear burst sequence is asserted), then the next linear burst sequence will be executed before the PCnet-32 controller releases the HOLD signal and relinquishes the bus. (If the T1 cycle in which the preemption occurred was to begin an basic transfer, then the basic transfer plus as many as two additional basic transfers will be executed before relinquishing the bus.)



***Note:** HOLD is always held inactive for 2 BLCK cycles before being reasserted.*

Figure 20. Linear Burst Read with Preemption that Occurs During the T1 Cycle

Effect of B0FF

Assertion of BOFF during a linear burst has two possible outcomes. In the case of BOFF asserted ***before*** the first BRDY (or RDYRTN) has been sampled, the PCnet-32 controller will restart the linear burst after the BOFF event has ended. A new ADS will be asserted with the original starting address for the halted linear burst. See Figure 21.

However, if the BOFF signal is asserted ***after*** the first BRDY (or RDYRTN) has been sampled, the PCnet-32 controller will revert to ordinary burst-cycle accesses following the BOFF event. In this case, the linear bursting will *next* occur when the memory address being accessed *next* meets the linear burst starting address requirements. If BOFF is sampled active on the same clock edge that BRDY or RDYRTN is sampled active, then the BOFF takes priority.

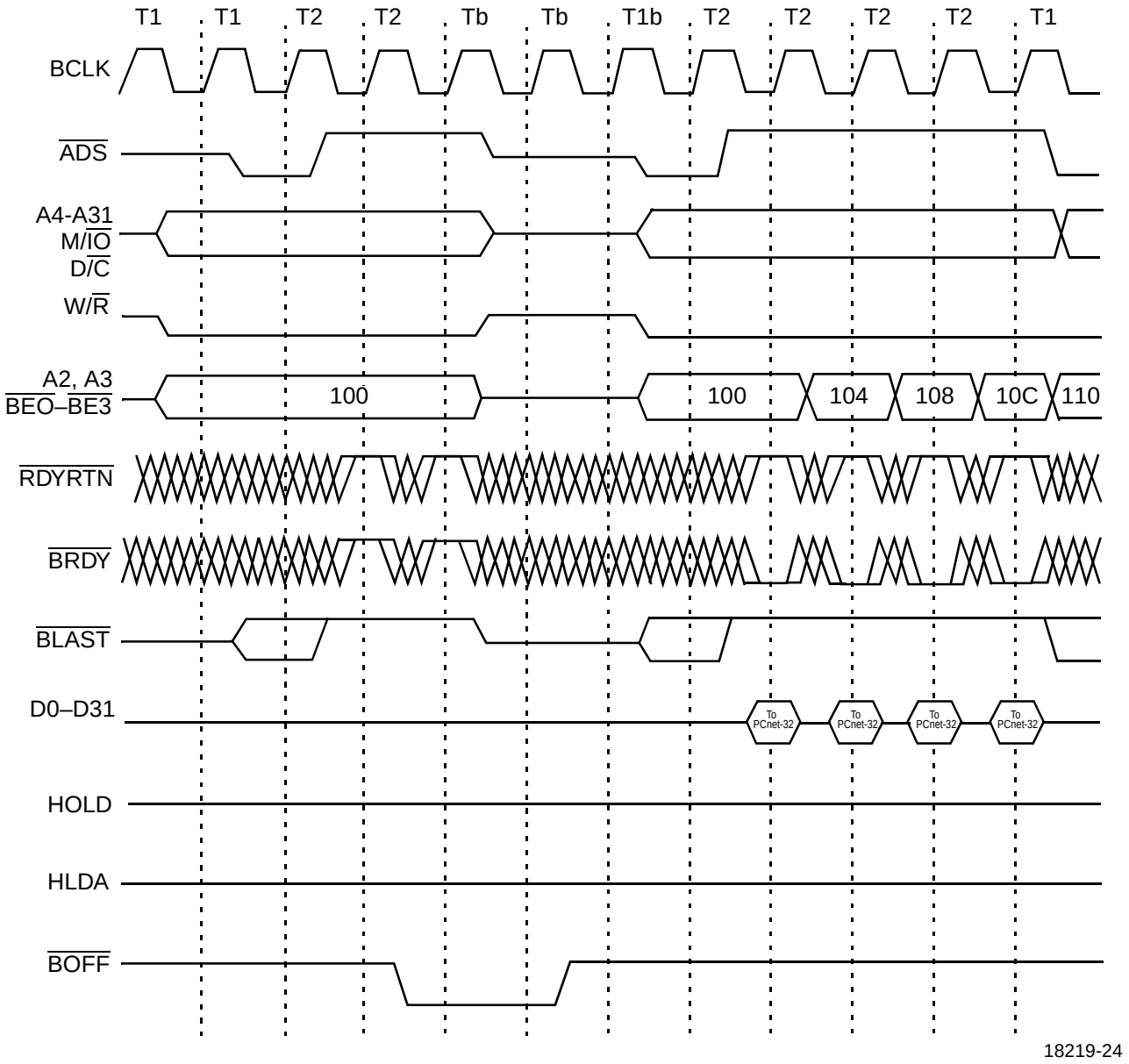


Figure 21. Restarted Linear Burst Read in which BOFF was Asserted Before BRDY of First Transfer in Linear Burst Sequency, Hence Linear Burst Sequence is Restarted When BOFF is De-asserted.

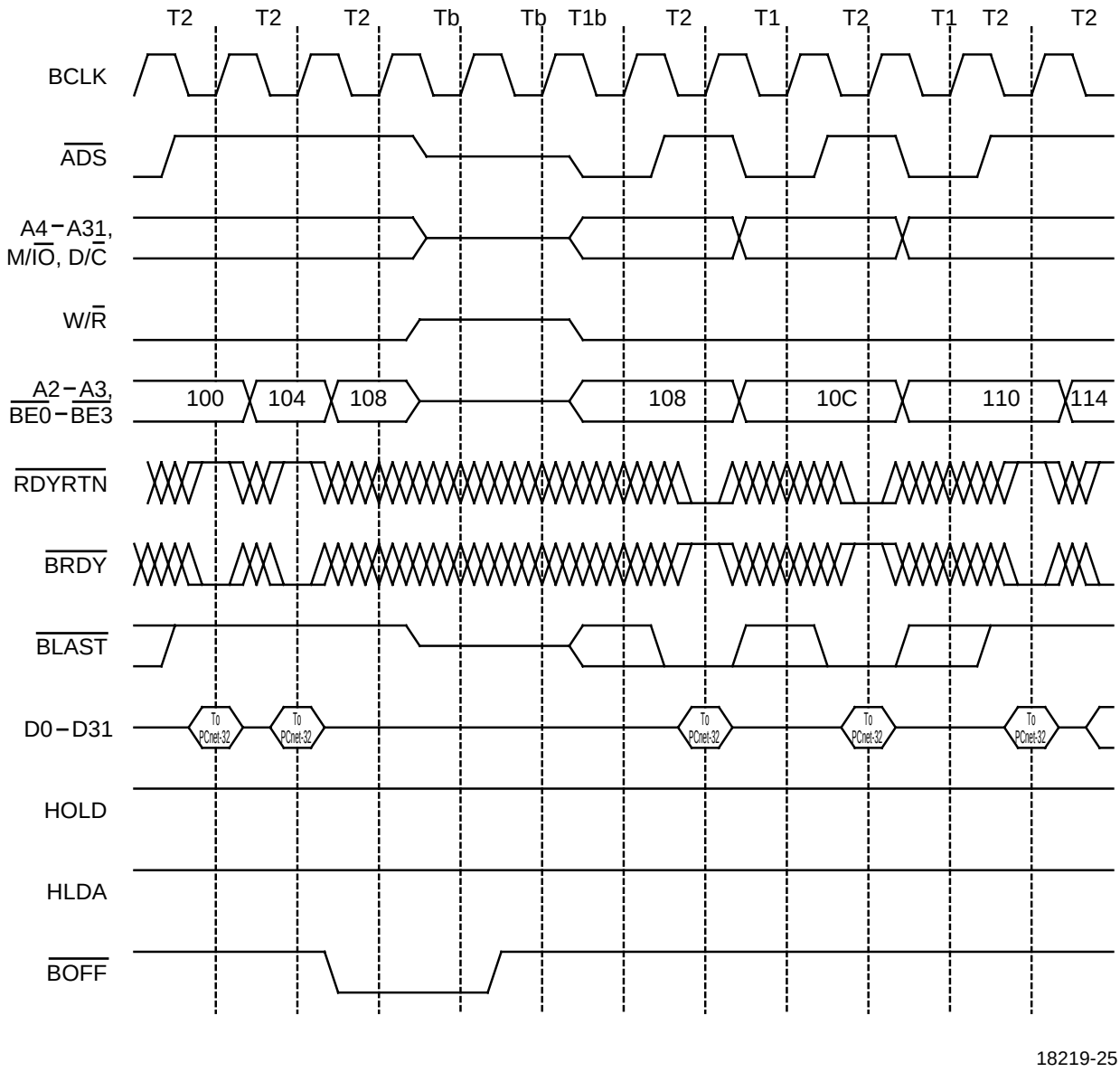


Figure 22. Restarted Linear Burst Read in which BOFF was Asserted After BRDY of First Transfer in Linear Burst Sequency. Hence, Linear Burst Reverts to Ordinary Cycles Until Next Legal Linear Burst Starting Address is Reached

In general, if the linear burst is suspended by another bus master (either because of BOFF or PCnet-32 controller releasing HOLD) then any partially completed linear burst access will not resume when the PCnet-32 controller regains bus ownership. But if the PCnet-32 controller linear burst is interrupted by the receipt of RDYRTN in place of BRDY, then the PCnet-32 controller will resume the linear burst operation as indicated by the BLAST signal.

Register accesses cannot be performed to the PCnet-32 device while BOFF is asserted.

Effect of AHOLD

Assertion of AHOLD during Linear Burst transfers will cause the PCnet-32 controller to float some portion of the address bus beginning at the next clock cycle. If $\overline{BRDY}$ is returned while AHOLD is active, then the linear burst sequence will continue until the current burst would otherwise normally terminate, since the data bus and the lower portion of the address bus may remain active during AHOLD. However, a new linear burst sequence, requiring a new $\overline{ADS}$ assertion, will not be started while AHOLD is active.

When AHOLD is asserted during T1 of a linear burst, then the linear burst operation will be suspended until AHOLD is de-asserted. Once AHOLD is de-asserted, then the PCnet-32 controller will start the suspended linear burst with the intended address. See Figure 23. (Note that the intended T1 of the linear burst sequence has been labeled Ta in the figure, since a T1 was never executed due to the suspension of the address bus required by the assertion of AHOLD.)

When AHOLD is asserted in the middle of a linear burst, the linear burst may proceed without stalling or halting. AHOLD requires that PCnet-32 controller float a portion of its address bus, but linear burst data cycles will still proceed, since the AHOLD signal only affects a portion of the address bus, and that portion of the address bus is not being used for the middle accesses of a linear burst.

However, if AHOLD is asserted in the middle of a linear burst operation, and the AHOLD signal is held long enough that a new linear burst sequence will start (a new $\overline{ADS}$ is to be issued by the PCnet-32 controller) then at the end of the current linear sequence, the PCnet-32 controller must wait for the AHOLD signal to become inactive before beginning the next linear sequence, since the AHOLD signal would now interfere with the PCnet-32 controller's wish to assert $\overline{ADS}$ and a new address on the entire address bus. During the time that the PCnet-32 controller is waiting for the release of the AHOLD signal, the PCnet-32 controller will continue to drive the command signals, but $\overline{ADS}$ will be driven inactive.

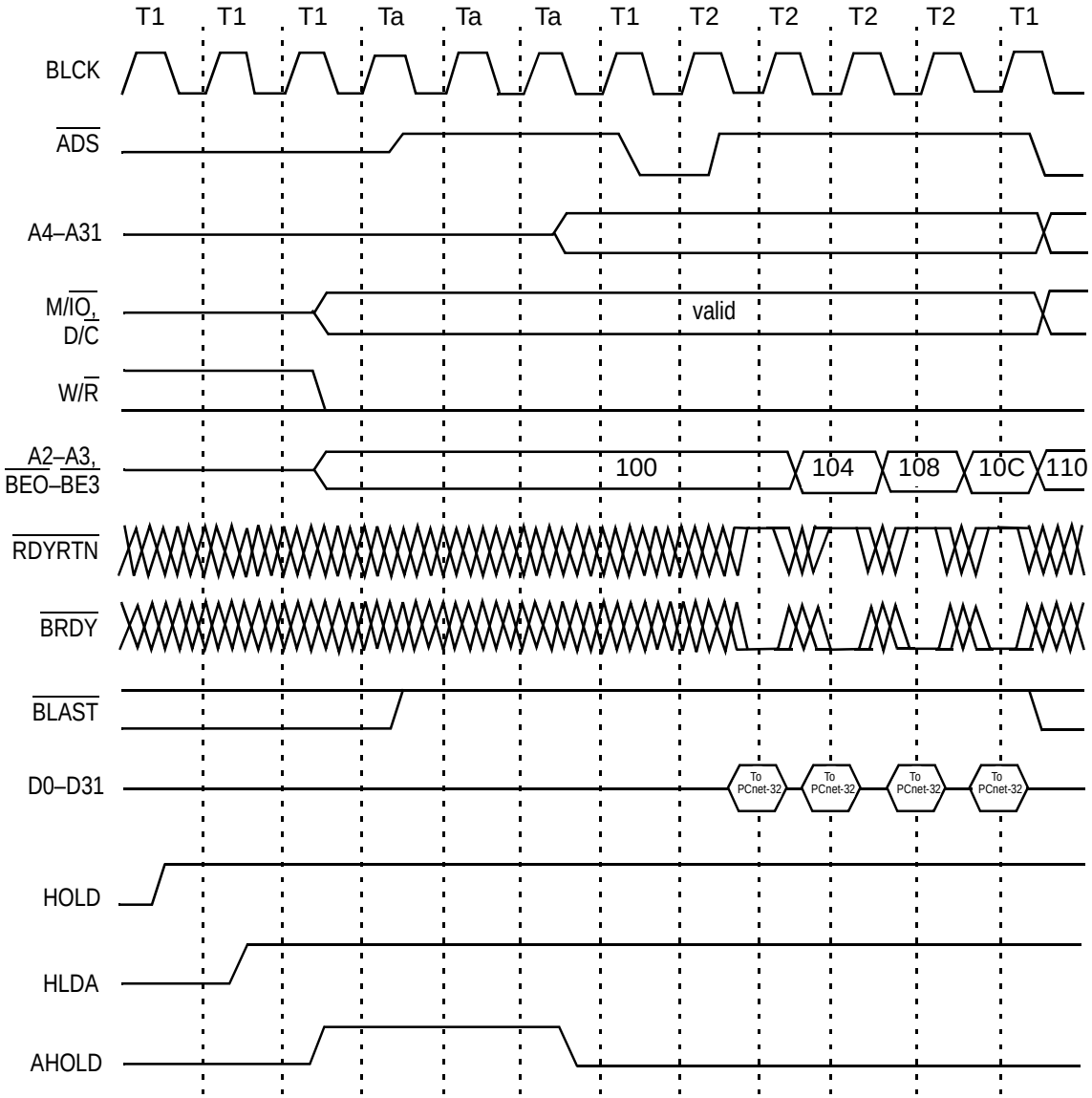
Note that if $\overline{RDYRTN}$ is asserted during a linear burst sequence while AHOLD is active, then no more access will be performed until AHOLD is de-asserted. This is because the assertion of $\overline{RDYRTN}$ will cause the PCnet-32 controller to insert a new T1 cycle into the linear burst. A T1 cycle requires assertion of $\overline{ADS}$, but $\overline{ADS}$ assertion is not allowed as long as AHOLD is still asserted. Therefore, the T1 cycle is delayed until the AHOLD is de-asserted.

The portion of the Address Bus that will be floated at the time of an address hold operation will be determined by the value of the Cache Line Length register (BCR18, bits 15–11). Table 23 lists all of the legal values of CLL showing the portion of the Address Bus that will become floated during an address hold operation.

Table 23. CLL Value of Floated Address in AHOLD

CLL Value	Floated Portion of Address Bus During AHOLD
00000	None
00001	A31–A2
00010	A31–A3
00011	Reserved CLL Value
00100	A31–A4
00101–00111	Reserved CLL Values
01000	A31–A5
01001–01111	Reserved CLL Values
10000	A31–A6
10001–11111	Reserved CLL Values

Note that the default value of CLL after H_RESET is 00100. All timing diagrams in this document are drawn with the assumption that this is the value of CLL.



18219-26

Figure 23. Linear Burst Read in which AHOLD was Asserted During T1 in the Linear Burst Sequence; Linear Burst Sequence is Started when AHOLD is De-asserted

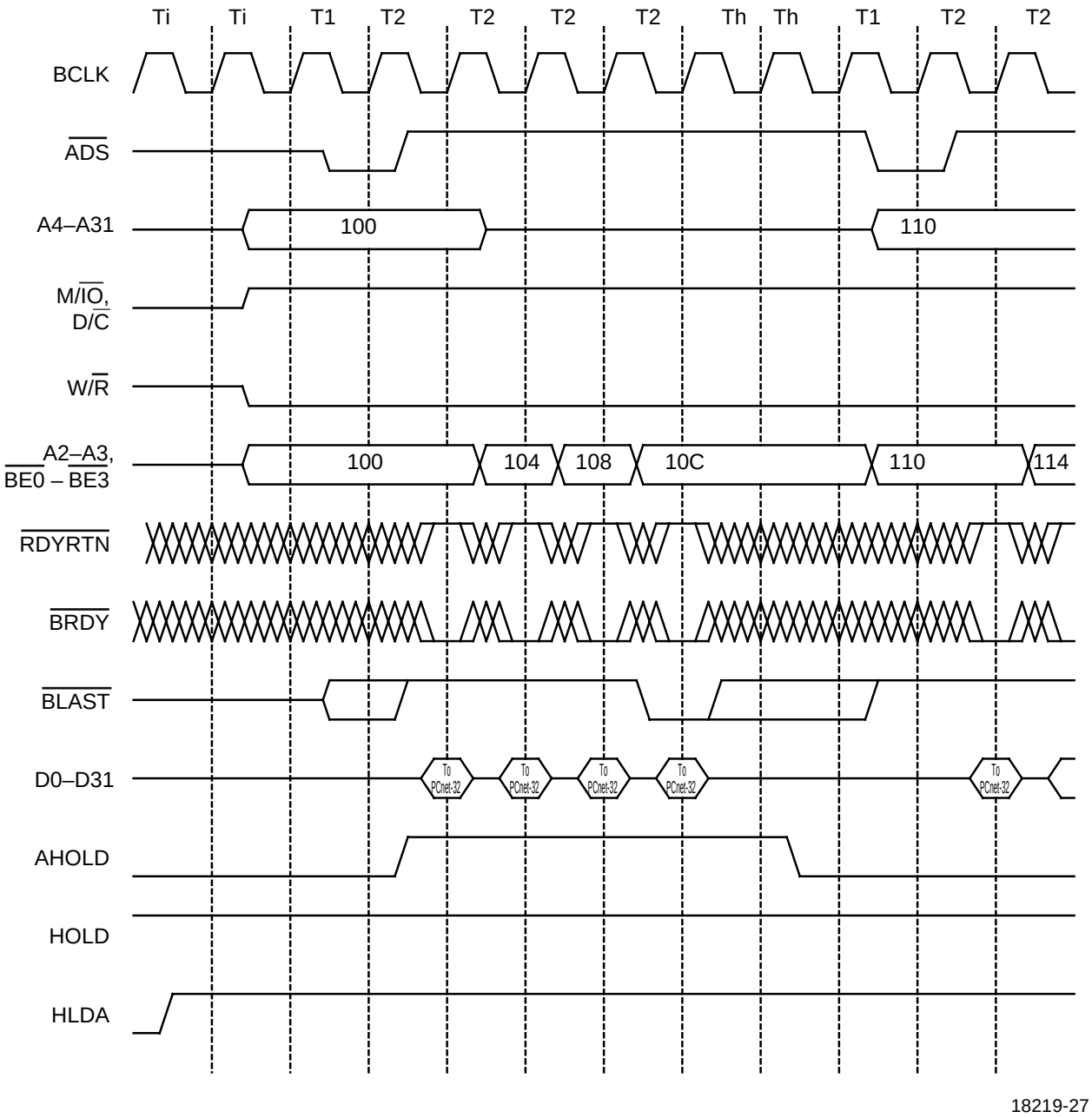


Figure 24. Linear Burst Read Cycle with AHOLD

Note that Linear Burst sequence is allowed to complete in spite of AHOLD, but next linear Burst sequence is prevented from beginning until AHOLD is de-asserted.

Note that if the BRDY (or RDYRTN) signal is not returned while AHOLD is active, then the PCnet-32 controller will resume driving the same address onto the address bus when AHOLD is released. The PCnet-32 controller will not reissue the ADS signal at this time.

Bus Activity Timer Register Time Out During Linear Burst

When the Bus Activity Timer Register (CSR82 bits [15:0]) times out before or concurrent with the execution of the first T2 cycle of the fourth from the last transfer cycle within a linear burst DMA sequence, then the linear burst will continue until a legal starting

address is reached, and then the PCnet-32 controller will relinquish the bus. If the Bus Activity Timer Register times out after the execution of the first T2 cycle of the fourth from the last transfer cycle within a linear burst DMA sequence, then the PCnet-32 controller will complete the current linear burst sequence and will execute a new linear burst sequence before releasing the HOLD signal and relinquishing the bus. (Effectively, the Bus Activity Timer time-out is treated in a manner identical to the occurrence of a preemption event.) Therefore, when programmed for Linear Burst mode, the PCnet-32 controller bus mastership time may exceed the Bus Activity Timer limit.

This is done because an immediate abort of the linear burst due to timer expiration would leave the current buffer pointer at an unaligned location. This would cause an address alignment of several ordinary cycles to be executed during the next FIFO DMA operation. Repeated occurrences of this nature would compromise the usefulness of the linear burst mode, since this would increase the number of non-linear burst cycles that are performed. This in turn would increase the bus bandwidth requirement of the PCnet-32 controller. Therefore, because the PCnet-32 controller Linear Burst operation does not strictly obey the Burst Timer, the user should program the Burst Timer value in such a manner as to include the expected linear burst release time. If the user has enabled the Linear Burst function, and wishes the PCnet-32 controller to limit bus activity to MAX_TIME ms, then the Burst Timer should be programmed to a value of:

MAX_TIME- [((3+lbs) x w + 10 + lbs) x (BCLK period)]

This is because the PCnet-32 controller may use as much as one “linear burst size” plus three transfers in order to complete the linear burst before releasing the bus.

As an example, if the linear burst size is 4 transfers, and the number of wait states for the system memory is 2, and the BCLK period is 30 ns and the MAX time allowed on the bus is 3 ms, then the Burst Timer should be programmed for:

MAX_TIME- [((3+lbs) x w + 10 + lbs)

x (BCLK period)];

3 ms – [(3 + 4) x 2 +10 + 4) x (30 ns)]

= 3 ms – (28 x 30 ns) = 3 – 0.84 ms = 2.16 ms.

Then, if the PCnet-32 controller’s Burst Timer times out after 2.16 ms when the PCnet-32 controller has completed all but the last three transfers of a linear burst, then the PCnet-32 controller *may* take as *much* as 0.84 ms to complete the bursts and release the bus. The bus release will occur at 2.16 + 0.84 = 3 ms.

Burst Cycle Time Out During Linear Burst

When the Burst Cycle (CSR80 bits [7:0]) times out in the middle of a linear burst, the linear burst will continue until a legal starting address is reached, and then the PCnet-32 controller will relinquish the bus.

The discussion for the Burst Cycle is identical to the discussion for the Bus Activity Timer Register, except that the quantities are in terms of transfers instead of in terms of time.

The equation for the proper burst register setting is:

Burst count setting = (desired_max DIV (length of linear burst in transfers)) x length of linear burst in transfers, where DIV is the operation that yields the INTEGER portion of the ^{3} operation.

Illegal Combinations of Watermark and LINBC

Certain combinations of watermark programming and LINBC programming may create situations where no linear bursting is possible, or where the FIFO may be excessively read or excessively written. Such combinations are declared as illegal.

Combinations of watermark settings and LINBC settings must obey the following relationship:

watermark (in bytes) Š LINBC (in bytes)

Combinations of watermark and LINBC settings that violate this rule may cause unexpected behavior.

Slave Timing

Slave timing in the PCnet-32 controller is designed to perform to both Am486 32-bit timing requirements and VESA VL-Bus timing requirements at the same time. Since the VESA VL-Bus is based upon Am486 bus timing, there is really little difference evident, except for hold-off requirements on the part of the slave driving the $\overline{RDY}$, $\overline{BRDY}$, and data signals when the high speed write signal is false. VESA VL-Bus requires that none of these signals are driven by the slave until the second T2 cycle when the high speed write signal is false. Since the PCnet-32 controller does not examine the high-speed write bit, it assumes that this signal is never true, and therefore always obeys the more stringent requirement of not being allowed to drive $\overline{RDY}$, $\overline{BRDY}$ and the data bus until the second T2. In addition, the PCnet-32 controller will drive $\overline{RDY}$ and $\overline{BRDY}$ inactive for one half BCLK cycle at the end of the slave access, immediately following the BCLK cycle in which the PCnet-32 controller asserted $\overline{RDY}$. Again, this behavior is required by the VESA VL-Bus specification, but it is not required for operation within an Am486 system. The PCnet-32 controller performs in this manner, regardless of the PCnet-32 controller mode setting.

Slave timing can generally be inferred from the bus master timing diagrams, with the exception of the following information:

PCnet-32 controller never responds with $\overline{BRDY}$ active during slave accesses. All PCnet-32 controller slave responses use only the $\overline{RDY}$ signal. $\overline{BRDY}$ will always be de-asserted during all PCnet-32 controller slave accesses. $\overline{RDY}$ is a PCnet-32 controller output signal. It is used during PCnet-32 controller slave accesses. $\overline{RDYRTN}$ is a PCnet-32 controller input signal. It is used during all PCnet-32 controller bus master accesses, as well as during PCnet-32 controller slave read accesses.

The typical number of wait states added to a slave access on the part of the PCnet-32 controller is 6 or 7 BCLK cycles, depending upon the relative phases of the internal Buffer Management Unit clock and the BCLK signal, since the internal Buffer Management Unit clock is a divide-by-two version of the BCLK signal.

The PCnet-32 controller $\overline{RDY}$ and $\overline{RDYRTN}$ signals may be wired together. This allows the PCnet-32 controller to operate within a system that has a single $\overline{READY}$ signal.

The $\overline{LDEV}$ signal is generated in response to a valid PCnet-32 controller I/O address on the bus together with a valid $\overline{ADS}$ signal. $\overline{LDEV}$ is generated in an asynchronous manner by the PCnet-32 controller. See the parameter listings for delay values of the $\overline{LDEV}$ signal. $\overline{RDY}$, $\overline{BRDY}$ and D[31:0] are never driven until the second T2 state of a slave access. Before that time, it is expected that a system pull-up device is holding the $\overline{RDY}$ and $\overline{BRDY}$ signals in a de-asserted state.

The $\overline{RDY}$ and $\overline{BRDY}$ signals are always driven high for one half BCLK cycle immediately following the BCLK period during which $\overline{RDY}$ was driven asserted. Then the $\overline{RDY}$ and $\overline{BRDY}$ signals are floated. This behavior is performed regardless of the PCnet-32 controller mode setting. See Figure 25.

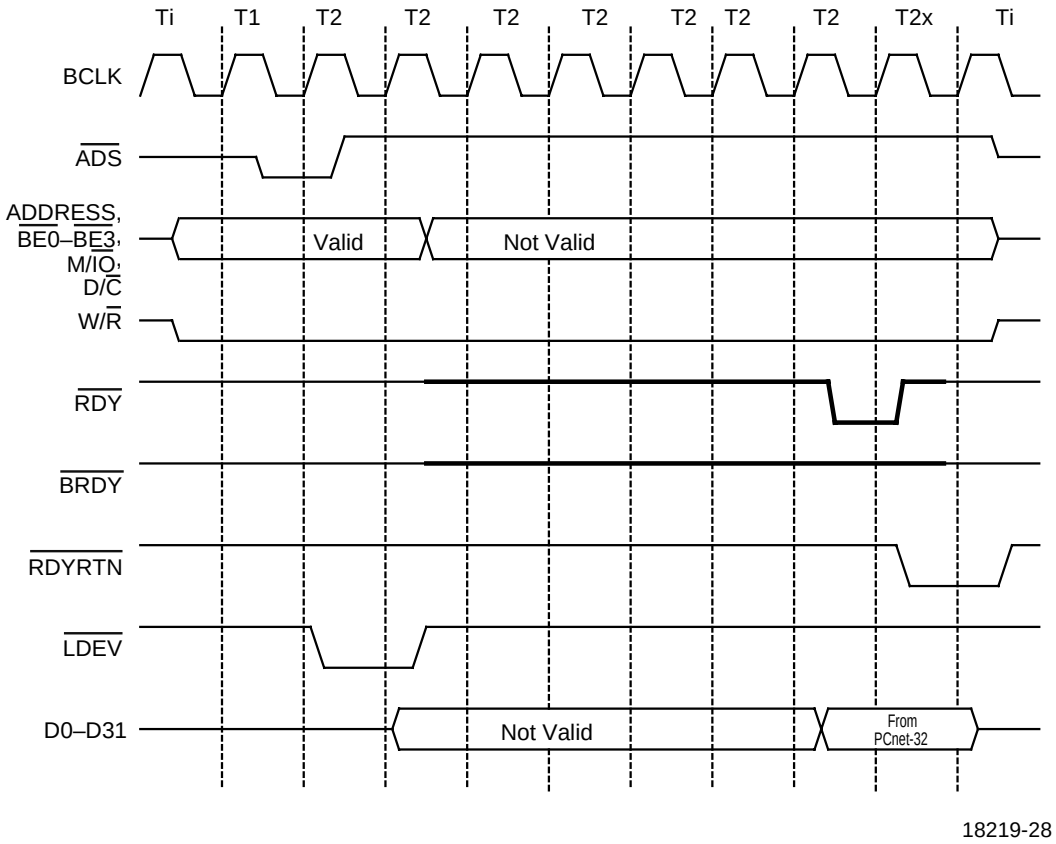


Figure 25. Slave $\overline{RDY}$ Timing

VESA VL-Bus Mode Timing

VESA VL-Bus mode functional timing is essentially identical to the timing of the Am486 32-bit mode, except that the bus request and bus acknowledge signals have inverted senses from those shown in the previous timing diagrams and the AHOLD signal does not exist while the PCnet-32 controller is programmed

for VL-Bus mode. In addition, dynamic bus sizing is supported in VESA VL-Bus mode, through the use of the $\overline{\text{LBS16}}$ signal. The following section describes possible $\overline{\text{LBS16}}$ interactions while programmed for the VESA VL-Bus mode of operation. Other differences exist between VL-Bus mode and Am486 mode, but these other differences are not directly related to the master or slave cycle timings.

Effect of $\overline{\text{LBS16}}$ (VL-Bus mode only)

Dynamic bus sizing is recognized by the PCnet-32 controller while operating in the VL-Bus mode. The $\overline{\text{LBS16}}$ signal is used to indicate to the PCnet-32 controller whether the VL-Bus target is a 16-bit or 32-bit peripheral. When the target device indicates that it is 16 bits in width by asserting the $\overline{\text{LBS16}}$ signal at least one LCLK period before asserting the $\overline{\text{BRDY}}$ signal, then the PCnet-32 controller will dynamically respond to the size constraints of the peripheral by performing additional accesses. Table 24 indicates the sequence of accesses that will be performed by the PCnet-32 controller in response to the assertion of $\overline{\text{LBS16}}$.

Table 24. Data Transfer Sequence from 32-Bit Wide to 16-Bit Wide

Current Access				Next with LBS16			
$\overline{\text{BE3}}$	$\overline{\text{BE2}}$	$\overline{\text{BE1}}$	$\overline{\text{BE0}}$	$\overline{\text{BE3}}$	$\overline{\text{BE2}}$	$\overline{\text{BE1}}$	$\overline{\text{BE0}}$
1	1	1	0	NR*			
1	1	0	0	NR*			
1	0	0	0	1	0	1	1
0	0	0	0	0	0	1	1
1	1	0	1	NR*			
1	0	0	1	1	0	1	1
0	0	0	1	0	0	1	1
1	0	1	1	NR*			
0	0	1	1	NR*			
0	1	1	1	NR*			

**NR = No second access Required for these cases.*

Figure 26 shows an example of an exchange between a 16-bit VL-Bus peripheral and the PCnet-32 controller during ordinary read cycles while programmed for VL-Bus mode of operation. Note that the $\overline{LBS16}$ signal is asserted during the LCLK that precedes the assertion of $\overline{RDYRTN}$. In this particular case, in order to maintain zero-wait state accesses, the 16-bit target must generate $\overline{LBS16}$ in a very short time in order to meet the required setup time of $\overline{LBS16}$ into the PCnet-32 controller. If the peripheral were incapable of meeting the required setup time, then a wait state would be needed in order to insure that $\overline{LBS16}$ is asserted at least one LCLK prior to the assertion of the $\overline{RDYRTN}$ signal. This situation is illustrated in the second double word access of the diagram. The wait state only needs to be inserted on the first access of the sequence, since from that point on, $\overline{LBS16}$ could be held active low until the entire double word transfer had completed, thus adequately satisfying the $\overline{LBS16}$ setup requirement for the second $\overline{RDYRTN}$ assertion. Note that only two bytes of data are transferred during each T2 cycle so that the total number of bytes transferred during each access is four.

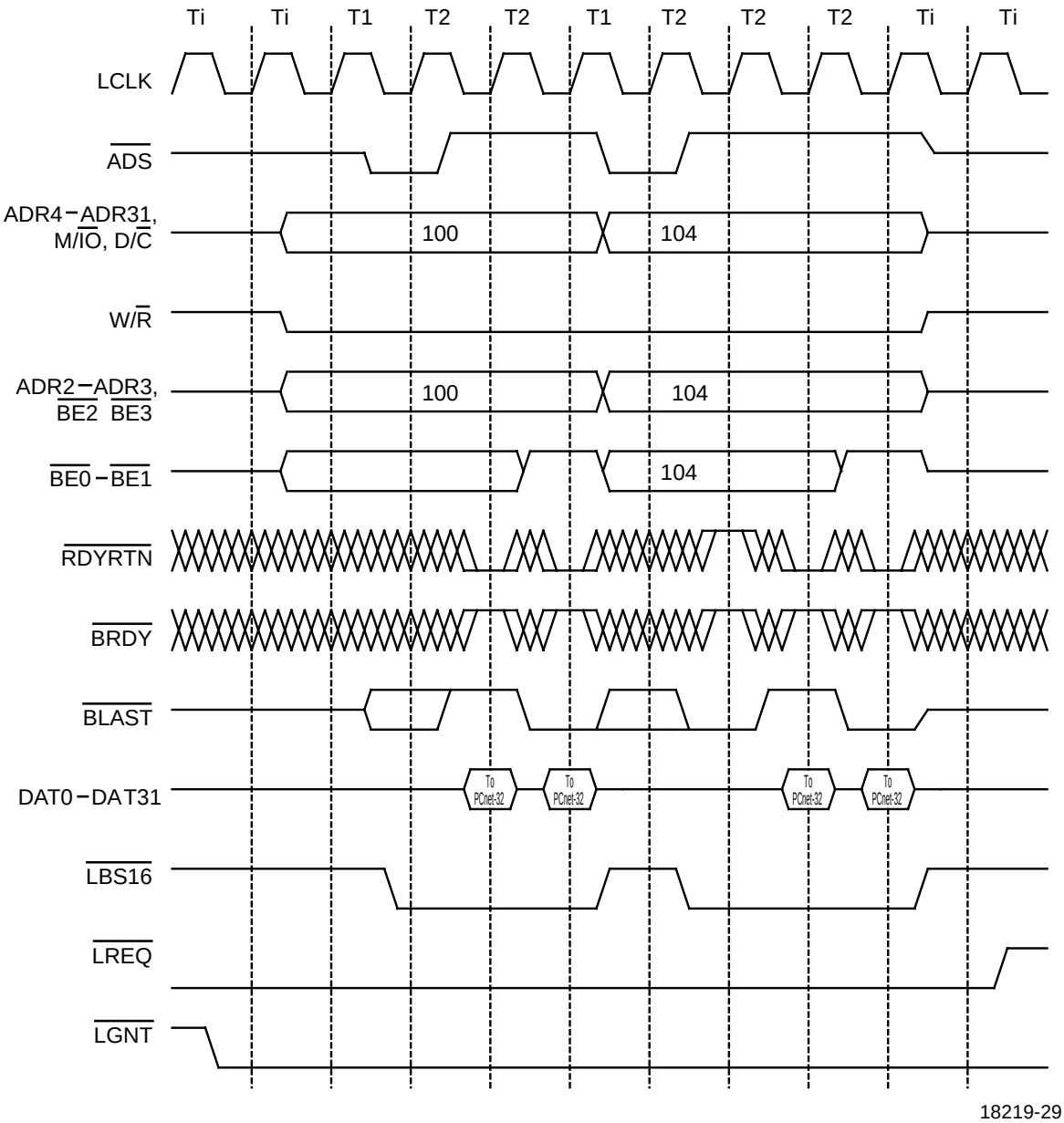


Figure 26. VL-Bus Basic Read with $\overline{LBS16}$

Figure 27 shows an example of an exchange between a 16-bit VL-Bus peripheral and the PCnet-32 controller during linear burst mode while programmed for VL-Bus mode of operation. Note that the $\overline{LBS16}$ signal is as-

serted during the LCLK that precedes the assertion of BRDY. In this particular case, in order to maintain zero-wait state accesses, the 16-bit target must generate LBS16 in a very short time in order to meet the required setup time of LBS16 into the PCnet-32 controller. If the peripheral were incapable of meeting the required setup time, then a wait state would be needed in order to insure that LBS16 is asserted at least one LCLK prior to the assertion of the BRDY signal. For linear burst sequences, this wait state would only need to be inserted on the first access of the sequence, since from that point on, LBS16 could be held active low until the entire sequence had completed, thus adequately satisfying the LBS16 setup requirement for each subsequent BRDY assertion. Note that only 2 bytes of data are transferred during each T2 cycle so that the total number of bytes transferred during the linear burst sequence is 16, even though 8 T2 cycles are executed.

When the assertion of LBS16 during a PCnet-32 controller master access has created the need for a second access as specified in the table above, **and** the WBACK signal becomes active during the second access, then when WBACK is de-asserted, the PCnet-32 controller will repeat both accesses of the pair.

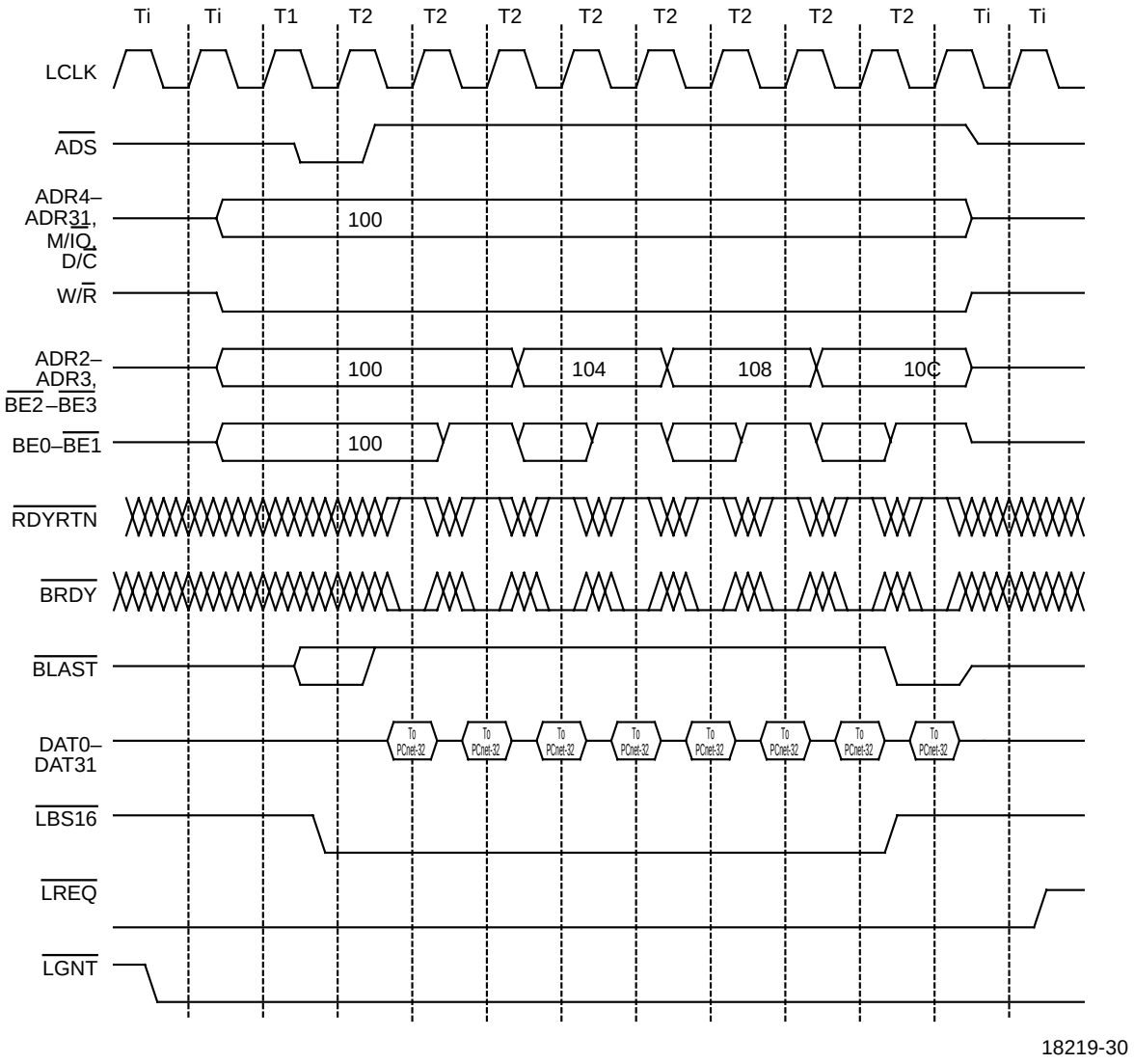


Figure 27. VL-Bus Linear Burst Read with LBS16

Bus Master and Bus Slave Data

Byte Placement

The general rule of data placement is that the active data byte lanes are indicated by the byte enable signals for all transfers. Note that during all master read operations, the PCnet-32 controller will always activate all byte enables, even though some byte lanes may not contain “valid” data as indicated by a buffer pointer value. In such instances, the PCnet-32 controller will internally discard unneeded bytes.

Note that in all 32-bit environments, regardless of the mode settings, the placement of data bytes on the data bus during all PCnet-32 controller bus operations (master and slave) will proceed in accordance with the data byte duplication rules of the Am386DX. The Am386DX requirement is for duplication of active data bytes in corresponding lower-half byte lanes when the access is a byte or word access that utilizes the upper half of the data bus. PCnet-32 controller performs data byte duplication in this manner. The Am386DX does not indicate byte duplication when the active data bytes of a byte or word access are exclusively contained in the lower half of the data bus, therefore, the PCnet-32 controller will not perform data byte duplication in this case. Byte duplication for bus master writes and bus slave reads will follow Table 25.

A[1:0] in the table refer to software pointers, since A[1:0] pins do not physically exist in the system. (Software pointers include I/O address software pointers in the driver code for I/O accesses to the PCnet-32 controller, or software pointers for the initialization block, descriptor areas or buffer areas that are used by the PCnet-32 controller during master accesses.)

For master read operations, the PCnet-32 controller expects data according to the byte enable signaling only. Byte lanes with inactive byte enables are expected to carry invalid data.

For slave write operations, the PCnet-32 controller expects data according to the byte enable signaling only. Byte lanes with inactive byte enables are expected to carry invalid data.

For master write operations, the PCnet-32 controller will produce data as indicated in Table 25.

For slave read operations, the PCnet-32 controller will produce data as indicated for the BSWP = 0 cases in Table 25, regardless of the actual setting of the BSWP bit.

Table 25. Master and Slave Byte Placement

Case No.	A[1:0]	BSWP	$\overline{BE3}$–$\overline{BE0}$	D[31:24]	D[23:16]	D[15:8]	D[7:0]
1a	00	0	0000	Byte3	Byte2	Byte1	Byte0
1b	00	1	0000	Byte0	Byte1	Byte2	Byte3
2b	01	1	1000	Undef	Byte0	Byte1	Byte2
3a	10	0	0011	Byte1	Byte0	Copy1	Copy0
3b	10	1	1100	Undef	Undef	Byte0	Byte1
4a	11	0	0111	Byte0	Undef	Copy0	Undef
4b	11	1	1110	Undef	Undef	Undef	Byte0
5a	00	0	1000	Undef	Byte2	Byte1	Byte0
5b	00	1	0001	Byte0	Byte1	Byte2	Undef
6a	01	0	0001	Byte2	Byte1	Byte0	Undef
6b	01	1	1000	Undef	Byte0	Byte1	Byte2
7a	10	0	0011	Byte1	Byte0	Copy1	Copy0
7b	10	1	1100	Undef	Undef	Byte0	Byte1
8a	11	0	0111	Byte0	Undef	Copy0	Undef
8b	11	1	1110	Undef	Undef	Undef	Byte0
9a	00	0	1100	Undef	Undef	Byte1	Byte0
9b	00	1	0011	Byte0	Byte1	Copy0	Copy1
10a	01	0	1001	Undef	Byte1	Byte0	Undef
10b	01	1	1001	Undef	Byte0	Byte1	Undef
11a	10	0	0011	Byte1	Byte0	Copy1	Copy0
11b	10	1	1100	Undef	Undef	Byte0	Byte1
12a	11	0	0111	Byte0	Undef	Copy0	Undef
12b	11	1	1110	Undef	Undef	Undef	Byte0
13a	00	0	1110	Undef	Undef	Undef	Byte0
13b	00	1	0111	Byte0	Undef	Copy0	Undef
14a	01	0	1101	Undef	Undef	Byte0	Undef
14b	01	1	1011	Undef	Byte0	Undef	Copy0
15a	10	0	1011	Undef	Byte0	Undef	Copy0
15b	10	1	1101	Undef	Undef	Byte0	Undef
16a	11	0	0111	Byte0	Undef	Copy0	Undef
16b	11	1	1110	Undef	Undef	Undef	Byte0

Note that cases 10, 12, and 15 will not normally be produced during PCnet-32 controller bus master operations. These cases will only occur during bus master operations if the software programs an extremely short buffer size into the descriptor BCNT field, where extremely short means exactly 4, 3, 2 or 1 bytes in length.

BSWP = 0 corresponds to little Endian byte ordering. BSWP = 1 corresponds to big Endian byte ordering.

Buffer Management Unit

The buffer management unit is a micro-coded state machine which implements the initialization procedure and manages the descriptors and buffers. The buffer management unit operates at a speed of BCLK 2.

Initialization

PCnet-32 controller initialization includes the reading of the initialization block in memory to obtain the operating parameters. The initialization block must be located on a double word (4-byte) address boundary, regardless of the setting of the SSIZE32, (CSR58[8]/ BCR20[8]) bit. The initialization block is read when the INIT bit in CSR0 is set. The INIT bit should be set before or concurrent with the STRT bit to insure correct operation. Two double-words are read during each period of bus mastership. When SSIZE32 = 1 (CSR58[8]/ BCR20[8]), this results in a total of 4 arbitration cycles (3 arbitration cycles if SSIZE32 = 0). Once the initialization block has been completely read in and internal registers have been updated, IDON will be set in CSR0, and an interrupt generated (if IENA is set). At this point, the BMU knows where the receive and transmit descriptor rings and hence, normal network operations will begin.

The Initialization Block is vectored by the contents of CSR1 (least significant 16 bits of address) and CSR2 (most significant 16 bits of address). The block contains the user defined conditions for PCnet-32 controller operation, together with the base addresses and length information of the transmit and receive descriptor rings.

There is an alternative method to initialize the PCnet-32 controller. Instead of initialization via the initialization block in memory, data can be written directly into the appropriate registers. Either method may be used at the discretion of the programmer. If the registers are written to directly, the INIT bit must not be set, or the initialization block will be read in, thus overwriting the previously written information. Please refer to Appendix C for details on this alternative method.

Re-Initialization

The transmitter and receiver sections of the PCnet-32 controller can be turned on via the initialization block (MODE Register DTX, DRX bits; CSR15[1:0]). The states of the transmitter and receiver are monitored by the host through CSR0 (RXON, TXON bits). The PCnet-32 controller should be reinitialized if the transmitter and/or the receiver were not turned on during the original initialization, and it was subsequently required to activate them or if either section was shut off due to the detection of an error condition (MER, UFLO, TX BUFF error).

Re-initialization may be done via the initialization block or by setting the STOP bit in CSR0, followed by writing to CSR15, and then setting the START bit in CSR0. Note that this form of restart will not perform the same in the PCnet-32 controller as in the LANCE. In particular, upon restart, the PCnet-32 controller reloads the transmit and receive descriptor pointers with their respective base addresses. This means that the software must clear the descriptor own bits and reset its descriptor ring pointers before the restart of the PCnet-32 controller. The reload of descriptor base addresses is performed in the LANCE only after initialization, so a restart of the LANCE without initialization leaves the LANCE pointing at the same descriptor locations as before the restart.

Buffer Management

Buffer management is accomplished through message descriptor entries organized as ring structures in memory. There are two rings, a receive ring and a transmit ring. The size of a message descriptor entry is 4 double-words, or 16 bytes, when SSIZE32 = 1. The size of a message descriptor entry is 4 words, or 8 bytes, when SSIZE32 = 0 (CSR58[8]/BCR20[8]).

Descriptor Rings

Each descriptor ring must be organized in a contiguous area of memory. At initialization time (setting the INIT bit in CSR0), the PCnet-32 controller reads the user-defined base address for the transmit and receive descriptor rings, as well as the number of entries contained in the descriptor rings. Descriptor ring base addresses must be on a 16-byte boundary when SSIZE32 = 1, or on an 8-byte boundary when SSIZE32 = 0. A maximum of 128 (or 512, depending upon the value of SSIZE32) ring entries is allowed when the ring length is set through the TLEN and RLEN fields of the initialization block. However, the ring lengths can be set beyond this range (up to 65535) by writing the transmit and receive ring length registers (CSR76, CSR78) directly.

Each ring entry contains the following information:

1. The address of the actual message data buffer in user or host memory
2. The length of the message buffer
3. Status information indicating the condition of the buffer

To permit the queuing and dequeuing of message buffers, ownership of each buffer is allocated to either the PCnet-32 controller or the host. The OWN bit within the descriptor status information, either TMD or RMD (see section on TMD or RMD), is used for this purpose. OWN = “1” signifies that the PCnet-32 controller currently has ownership of this ring descriptor and its associated buffer. Only the owner is permitted to relinquish ownership or to write to any field in the

descriptor entry. A device that is not the current owner of a descriptor entry cannot assume ownership or change any field in the entry. A device may, however, read from a descriptor that it does not currently own. Software should always read descriptor entries in sequential order. When software finds that the current descriptor is owned by the PCnet-32 controller, then the software must not read "ahead" to the next descriptor. The software should wait at the unOWNed descriptor until ownership has been granted to the software (when SPRINTEN = 1 (CSR3, bit5), then this rule is modified. See the SPRINTEN description). Strict adherence to these rules insures that "Deadly Embrace" conditions are avoided.

Descriptor Ring Access Mechanism

At initialization, the PCnet-32 controller reads the base address of both the transmit and receive descriptor rings into CSRs for use by the PCnet-32 controller during subsequent operations.

As the final step in the self-initialization process, the base address of each ring is loaded into each of the current descriptor address registers and the address of the next descriptor entry in the transmit and receive rings is computed and loaded into each of the next descriptor address registers.

When SSIZE32 = 0, software data structures are 16 bits wide. Figure 28 illustrates the relationship between the Initialization Base Address, the Initialization Block, the Receive and Transmit Descriptor Ring Base Addresses, the Receive and Transmit Descriptors and the Receive and Transmit Data Buffers, for the case of SSIZE32 = 0.

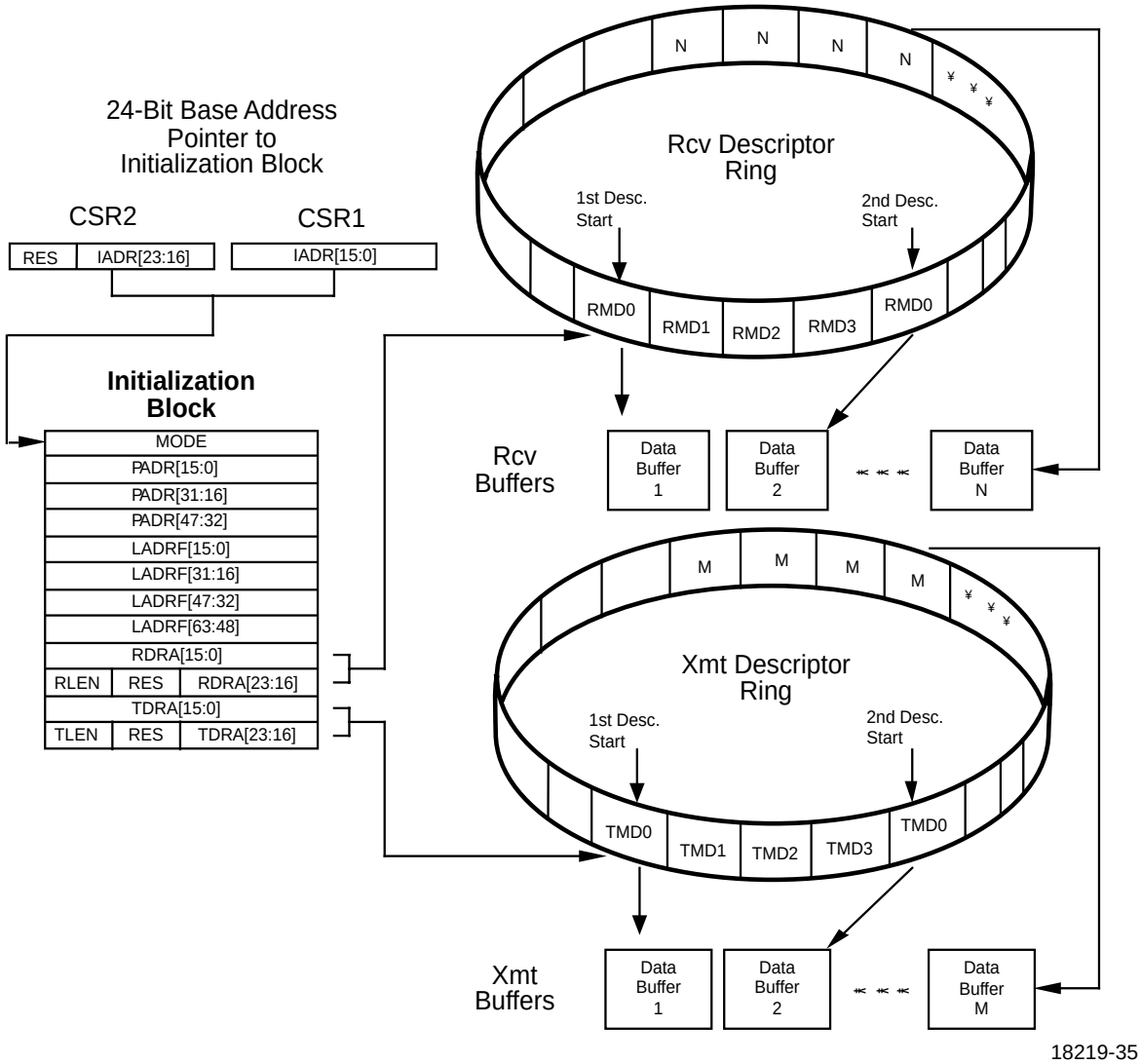


Figure 28. 16-Bit Initialization Block Descriptor Rings

When SSIZE32 = 1, software data structures are 32 bits wide. Figure 29 illustrates the relationship between the Initialization Base Address, the Initialization Block, the Receive and Transmit Descriptor Ring Base Addresses (TDRA/RDRA), the Receive and Transmit Descriptors and the Receive and Transmit Data Buffers, for the case of SSIZE32 = 1.

Polling

If there is no network channel activity and there is no pre- or post-receive or pre- or post-transmit activity being performed by the PCnet-32 controller, then the PCnet-32 controller will periodically poll the current receive and transmit descriptor entries in order to ascertain their ownership. If the DPOLL bit in CSR4 is set, then transmit polling function is disabled.

A typical polling operation consists of the following: The PCnet-32 controller will use the current receive descriptor address stored internally to vector to the appropriate Receive Descriptor Table Entry (RDTE). It will then use the current transmit descriptor address (stored internally) to vector to the appropriate Transmit Descriptor Table Entry (TDTE). The accesses will be made in the following order: RMD1, then RMD0 of the current RDTE during one bus arbitration, and after that, TMD1, then TMD0 of the current TDTE during a second bus arbitration. All information collected during polling activity will be stored internally in the appropriate CSRs (i.e. CSR18, CSR19, CSR20, CSR21, CSR40, CSR42, CSR50, CSR52). UnOWNed descriptor status will be internally ignored.

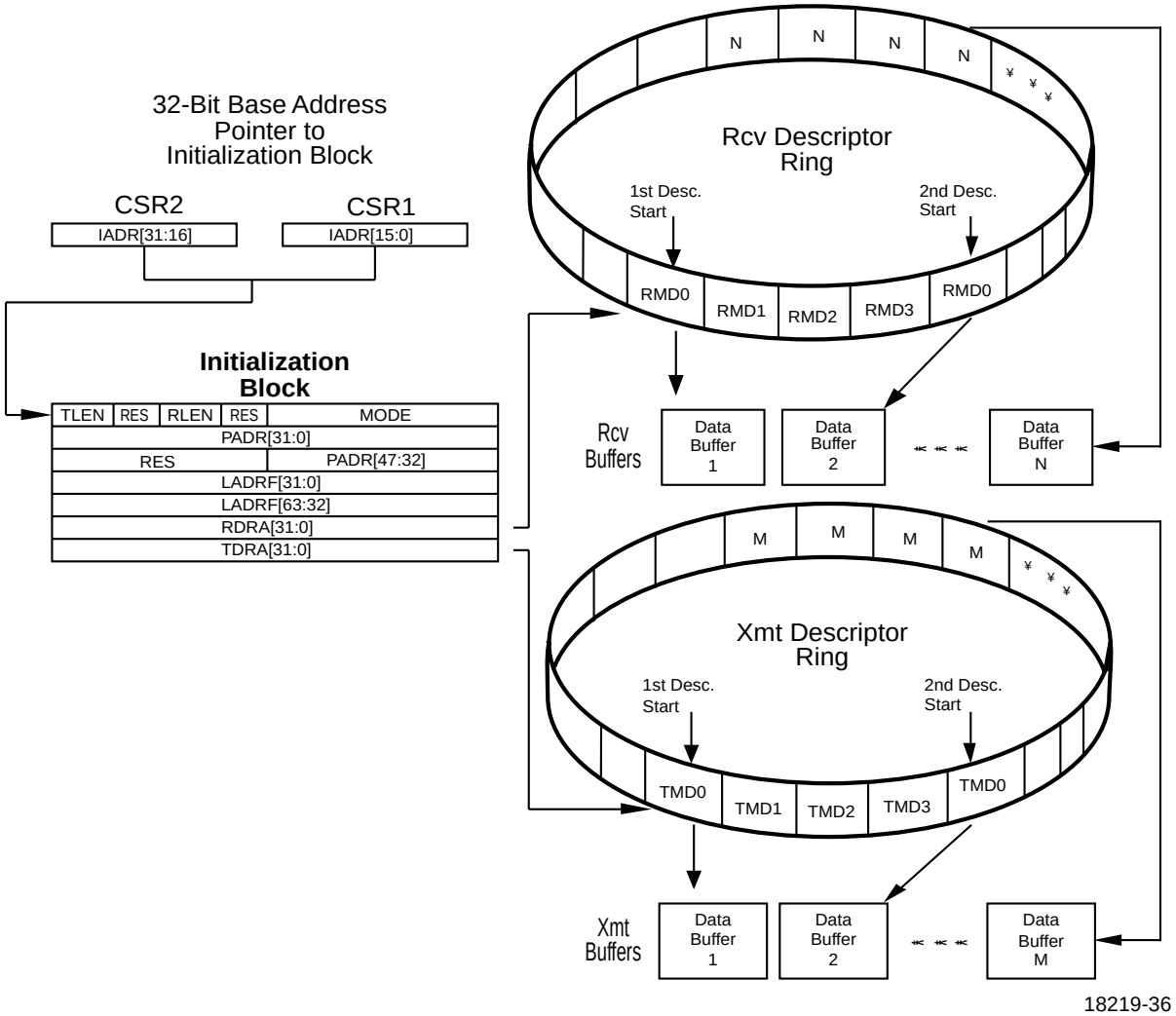


Figure 29. 32-Bit Initialization Block and Descriptor Ring

A typical receive poll is the product of the following conditions:

1. PCnet-32 controller does not possess ownership of the current RDTE and the poll time has elapsed and RXON = 1, or

2. PCnet-32 controller does not possess ownership of the next RDTE the poll time has elapsed and RXON = 1.

If RXON = 0 the PCnet-32 controller will never poll RDTE locations.

The ideal system should always have at least one RDTE available for the possibility of an unpredictable receive event. (This condition is not a requirement. If this condition is not met, it simply means that frames will be missed by the system because there was no buffer space available.) But the typical system usually has at least one or two RDTEs available for the possibility of an unpredictable receive event. Given that this condition is satisfied, the current and next RDTE polls are rarely seen and hence, the typical poll operation simply consists of a check of the status of the current TDTE. When there is only one RDTE (because the RLEN was set to zero), then there is no "next RDTE" and ownership of "next RDTE" cannot be checked. If there is at least one RDTE, the RDTE poll will rarely be seen and the typical poll operation simply consists of a check of the current TDTE.

A typical transmit poll is the product of the following conditions:

1. PCnet-32 controller does not possess ownership of the current TDTE and
   DPOLL = 0 and
   TXON = 1 and
   the poll time has elapsed, or
2. PCnet-32 controller does not possess ownership of the current TDTE and
   DPOLL = 0 and
   TXON = 1 and
   a frame has just been received, or
3. PCnet-32 controller does not possess ownership of the current TDTE and
   DPOLL = 0 and
   TXON = 1 and
   a frame has just been transmitted.

Setting the TDMD bit of CSR0 will cause the microcode controller to exit the poll counting code and immediately perform a polling operation. If RDTE ownership has not been previously established, then an RDTE poll will be performed ahead of the TDTE poll. If the microcode is not executing the poll counting code when the TDMD bit is set, then the demanded poll of the TDTE will be delayed until the microcode returns to the poll counting code.

The user may change the poll time value from the default of 65,536 BCLK periods by modifying the value in the Polling Interval register (CSR47). Note that if a non-default value is desired, then a strict sequence of setting the INIT bit in CSR0, waiting for IDON (CSR0[8]), then writing to CSR47, and then setting STRT in CSR0 must be observed, otherwise the default value will not be overwritten. See the CSR47 section for details.

Transmit Descriptor Table Entry (TDTE)

If, after a TDTE access, the PCnet-32 controller finds that the OWN bit of that TDTE is not set, then the PCnet-32 controller resumes the poll time count and reexamines the same TDTE at the next expiration of the poll time count.

If the OWN bit of the TDTE is set, but STP = 0, the PCnet-32 controller will immediately request the bus in order to reset the OWN bit of this descriptor. (This condition would normally be found following a LCOL or RETRY error that occurred in the middle of a transmit frame chain of buffers.) After resetting the OWN bit of this descriptor, the PCnet-32 controller will again immediately request the bus in order to access the next TDTE location in the ring.

If the OWN bit is set and the buffer length is 0, the OWN bit will be reset. In the LANCE the buffer length of 0 is interpreted as a 4096-byte buffer. It is acceptable to have a 0 length buffer on transmit with STP = 1 or STP = 1 *and* ENP = 1. It is not acceptable to have 0 length buffer with STP = 0 and ENP =1.

If the OWN bit is set and the start of packet (STP) bit is set, then microcode control proceeds to a routine that will enable transmit data transfers to the FIFO. The PCnet-32 controller will look ahead to the next transmit descriptor after it has performed at least one transmit data transfer from the first buffer. (More than one transmit data transfer may possibly take place, depending upon the state of the transmitter.) The contents of TMD0 and TMD1 will be stored in Next Xmt Buffer Address (CSR64 and CSR65), Next Xmt Byte Count (CSR66) and Next Xmt Status (CSR67) regardless of the state of the OWN bit. This transmit descriptor look-ahead operation is performed only once.

If the PCnet-32 controller does not own the next TDTE (i.e. the second TDTE for this frame), then it will complete transmission of the current buffer and then update the status of the current (first) TDTE with the BUFF and UFLO bits being set. This will cause the transmitter to be disabled (CSR0, TXON=0). The PCnet-32 controller will have to be re-initialized to restore the transmit function. The situation that matches this description implies that the system has not been able to stay ahead of the PCnet-32 controller in the transmit descriptor ring and therefore, the condition is treated as a fatal error. (To avoid this situation, the system should always set the transmit chain descriptor own bits in reverse order.)

If the PCnet-32 controller does own the second TDTE in a chain, it will gradually empty the contents of the first buffer (as the bytes are needed by the transmit opera-

tion), perform a single-cycle DMA transfer to update the status of the first descriptor (reset the OWN bit in TMD1), and then it may perform one data DMA access on the second buffer in the chain before executing another look-ahead operation (i.e. a look-ahead to the third descriptor.)

The PCnet-32 controller can queue up to two frames in the transmit FIFO. Call them frame “X” and frame “Y”, where “Y” is after “X”. Assume that frame “X” is currently being transmitted. Because the PCnet-32 controller can perform look-ahead data transfer past the ENP of frame “X”, it is possible for the PCnet-32 controller to completely transfer the data from a buffer belonging to frame “Y” into the FIFO even though frame “X” has not yet been completely transmitted. At the end of this “Y” buffer data transfer, the PCnet-32 controller will write intermediate status (change the OWN bit to a zero) for the “Y” frame buffer, if frame “Y” uses data chaining. The last TDTE for the “X” frame (containing ENP) has not yet been written, since the “X” frame has not yet been completely transmitted. Note that the PCnet-32 controller has, in this instance, returned ownership of a TDTE to the host out of a “normal” sequence. For this reason, it becomes imperative that the host system should never read the Transmit DTE ownership bits out of order.

There should be no problems for software which processes buffers in sequence, waiting for ownership before proceeding.

If an error occurs in the transmission before all of the bytes of the current buffer have been transferred, then TMD2 and TMD1 of the current buffer will be written. In such a case, data transfers from the next buffer will not commence. Instead, following the TMD2/TMD1 update, the PCnet-32 controller will go to the next transmit frame, if any, skipping over the rest of the frame which experienced an error, including chained buffers. This is done by returning to the polling microcode where PCnet-32 controller will immediately access the next descriptor and find the condition OWN=1 and STP=0 as described earlier. As described for that case, the PCnet-32 controller will reset the own bit for this descriptor and continue in like manner until a descriptor with OWN=0 (no more transmit frames in the ring) or OWN=1 and STP=1 (the first buffer of a new frame) is reached.

At the end of any transmit operation, whether successful or with errors, immediately following the completion of the descriptor updates, the PCnet-32 controller will al ways perform another poll operation. As described earlier, this poll operation will begin with a check of the current RDTE, unless the PCnet-32 controller already owns that descriptor. Then the PCnet-32 controller will proceed to polling the next TDTE. If the transmit descriptor OWN bit has a zero value, then the PCnet-32 controller will resume poll time count incrementing. If the transmit descriptor OWN bit has a value of ONE, then the PCnet-32 controller will begin filling the FIFO with transmit data and initiate a transmission. This end-of-operation poll coupled with the TDTE look-ahead operation allows the PCnet-32 controller to avoid inserting poll time counts between successive transmit frames.

Whenever the PCnet-32 controller completes a transmit frame (either with or without error) and writes the status information to the current descriptor, then the TINT bit of CSR0 is set to indicate the completion of a transmission. This causes an interrupt signal if the IENA bit of CSR0 has been set and the TINTM bit of CSR3 is reset.

Receive Descriptor Table Entry (RDTE)

If the PCnet-32 controller does not own both the current and the next Receive Descriptor Table Entry then the PCnet-32 controller will continue to poll according to the polling sequence described above. If the receive descriptor ring length is 1, then there is no next descriptor to be polled.

If a poll operation has revealed that the current and the next RDTE belong to the PCnet-32 controller then additional poll accesses are not necessary. Future poll operations will not include RDTE accesses as long as the PCnet-32 controller retains ownership of the current and the next RDTE.

When receive activity is present on the channel, the PCnet-32 controller waits for the complete address of the message to arrive. It then decides whether to accept or reject the frame based on all active addressing schemes. If the frame is accepted the PCnet-32 controller checks the current receive buffer status register CRST (CSR41) to determine the ownership of the current buffer.

If ownership is lacking, then the PCnet-32 controller will immediately perform a (last ditch) poll of the current RDTE. If ownership is still denied, then the PCnet-32 controller has no buffer in which to store the incoming message. The MISS bit will be set in CSR0 and an interrupt will be generated if INEA=1 (CSR0) and MISSM=0 (CSR3). Another poll of the current RDTE will not occur until the frame has finished.

If the PCnet-32 controller sees that the last poll (either a normal poll, or the last-ditch effort described in the above paragraph) of the current RDTE shows valid ownership, then it proceeds to a poll of the next RDTE. Following this poll, and regardless of the outcome of this poll, transfers of receive data from the FIFO may begin.

Regardless of ownership of the second receive descriptor, the PCnet-32 controller will continue to perform receive data DMA transfers to the first buffer, using burst-cycle DMA transfers. If the frame length

exceeds the length of the first buffer, and the PCnet-32 controller does not own the second buffer, ownership of the current descriptor will be passed back to the system by writing a zero to the OWN bit of RMD1 and status will be written indicating buffer (BUFF=1) and possibly overflow (OFLO=1) errors.

If the frame length exceeds the length of the first (current) buffer, and the PCnet-32 controller does own the second (next) buffer, ownership will be passed back to the system by writing a zero to the OWN bit of RMD1 when the first buffer is full. Receive data transfers to the second buffer may occur before the PCnet-32 controller proceeds to look ahead to the ownership of the third buffer. Such action will depend upon the state of the FIFO when the status has been updated on the first descriptor. In any case, look-ahead will be performed to the third buffer and the information gathered will be stored in the chip, regardless of the state of the ownership bit. As in the transmit flow, look-ahead operations are performed only once.

This activity continues until the PCnet-32 controller recognizes the completion of the frame (the last byte of this receive message has been removed from the FIFO). The PCnet-32 controller will subsequently update the current RDTE status with the end of frame (ENP) indication set, write the message byte count (MCNT) of the complete frame into RMD2 and overwrite the “current” entries in the CSRs with the “next” entries.

Media Access Control

The Media Access Control engine incorporates the essential protocol requirements for operation of a compliant Ethernet/802.3 node, and provides the interface between the FIFO subsystem and the Manchester Encoder/ Decoder (MENDEC).

The MAC engine is fully compliant to Section 4 of ISO/IEC 8802-3 (ANSI/IEEE Standard 1990 Second edition) and ANSI/IEEE 802.3 (1985).

The MAC engine provides programmable enhanced features designed to minimize host supervision, bus utilization, and pre- or post-message processing. These include the ability to disable retries after a collision, dynamic FCS generation on a frame-by-frame basis, and automatic pad field insertion and deletion to enforce minimum frame size attributes and reduces bus bandwidth use.

The two primary attributes of the MAC engine are:

- Transmit and receive message data encapsulation.
  - — Framing (frame boundary delimitation, frame synchronization)
  - — Addressing (source and destination address handling)
  - — Error detection (physical medium transmission errors)
- Media access management.
  - — Medium allocation (collision avoidance)
  - — Contention resolution (collision handling)

Transmit and Receive Message Data Encapsulation

The MAC engine provides minimum frame size enforcement for transmit and receive frames. When APAD_XMT = 1 (CSR4[11]), transmit messages will be padded with sufficient bytes (containing 00h) to ensure that the receiving station will observe an information field (destination address, source address, length/type, data and FCS) of 64 bytes. When ASTRP_RCV = 1 (CSR4[10]), the receiver will automatically strip pad bytes from the received message by observing the value in the length field, and stripping excess bytes if this value is below the minimum data size (46 bytes). Both features can be independently overridden to allow illegally short (less than 64 bytes of frame data) messages to be transmitted and/or received. Use of this feature decreases bus usage because the pad bytes are not transferred into or out of host memory.

Framing (Frame Boundary Delimitation, Frame Synchronization)

The MAC engine will autonomously handle the construction of the transmit frame. Once the Transmit FIFO has been filled to the predetermined threshold (set by XMTSP in CSR80), and providing access to the channel is currently permitted, the MAC engine will commence the 7-byte preamble sequence (10101010b, where first bit transmitted is a 1). The MAC engine will subsequently append the Start Frame Delimiter (SFD) byte (10101011b) followed by the serialized data from the Transmit FIFO. Once the data has been completed, the MAC engine will append the FCS (most significant bit first) which was computed on the message (destination address, source address, length field, data field, and pad (if applicable)).

Note that the user is responsible for the correct ordering and content in each of the fields in the frame, including the destination address, source address, length/type and frame data.

The receive section of the MAC engine will detect an incoming preamble sequence and lock to the encoded clock. The internal MENDEC will decode the serial bit stream and present this to the MAC engine. The MAC will discard the first 8-bits of information before searching for the SFD sequence. Once the SFD is detected, all subsequent bits are treated as part of the frame. The MAC engine will inspect the length field to ensure minimum frame size, strip unnecessary pad characters (if enabled), and pass the remaining bytes through the Receive FIFO to the host. If pad stripping is performed, the MAC engine will also strip the received FCS bytes,

although the normal FCS computation and checking will occur. Note that apart from pad stripping, the frame will be passed unmodified to the host. If the length field has a value of 46 or greater, the MAC engine will not attempt to validate the length against the number of bytes contained in the message.

If the frame terminates or suffers a collision before 64 bytes of information (after SFD) have been received, the MAC engine will automatically delete the frame from the Receive FIFO, without host intervention.

Addressing (Source and Destination Address Handling)

The first 6 bytes of information after SFD will be interpreted as the destination address field. The MAC engine provides facilities for physical, logical (multicast) and broadcast address reception. In addition, multiple physical addresses can be constructed (perfect address filtering) using external logic in conjunction with the EADI interface.

Error Detection (Physical Medium Transmission Errors)

The MAC engine provides several facilities which report and recover from errors on the medium. In addition, the network is protected from gross errors due to inability of the host to keep pace with the MAC engine activity.

On completion of transmission, the following transmit status is available in the appropriate TMD and CSR areas:

- The number of transmission retry attempts (ONE, MORE or RTRY).
- Whether the MAC engine had to Defer (DEF) due to channel activity.
- Excessive deferral (EXDEF), indicating that the transmitter has experienced Excessive Deferral on this transmit frame, where Excessive Deferral is defined in ISO 8802-3 (IEEE/ANSI 802.3).
- Loss of Carrier (LCAR) indicates that there was an interruption in the ability of the MAC engine to monitor its own transmission. Repeated LCAR errors indicate a potentially faulty transceiver or network connection.
- Late Collision (LCOL) indicates that the transmission suffered a collision after the slot time. This is indicative of a badly configured network. Late collisions should not occur in a normal operating network.
- Collision Error (CER) indicates that the transceiver did not respond with an SQE Test message within the predetermined time after a transmission completed. This may be due to a failed transceiver, disconnected or faulty transceiver drop cable, or the fact the transceiver does not support this feature (or it is disabled).

In addition to the reporting of network errors, the MAC engine will also attempt to prevent the creation of any network error due to the inability of the host to service the MAC engine. During transmission, if the host fails to keep the Transmit FIFO filled sufficiently, causing an underflow, the MAC engine will guarantee the message is either sent as a runt packet (which will be deleted by the receiving station) or has an invalid FCS (which will also cause the receiver to reject the message).

The status of each receive message is available in the appropriate RMD and CSR areas. FCS and Framing errors (FRAM) are reported, although the received frame is still passed to the host. The error will only be reported if an FCS error is detected and there are a non integral number of bytes in the message. The MAC engine will ignore up to 7 additional bits at the end of a message (dribbling bits), which can occur under normal network operating conditions. The reception of 8 additional bits will cause the MAC engine to deserialize the entire byte, and will result in the received message and FCS being modified.

The PCnet-32 controller can handle up to 7 dribbling bits when a received frame terminates. During the reception, the FCS is generated on every serial bit (including the dribbling bits) coming from the cable, although the internally saved FCS value is only updated on the eighth bit (on each byte boundary). The framing error is reported to the user as follows:

- If the number of dribbling bits are 1 to 7 and there is no CRC (FCS) error, then there is no Framing error (FRAM = 0).
- If the number of dribbling bits are 1 to 7 and there is a CRC (FCS) error, then there is also a Framing error (FRAM = 1).
- If the number of dribbling bits = 0, then there is no Framing error. There may or may not be a CRC (FCS) error.

Counters are provided to report the Receive Collision Count and Runt Packet Count for network statistics and utilization calculations.

Note that if the MAC engine detects a received frame which has a 00b pattern in the preamble (after the first 8-bits which are ignored), the entire frame will be ignored. The MAC engine will wait for the network to go inactive before attempting to receive additional frames.

Media Access Management

The basic requirement for all stations on the network is to provide fairness of channel allocation. The 802.3/Ethernet protocols define a media access mechanism which permits all stations to access the channel with equality. Any node can attempt to contend for the channel by waiting for a predetermined time (Inter

Packet Gap internal) after the last activity, before transmitting on the media. The channel is a multidrop communications media (with various topological configurations permitted) which allows a single station to transmit and all other stations to receive. If two nodes simultaneously contend for the channel, their signals will interact causing loss of data, defined as a collision. It is the responsibility of the MAC to attempt to avoid and recover from a collision, to guarantee data integrity for the end-to-end transmission to the receiving station.

Medium Allocation

The IEEE/ANSI 802.3 Standard (ISO/IEC 8802-3 1990) requires that the CSMA/CD MAC monitor the medium for traffic by watching for carrier activity. When carrier is detected, the media is considered busy, and the MAC should defer to the existing message.

The ISO 8802-3 (IEEE/ANSI 802.3) Standard also allows optional two-part deferral after a receive message.

See ANSI/IEEE Std 802.3-1990 Edition, 4.2.3.2.1:

"Note: *It is possible for the PLS carrier sense indication to fail to be asserted during a collision on the media. If the deference process simply times the inter packet gap based on this indication it is possible for a short inter packet gap to be generated, leading to a potential reception failure of a subsequent frame. To enhance system robustness the following optional measures, as specified in 4.2.8, are recommended when Inter Frame Spacing Part 1 is other than zero:*

1. *Upon completing a transmission, start timing the interpacket gap, as soon as transmitting and carrier sense are both false.*
2. *When timing an inter packet gap following reception, reset the inter packet gap timing if carrier sense becomes true during the first 2/3 of the inter-packet gap timing interval. During the final 1/3 of the interval the timer shall not be reset to en sure fair access to the medium. An initial period shorter than 2/3 of the interval is permissible in cluding zero.*

The MAC engine implements the optional receive two part deferral algorithm, with a first part inter-frame-spacing time of 6.0 µs. The second part of the inter-frame-spacing interval is therefore 3.6 µs.

The PCnet-32 controller will perform the two part deferral algorithm as specified in Section 4.2.8 (Process Deference). The Inter Packet Gap (IPG) timer will start timing the 9.6 µs Inter Frame Spacing after the receive carrier is de-asserted. During the first part deferral (Inter Frame Spacing Part1 - IFS1) the PCnet-32 controller will defer any pending transmit frame and respond to the receive message. The IPG counter will be reset to zero continuously until the carrier de-asserts, at which point the IPG counter will resume the 9.6 µs count once again. Once the IFS1 period of 6.0 µs has elapsed, the PCnet-32 controller will begin timing the second part deferral (Inter Frame Spacing Part 2 - IFS2) of 3.6 µs. Once IFS1 has completed, and IFS2 has commenced, the PCnet-32 controller will not defer to a receive frame if a transmit frame is pending. This means that the PCnet-32 controller will not attempt to receive the receive frame, since it will start to transmit, and generate a collision at 9.6 µs. The PCnet-32 controller will guarantee to complete the preamble (64-bit) and jam (32-bit) sequence before ceasing transmission and invoking the random back-off algorithm.

This transmit two part deferral algorithm is implemented as an option which can be disabled using the DXMT2PD bit in CSR3. Two part deferral after transmission is useful for ensuring that severe IPG shrinkage cannot occur in specific circumstances, causing a transmit message to follow a receive message so closely as to make them indistinguishable.

During the time period immediately after a transmission has been completed, the external transceiver (in the case of a standard AUI connected device), should generate the SQE Test message (a nominal 10 MHz burst of 5-15 Bit Times duration) on the CI± pair (within 0.6 - 1.6 µs after the transmission ceases). During the time period in which the SQE Test message is expected the PCnet-32 controller will not respond to receive carrier sense.

See ANSI/IEEE Std 802.3-1990 Edition, 7.2.4.6 (1)):

> *"At the conclusion of the output function, the DTE opens a time window during which it expects to see the signal_quality_error signal asserted on the Control In circuit. The time window begins when the CARRIER_ STATUS becomes CARRIER_OFF. If execution of the output function does not cause CARRIER_ON to occur, no SQE test occurs in the DTE. The duration of the window shall be at least 4.0 µs but no more than 8.0 µs. During the time window, the Carrier Sense Function is inhibited."*

The PCnet-32 controller implements a carrier sense "blinding" period within 0 µs–4.0 µs from de-assertion of carrier sense after transmission. This effectively means that when transmit two part deferral is enabled (DXMT2PD is cleared) the IFS1 time is from 4 µs to 6 µs after a transmission. However, since IPG shrinkage below 4 µs will rarely be encountered on a correctly configured networks, and since the fragment size will be larger than the 4 µs blinding window, then the IPG counter will be reset by a worst case IPG shrinkage/ fragment scenario and the PCnet-32 controller will defer its transmission. In addition, the PCnet-32 controller will not restart the "blinding" period if carrier

is detected within the 4.0 µs – 6.0 µs IFS1 period, but will commence timing of the entire IFS1 period.

Contention Resolution (Collision Handling)

Collision detection is performed and reported to the MAC engine by the integrated Manchester Encoder/ Decoder (MENDEC).

If a collision is detected before the complete preamble/ SFD sequence has been transmitted, the MAC Engine will complete the preamble/SFD before appending the jam sequence. If a collision is detected after the preamble/SFD has been completed, but prior to 512 bits being transmitted, the MAC Engine will abort the transmission, and append the jam sequence immediately. The jam sequence is a 32-bit all zeroes pattern.

The MAC Engine will attempt to transmit a frame a total of 16 times (initial attempt plus 15 retries) due to normal collisions (those within the slot time). Detection of collision will cause the transmission to be rescheduled, dependent on the back-off time that the MAC Engine computes. If a single retry was required, the ONE bit will be set in the Transmit Frame Status. If more than one retry was required, the MORE bit will be set. If all 16 attempts experienced collisions, the RTRY bit will be set (ONE and MORE will be clear), and the transmit message will be flushed from the FIFO. If retries have been disabled by setting the DRTY bit in CSR15, the MAC Engine will abandon transmission of the frame on detection of the first collision. In this case, only the RTRY bit will be set and the transmit message will be flushed from the FIFO.

If a collision is detected after 512 bit times have been transmitted, the collision is termed a late collision. The MAC Engine will abort the transmission, append the jam sequence and set the LCOL bit. No retry attempt will be scheduled on detection of a late collision, and the transmit message will be flushed from the FIFO.

The ISO 8802-3 (IEEE/ANSI 802.3) Standard requires use of a “truncated binary exponential back-off” algorithm which provides a controlled pseudo random mechanism to enforce the collision back-off interval, be-fore retransmission is attempted.

See ANSI/IEEE Std 802.3-1990 Edition, 4.2.3.2.5:

> *“At the end of enforcing a collision (jamming), the CSMA/CD sublayer delays before attempting to retransmit the frame. The delay is an integer multiple of slotTime. The number of slot times to delay before the nth retransmission attempt is chosen as a uniformly distributed random integer r in the range:*
>
> $$0 \pounds r < 2k - 1$$
>
> *where k = min (n,10).”*

The PCnet-32 controller provides an alternative algorithm, which suspends the counting of the slot time/IPG during the time that receive carrier sense is detected. This aids in networks where large numbers of nodes are present, and numerous nodes can be in collision. It effectively accelerates the increase in the back-off time in busy networks, and allows nodes not involved in the collision to access the channel whilst the colliding nodes await a reduction in channel activity. Once channel activity is reduced, the nodes resolving the collision time out their slot time counters as normal.

Manchester Encoder/Decoder (MENDEC)

The integrated Manchester Encoder/Decoder provides the PLS (Physical Layer Signaling) functions required for a fully compliant ISO 8802-3 (IEEE/ANSI 802.3) station. The MENDEC provides the encoding function for data to be transmitted on the network using the high accuracy on-board oscillator, driven by either the crystal oscillator or an external CMOS level compatible clock. The MENDEC also provides the decoding function from data received from the network. The MENDEC contains a Power On Reset (POR) circuit, which ensures that all analog portions of the PCnet-32 controller are forced into their correct state during power up, and prevents erroneous data transmission and/or reception during this time.

External Crystal Characteristics

When using a crystal to drive the oscillator, the crystal specification shown in Table 26 may be used to ensure less than ±0.5 ns jitter at the transmit outputs.

Table 26. External Crystal Specification

Parameter	Min	Nom	Max	Unit
1.Parallel Resonant Frequency		20		MHz
2.Resonant Frequency Error	–50		+50	PPM
3.Change in Resonant Frequency With Respect To Temperature (0×C–70×C)*	–40		+40	PPM
4.Crystal Load Capacitance	20		50	pF
5.Motional Crystal Capacitance (C1)		0.022		pF
6.Series Resistance			25	ohm
7.Shunt Capacitance			7	pF
8.Drive Level			TBD	mW

**Requires trimming spec,: no trim is 50 ppm total.*

External Clock Drive Characteristics

When driving the oscillator from an external clock source, XTAL2 must be left floating (unconnected). An external clock having the following characteristics must be used to ensure less than ±0.5 ns jitter at the transmit outputs. See Table 27.

Table 27. External Clock Drive Characteristics

Clock Frequency:	20 MHz ± 0.01%
Rise/Fall Time (t_R/t_F):	< 6 ns from 0.5 V to V_{DD}–0.5
XTAL1 HIGH/LOW Time (t_{HIGH}/t_{LOW}):	20 ns min
XTAL1 Falling Edge to Falling Edge Jitter:	< ±0.2 ns at 2.5 V input ($V_{DD}/2$)

MENDEC Transmit Path

The transmit section encodes separate clock and NRZ data input signals into a standard Manchester encoded serial bit stream. The transmit outputs (DO±/TXD±) are designed to operate into terminated transmission lines. When operating into a 78 W terminated transmission line, the transmit signaling meets the required output levels and skew for Cheapernet, Ethernet and IEEE-802.3.

Transmitter Timing and Operation

A 20 MHz fundamental mode crystal oscillator provides the basic timing reference for the MENDEC portion of the PCnet-32 controller. The crystal is divided by two, to create the internal transmit clock reference. Both clocks are fed into the MENDEC's Manchester Encoder to generate the transitions in the encoded data stream. The internal transmit clock is used by the MENDEC to internally synchronize the Internal Transmit Data (ITXDAT) from the controller and Internal Transmit Enable (ITXEN). The internal transmit clock is also used as a stable bit rate clock by the receive section of the MENDEC and controller.

The oscillator requires an external ±0.01% timing reference. The accuracy requirements, if an external crystal is used are tighter because allowance for the on-board parasitics must be made to deliver a final accuracy of 0.01%.

Transmission is enabled by the controller. As long as the ITXEN request remains active, the serial output of the controller will be Manchester encoded and appear at DO±/TXD±. When the internal request is dropped by the controller, the differential transmit outputs go to one of two idle states, dependent on TSEL in the Mode Register (CSR15, bit 9):

TSEL LOW:	The idle state of DO±/TXD± yields "zero" differential to operate transformer-coupled loads.
TSEL HIGH:	In this idle state, DO+/TXD+ is positive with respect to DO–/TXD– (logical HIGH).

Receiver Path

The principal functions of the Receiver are to signal the PCnet-32 controller that there is information on the receive pair, and separate the incoming Manchester encoded data stream into clock and NRZ data.

The Receiver section (see Figure 30) consists of two parallel paths. The receive data path is a zero threshold, wide bandwidth line receiver. The carrier path is an offset threshold bandpass detecting line receiver. Both receivers share common bias networks to allow operation over a wide input common mode range.

Input Signal Conditioning

Transient noise pulses at the input data stream are rejected by the Noise Rejection Filter. Pulse width rejection is proportional to transmit data rate.

The Carrier Detection circuitry detects the presence of an incoming data frame by discerning and rejecting noise from expected Manchester data, and controls the stop and start of the phase-lock loop during clock acquisition.Clock acquisition requires a valid Manchester bit pattern of 1010b to lock onto the incoming message.

When input amplitude and pulse width conditions are met at DI±/RXD±, the internal enable signal from the MENDEC to controller (IRXCRS) is asserted and clock acquisition cycle is initiated.

Clock Acquisition

When there is no activity at DI± (receiver is idle), the receive oscillator is phase locked to the internal transmit clock. The first negative clock transition (bit cell center of first valid Manchester "0") after IRXCRS is asserted interrupts the receive oscillator. The oscillator is then restarted at the second Manchester "0" (bit time 4) and phase locked to it. As a result, the MENDEC acquires the clock from the incoming Manchester bit pattern in bit times with a "1010" Manchester bit pattern.

ISRDCLK and IRXDAT are enabled 1/4 bit time after clock acquisition in bit cell 5. IRXDAT is at a HIGH state when the receiver is idle (no ISRDCLK). IRXDAT however, is undefined when clock is acquired and may remain HIGH or change to LOW state whenever ISRDCLK is enabled. At 1/4 bit time through bit cell 5,the controller portion of the PCnet-32 controller sees the first ISRDCLK transition. This also strobes in the incoming fifth bit to the MENDEC as Manchester "1". IRXDAT may make a transition after the ISRDCLK rising edge bit cell 5, but its state is still undefined. The Manchester"1" at bit 5 is clocked to IRXDAT output at 1/4 bit time bit cell 6.

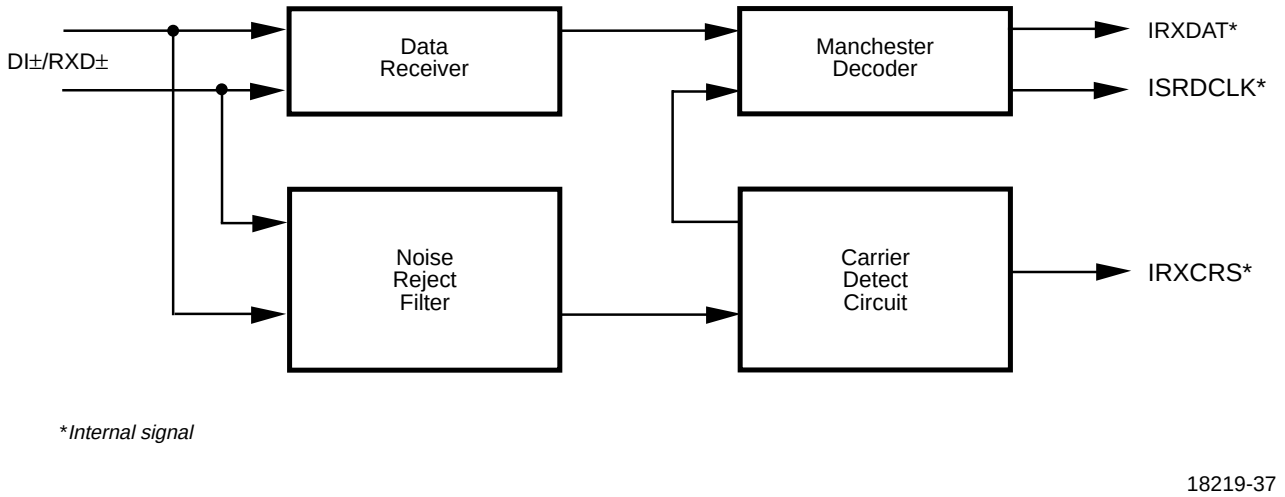


Figure 30. Receive Block Diagram

PLL Tracking

After clock acquisition, the phase-locked clock is compared to the incoming transition at the bit cell center (BCC) and the resulting phase error is applied to a correction circuit. This circuit ensures that the phase-locked clock remains locked on the received signal. Individual bit cell phase corrections of the Voltage Controlled Oscillator (VCO) are limited to 10% of the phase difference between BCC and phase-locked clock. Hence, input data jitter is reduced in ISRDCLK by 10 to 1.

Carrier Tracking and End of Message

The carrier detection circuit monitors the DI± inputs after IRXCRS is asserted for an end of message. IRXCRS de-asserts 1 to 2 bit times after the last positive transition on the incoming message. This initiates the end of reception cycle. The time delay from the last rising edge of the message to IRXCRS de-assert allows the last bit to be strobed by ISRDCLK and transferred to the controller section, but prevents any extra bit(s) at the end of message.

Data Decoding

The data receiver is a comparator with clocked output to minimize noise sensitivity to the DI±/RXD± inputs. Input error is less than ± 35 mV to minimize sensitivity to input rise and fall time. ISRDCLK strobes the data receiver output at 1/4 bit time to determine the value of the Manchester bit, and clocks the data out on IRXDAT on the following ISRDCLK. The data receiver also generates the signal used for phase detector comparison to the internal MENDEC voltage controlled oscillator (VCO).

Jitter Tolerance Definition

The MENDEC utilizes a clock capture circuit to align its internal data strobe with an incoming bit stream. The clock acquisition circuitry requires four valid bits with the values 1010b. Clock is phase-locked to the negative transition at the bit cell center of the second “0” in the pattern.

Since data is strobed at 1/4 bit time, Manchester transitions which shift from their nominal placement through 1/4 bit time will result in improperly decoded data. With this as the criteria for an error, a definition of “Jitter Handling” is:

> *The peak deviation approaching or crossing 1/4 bit cell position from nominal input transition, for which the MENDEC section will properly decode data.*

Attachment Unit Interface(AUI)

The AUI is the PLS (Physical Layer Signaling) to PMA (Physical Medium Attachment) interface which effectively connects the DTE to a MAU. The differential interface provided by the PCnet-32 controller is fully compliant to Section 7 of ISO 8802-3 (ANSI/IEEE 802.3).

After the PCnet-32 controller initiates a transmission it will expect to see data “looped-back” on the DI± pair (when the AUI port is selected). This will internally generate a “carrier sense”, indicating that the integrity of the data path to and from the MAU is intact, and that the MAU is operating correctly. This “carrier sense” signal must be asserted within TBD bit times after the first transmitted bit on DO± (when using the AUI port). If “carrier sense” does not become active in response to the data transmission, or becomes inactive before the end of transmission, the loss of carrier (LCAR) error bit

will be set in the Transmit Descriptor Ring (TMD2, bit27) after the frame has been transmitted.

Differential Input Terminations

The differential input for the Manchester data (DI±) is externally terminated by two 40.2 W ±1% resistors and one optional common-mode bypass capacitor, as shown in the Differential Input Termination diagram below. The differential input impedance, ZIDF, and the common-mode input impedance, ZICM, are specified so that the Ethernet specification for cable termination impedance is met using standard 1% resistor terminators. If SIP devices are used, 39 W is also a suitable value. The CI± differential inputs are terminated in exactly the same way as the DI± pair.

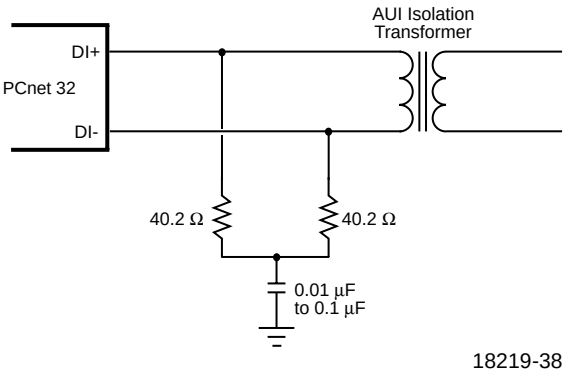


Figure 31. AUI Differential Input Termination

Collision Detection

A MAU detects the collision condition on the network and generates a differential signal at the CI± inputs. This collision signal passes through an input stage which detects signal levels and pulse duration. When the signal is detected by the MENDEC it sets the ICLSN line HIGH. The condition continues for approximately 1.5 bit times after the last LOW-to-HIGH transition on CI±.

Twisted-Pair Transceiver (T-MAU)

The T-MAU implements the Medium Attachment Unit (MAU) functions for the Twisted Pair Medium, as specified by the supplement to ISO 8802-3 (IEEE/ANSI 802.3) standard (Type 10BASE-T). The T-MAU provides twisted pair driver and receiver circuits, including on-board transmit digital predistortion and receiver squelch and a number of additional features including Link Status indication, Automatic Twisted Pair Receive Polarity Detection/Correction and Indication, Receive Carrier Sense, Transmit Active and Collision Present indication.

T-MAU gets reset during power-up by H_RESET, by S_RESET when reset port is read, or by asserting the RESET pin. T-MAU is not reset by STOP.

Twisted Pair Transmit Function

The differential driver circuitry in the TXD± and TXP± pins provides the necessary electrical driving capability and the pre-distortion control for transmitting signals over maximum length Twisted Pair cable, as specified by the 10BASE-T supplement to the ISO 8802-3 (IEEE/ANSI 802.3) Standard. The transmit function for data output meets the propagation delays and jitter specified by the standard.

Twisted Pair Receive Function

The receiver complies with the receiver specifications of the ISO 8802-3 (IEEE/ANSI 802.3) 10BASE-T Standard, including noise immunity and received signal rejection criteria ('Smart Squelch'). Signals meeting this criteria appearing at the RXD± differential input pair are routed to the MENDEC. The receiver function meets the propagation delays and jitter requirements specified by the standard. The receiver squelch level drops to half its threshold value after unsquelch to allow reception of minimum amplitude signals and to offset carrier fade in the event of worst case signal attenuation and crosstalk noise conditions.

Note that the 10BASE-T Standard defines the receive input amplitude at the external Media Dependent Interface (MDI). Filter and transformer loss are not specified. The T-MAU receiver squelch levels are defined to account for a 1 dB insertion loss at 10 MHz, which is typical for the type of receive filters/transformers employed.

Normal 10BASE-T compatible receive thresholds are employed when the LRT (CSR15[9]) bit is LOW. When the LRT bit is set (HIGH), the Low Receive Threshold option is invoked, and the sensitivity of the T-MAU receiver is increased. This allows longer line lengths to be employed, exceeding the 100 m target distance of normal 10BASE-T (assuming typical 24 AWG cable). The increased receiver sensitivity compensates for the increased signal attenuation caused by the additional cable distance.

However, making the receiver more sensitive means that it is also more susceptible to extraneous noise, primarily caused by coupling from co-resident services (crosstalk). For this reason, it is recommended that when using the Low Receive Threshold option that the service should be installed on 4-pair cable only. Multipair cables within the same outer sheath have lower crosstalk attenuation, and may allow noise emitted from adjacent pairs to couple into the receive pair, and be of sufficient amplitude to falsely unsquelch the T-MAU.

Link Test Function

The link test function is implemented as specified by 10BASE-T standard. During periods of transmit pair inactivity, 'Link beat pulses' will be periodically sent over the twisted pair medium to constantly monitor medium integrity.

When the link test function is enabled (DLNKTST bit in CSR15 is cleared), the absence of link beat pulses and receive data on the RXD± pair will cause the T-MAU to go into a link fail state. In the link fail state, data transmission, data reception, data loopback and the collision detection functions are disabled, and remain disabled until valid data or >5 consecutive link pulses appear on the RXD± pair. During link fail, the Link Status signal is inactive. When the link is identified as functional, the Link Status signal is asserted. The $\overline{\text{LNKST}}$ pin displays the Link Status signal by default.

Transmission attempts during Link Fail state will produce no network activity and will produce LCAR and CERR error indications.

In order to inter-operate with systems which do not implement Link Test, this function can be disabled by setting the DLNKTST bit in CSR15. With link test disabled, the data driver, receiver and loopback functions as well as collision detection remain enabled regardless of the presence or absence of data or link pulses on the RXD± pair. Link Test pulses continue to be sent regardless of the state of the DLNKTST bit.

Polarity Detection and Reversal

The T-MAU receive function includes the ability to invert the polarity of the signals appearing at the RXD± pair if the polarity of the received signal is reversed (such as in the case of a wiring error). This feature allows data frames received from a reverse wired RXD± input pair to be corrected in the T-MAU prior to transfer to the MENDEC. The polarity detection function is activated following H_RESET or Link Fail, and will reverse the receive polarity based on both the polarity of any previous link beat pulses and the polarity of subsequent frames with a valid End Transmit Delimiter (ETD).

When in the Link Fail state, the T-MAU will recognize link beat pulses of either positive or negative polarity. Exit from the Link Fail state is made due to the reception of 5–6 consecutive link beat pulses of identical polarity. On entry to the Link Pass state, the polarity of the last 5 link beat pulses is used to determine the initial receive polarity configuration and the receiver is reconfigured to subsequently recognize only link beat pulses of the previously recognized polarity.

Positive link beat pulses are defined as received signal with a positive amplitude greater than 585 mV (LRT = HIGH) with a pulse width of 60 ns–200 ns. This positive excursion may be followed by a negative excursion. This definition is consistent with the expected received signal at a correctly wired receiver, when a link beat pulse which fits the template of Figure 14-12 of the 10BASE-T Standard is generated at a transmitter and passed through 100 m of twisted pair cable.

Negative link beat pulses are defined as received signals with a negative amplitude greater than 585 mV with a pulse width of 60 ns–200 ns. This negative excursion may be followed by a positive excursion. This definition is consistent with the expected received signal at a reverse wired receiver, when a link beat pulse which fits the template of Figure 14-12 in the 10BASE-T Standard is generated at a transmitter and passed through 100 m of twisted pair cable.

The polarity detection/correction algorithm will remain "armed" until two consecutive frames with valid ETD of identical polarity are detected. When "armed", the receiver is capable of changing the initial or previous polarity configuration based on the ETD polarity.

On receipt of the first frame with valid ETD following H_RESET or link fail, the T-MAU will utilize the inferred polarity information to configure its RXD± input, regardless of its previous state. On receipt of a second frame with a valid ETD with correct polarity, the detection/correction algorithm will "lock-in" the received polarity. If the second (or subsequent) frame is not detected as confirming the previous polarity decision, the most recently detected ETD polarity will be used as the default. Note that frames with invalid ETD have no effect on updating the previous polarity decision. Once two consecutive frames with valid ETD have been received, the T-MAU will disable the detection/correction algorithm until either a Link Fail condition occurs or H_RESET is activated.

During polarity reversal, an internal POL signal will be active. During normal polarity conditions, this internal POL signal is inactive. The state of this signal can be read by software and/or displayed by LED when enabled by the LED control bits in the Bus Configuration Registers (BCR4–BCR7).

Twisted Pair Interface Status

Three signals (XMT, RCV and COL) indicate whether the T-MAU is transmitting, receiving, or in a collision state with both functions active simultaneously. These signals are internal signals and the behavior of the LED outputs depends on how the LED Output circuiting is programmed.

The T-MAU will power up in the Link Fail state and the normal algorithm will apply to allow it to enter the Link Pass state. In the Link Pass state, transmit or receive activity will be indicated by assertion of RCV signal going active. If T-MAU is selected using the PORTSEL bits in CSR15, then when moving from AUI to T-MAU

selection the T-MAU will be forced into the Link Fail state. In the Link Fail state, XMT, RCV and COL are inactive.

Collision Detect Function

Activity on both twisted pair signals RXD± and TXD± constitutes a collision, thereby causing the COL signal to be activated. (COL is used by the LED control circuits) COL will remain active until one of the two colliding signals changes from active to idle. However, transmission attempt in Link Fail state results in LCAR and CERR indication. COL stays active for 2-bit times at the end of a collision.

Signal Quality Error (SQE) Test (Heartbeat) Function

The SQE function is disabled when the 10BASE-T port is selected.

Jabber Function

The Jabber function inhibits the twisted pair transmit function of the T-MAU if the TXD± circuit is active for an excessive period (20 ms–150 ms). This prevents any one node from disrupting the network due to a 'stuck-on' or faulty transmitter. If this maximum transmit time is exceeded, the T-MAU transmitter circuitry is disabled, the JAB bit is set (CSR4, bit1), the COL signal is asserted. Once the transmit data stream to the T-MAU is removed, an “unjab” time of 250 ms–750 ms will elapse before the T-MAU COL and re-enables the transmit circuitry.

Power Down

The T-MAU circuitry can be made to go into power savings mode. This feature is useful in battery powered or low duty cycle systems. The T-MAU will go into power down mode when H_RESET is active, coma mode is active, or the T-MAU is not selected. Refer to the Power Savings Modes section for descriptions of the various power down modes.

Any of the three conditions listed above resets the internal logic of the T-MAU and places the device into power down mode. In this mode, the Twisted Pair driver pins (TXD±,TXP±) are driven LOW, and the internal T-MAU status signals (LNKST, RCVPOL, XMT, RCV and COL) signals are inactive.

Once H_RESET ends, coma mode is disabled, and the T-MAU is selected. The T-MAU will remain in the reset state for up to 10 µs. Immediately after the reset condition is removed, the T-MAU will be forced into the Link Fail state. The T-MAU will move to the Link Pass state only after 5 – 6 link beat pulses and/or a single received message is detected on the RD± pair.

In snooze mode, the T-MAU receive circuitry will remain enabled even while the $\overline{\text{SLEEP}}$ pin is driven LOW.

The T-MAU circuitry will always go into power down mode if H_RESET is asserted, coma mode is enabled, or the T-MAU is not selected.

10BASE-T Interface Connection

Figure 32 shows the proper 10BASE-T network interface design. Refer to the PCnet Family Technical Manual (PID # 18216A) for more design details, and refer to Appendix A for a list of compatible 10BASE-T filter/ transformer modules.

Note that the recommended resistor values and filter and transformer modules are the same as those used by the IMR (Am79C980) and the IMR+ (Am79C981).

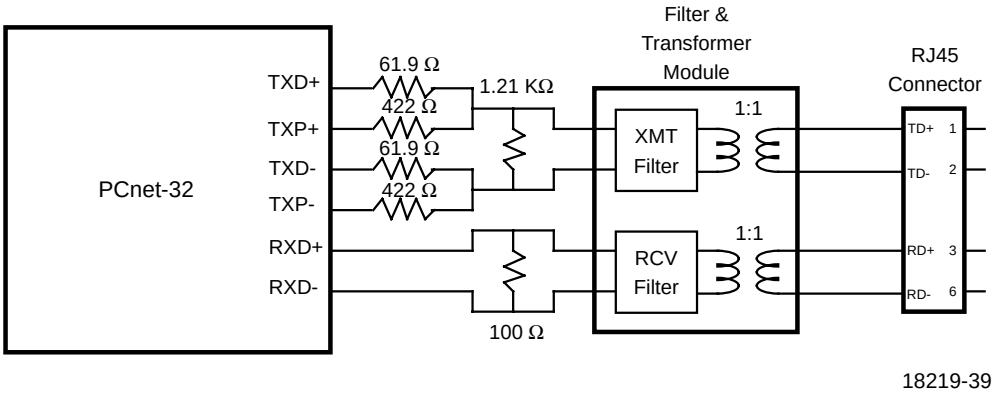


Figure 32. 10BASE-T Interface Connection

IEEE 1149.1 Test Access Port Interface

An IEEE 1149.1 compatible boundary scan Test Access Port is provided for board level continuity test and diagnostics. All digital input, output and input/output pins are tested. Analog pins, including the AUI differential driver (DO±) and receivers (DI±, CI±), and the crystal input (XTAL1/XTAL2) pins, are tested. The T-MAU drivers TXD±, TXP± and receiver RXD± are also tested.

The following is a brief summary of the IEEE 1149.1 compatible test functions implemented in the PCnet-32 controller.

Boundary Scan Circuit

The boundary scan test circuit requires four pins (TCK, TMS, TDI and TDO), defined as the Test Access Port (TAP). It includes a finite state machine (FSM), an instruction register, a data register array, and a power-on reset circuit. Internal pull-up resistors are provided for the TDI, TCK, and TMS pins. The TCK pin must not be left unconnected. The boundary scan circuit remains active during Sleep.

TAP FSM

The TAP engine is a 16-state FSM, driven by the Test Clock (TCK) and the Test Mode Select (TMS) pins. This FSM is in its reset state at power-up or after H_RESET. An independent power-on reset circuit is provided to ensure the FSM is in the TEST_LOGIC_RESET state at power-up.

Supported Instructions

In addition to the minimum IEEE 1149.1 requirements (BYPASS, EXTEST, and SAMPLE instructions), three additional instructions (IDCODE, TRIBYP and SETBYP) are provided to further ease board-level testing. All unused instruction codes are reserved. See Table 28 for a summary of supported instructions.

Instruction Register and Decoding Logic

After H_RESET or S_RESET or STOP, the IDCODE instruction is always invoked. The decoding logic gives signals to control the data flow in the DATA registers according to the current instruction.

Boundary Scan Register (BSR)

Each BSR cell has two stages. A flip-flop and a latch are used for the SERIAL SHIFT STAGE and the PARALLEL OUTPUT STAGE, respectively.

Table 28. IEEE 1149.1 Supported Instruction Summary

Instruction Name	Description	Selected Data Reg	Mode	Instruction Code
EXTEST	External Test	BSR	Test	0000
IDCODE	ID Code Inspection	ID REG	Normal	0001
SAMPLE	Sample Boundary	BSR	Normal	0010
TRIBYP	Force Tristate	Bypass	Normal	0011
SETBYP	Control Boundary To 1/0	Bypass	Test	0100
BYPASS	Bypass Scan	Bypass	Normal	1111

There are four possible operation modes in the BSR cell:

1	Capture
2	Shift
3	Update
4	System Function

Other Data Register

1) BYPASS REG (1 Bit)

2) DEV ID REG (32 bits)

Bits 31–28:	Version
Bits 27–12:	Part number (0010 0100 0011 0000)
Bits 11–1:	Manufacturer ID. The 11 bit manufacturer ID code for AMD is 00000000001 according to JEDEC Publication 106-A.
Bit 0:	Always a logic 1

3) INSCAN0

This is an internal scan path for AMD internal testing use.

EADI (External Address Detection Interface)

This interface is provided to allow external address filtering. It is selected by setting the EADISEL bit in

BCR2 to a ONE. This feature is typically utilized for terminal servers, bridges and/or router products. The EADI interface can be used in conjunction with external logic to capture the packet destination address from the serial bit stream as it arrives at the PCnet-32 controller, compare the captured address with a table of stored addresses or identifiers, and then determine whether or not the PCnet-32 controller should accept the packet.

The EADI interface outputs are delivered directly from the NRZ decoded data and clock recovered by the Manchester decoder or input into the GPSI port. This allows the external address detection to be performed in parallel with frame reception and address comparison in the MAC Station Address Detection (SAD) block of the PCnet-32 controller.

SRDCLK is provided to allow clocking of the receive bit stream into the external address detection logic. SRDCLK runs only during frame reception activity. Once a received frame commences and data and clock are available from the decoder, the EADI logic will monitor the alternating (“1,0”) preamble pattern until the two ones of the Start Frame Delimiter (“1,0,1,0,1,0,1,1”) are detected, at which point the SF/BD output will be driven HIGH.

The SF/BD signal will initially be LOW. The assertion of SF/BD is a signal to the external address detection logic that the SFD has been detected and that subsequent SRDCLK cycles will deliver packet data to the external logic. Therefore, when SF/BD is asserted, the external address matching logic should begin deserialization of the SRD data and send the resulting destination address to a content addressable memory (CAM) or other address detection device.

In order to reduce the amount of logic external to the PCnet-32 controller for multiple address decoding systems, the SF/BD signal will toggle at each new byte boundary within the packet, subsequent to the SFD. This eliminates the need for externally supplying byte framing logic.

The $\overline{\text{EAR}}$ pin should be driven LOW by the external address comparison logic to reject a frame.

If an address match is detected by internal address comparison with either the Physical or Logical or broadcast Address contained within the PCnet-32 controller, then the frame will be accepted regardless of the condition of $\overline{\text{EAR}}$. Internal address match is disabled when PROM (CSR15[15]) = 0, DRCVBC (CSR15[14]) = 1, DRCVPA (CSR15[13]) = 1 and Logical Address Filter (CSR8–CSR11) = 0.

When the EADISEL bit of BCR2 is set to a ONE and internal address match is disabled, then all incoming frames will be accepted by the PCnet-32 controller, unless the $\overline{\text{EAR}}$ pin becomes active during the first 64 bytes of the frame (excluding preamble and SFD). This allows external address lookup logic approximately 58 byte times after the last destination address bit is available to generate the $\overline{\text{EAR}}$ signal, assuming that the PCnet-32 controller is not configured to accept runt packets. $\overline{\text{EAR}}$ will be ignored after 64 byte times after the SFD. The frame will be accepted if EAR has not been asserted before this time. If Runt Packet Accept is enabled, then the $\overline{\text{EAR}}$ signal must be generated prior to the receive message completion, if packet rejection is to be guaranteed. Runt packet sizes could be as short as 12 byte times (assuming 6 bytes for source address, 2 bytes for length, no data, 4 bytes for FCS) after the last bit of the destination address is available. $\overline{\text{EAR}}$ must have a pulse width of at least 150 ns.

When the EADISEL bit of BCR2 is set to a ONE and the PROM bit of the Mode Register is set to a ONE, then all incoming frames will be accepted by the PCnet-32 controller, regardless of any activity on the $\overline{\text{EAR}}$ pin.

The EADI outputs continue to provide data throughout the reception of a packet. This allows the external logic to capture packet header information to determine protocol type, inter-networking information, and other useful data.

The EADI interface will operate as long as the STRT bit in CSR0 is set, even if the receiver and/or transmitter are disabled by software (DTX and DRX bits in CSR15 are set). This configuration is useful as a semi-power-down mode in that the PCnet-32 controller will not perform any power-consuming DMA operations. However, external circuitry can still respond to “control” frames on the network to facilitate remote node control.

Table 29 summarizes the operation of the EADI interface.

Table 29. EADI Operations

PROM	$\overline{\text{EAR}}$	Required Timing	Received Messages
1	X	No timing requirements	All Received Frames
0	1	No timing requirements	All Received Frames
0	0	Low for 150 ns within 512 bits after SFD	PCnet-32 Controller Internal Physical and Logical Address Matches

General Purpose Serial Interface (GPSI)

The PCnet-32 controller is capable of providing a purely digital network interface in the form of the General Purpose Serial Interface. This interface matches the functionality provided by earlier AMD network controller products, such as the Am7990 LANCE, Am79C90 C-LANCE, Am79C900 ILACC controller, Am79C940 MACE controller, and Am79C960 PCnet-ISA controller.

The GPSI interface is selected through the PORTSEL bits of the Mode register (CSR15) and enabled through the TSTSHDW bits of BCR18 (bits 3 and 4) or enabled through the GPSIEN bit in CSR124 (bit 4). The possible settings to invoke the GPSI mode are shown in Table 30.

Table 30. GPSI Mode Selection

TSTSHDW Value (BCR18 [4:3])	PVALID (BCR19 [15])	GPSIEN (CSR124[4])	Operating Mode
00	1	0	Normal Operating Mode
10	1	X	GPSI Mode
01	1	0	Reserved
11	1	X	Reserved
XX	0	0	Normal Operating Mode
XX	0	1	GPSI Mode

Note that the TSTSHDW bit values are only active when PVALID is TRUE.

The GPSI interface is multiplexed through 7 of the upper 8 address bits of the system bus interface. Therefore, applications that require the use of the GPSI interface will be limited to the use of only 24 address bits (A[23] through A[2] plus the byte enables, which decode into two effective address bits).

In order to prevent the PCnet-32 controller from interpreting the GPSI signals as address bits during the Software Relocatable Mode and during slave accesses, the 24-bit Software Relocatable Mode Address 24 mode and the I/O Address Width 24 mode of the PCnet-32 controller must be invoked. Note that if Software Relocatable Mode is invoked, then PVALID must have been set to ZERO, and, therefore, the GPSI mode is not active and therefore, the Software Relocatable Mode might assume that all 30 address bits are visible. But in a system that uses GPSI mode, the GPSI signals would likely all be hard-wired to the address pins, and, therefore, even though the device never made it into GPSI mode, it will still not be able to see the upper address bits. Therefore, it is always recommended that SRMA24 mode be invoked as described in the next paragraph:

Software Relocatable Mode Address 24 mode is invoked by connecting the LED2/SRDCLK pin to a LOW level during H_RESET and during the execution of the Software Relocation operation. When the LED2/SRDCLK pin is LOW during H_RESET and Software Relocatable Mode, then the device will be programmed to use 24 bits of addressing while snooping accesses on the bus during Software Relocatable Mode; In this case, the PCnet-32 controller will assume that bits A[31] through A[24] are matched at all times, regardless of the actual values on these pins.

I/O Address Width 24 mode is invoked by writing a ONE to the IOAW24 bit of BCR21 (bit 8 of BCR21). This can be accomplished *safely* in either of two ways:

1. A Software Relocation operation can write a ONE to BCR21, bit 8.
2. A read of the EEPROM contents can write a ONE to BCR21, bit 8, if the EEPROM contents are correctly programmed.

These two methods do NOT require a slave access to the PCnet-32 controller, and therefore may be performed in a system in which the GPSI signals are permanently connected to the A[31] through A[23] pins (assuming SRMA24 mode is invoked via the LED2/SRDCLK pin to use method 1).

The PCnet-32 controller upper address pins are reconfigured during GPSI mode to the functions listed in Table 31. Note that pin number 143 (A24) has no equivalent GPSI function and should be left unconnected when GPSI mode is enabled.

GPSI signal functions are described in the pin description section under the GPSI subheading.

At the time that GPSI mode is entered, the internal MAC clock is switched from a XTAL1-derived clock to a clock derived from the STDCLK input. The STDCLK input determines the network data rate and therefore must meet frequency and stability specifications.

Table 31. GPSI Pin Configurations

GPSI Function	GPSI I/O Type	LANCE GPSI Pin	ILACC GPSI Pin	PCnet-32/ PCnet-ISA GPSI Pin	PCnet-32 Pin Number	PCnet-32 Normal Pin Function
Transmit Data	O	TX	TXD	TXDAT	132	A31
Transmit Enable	O	TENA	RTS	TXEN	133	A30
Transmit Clock	I	TCLK	TXC	STDCLK	134	A29
Collision	I	CLSN	CDT	CLSN	137	A28
Receive Clock	I	RCLK	RXC	SRDCLK	140	A26
Receive Data	I	RX	RXD	RXDAT	141	A25

Note that the XTAL1 input must always be driven with a clock source, even if GPSI mode is to be used. It is not necessary for the XTAL1 clock to meet the normal frequency and stability requirements in this case. Any frequency between 8 MHz and 20 MHz is acceptable. However, voltage drive requirements do not change. When GPSI mode is used, XTAL1 must be driven for several reasons:

1. The default pin RESET configuration for the PCnet-32 controller is “AUI port selected,” and until GPSI mode is selected, the XTAL1 clock is needed for some internal operations (namely, RESET).
2. The XTAL1 clock drives the EEPROM read operation, regardless of the network mode selected.
3. The XTAL1 clock determines the length of a Reset Register read operation, regardless of the network mode selected (due to the internal RESET caused by the read).

Note that if a clock slower than 20 MHz is provided at the XTAL1 input, the time needed for EEPROM read and Reset Register Read operations will increase.

Power Savings Modes

The PCnet-32 controller supports two hardware power savings modes. Both are entered by driving the SLEEP pin LOW.

In **coma** mode, the PCnet-32 controller will go into a deep sleep with no means to use the network to automatically wake itself up. Coma mode is enabled when the AWAKE bit in BCR2 is reset. Coma mode is the default power down mode. When coma mode is invoked, the T-MAU circuitry will go into power down mode. The system bus interface will be floated and inactive during coma mode. LDEV and the selected interrupt pin will be driven to inactive levels. While in coma mode, if the PCnet-32 controller is configured for a daisy chain (HOLDI and HLDAO or LREQI and LGNTO signals have been selected with the JTAGSEL pin), then the daisy chain signal LREQI/HOLDI will be passed directly to LREQ/HOLD and the system arbitration signal LGNT/ HLDA will be passed directly to the daisy-chain signal LGNTO/HLDAO.

In **snooze** mode, enabled by setting the AWAKE bit in BCR2 and driving the SLEEP pin LOW, the T-MAU receive circuitry will remain enabled even while the SLEEP pin is driven LOW. The LNKST output will also continue to function, indicating a good 10BASE-T link if there are link beat pulses or valid frames present. This LNKST pin can be used to drive an LED and/or external hardware that directly controls the SLEEP pin of the PCnet-32 controller. This configuration effectively wakes the system when there is any activity on the 10BASE-T link. Auto wake mode can be used only if the T-MAU is the selected network port. Link beat pulses are not transmitted during Auto-wake mode.

The system bus interface will be floated and inactive during snooze mode. LDEV and the selected interrupt pin will be driven to inactive levels. While in snooze mode, if the PCnet-32 controller is configured for a daisy chain (HOLDI and HLDAO or LREQI and LGNTO signals have been selected with the JTAGSEL pin), then the daisy chain signal LREQI/HOLDI will be passed directly to LREQ/HOLD and the system arbitration signal LGNT/HLDA will be passed directly to the daisy-chain signal LGNTO/HLDAO.

If the HOLD output is active when the SLEEP pin is asserted, then the PCnet-32 controller will wait until the HLDA input is asserted. Then the PCnet-32 controller will de-assert the HOLD pin and finally, it will internally enter either the coma or snooze sleep mode.

Before the sleep mode is invoked, the PCnet-32 controller will perform an internal S_RESET. This S_RESET operation will not affect the values of the BCR registers.

The SLEEP pin should not be asserted during power supply ramp-up. If it is desired that SLEEP be asserted at power up time, then the system must delay the assertion of SLEEP until three BCLK cycles after the completion of a valid pin RESET operation.

Software Access

I/O Resources

PCnet-32 Controller I/O Resource Mapping

The PCnet-32 controller has several I/O resources. These resources use 32 bytes of I/O space that begin at the PCnet-32 controller I/O Base Address. The PCnet-32 controller allows two modes of slave access. Word I/O mode treats all PCnet-32 controller I/O Resources as two-byte entities spaced at two-byte address intervals. Double Word I/O mode treats all PCnet-32 controller I/O Resources as four-byte entities spaced at four-byte address intervals. The selection of WIO or DWIO mode is accomplished by one of two ways:

1. H_RESET function.
2. The PCnet-32 controller I/O mode setting will default to WIO after H_RESET (i.e. DWIO = 0).
3. Automatic determination of DWIO mode due to double-word I/O write access to offset 10h.

DWIO is automatically programmed as active when the system attempts a double word write access to offset 10h of the PCnet-32 controller I/O space. Note that this space corresponds to RDP, regardless of whether DWIO or WIO mode has been programmed. The power up H_RESET value of DWIO will be ZERO, and this value will be maintained until a double word access is performed to PCnet-32 controller I/O space.

Therefore, if DWIO mode is desired, it is imperative that the first access to the PCnet-32 controller be a double word write access to offset 10h.

Alternatively, if DWIO mode is not desired, then it is imperative that the software never executes a double word write access to offset 10h of the PCnet-32 controller I/O space.

Once the DWIO bit has been set to a ONE, only a H_RESET can reset it to a ZERO.

The DWIO mode setting is unaffected by S_RESET or the STOP bit.

WIO I/O Resource Map

When the PCnet-32 controller I/O space is mapped as Word I/O, then the resources that are allotted to the PCnet-32 controller occur on word boundaries that are offset from the PCnet-32 controller I/O Base Address as shown in Table 32.

Table 32. Word I/O Mapping

Offset (Hex)	No. of Bytes	Register
0	2	Address PROM
2	2	Address PROM
4	2	Address PROM
8	2	Address PROM
A	2	Address PROM
C	2	Address PROM
E	2	Address PROM
10	2	RDP
12	2	RAP (shared by RDP and BDP)
14	2	Reset Register
16	2	BDP
18	2	Vendor Specific Word
1A	2	Reserved
1C	2	Reserved
1E	2	Reserved

When PCnet-32 controller I/O space is Word mapped, all I/O resources fall on word boundaries and all I/O resources are word quantities. However, while in Word I/O mode, address PROM accesses may also be accessed as individual bytes on byte addresses.

Attempts to write to any PCnet-32 controller I/O resources (except to offset 10h, RDP) as 32 bit quantities while in Word I/O mode are illegal and may cause unexpected reprogramming of the PCnet-32 controller control registers. Attempts to read from any PCnet-32 controller I/O resources as 32-bit quantities while in Word I/O mode are illegal and will yield undefined values.

An attempt to *write* to offset 10H (RDP) as a 32 bit quantity while in Word I/O mode will cause the PCnet-32 controller to exit WIO mode and immediately thereafter, to enter DWIO mode.

Accesses to non-word address boundaries are not allowed while in WIO mode, with the exception of the APROM locations. The PCnet-32 controller may or may not produce an $\overline{\text{LDEV}}$ and a $\overline{\text{RDY}}$ signal in response to such accesses, but data will be undefined.

Accesses of non-word quantities to any I/O resource are not allowed while in WIO mode, *with the exception of byte reads from the APROM locations.* PCnet-32 controller may or may not produce an $\overline{\text{LDEV}}$ and will not produce a $\overline{\text{RDY}}$ signal in response to such accesses, but data will be undefined.

The Vendor Specific Word (VSW) is not implemented by the PCnet-32 controller. This particular I/O address is reserved for customer use and will not be used by future AMD Ethernet controller products. If more than

one Vendor Specific Word is needed, it is suggested that the VSW location should be divided into a VSW Register Address Pointer (VSWRAP) at one location (e.g. VSWRAP at byte location 18h) and a VSW Data Port (VSWDP) at the other location (e.g. VSWDP at byte location 19h). Alternatively, the system may capture RAP data accesses in parallel with the PCnet-32 controller and therefore share the PCnet-32 controller RAP to allow expanded VSW space. PCnet-32 controller will not respond to access to the VSW I/O address.

DWIO I/O Resource Map

When the PCnet-32 controller I/O space is mapped as Double Word I/O, then all of the resources that are allotted to the PCnet-32 controller occur on double word boundaries that are offset from the PCnet-32 controller I/O Base Address as shown in Table 33.

Table 33. Double Word I/O Mapping

Offset (Hex)	No. of Bytes	Register
0	4	Address PROM
4	4	Address PROM
8	4	Address PROM
C	4	Address PROM
10	4	RDP
14	4	RAP (shared by RDP and BDP)
18	4	Reset Register
1C	4	BDP

When PCnet-32 controller I/O space is Double Word-mapped, all I/O resources fall on double word boundaries. Address PROM resources are double word quantities in DWIO mode. RDP, RAP and BDP contain only two bytes of valid data. The other two bytes of these resources are **reserved** for future use. The reserved bits must be written as zeros, and when read, are considered *undefined*.

Accesses to non-double word address boundaries are not allowed while in DWIO mode. The PCnet-32 controller may or may not produce an $\overline{\text{LDEV}}$ and a $\overline{\text{RDY}}$ signal in response to such accesses, but data will be undefined.

Accesses of less than 4 bytes to any I/O resource are not allowed while in DWIO mode (i.e. PCnet-32 controller will not respond to such accesses. PCnet-32 controller will not produce an $\overline{\text{LDEV}}$ and a $\overline{\text{RDY}}$ signal in response to such accesses), but data will be undefined. A double word write access to the RDP offset of 10h will automatically program DWIO mode.

Note that in all cases when I/O resource width is defined as 32 bits, the upper 16 bits of the I/O resource is reserved and written as ZEROs and read as undefined, except for the APROM locations and CSR88.

DWIO mode is exited by asserting the RESET pin. Assertion of S_RESET or setting the STOP bit of CSR0 will have no effect on the DWIO mode setting.

I/O Space Comments

The following statements apply to both WIO and DWIO mapping:

The RAP is shared by the RDP and the BDP.

The PCnet-32 controller does not respond to any addresses outside of the offset range 0h-17h when DWIO = 0 or 0h-1F when DWIO = 1. I/O offsets 18h through 1Fh are not used by the PCnet-32 controller when programmed for DWIO = 0 mode. Locations 1Ah through 1Fh are reserved for future AMD use and therefore should not be implemented by the user if upward compatibility to future AMD devices is desired.

Note that Address PROM accesses do not directly access the EEPROM, but are redirected to a set of shadow registers on board the PCnet-32 controller that contain a copy of the EEPROM contents that was obtained during the automatic EEPROM read operation that follows the RESET operation.

PCnet-32 Controller I/O Base Address

The I/O Base Address Registers (BCR16 and BCR17) will reflect the current value of the base of the PCnet-32 controller I/O address space. BCR16 contains the lower 16 bits of the 32-bit I/O base address for the PCnet-32 controller. BCR17 contains the upper 16 bits of the 32-bit I/O base address for the PCnet-32 controller. This set of registers is both readable and writable by the host. The value contained in these registers is affected through three means:

1. Immediately *following* the H_RESET operation, the I/O Base Address will be determined by the EEPROM read operation. During this operation, the I/O Base Address register will become programmed with the value of the I/O Base Address field of the EEPROM.
2. If no EEPROM exists, or if an error is detected in the EEPROM data, then the PCnet-32 controller will enter **Software Relocatable Mode**. While in this mode, the PCnet-32 controller will not respond to any I/O accesses directly. However, the PCnet-32 controller will snoop accesses on the system bus. When the PCnet-32 controller detects a specific sequence of four write accesses to I/O address 378h, then the PCnet-32 controller will assume that the software is attempting to relocate the PCnet-32 controller. On eight subsequent write accesses to I/O address 378h, the PCnet-32 controller will accept the data on the bus as a new I/O Base Address and other programming infor-

mation, and it will leave Software Relocatable Mode. At this point, the PCnet-32 controller will begin responding to I/O accesses directly.

While the PCnet-32 controller is in software relocatable mode, if the LED2 pin is pulled LOW, then the SRMA24 mode is entered and only the lower 24 bits of address are matched to 378h.

3. The I/O Base Address Registers may be directly written to, provided that the PCnet-32 controller is not currently in the Software Relocatable Mode.

Software Relocation of I/O Resources

In order to allow for jumperless Ethernet implementations, the I/O Base Address register value will be automatically altered by the PCnet-32 controller during an EEPROM read operation. In this case, the value of the I/O Base Address for the PCnet-32 controller will be directly dependent upon the value of the BCR16 and BCR17 fields that are stored in the EEPROM. If no EEPROM exists and an EEPROM read is attempted, then the PCnet-32 controller will enter Software Relocatable Mode.

Software Relocatable Mode

While in Software Relocatable Mode, the PCnet-32 controller will not respond to any access on the system bus. However, the PCnet-32 controller will snoop any I/O write accesses that may be present. The PCnet-32 controller will watch for a specific sequence of accesses in order to determine a value for the I/O Base Address Registers. Specifically, the PCnet-32 controller will wait for a sequence of 12 uninterrupted byte-write accesses to I/O address 378h.

The 12 byte-write accesses must occur without intervening I/O accesses to other locations, and they must contain the data in the order shown in Table 34.

$\overline{BE0}$ is required to be active during all Software Relocatable Mode snoop accesses. $\overline{BE3}$–$\overline{BE1}$ may have any value during Software Relocatable Mode snoop accesses.

Table 34. I/O Base Address Write Sequence in SRM

Access No.	I/O Address (Hex)	D[7:0]* (Hex)	ASCII Inter–pretation
1	0000 0378	41	A
2	0000 0378	4D	M
3	0000 0378	44	D
4	0000 0378	01	NA
5	0000 0378	IOBASEL[7:0]	NA
6	0000 0378	IOBASEL[15:8]	NA
8	0000 0378	IOBASEU[15:8]	NA
9	0000 0378	BCR2[7:0]	NA
10	0000 0378	BCR2[15:8]	NA
11	0000 0378	BCR21[7:0]	NA
12	0000 0378	BCR21[15:8]	NA

**Note that D[31:8] are don't care, since the accesses are required to one byte in width.*

Immediately following the 12th write access in the sequence, the PCnet-32 controller will leave Software Relocatable Mode, and it will then respond to I/O accesses to the 32 bytes of I/O space that begins at the I/O Base Address location.

Note that in Figure 33 (Software Relocatable Mode Snoop Accesses), the data that is accepted by the PCnet-32 controller will always be the data that is presented during the first T2 cycle, regardless of the state of the $\overline{RDYRTN}$ input.

Since the PCnet-32 controller will not respond to the Software Relocatable Mode snoop accesses, some other device must drive the $\overline{RDYRTN}$ signal during these accesses. In a typical PC environment, these I/O accesses will be directed toward the data port of a parallel port. Therefore, the $\overline{RDYRTN}$ will typically be generated by the parallel port controller. In systems in which the parallel port device does not exist, or is at a different location, the 378h accesses will go unclaimed by any device on the local bus or on the expansion bus. In this case, the chipset will typically terminate the access by providing the $\overline{RDYRTN}$ signal after some access-time-out counter has elapsed.

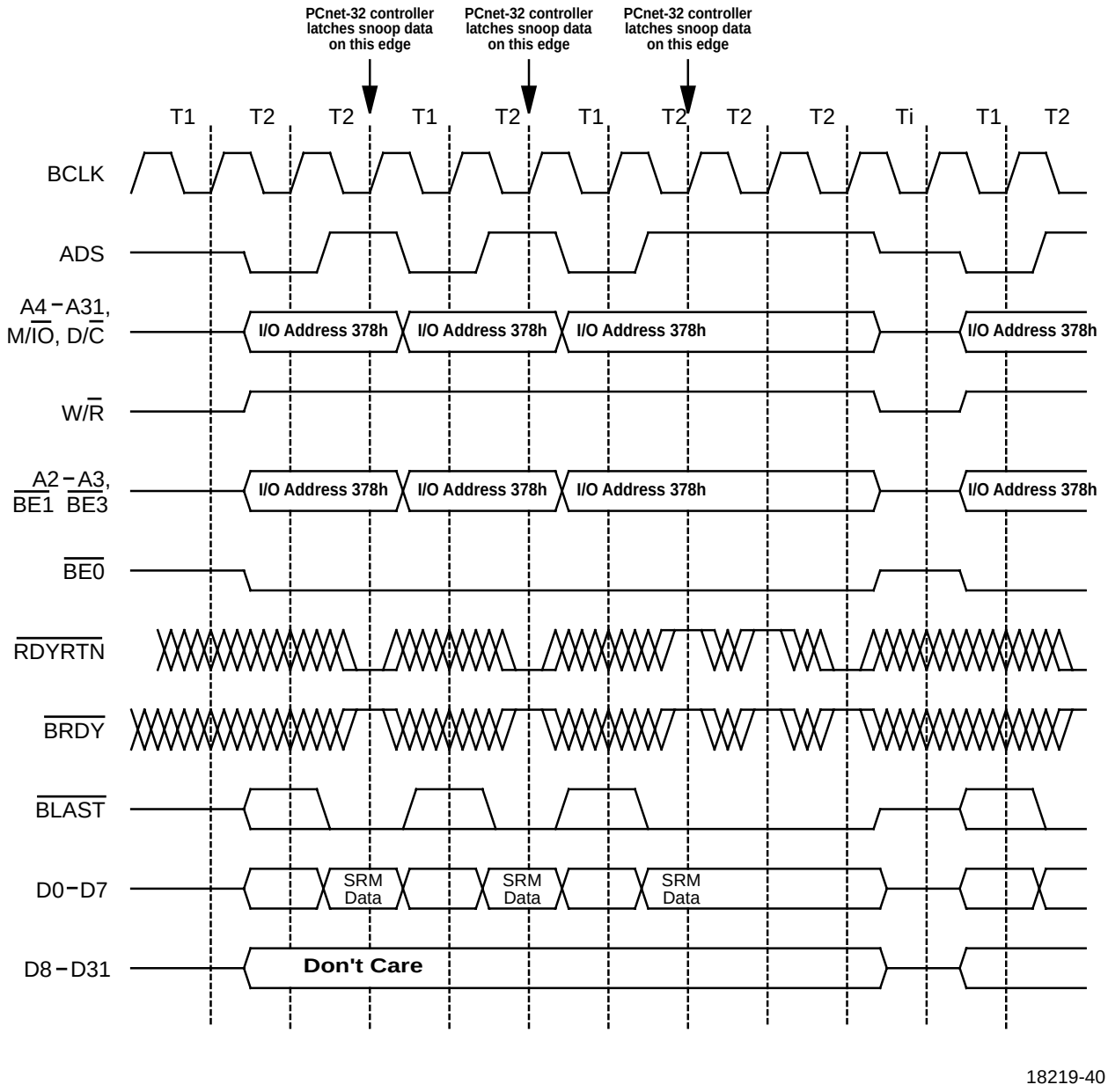


Figure 33. Software Relocatable Mode Snoop Cycles

The I/O Base Address register value may be altered by a direct write to BCR16 and BCR17 by the host. Slave accesses to the PCnet-32 controller should not be performed until both BCR16 and BCR17 have been written with new values. An intermediate I/O Base address will be created when only one of the two registers has been written. Therefore, this method of I/O Base address reassignment is not recommended.

I/O Register Access

All I/O resources are accessed with similar I/O bus cycles.

I/O accesses to the PCnet-32 controller begin with a valid $\overline{ADS}$ strobe and an address on the A[31:2] lines that falls within the I/O space of the PCnet-32 controller. The PCnet-32 controller I/O space will be determined by the I/O Base Address Registers. EEPROM read operations will determine the value of the I/O Base Address Registers.

The PCnet-32 controller will respond to an access to its I/O space by asserting the $\overline{LDEV}$ signal and eventually, by asserting the $\overline{RDY}$ signal.

Typical I/O access times are 6 or 7 BCLK periods. The

exact number of clock cycles for any particular access will depend upon the relative phases of the signal on the BCLK pin and the internal clock that is used to drive the PCnet-32 controller buffer management unit. Since the PCnet-32 controller buffer management unit operates on a clock that is a ÷2 version of the interface clock, it possible for the buffer management unit to introduce one or more BCLK wait delays to allow for synchronization of the two state machines before proceeding with the access. The PCnet-32 buffer management unit uses BCLK ÷2, so the maximum number of BCLK cycles needed to synchronize the BIU and buffer management units is one BCLK period.

APROM Access

The APROM space is a convenient place to store the value of the IEEE station address. This space is automatically loaded from the serial EEPROM, if an EEPROM is present. Its contents have no effect on the operation of the controller. The software must copy the station address from the APROM space to the Initialization Block or to CSR12–CSR14 in order for the receiver to accept unicast frames directed to this station.

When programmed for WIO mode, any byte or word address from an offset of 0h to an offset of Fh may be read. An appropriate byte or word of APROM contents will be delivered by the PCnet-32 controller in response to accesses that fall within the APROM range of 0h to Fh.

When programmed for DWIO mode, only double word addresses from an offset of 0h to an offset of Fh may be read. An appropriate double word of APROM contents will be delivered in response to accesses that fall within the APROM range of 0h to Fh.

Reads of non-double-word *quantities* are not allowed in DWIO mode, even though such an access may be properly aligned to a double word address boundary. Write access to any of the APROM locations is allowed, but only 4 bytes on double-word boundaries in DWIO mode or 2 bytes on word boundaries in WIO mode. The IESRWE bit (see BCR2) must be set in order to enable such a write. Only the PCnet-32 controller on-board IEEE Shadow registers are modified by writes to APROM locations. The EEPROM is unaffected by writes to APROM locations.

Note that the APROM locations occupy 16 bytes of space, yet the IEEE station address requirement is for 6 bytes. The 6 bytes of IEEE station address occupy the first 6 locations of the APROM space. The next six bytes are reserved. Bytes 12 and 13 should match the value of the checksum of bytes 1 through 11 and 14 and 15. Bytes 14 and 15 should each be ASCII “W” (57 h) if compatibility to AMD driver software is desired.

RDP Access (CSR Register Space)

RDP = Register Data Port. The RDP is used with the RAP to gain access to any of the PCnet-32 controller CSR locations.

Access to any of the CSR locations of the PCnet-32 controller is performed through the PCnet-32 controller’s Register Data Port (RDP). In order to access a particular CSR location, the Register Address Port (RAP) should first be written with the appropriate CSR address. The RDP now points to the selected CSR. A read of the RDP will yield the selected CSR’s data. A write to the RDP will write to the selected CSR.

When programmed for WIO mode, the RDP has a width of 16 bits, hence, all CSR locations have 16 bits of width. Note that when accessing RDP, the upper two bytes of the data bus will be undefined since the byte masks will not be active for those bytes.

If DWIO mode has been invoked, then the RDP has a width of 32 bits, hence, all CSR locations have 32 bits of width and the upper two bytes of the data bus will be active, as indicated by the byte mask. In this case, note that the upper 16 bits of all CSR locations (except CSR88) are reserved and written as zeros and read as undefined values. Therefore, during RDP write operations in DWIO mode, the upper 16 bits of all CSR locations should be written as ZEROs.

RAP Access

RAP = Register Address Port. The RAP is used with the RDP and with the BDP to gain access to any of the CSR and BCR register locations, respectively. The RAP contains the address pointer that will be used by an access to either the RDP or BDP. Therefore, it is necessary to set the RAP value before accessing a specific CSR or BCR location. Once the RAP has been written with a value, the RAP value remains unchanged until another RAP write occurs, or until an H_RESET or S_RESET occurs. RAP is set to all zeros when an H_RESET or S_RESET occurs. RAP is unaffected by the STOP bit.

When programmed for WIO mode, the RAP has a width of 16 bits. Note that when accessing RAP, the lower two bytes of the data bus will be undefined since the byte masks will not be active for those bytes

When programmed for DWIO mode, the RAP has a width of 32 bits. In DWIO mode, the upper 16 bits of the RAP are reserved and written as zeros and read as undefined. These bits should be written as zeros.

BDP Access (BCR Register Space)

BDP = Bus Configuration Register Data Port. The BDP is used with the RAP to gain access to any of the PCnet-32 controller BCR locations.

Access to any of the BCR locations of the PCnet-32 controller is performed through the PCnet-32

controller's BCR Data Port (BDP). In order to access a particular BCR location, the Register Address Port (RAP) should first be written with the appropriate BCR address. The BDP now points to the selected BCR. A read of the BDP will yield the selected BCR's data. A write to the BDP will write to the selected BCR.

When programmed for WIO mode, the BDP has a width of 16 bits, hence, all BCR locations have 16 bits of width in WIO mode. Note that when operating in WIO mode, the upper two bytes of the data bus will be undefined since the byte mask will not be active for those bytes.

If DWIO mode has been invoked, then the BDP has a width of 32 bits, hence, all BCR locations have 32 bits of width and the upper two bytes of the data bus will be active, as indicated by the byte mask. In this case, note that the upper 16 bits of all BCR locations are reserved and written as zeros and read as undefined. Therefore, during BDP write operations in DWIO mode, the upper 16 bits of all BCR locations should be written as zeros.

Reset Register (S_RESET)

A read of the reset register creates an internal S_RESET pulse in the PCnet-32 controller. This read access cycle must be 16 bits wide in WIO mode and 32 bits wide in DWIO mode. The internal S_RESET pulse that is generated by this access is different from both the assertion of the hardware RESET pin (H_RESET) and from the assertion of the software STOP bit. Specifically, the Reset register's S_RESET will be the equivalent of the assertion of the RESET pin (H_RESET) assertion for all CSR locations, but S_RESET will have no effect at all on the BCR locations, and S_RESET will not cause a de-assertion of the HOLD pin.

The NE2100 LANCE based family of Ethernet cards requires that a write access to the reset register follows each read access to the reset register. The PCnet-32 controller does not have a similar requirement. The write access is not required but it does not have any harmful effects.

Write accesses to the reset register will have no effect on the PCnet-32 controller.

Note that a read of the Reset register will take longer than the normal I/O access time of the PCnet-32 controller. This is because an internal S_RESET pulse will be generated due to this access, and the access will not be allowed to complete on the system bus until the internal S_RESET operation has been completed. This is to avoid the problem of allowing a new I/O access to proceed while the S_RESET operation has not yet completed, which would result in erroneous data being returned by (or written into) the PCnet-32 controller. The length of a read of the Reset register can be as long as 128 BCLK cycles when Am386 mode has been selected and 64 BCLK cycles when Am486 mode or VESA VL-Bus mode has been selected.

Note that a read of the Reset register will **not** cause a de-assertion of the HOLD signal, if it happens to be active at the time of the read to the reset register. The HOLD signal will remain active until the HLDA signal is synchronously sampled as asserted. Following the read of the RESET register, on the next clock cycle after the HLDA signal is synchronously sampled as asserted, the PCnet-32 controller will de-assert the HOLD signal). No bus master accesses will have been performed during this brief bus ownership period.

Note that this behavior differs from that which occurs following the assertion of a minimum-width pulse on the RESET pin (H_RESET). A RESET pin assertion will cause the HOLD signal to de-assert within six clock cycles following the assertion. In the RESET pin case, the PCnet-32 controller will not wait for the assertion of the HLDA signal before de-asserting the HOLD signal.

Vendor Specific Word

This I/O offset is reserved for use by the system designer. The PCnet-32 controller will not respond to accesses directed toward this offset.

Reserved I/O Space

These locations are reserved for future use by AMD. The PCnet-32 controller does not respond to accesses directed toward these locations, but future AMD products that are intended to be upward compatible with the PCnet-32 controller may decode accesses to these locations. Therefore, the system designer may not utilize these I/O locations.

Hardware Access

PCnet-32 Controller Master Accesses

The particular signals involved in a PCnet-32 controller bus master transfer depends upon the bus mode that has been selected. There are two bus modes to choose from. They are:

- Am486 32-bit mode
- VESA VL-Bus mode

Complete descriptions of the signals involved in bus master transactions for each mode may be found in the pin description section of this document. Timing diagrams for master accesses may be found in the block description section for the Bus Interface Unit. This section simply lists the types of master accesses that will be performed by the PCnet-32 controller with respect to data size and address information.

The PCnet-32 controller will support master accesses only to 32-bit peripherals in Am486 environments. The PCnet-32 controller does not support master accesses to 16-bit peripherals in the 486 local bus mode.

The PCnet-32 controller will support master accesses to either 32-bit or 16-bit peripherals in VESA VL-Bus mode. Support of master accesses to 16-bit peripherals in VESA VL-Bus mode is provided through the $\overline{\text{LBS16}}$ input pin.

The PCnet-32 controller is not compatible with 8-bit systems, since there is no mode that supports PCnet-32 controller accesses to 8-bit peripherals.

Table 35 describes all possible bus master accesses that the PCnet-32 controller will perform. The rightmost column lists all operations that may execute the given access.

Table 35. Master Accesses

Access	R/W	$\overline{\text{BE3}}$–$\overline{\text{BE0}}$	Possible Instance
4-Byte Read	RD	0000	Descriptor Read or Initialization Block Read or Transmit Data Buffer Read
4-Byte Write	WR	0000	Descriptor Write or Receive Data Buffer Write
3-Byte Write	WR	1000	Receive Data Buffer Write
3-Byte Write	WR	0001	Receive Data Buffer Write
2-Byte Write	WR	1100	Receive Data Buffer Write
2-Byte Write	WR	1001*	Receive Data Buffer Write
2-Byte Write	WR	0011	Receive Data Buffer Write
1-Byte Write	WR	1110	Receive Data Buffer Write
1-Byte Write	WR	1101*	Receive Data Buffer Write
1-Byte Write	WR	1011*	Receive Data Buffer Write
1-Byte Write	WR	0111	Descriptor Write or Receive Data Buffer Write

**Cases marked with an asterisk represent extreme boundary conditions that are the result of programming one- and two-byte buffer sizes, and therefore will not be seen under normal circumstances.*

Note that all PCnet-32 controller master read operations will always activate all byte enables. (Note the exception, when $\overline{\text{LBS16}}$ has been asserted in VL-Bus mode, requiring a second access. In all $\overline{\text{LBS16}}$ cases, the second access (if required) will have $\overline{\text{BE1}}$ and $\overline{\text{BE0}}$ disabled. Therefore, no one-, two- or three-byte read operations are indicated in the table.

In the instance where a transmit buffer pointer address begins on a non-double-word boundary, the pointer will be truncated to the next double-word boundary address that lies below the given pointer address and the first read access from the transmit buffer will be indicated on the byte enable signals as a four-byte read from this address. Any data from byte lanes that lie outside of the boundary indicated by the buffer pointer will be discarded inside of the PCnet-32 controller.

Similarly, if the end of a transmit buffer occurs on a non-double-word boundary, then all byte lanes will be indicated as active by the byte enable signals, and any data from byte lanes that lie outside of the boundary indicated by the buffer pointer will be discarded inside of the PCnet-32 controller.

Slave Access to I/O Resources

The PCnet-32 controller is a 32-bit peripheral device on the system bus. However, the width of individual software resources on board the PCnet-32 controller may be either 16-bits or 32-bits. The PCnet-32 controller I/O resource widths are determined by the value of the DWIO bit (BCR 18, bit 7) as indicated in Table 36.

Note that when I/O resource width is defined as 32 bits (DWIO mode), the upper 16 bits of the I/O resource is reserved and written as ZEROs and read as undefined, *except for the APROM locations*. The APROM locations are the only I/O resources for which all 32 bits will have defined values. However, this is true only when the PCnet-32 controller is in DWIO mode.

Configuring the PCnet-32 controller for DWIO mode is accomplished whenever there is any attempt to perform a 32-bit write access to the RDP location (offset 10h). See the DWIO section for more details.

Table 37 describes all possible bus slave accesses that may be directed toward the PCnet-32 controller. (i.e. the PCnet-32 controller is the slave device during the transfer.) The four byte columns (D31–D24, D23–D16, D15–D8, D7–D0) indicate the value on the data bus during the access.

Table 36. I/O Resource Access

DWIO Setting	PCnet-32 Controller I/O Resource Width	Example Application
DWIO = 0	16-bit	Existing PCnet-ISA driver that assumes 16-bit I/O mapping and 16-bit resource widths
DWIO = 1	32-bit	New drivers written specifically for the PCnet-32 controller

Table 37. Slave Accesses

R/W	$\overline{BE3}$ – $\overline{BE0}$	D31–D24	D23–D16	D15–D8	D7–D0	Comments
RD	0000	Data	Data	Data	Data	Double word access to double word address, e.g. 300, 30C, 310 (DWIO mode only)
RD	1100	Undef	Undef	Data	Data	*WIO Mode Only:* Word access to even word address, e.g. 300, 30C, 310
RD	0011	Data	Data	Copy	Copy	*WIO Mode Only:* Word access to odd word address, e.g. 302, 30E, 312
RD	1110	Undef	Undef	Undef	Data	*WIO Mode APROM Read Only:* Byte access to lower byte of even word address, e.g. 300, 304
RD	1101	Undef	Undef	Data	Undef	*WIO Mode APROM Read Only:* Byte access to upper byte of even word address, e.g. 301, 305
RD	1011	Undef	Data	Undef	Copy	*WIO Mode APROM Read Only:* Byte access to lower byte of odd word address, e.g. 302, 306
RD	0111	Data	Undef	Copy	Undef	*WIO Mode APROM Read Only:* Byte access to upper byte of odd word address, e.g. 303, 307
WR	0000	Data	Data	Data	Data	Double word access to double word address, e.g. 300, 30C, 310 (DWIO mode only)
WR	1100	Undef	Undef	Data	Data	*WIO Mode Only:* Word access to even word address, e.g. 300, 30C, 310
WR	0011	Data	Data	Undef	Undef	*WIO Mode Only:* Word access to odd word address, e.g. 302, 30E, 312

Data = *indicates the position of the active bytes.*

Copy = *indicates the positions of copies of the active bytes.*

Undef = *indicates byte locations that are undefined during the transfer.*

EEPROM Microwire Access

The PCnet-32 controller contains a built-in capability for reading and writing to an external EEPROM. This built-in capability consists of a microwire interface for direct connection to a microwire compatible EEPROM, an automatic EEPROM read feature, and a user-programmable register that allows direct access to the microwire interface pins.

Automatic EEPROM Read Operation

Shortly after the de-assertion of the RESET pin, the PCnet-32 controller will read the contents of the EEPROM that is attached to the microwire interface.The automatic EEPROM read begins with EECS being asserted approximately 2.5 EESK periods $(\frac{2.5}{t_{42}})$ following the de-assertion of the $\overline{RESET}$ pin.

Because of this automatic-read capability of the PCnet-32 controller, an EEPROM can be used to program many of the features of the PCnet-32 controller at power-up, allowing system-dependent configuration information to be stored in the hardware, instead of inside of operating code. PCnet-32 controller interrupt pins will be floated during H_RESET and will remain floated until either the EEPROM has been successfully read, or, following an EEPROM read failure, a Software Relocatable Mode sequence has been successfully executed.

If an EEPROM exists on the microwire interface, the PCnet-32 controller will read the EEPROM contents at the end of the H_RESET operation. The EEPROM contents will be serially shifted into a temporary register and then sent to various register locations on board the PCnet-32 controller. System bus interaction will not occur during the EEPROM read operation after H_RESET, since H_RESET will put the PCnet-32 controller into a state that will not recognize any I/O accesses and the PCnet-32 controller will not yet be at an operating point that requires it to request the bus for bus mastering.

Thirty-four bytes of the EEPROM are set aside for PCnet-32 configuration programming and 2 bytes of the EEPROM are set aside for user programming of logic that is external to the PCnet-32 controller and may or may not be pertinent to the operation of the PCnet-32 controller within the system. The user may gain access to the EEPROM data by snooping the PCnet-32 controller automatic read operation. Logic may be attached to the EEPROM interface to snoop the entire EEPROM read operation using the microwire signals directly, or a simpler scheme may be invoked by taking advantage of an additional signal provided by the PCnet-32 controller (i.e. the SHFBUSY signal).

A checksum verification is performed on the data that is read from the EEPROM. If the checksum verification

of the EEPROM data fails, then at the end of the EEPROM read sequence, the PCnet-32 controller will force all EEPROM-programmable register locations back to their H_RESET default values and then the PCnet-32 controller will enter Software Relocatable Mode. The 8-bit checksum for the entire 36 bytes of the EEPROM should be FFh. In the event of a checksum failure, Software Relocatable Mode is entered (PCnet-32 controller begins snooping for a 12-byte sequence) within 1 EESK period following the de-assertion of EECS ($\frac{1}{t_{42}}$).

If the absence of an EEPROM has been signaled by the EESK/LED1/SFBD pin at the time of the automatic read operation, then the PCnet-32 controller will recognize this condition and will abort the automatic read operation and reset both the PREAD and PVALID bits in BCR19. At this point, the PCnet-32 controller will enter the Software Relocatable Mode, and the EEPROM-programmable registers will be assigned their H_RESET default values. Software Relocatable Mode is entered (PCnet-32 controller begins snooping for 12-byte sequence) within 2.5 EESK periods ($\frac{2.5}{t_{42}}$) following the de-assertion of the RESET pin when absence of an EEPROM is signalled by the EESK/LEDI SFBD pin.

If the user wishes to modify any of the configuration bits that are contained in the EEPROM, then the 7 command, data and status bits of BCR19 can be used to write to the EEPROM. After writing to the EEPROM, the host should set the PREAD bit of BCR19. This action forces a PCnet-32 controller re-read of the EEPROM so that the new EEPROM contents will be loaded into the EEPROM-programmable registers on board the PCnet-32 controller. (The EEPROM-programmable registers may also be reprogrammed directly, but only information that is stored in the EEPROM will be preserved at system power-down.) When the PREAD bit of BCR19 is set, it will cause the PCnet-32 controller to ignore further accesses on the system interface bus until the completion of the EEPROM read operation.

EEPROM Auto-Detection

The PCnet-32 controller uses the EESK/LED1/SFBD pin to determine if an EEPROM is present in the system. At all rising BCLK edges during the assertion of the RESET pin, the PCnet-32 controller will sample the value of the EESK/LED1/SFBD pin. If the sampled value is a ONE, then the PCnet-32 controller assumes that an EEPROM is present, and the EEPROM read operation begins shortly after the RESET pin is de-asserted. If the sampled value of EESK/LED1/SFBD is a ZERO, then the PCnet-32 controller assumes that an external pull-down device is holding the EESK/LED1/SFBD pin low, and therefore, there is no EEPROM in the system. In this case, the PCnet-32 controller will enter Software Relocatable Mode. Note that if the designer creates a system that contains an LED circuit on the EESK/LED1/SFBD pin but has no EEPROM present, then the EEPROM auto-detection function will incorrectly conclude that an EEPROM is present in the system. However, this will not pose a problem for the PCnet-32 controller, since it will recognize the lack of an EEPROM at the end of the read operation, when the checksum verification fails. At this point, the PCnet-32 controller will enter Software Relocatable Mode.

The real intention of the EEPROM auto-detection feature is to allow a user to preempt a “good EEPROM” by temporarily resistively shorting the EESK/LED1/SFBD pin to ground. This may need to be done if an add-in card containing the PCnet-32 controller and its EEPROM has been programmed in one system and then later moved to a different system without also moving configuration information that indicates the I/O Base address of the card. The card would power up in the second system with an unknown I/O Base address if the configuration information were not carried with the card to the new system. By allowing the EESK/LED1/SFBD pin to be temporarily resistively shorted to ground, the PCnet-32 controller is fooled into believing that the EEPROM does not exist, and it will enter Software Relocatable Mode. This allows the new system to reconfigure the I/O Base address of the PCnet-32 controller to a location that is compatible to the parameters of the new system. This information will then be written into the EEPROM by a configuration utility through the EEPROM access port (BCR19), in spite of the fact that the PCnet-32 controller believes that there is no EEPROM. The resistive short to ground may now be removed, and the next power-up of the system will place the PCnet-32 controller into a I/O location that is known by this system. When the PREAD bit of BCR19 is set, an EEPROM read operation will be performed, regardless of the value of the EESK/ LED1/SFBD pin. Note that the H_RESET-generated EEPROM read operation always obeys the EESK/ LED1/SFBD indication.

Table 38 indicates the possible combinations of EEDET and the existence of an EEPROM and the resulting operations that are possible on the EEPROM microwire interface. *Note that the EEDET value (BCR19, bit 3) is determined from EESK/LED1/SFBD pin setting, and it may be set even though there is no EEPROM present.*

Table 38. Effect of EEDET on EEPROM Operation

EEDET Value (BCR19[3])	EEPROM Connected?	Result if PREAD is set to ONE	Result of Automatic EEPROM Read Operation Following H_RESET
0	No	EEPROM read operation is attempted. Entire read sequence will occur, checksum failure will result, PVALID is reset to ZERO.	First TWO EESK clock cycles are generated, then EEPROM read operation is aborted and PVALID is reset to ZERO.
0	Yes	EEPROM read operation is attempted. Entire read sequence will occur, checksum operation will pass, PVALID is set to ONE.	First TWO EESK clock cycles are generated, then EEPROM read operation is aborted and PVALID is reset to ZERO.
1	No	EEPROM read operation is attempted. Entire read sequence will occur, checksum failure will result, PVALID is reset to ZERO.	EEPROM read operation is attempted. Entire read sequence will occur, checksum failure will result, PVALID is reset to ZERO.
1	Yes	EEPROM read operation is attempted. Entire read sequence will occur, checksum operation will pass, PVALID is set to ONE.	EEPROM read operation is attempted. Entire read sequence will occur, checksum operation will pass, PVALID is set to ONE.

Systems Without an EEPROM

Some systems may be able to save the cost of an EEPROM by storing the ISO 8802-3 (IEEE/ANSI 802.3) station address and other configuration information somewhere else in the system. For such a system, a two step process is required. The first step will get the PCnet-32 controller into its normal operating mode within the system. The second step will load the IEEE station address. The designer has several choices:

1) *If the LED1 and SFBD functions are not needed in the system*, then the system designer may connect the EESK/LED1/SFBD pin to a resistive pull-down device. This will indicate to the EEPROM auto-detection function that no EEPROM is present, and the PCnet-32 controller will use Software Relocatable Mode to acquire its I/O Base Address and other configuration information (See the section on Software Relocatable Mode).

2) *If either of the LED1 or SFBD functions is needed in the system*, then the system designer will connect the EESK/LED1/SFBD pin to a resistive pull-up device and the EEPROM auto-detection function will incorrectly conclude that an EEPROM is present in the system. However, this will not pose a problem for the PCnet-32 controller, since it will recognize the lack of an EEPROM at the end of the read operation, when the checksum verification fails. At this point, the PCnet-32 controller into the Software Relocatable Mode to acquire its I/O Base Address and other configuration information (See the section on Software Relocatable Mode).

In either case, following the execution of Software Relocatable Mode, additional information, including the ISO 8802-3 (IEEE/ANSI 802.3) station address, may be loaded into the PCnet-32 controller. Note that the IESRWE bit (bit 8 of BCR2) must be set before the PCnet-32 controller will accept writes to the APROM offsets within the PCnet-32 controller I/O resources map. Startup code in the system BIOS can perform the Software Relocatable Mode accesses, the IESRWE bit write, and the APROM writes.

If compatibility to existing driver code is desired, then it is not recommended that the ISO 8802-3 (IEEE/ANSI 802.3) station address be loaded into the Initialization Block structure in memory instead of the APROM locations, since existing code typically expects to find the ISO 8802-3 (IEEE/ANSI 802.3) station address at the APROM offsets from the PCnet-32 controller I/O Base Address.

Direct Access to the Microwire Interface

The user may directly access the microwire port through the EEPROM register, BCR19. This register contains bits that can be used to control the microwire interface pins. By performing an appropriate sequence of I/O accesses to BCR19, the user can effectively write to and read from the EEPROM. This feature may be used by a system configuration utility to program hardware configuration information into the EEPROM.

EEPROM-Programmable Registers

The following registers contain configuration information that will be programmed automatically during the EEPROM read operation:

1) I/O offsets 0h–Fh	Address PROM locations
2) BCR2	Miscellaneous Configuration register
3) BCR16	I/O Base Address Lower
4) BCR17	I/O Base Address Upper
5) BCR18	Burst Size and Bus Control Register
6) BCR21	Interrupt Control Register

If the PREAD bit (BCR19) is reset to ZERO and the PVALID bit (BCR19) is reset to ZERO, then the EEPROM read has experienced a failure and the contents of the EEPROM programmable register will be set to default H_RESET values. At this point, the PCnet-32 controller will enter Software Relocatable Mode.

Note that accesses to the Address PROM I/O locations do not directly access the Address EEPROM itself. Instead, these accesses are routed to a set of “shadow” registers on board the PCnet-32 controller that are loaded with a copy of the EEPROM contents during the automatic read operation that immediately follows the H_RESET operation.

EEPROM MAP

The automatic EEPROM read operation will access 18 words (i.e. 36 bytes) of the EEPROM. The format of the EEPROM contents is shown in Table 39, beginning with the byte that resides at the lowest EEPROM address.

Table 39. EEPROM Content

EEPROM Word Addr	EEPROM Contents Byte Addr			
	Byte Addr	MSB (Most Significant Byte)	Byte Addr	LSB (Lease Significant Byte)
0 (Lowest Address)	1	2nd byte of the ISO 8802-3 (IEEE/ANSI 802.3) station physical address for this node	0	First byte of the ISO 8802-3 (IEEE/ANSI 802.3) station physical address for this node, where first byte refers to the first byte to appear on the 802.3 medium
1	3	4th byte of the node address	2	3rd byte of the node address
2	5	6th byte of the node address	4	5th byte of the node address
3	7	Reserved location: *must be 00h*	6	Reserved location: *must be 00h*
4	9	Hardware ID: must be 10h if compatibility to AMD drivers is desired	8	Driver IRQ: Must be programmed to the system IRQ channel number being used if AMD drivers are used.
5	B	User programmable space	A	User programmable space
6	D	MSB of two-byte checksum, which is the sum of bytes 0-B and bytes E and F	C	LSB of two-byte checksum, which is the sum of bytes 0-B and bytes E and F
7	F	Must be ASCII “W” (57h) if compatibility to AMD driver software is desired	E	Must be ASCII “W” (57h) if compatibility to AMD driver software is desired
8	11	BCR16[15:8] (I/O Base Address Lower)	10	BCR16[7:0] (I/O Base Address Lower)
9	13	BCR17[15:8] (I/O Base Address Upper)	12	BCR17[7:0] (I/O Base Address Upper)
A	15	BCR18[15:8] (Burst Size and Bus Control)	14	BCR18[7:0] (Burst Size and Bus Control)
B	17	BCR2[15:8] (Miscellaneous configuration)	16	BCR2[7:0] (Miscellaneous configuration)
C	19	BCR21[15:8] (Interrupt Control)	18	BCR21[7:0] (Interrupt Control)
D	1B	Reserved location: *must be 00h*	1A	Reserved location: *must be 00h*
E	1D	Reserved location: *must be 00h*	1C	Reserved location: *must be 00h*
F	1F	checksum *adjust* byte for the first 36 bytes of the EEPROM contents; checksum of the first 36 bytes of the EEPROM should total to FFh	1E	Reserved location: *must be 00h*
10	21	Reserved location: *must be 00h*	20	Reserved location: *must be 00h*
11	23	User programmable byte locations	22	User programmable byte locations

Note that the first bit out of any WORD location in the EEPROM is treated as the MSB of the register that is being programmed. For example, the first bit out of EEPROM WORD location 08h will be written into BCR16[15], the second bit out of EEPROM WORD location 08h will be written into BCR16[14], etc.

There are two checksum locations within the EEPROM. The first is required for the EEPROM address. This checksum will be used by AMD driver software to verify that the ISO 8802-3 (IEEE/ANSI 802.3) station address has not been corrupted. The value of bytes C and D should match the 16-bit sum of bytes 0 through B and E and F. The second checksum location “byte 1F ” is not a checksum total, but is, instead, a checksum adjustment. The value of this byte should be such that the total 8-bit checksum for the entire 36 bytes of EEPROM data equals the value FFh. The checksum adjust byte is needed by the PCnet-32 controller in order to verify that the EEPROM contents have not been corrupted.

Byte address 8h of the EEPROM map contains the driver IRQ field. The content of this field is used by AMD drivers to program the interrupt channel being used by the PCnet-32. Note that the PCnet-32 interrupt pin selection is NOT effected by this field. The interrupt pin selection is controlled only by the appropriate bits in BCR21. The system interrupt channel associated with each of the PCnet-32 INTR pins is application-dependent. AMD drivers utilize byte location 8h of the EEPROM to resolve this dependency.

EEPROM-Programming of System Logic

When the user has shareable hardware resources in the system and wishes to have these resources programmed at power up, the user may desire to take advantage of the extra space in the EEPROM that is used to configure the PCnet-32 controller in order to store the additional configuration information. The PCnet-32 controller provides a convenient means of access for the user who wishes to utilize this space. The schematic in Figure 34 illustrates an example of logic that is used to generate static control signals for some programmable features of the system, where the programming information is stored on board the PCnet-32 controller’s EEPROM and the logic is to be automatically programmed after RESET.

Note that the EECS signal pulses low during the EEPROM read operation and is therefore unsuitable for use as a gate signal for the programmable logic outputs. PCnet-32 controller provides an additional signal, SHFBUSY, which will remain active HIGH during the entire EEPROM read operation. This signal will therefore be suitable for use as the gate of the programmable logic outputs as shown in the diagram. Note that since most of the EEPROM microwire interface signals are multiplexed with other PCnet-32 controller functions, it is necessary for the SHFBUSY pin to enable the shift path of the programmable logic, otherwise the shift path would become active when the EESK and EEDO functions were operating as their alternate functions.

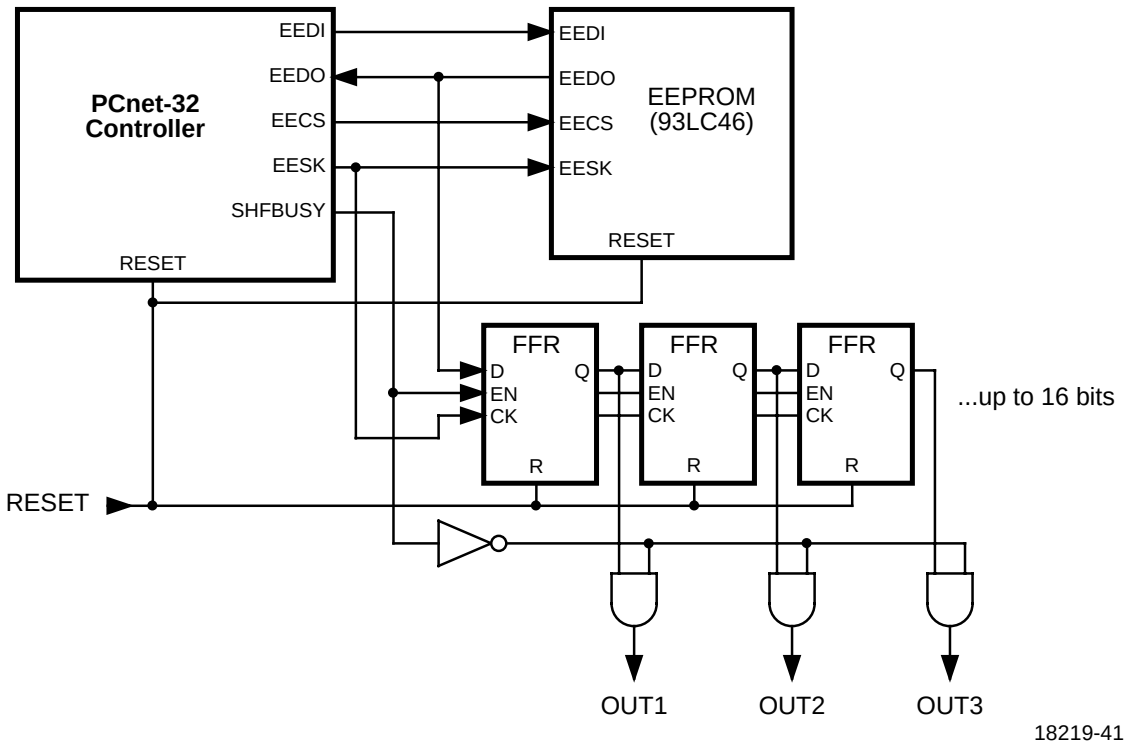


Figure 34. Programming System Logic Through the PCnet-32 EEPROM Read Operation

SHFBUSY will be HIGH for the entire EEPROM read operation, and therefore all EEPROM contents will have been shifted through the external logic before it settles on its final, programmed value. If the EEPROM checksum verification fails, then the EEPROM contents are assumed to be invalid, and the SHFBUSY signal will remain HIGH after the completion of the EEPROM read operation. This action will prevent incorrect system logic values from being driven into the system. If the EEPROM checksum verification passes, then the EEPROM contents are assumed to be valid, and the SHFBUSY signal will return to a LOW state after the completion of the EEPROM read operation.

Transmit Operation

The transmit operation and features of the PCnet-32 controller are controlled by programmable options.

Transmit Function Programming

Automatic transmit features such as retry on collision, FCS generation/transmission, and pad field insertion can all be programmed to provide flexibility in the (re-) transmission of messages.

Disable retry on collision (DRTY) is controlled by the DRTY bit of the Mode register (CSR15) in the initialization block.

Automatic pad field insertion is controlled by the APAD_XMT bit in CSR4. If APAD_XMT is set, automatic pad field insertion is enabled, the DXMTFCS feature is over-ridden, and the 4 byte FCS will be added to the transmitted frame unconditionally. If APAD_XMT is clear, no pad field insertion will take place and runt packet transmission is possible.

The disable FCS generation/transmission feature can be programmed dynamically on a frame by frame basis. See the ADD_FCS description of TMD1.

Transmit FIFO Watermark (XMTFW in CSR80) sets the point at which the BMU requests more data from the transmit buffers for the FIFO. A minimum of “XMTFW” empty spaces must be available in the transmit FIFO before the BMU will request the system bus in order to transfer transmit packet data into the transmit FIFO.

Transmit Start Point (XMTSP in CSR80) sets the point when the transmitter actually attempts to transmit a frame onto the media. A minimum of “XMTSP” bytes must be written to the transmit FIFO for the current frame before transmission of the current frame will begin. (When automatically padded packets are being sent, it is conceivable that the XMTSP is not reached when all of the data has been transferred to the FIFO. In this case, the transmission will begin when all of the packet data has been placed into the transmit FIFO.)

When the entire frame is in the FIFO, attempts at transmission of preamble will commence regardless of the

value in XMTSP. The default value of XMTSP is 10b, meaning 64 bytes full.

Automatic Pad Generation

Transmit frames can be automatically padded to extend them to 64 data bytes (excluding preamble). This allows the minimum frame size of 64 bytes (512 bits) for 802.3/Ethernet to be guaranteed with no software intervention from the host/controlling process. Setting the APAD_XMT bit in CSR4 enables the automatic padding feature. The pad is placed between the LLC data field and FCS field in the 802.3 frame (see Figure 35). FCS is always added if the frame is padded, regardless of the state of DXMTFCS. The transmit frame will be padded by bytes with the value of 00h. The default value of APAD_XMT is 0; this will disable auto pad generation after H_RESET.

It is the responsibility of upper layer software to correctly define the actual length field contained in the message to correspond to the total number of LLC Data bytes encapsulated in the packet (length field as defined in the ISO 8802-3 (IEEE/ANSI 802.3) standard). The length value contained in the message is not used by the PCnet-32 controller to compute the actual number of pad bytes to be inserted. The PCnet-32 controller will append pad bytes dependent on the actual number of bits transmitted onto the network. Once the last data byte of the frame has completed, prior to appending the FCS, the PCnet-32 controller will check to ensure that 544 bits have been transmitted. If not, pad bytes are added to extend the frame size to this value, and the FCS is then added.

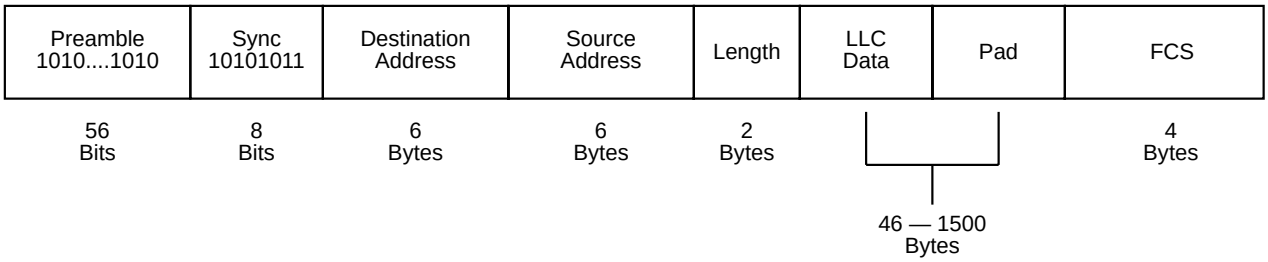


Figure 35. ISO 8802-3 (IEEE/ANSI 802.3) Data Frame

The 544 bit count is derived from the following:

Minimum frame size (excluding preamble, including FCS)	64 bytes	512 bits
Preamble/SFD size	8 bytes	64 bits
FCS size	4 bytes	32 bits

To be classed as a minimum size frame at the receiver the transmitted frame must contain:

Preamble + (Min Frame Size + FCS) bits

At the point that FCS is to be appended, the transmitted frame should contain:

Preamble + (Min Frame Size–FCS) bits

64 + (512–32) bits

A minimum length transmit frame from the PCnet-32 controller will, therefore, be 576 bits, after the FCS is appended.

The Ethernet specification assumes that minimum length messages will be at least 64 bytes in length.

Transmit FCS Generation

Automatic generation and transmission of FCS for a transmit frame depends on the value of DXMTFCS bit in CSR15. When DXMTFCS = 0 the transmitter will generate and append the FCS to the transmitted frame. If the automatic padding feature is invoked (APAD_XMT is set in CSR4), the FCS will be appended by the PCnet-32 controller regardless of the state of DXMTFCS. Note that the calculated FCS is transmitted most significant bit first. The default value of DXMTFCS is 0 after H_RESET.

Transmit Exception Conditions

Exception conditions for frame transmission fall into two distinct categories. Those which are the result of normal network operation, and those which occur due to abnormal network and/or host related events.

Normal events which may occur and which are handled autonomously by the PCnet-32 controller include collisions within the slot time with automatic retry. The PCnet-32 controller will ensure that collisions which occur within 512 bit times from the start of transmission (including preamble) will be automatically retired with no host intervention. The transmit FIFO ensures this by

guaranteeing that data contained within the FIFO will not be overwritten until at least 64 bytes (512 bits) of preamble plus address, length and data fields have been transmitted onto the network without encountering a collision.

If 16 total attempts (initial attempt plus 15 retries) fail, the PCnet-32 controller sets the RTRY bit in the current transmit TDTE in host memory (TMD2), gives up ownership (resets the OWN bit to zero) for this frame, and processes the next frame in the transmit ring for transmission. Abnormal network conditions include:

- Loss of carrier.
- Late collision.
- SQE Test Error (does not apply to 10BASE-T port)

None of the abnormal network conditions should not occur on a correctly configured 802.3 network, and will be reported if they do.

When an error occurs in the middle of a multi-buffer frame transmission, the error status will be written in the current descriptor. The OWN bit(s) in the subsequent descriptor(s) will be reset until the STP (the next frame) is found.

Loss of Carrier

A loss of carrier condition will be reported if the PCnet-32 controller cannot observe receive activity while it is transmitting on the AUI port. After the PCnet-32 controller initiates a transmission it will expect to see data “looped-back” on the DI± pair. This will internally generate a “carrier sense,” indicating that the integrity of the data path to and from the MAU is intact, and that the MAU is operating correctly. This “carrier sense” signal must be asserted about 6 bit times before the last transmitted bit on DO±. If “carrier sense” does not become active in response to the data transmission, or becomes inactive before the end of transmission, the loss of carrier (LCAR) error bit will be set in TMD2 after the frame has been transmitted. The frame will not be retried on the basis of an LCAR error.

When the 10BASE-T port is selected, LCAR will be reported for every frame transmitted during the Link fail condition.

Late Collision

A late collision will be reported if a collision condition occurs after one slot time (512 bit times) after the transmit process was initiated (first bit of preamble commenced). The PCnet-32 controller will abandon the transmit process for the particular frame, set Late Collision (LCOL) in the associated TMD2, and process the next transmit frame in the ring. Frames experiencing a late collision will not be retried. Recovery from this condition must be performed by upper layer software.

SQE Test Error

During the inter packet gap time following the completion of a transmitted message, the AUI CI± pair is asserted by some transceivers as a self-test. The integral Manchester Encoder/Decoder will expect the SQE Test Message (nominal 10 MHz sequence) to be returned via the CI± pair, within a 40 network bit time period after DI± goes inactive (this does not apply if the 10BASE-T port is selected). If the CI± input is not asserted within the 40 network bit time period following the completion of transmission, then the PCnet-32 controller will set the CERR bit in CSR0. CERR will be asserted in 10BASE-T mode after transmit if T-MAU is in Link Fail state. CERR will never cause INTR to be activated. It will, however, set the ERR bit in CSR0.

Receive Operation

The receive operation and features of the PCnet-32 controller are controlled by programmable options.

Receive Function Programming

Automatic pad field stripping is enabled by setting the ASTRP_RCV bit in CSR4. this can provide flexibility in the reception of messages using the 802.3 frame format.

All receive frames can be accepted by setting the PROM bit in CSR15. When PROM is set, the PCnet-32 controller will attempt to receive all messages, subject to minimum frame enforcement. Promiscuous mode overrides the effect of the Disable Receive Broadcast bit on receiving broadcast frames.

The point at which the BMU will start to transfer data from the receive FIFO to buffer memory is controlled by the RCVFW bits in CSR80. The default established during H_RESET is 10b which sets the threshold flag at 64 bytes empty.

Automatic Pad Stripping

During reception of an 802.3 frame the pad field can be stripped automatically.

ASTRP_RCV (CSR4, bit 10) = 1 enables the automatic pad stripping feature. The pad field will be stripped before the frame is passed to the FIFO, thus preserving FIFO space for additional frames. The FCS field will also be stripped, since it is computed at the transmitting station based on the data and pad field characters, and will be invalid for a receive frame that has had the pad characters stripped.

The number of bytes to be stripped is calculated from the embedded length field, as defined in the ISO 8802-3 (IEEE/ANSI 802.3) definition, contained in the frame. The length indicates the actual number of LLC data bytes contained in the message. Any received frame which contains a length field less than 46 bytes will have the pad field stripped (if ASTRP_RCV is set).

Receive frames which have a length field of 46 bytes or greater will be passed to the host unmodified.

Note that for some network protocols, the value passed in the Ethernet Type and/or 802.3 Length field is not compliant with either standard and may cause problems.

Figure 36 shows the byte/bit ordering of the received length field for an 802.3 compatible frame format.

Receive FCS Checking

Reception and checking of the received FCS is performed automatically by the PCnet-32 controller. Note that if the Automatic Pad Stripping feature is enabled, the FCS for padded frames will be verified against the value computed for the incoming bit stream including pad characters, but the FCS value for a padded frame will not be passed to the host. If an FCS error is detected in any frame, the error will be reported in the CRC bit in RMD1.

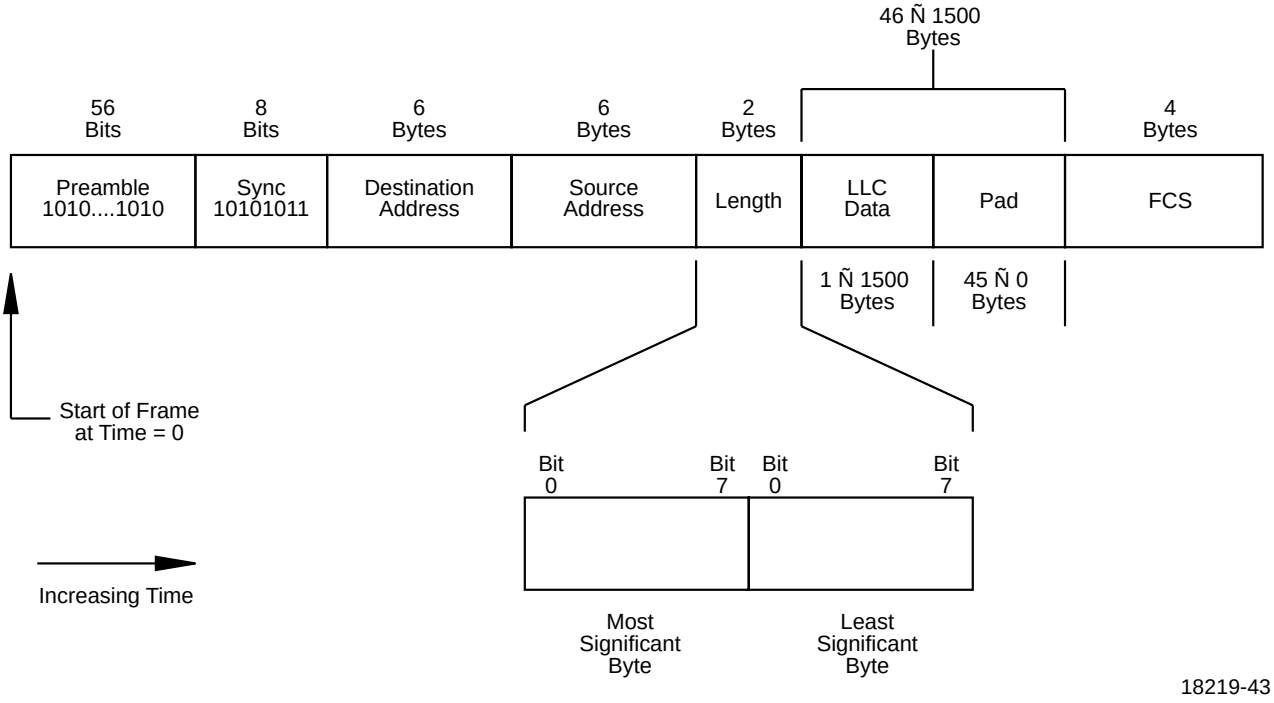


Figure 36. ISO 8802.3 (IEEE/ANSI 802.3) Data Frame

Receive Exception Conditions

Exception conditions for frame reception fall into two distinct categories: Those which are the result of normal network operation, and those which occur due to abnormal network and/or host related events.

Normal events which may occur and which are handled autonomously by the PCnet-32 controller are basically collisions within the slot time and automatic runt packet rejection. The PCnet-32 controller will ensure that collisions which occur within 512 bit times from the start of reception (excluding preamble) will be automatically deleted from the receive FIFO with no host intervention. The receive FIFO will delete any frame which is composed of fewer than 64 bytes provided that the Runt Packet Accept (RPA bit in CSR124) feature has not been enabled. This criterion will be met regardless of whether the receive frame was the first (or only) frame in the FIFO or if the receive frame was queued behind a previously received message.

Abnormal network conditions include:

- FCS error
- Late collision

Host-related receive exception conditions include MISS, BUFF, and OFLO. These are described in the BMU section.

Loopback Operation

Loopback is a mode of operation intended for system testing. In this mode the transmitter and receiver are both operating at the same time so that the controller receives its own transmissions. The controller provides two types of internal loopback and one type of external loopback. In internal loopback mode the transmitter

data can be looped back to the receiver at one of two places inside the controller without actually transmitting any data to the external network. The receiver will move the received data to the external network. The receiver will move the received data to the next receive buffer, where it can be examined by software. Alternatively, in external loopback mode, data can be transmitted to and received from the external network.

There are restrictions on loopback operation. The PCnet-32 controller has only one FCS generator circuit. The FCS generator can be used by the transmitter to generate the FCS that is appended to the frame, or it can be used by the receiver to verify the FCS of the received frame. It can not be used by the receiver and transmitter at the same time.

If the FCS generator is connected to the receiver, the transmitter will not append an FCS to the frame, but the receiver will check for one. The user can, however, calculate the FCS value for a frame and include this four-byte number in the transmit buffer.

If the FCS generator is connected to the transmitter, the transmitter will append an FCS to the frame, but the receiver will not check it. However, the user can verify the FCS by software.

During loopback the FCS logic can be allocated to the receiver by setting DXMTFCS = 1 in CSR15.

If DXMTFCS=0, the MAC Engine will calculate and append the FCS to the transmitted message. The receive message passed to the host will therefore contain an additional 4 bytes of FCS. In this loopback configuration, the receive circuitry cannot detect FCS errors if they occur.

If DXMTFCS=1, the last four bytes of the transmit message must contain the (software generated) FCS computed for the transmit data preceding it. The MAC Engine will transmit the data without addition of an FCS field, and the FCS will be calculated and verified at the receiver.

The loopback facilities of the MAC Engine allow full operation to be verified without disturbance to the network. Loopback operation is also affected by the state of the Loopback Control bits (LOOP, MENDECL, and INTL) in CSR15. This affects whether the internal MENDEC is considered part of the internal or external loop-back path.

The multicase address detection logic uses the FCS generator. Therefore, when in the loopback mode(s), the multicast address detection feature of the MAC Engine, programmed by the contents of the Logical Address Filter (LADRF [63:0] in CSRs 8-11) can only be tested when DXMTFCS=1, allocating the FCS generator to the receiver. All other features operate identically in loopback as in normal operation, such as automatic transmit padding and receive pad stripping.

When performing an internal loopback, no frame will be transmitted to the network. However, when the PCnet-32 controller is configured for internal loopback the receiver will not be able to detect network traffic. External loopback tests will transmit frames onto the network when the AUI port is selected. Runt Packet Accept is automatically enabled when any loopback mode is invoked.

Loopback mode can be performed with any frame size. Runt Packet Accept is internally enabled (RPA bit in CSR 124 is not affected) when any loopback mode is invoked. This is to be backwards compatible to the LANCE (Am7990) software.

When external loopback is performed while the 10BASE-T MAU is selected, collision detection is disabled. This is necessary, because a collision in a 10BASE-T system is defined as activity on the transmitter outputs and receiver inputs at the same time, which is exactly what happens during external loopback.

Since a 10BASE-T hub does not normally feed the station's transmitter outputs back into the station's receiver inputs, the use of external loopback in a 10BASE-T system usually requires some sort of external hardware that connects the outputs of the 10BASE-T MAU to its inputs.

LED Support

The PCnet-32 controller can support up to 4 LEDs.

LED outputs $\overline{\text{LNKST}}$, LED1 and LED2 allow for direct connection of an LED and its supporting pull-up device. LED output LEDPRE3 may require an additional buffer between the PCnet-32 controller output pin and the LED and its supporting pull-up device.

Because the LEDPRE3 output is multiplexed with other PCnet-32 controller functions, it may not always be possible to connect an LED circuit directly to the LEDPRE3 pin. For example, when an LED circuit is directly connected to the EEDO/LEDPRE3/SRD pin, then it is not possible for most serial EEPROM devices to sink enough I_{OL} to maintain a valid low level on the EEDO input to the PCnet-32 controller. Therefore, in applications that require both an EEPROM and a fourth LED, then it is necessary to buffer the LEDPRE3 circuit from the EEPROM-PCnet-32 controller connection. The LED registers in the BCR resource space allow each LED output to be programmed for either active high or active low operation, so that both inverting and non-inverting buffering choices are possible.

In applications where an EEPROM is not needed, the LEDPRE3 pin may be directly connected to an LED circuit. The PCnet-32 controller LEDPRE3 pin driver will be able to sink enough current to properly drive the LED circuit.

By default, after H_RESET, the 4 LED outputs are configured as shown in Table 40.

Table 40. LED Configuration

LED Output	Default Interpretation	Default Drive Enable	Default Output Polarity
$\overline{\text{LNKST}}$	Link Status	Enabled	Active LOW
LED1	Receive	Enabled	Active LOW
LED2	Receive Polarity	Enabled	Active LOW
LEDPRE3	Transmit	Enabled	Active LOW

For each LED register, each of the status signals is ANDed with its enable signal, and these signals are all OR'd together to form a combined status signal. Each LED pin's combined status signal runs to a pulse stretcher, which consists of a 3-bit shift register clocked at 38 Hz (26 ms). The data input of each shift register is normally at logic 0. The OR gate output for each LED register asynchronously sets all three bits of its shift register when the output becomes asserted. The inverted output of each shift register is used to control an LED pin. Thus the pulse stretcher provides 2–3 clocks of stretched LED output, or 52 ms to 78 ms.

Figure 37 shows the LED signal circuit that exists for each LED pin.

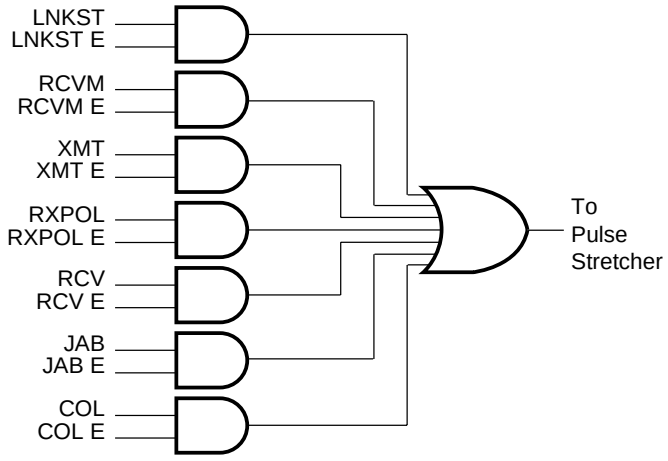


Figure 37. On-Chip LED Control Logic

H_RESET, S_RESET and STOP

There are three different types of RESET operations that may be performed on the PCnet-32 device, H_RESET, S_RESET and STOP. These names have been used throughout the document. The following is a description of each type of RESET operation:

H_RESET

H_RESET= HARDWARE_RESET is a PCnet-32 RESET operation that has been created by the proper assertion of the RESET pin of the PCnet-32 device. When the minimum pulse width timing as specified in the RESET pin description has been satisfied, then an internal RESET operation will be performed.

H_RESET will RESET all of or some portions of CSR0, 3, 4, 15, 58, 80, 82, 100, 112, 114, 122, 124 and 126 to default values; H_RESET will RESET all of or some portions of BCR 2, 4, 5, 6, 7, 18, 19, 20, 21 to default values. H_RESET will cause the microcode program to jump to its RESET state. Following the end of the H_RESET operation, the PCnet-32 controller will attempt to read the EEPROM device through the EEPROM microwire interface. The H_RESET operation will unconditionally cause all INTR pins to become inactive. (Note that there may be either 2 or 4 INTR pins, depending upon the JTAGSEL pin setting.) The H_RESET operation will unconditionally cause the HOLD signal to become de-asserted.

H_RESET will reset T-MAU to Link Fail state. H_RESET is generated by proper assertion of either the RESET or $\overline{\text{RESET}}$ pin, depending upon the mode that has been selected through the LB/$\overline{\text{VESA}}$ pin.

S_RESET

S_RESET = SOFTWARE_RESET is a PCnet-32 RESET operation that has been created by a read access to the RESET REGISTER which is located at offset 14h from the PCnet-32 controller I/O base address.

S_RESET will RESET all of or some portions of CSR0, 3, 4, 15, 80, 100 and 124 to default values. S_RESET will RESET *NONE* of the BCR locations to default values. S_RESET will cause the microcode program to jump to its RESET state. Following the end of the S_RESET operation, the PCnet-32 controller will NOT attempt to read the EEPROM device. See also the subsection on RESET Register in the I/O Register Access section under Software Access. The S_RESET operation will not cause INTR pins to become inactive. S_RESET will set the T-MAU into Link Fail state.

Note that S_RESET will **not** cause a de-assertion of the HOLD signal, if it happens to be active at the time of the read to the reset register. The HOLD signal will remain active until the HLDA signal is synchronously sampled as asserted. Following the read of the RESET register, on the next clock cycle after the HLDA signal is synchronously sampled as asserted, (except in Am386 mode, when two cycles are needed) the PCnet-32 controller will de-assert the HOLD signal; No bus master accesses will have been performed during this brief bus ownership period.

STOP

STOP is a PCnet-32 RESET operation that has been created by the ASSERTION of the STOP bit in CSR0. That is, a STOP RESET is generated by writing a ONE to the STOP bit of CSR0 *when the STOP bit currently has a value of ZERO.* If the STOP bit value is currently a ONE, and a ONE is rewritten to the STOP bit, then NO STOP RESET will be generated. The STOP operation will not cause INTR pins to become inactive.

STOP will RESET all or some portions of CSR0, 3, and 4 to default values; STOP will RESET *NONE* of the BCR locations to default values. STOP will cause the microcode program to jump to its RESET state. Following the end of the STOP operation, the PCnet-32 controller will not attempt to read the EEPROM device. For the identity of individual CSRs and bit locations that are affected by STOP, see the individual CSR register descriptions. Setting the STOP bit does not affect the T-MAU.

USER ACCESSIBLE REGISTERS

The PCnet-32 controller implements all PCnet-ISA (Am79C960) registers, all LANCE (Am7990) registers, all ILACC (Am79C900) registers, plus a number of additional registers. The PCnet-32 controller registers are compatible with both the PCnet-ISA (Am79C960) registers and all of the LANCE (Am7990) registers upon power up. Compatibility to the ILACC set of registers requires one access to the Software Style register (BCR20, bits 7–0) to be performed. By setting an appropriate value of the Software Style register (BCR20, bits 7–0) the user can select a set of registers that are compatible with the ILACC set of registers.

Note that all register locations are defined to be 16 bits in width when WIO mode is selected. When DWIO mode is selected, all register locations are defined to be 32 bits in width. When performing register write operations in DWIO mode, the upper 16 bits should always be written as zeros, except APROM locations. When performing register read operations in DWIO mode, the upper 16 bits of I/O resources should always be written as ZEROs, except for APROM locations and CSR88. When performing register read operations in DWIO mode, the upper 16 bits of I/O resources should always be regarded as having undefined values, except for the APROM locations and CSR88.

PCnet-32 controller registers can be divided into three groups:

Setup registers: Registers that are intended to be initialized by the system initialization procedure (e.g. BIOS device initialization routine) or by the device driver to program the operation of various PCnet-32 controller features

Running registers: Registers that are intended to be used by the device driver software once the PCnet-32 controller is running to access status information and to pass control information

Test registers: Registers that are intended to be used only for testing and diagnostic purposes

Below is a list of the registers that fall into each of the first two categories. Those registers that are not included in either of these lists can be assumed to be intended for diagnostic purposes.

Setup Registers

The following is a list of those registers that would typically need to be programmed once during the setup of the PCnet-32 controller within a system. The control bits in each of these registers typically do not need to be modified once they have been written. However, there are no restrictions as to how many times these registers may actually be accessed. Note that if the default power up values of any of these registers is acceptable to the application, then such registers need never be accessed at all. Also note that some of these registers may be programmable through the EEPROM read operation, and therefore do not necessarily need to be written to by the system initialization procedure or by the driver software.

CRS1	Initialization Address[15:0]
CSR2	Initialization Address[31:16]
CSR3	Interrupt Masks and Deferral Control
CSR4	Test and Features Control
CSR8	Logical Address Filter [15:0]
CSR9	Logical Address Filter [31:16]
CSR10	Logical Address Filter [47:32]
CSR11	Logical Address Filter [63:48]
CSR12	Physical Address Filter [15:0]
CSR13	Physical Address Filter [31:16]
CSR14	Physical Address Filter [47:32]
CSR15	Mode Register
CSR24	Base Address of Receive Ring Lower
CSR25	Base Address of Receive Ring Upper
CSR30	Base Address of Transmit Ring Lower
CRS31	Base Address of Transmit Ring Upper
CSR47	Polling Interval
CSR58	Software Style
CSR76	Receive Ring Length
CSR78	Transmit Ring Length
CSR80	Cycle Register and FIFO Threshold Control
CSR82	Bus Activity Timer
CSR100	Memory Error Time-out Register
CSR122	Receiver Packet Alignment Control
BCR2	MAU configuration
BCR16	I/O Base Address Lower
BCR17	I/O Base Address Upper
BCR18	Bus Size and Burst Control Register
BCR19	EEPROM Control and Status Register
BCR20	Software Style
BCR21	Interrupt Control

Running Registers

The following is a list of those registers that would typically need to be periodically read and perhaps written during the normal running operation of the PCnet-32

controller within a system. Each of these registers contains control bits or status bits or both.

RAP	Register Address Port Register
CSR0	PCnet-32 Controller Status Register
CSR4	Test and Features Control
CSR112	Missed Frame Count
CSR114	Receive Collision Count

RAP Register

The RAP (Register Address Pointer) register is used to gain access to CSR and BCR registers on board the PCnet-32 controller. The value of the RAP indicates the address of a CSR or BCR whenever an RDP or BDP access is performed. That is to say, RAP serves as a pointer to CSR and BDP space.

As an example of RAP use, consider a read access to CSR4. In order to access this register, it is necessary to first load the value 0004 into the RAP by performing a write access to the RAP offset of 12h (12h when WIO mode has been selected, 14h when DWIO mode has been selected). The data for the RAP write would be 0004. Then a second access is performed on the PCnet-32 controller, this time to the RDP offset of 10h (for either WIO or DWIO mode). The RDP access is a read access, and since RAP has just been loaded with the value of 0004, the RDP read will yield the contents of CSR4. A read of the BDP at this time (offset of 16h when WIO mode has been selected, 1Ch when DWIO mode has been selected) will yield the contents of BCR4, since the RAP is used as the pointer into both BDP and RDP space.

RAP: Register Address Port

Bit	Name	Description
31-16	RES	Reserved locations. Written as zeros and read as undefined.
15-8	RES	Reserved locations. Read and written as zeros.
7-0	RAP	Register Address Port. The value of these 8 bits determines which CSR or BCR will be accessed when an I/O access to the RDP or BDP port, respectively, is performed. RAP is cleared by H_RESET or S_RESET and is unaffected by the STOP bit.

Control and Status Registers

The CSR space is accessible by performing accesses to the RDP (Register Data Port). The particular CSR that is read or written during an RDP access will depend upon the current setting of the RAP. RAP serves as a pointer into the CSR space. RAP also serves as the pointer to BCR space, which is described in a later section.

CSR0: PCnet-32 Controller Status

Bit	Name	Description
31-16	RES	Reserved locations. Written as zeros and read as undefined.
15	ERR	Error is set by the ORing of BABL, CERR, MISS, and MERR. ERR remains set as long as any of the error flags are true. ERR is read only. Write operations are ignored.
14	BABL	Babble is a transmitter time-out error. It indicates that the transmitter has been on the channel longer than the time required to send the maximum length frame. BABL will be set if 1519 bytes or greater are transmitted. When BABL is set, INTR is asserted if IENA = 1 and the mask bit BABLM in CSR3 is clear. BABL assertion will set the ERR bit. BABL is set by the MAC layer and cleared by writing a “1”. Writing a “0” has no effect. BABL is cleared by H_RESET or S_RESET or setting the STOP bit.
13	CERR	Collision Error indicates that the collision inputs to the AUI port failed to activate within 20 network bit times after chip terminated transmission (SQE Test). This feature is a transceiver test feature. In 10BASE-T mode CERR will be set if a transmission is attempted while the T-MAU is in Link Fail state. CERR assertion will not result in an interrupt being generated. CERR assertion will set the ERR bit. CERR is set by the MAC layer and cleared by writing a “1”. Writing a “0” has no effect. CERR is cleared by H_RESET or S_RESET or setting the STOP bit.
12	MISS	Missed Frame is set when PCnet-32 controller has lost an incoming receive frame resulting from a Receive Descriptor not being available. This bit is the only immediate indication that receive data has been lost since there is no

current receive descriptor to write status to.

When MISS is set, INTR is asserted if IENA = 1 and the mask bit MISSM in CSR3 is clear. MISS assertion will set the ERR bit.

MISS is set by the Buffer Management Unit and cleared by writing a “1”. Writing a “0” has no effect. MISS is cleared by H_RESET or S_RESET or setting the STOP bit.

11 MERR Memory Error is set when PCnet-32 controller requests the use of the system interface bus by asserting HOLD and has not received HLDA assertion after a programmable length of time.The length of time in microseconds before MERR is asserted will depend upon the setting of the Bus Time-Out Register (CSR100). The default setting of CSR100 will give a MERR after 51.2 µs of bus latency.

When MERR is set, INTR is asserted if IENA = 1 **and** the mask bit MERRM in CSR3 is clear. MERR assertion will set the ERR bit, regardless of the settings of IENA and MERRM.

MERR is set by the Bus Interface Unit and cleared by writing a “1”. Writing a “0” has no effect. MERR is cleared by H_RESET or S_RESET or by setting the STOP bit.

10 RINT Receive interrupt. RINT is set by the Buffer Management Unit of the PCnet-32 controller after the last descriptor of a receive packet has been updated by writing a ZERO to the ownership bit. RINT may *also* be set when the first descriptor of a receive packet has been updated by writing a ZERO to the ownership bit *if* the SPRINTEN bit of CSR3 has been set to a ONE.

When RINT is set, INTR is asserted if IENA = 1 and the mask bit RINTM in CSR3 is clear.

RINT is cleared by the host by writing a “1”. Writing a “0” has no effect. RINT is cleared by H_RESET or S_RESET or by setting the STOP bit.

9 TINT Transmit interrupt is set after completion of a transmit frame and toggling of the OWN bit in the last buffer in the Transmit Descriptor Ring.

When TINT is set, INTR is asserted if IENA = 1 and the mask bit TINTM in CSR3 is clear.

TINT is set by the Buffer Management Unit after the last transmit buffer has been updated and cleared by writing a “1”. Writing a “0” has no effect. TINT is cleared by H_RESET or S_RESET or setting the STOP bit.

8 IDON Initialization Done indicates that the initialization sequence has completed. When IDON is set, PCnet-32 controller has read the Initialization block from memory.

When IDON is set, INTR is asserted if IENA = 1 and the mask bit IDONM in CSR3 is clear.

IDON is set by the Buffer Management Unit after the initialization block has been read from memory and cleared by writing a “1”. Writing a “0” has no effect. IDON is cleared by H_RESET or S_RESET or setting the STOP bit.

7 INTR Interrupt Flag indicates that one or more following interrupt causing conditions has occurred: BABL, MISS, MERR, MFCO, RCVCCO, RINT, RPCO, TINT, IDON, JAB or TXSTRT. and its associated mask bit is clear. If IENA = 1 and INTR is set, INTR will be active.

INTR is read only. INTR is cleared by H_RESET or S_RESET or by setting the STOP bit or by clearing all of the active individual interrupt bits that have not been masked out.

6 IENA Interrupt Enable allows INTR to be active if the interrupt Flag is set. If IENA = “0” then INTR will be disabled regardless of the state of INTR.

IENA is set by writing a “1” and cleared by writing a “0”. IENA is cleared by H_RESET or S_RESET or setting the STOP bit.

5 RXON Receive On indicates that the Receive function is enabled. RXON is set if DRX (CSR15[0]) = “0” after the START bit is set. If INIT and START are set together, RXON will not be set until after the initialization block has been read in.

RXON is read only. RXON is cleared by H_RESET or S_RESET or setting the STOP bit.

4 TXON Transmit On indicates that the Transmit function is enabled. TXON is set if DTX (CSR15[1]) = “0” after the START bit is set. If INIT and START are set together, TXON will not be set until after the initialization block has been read in. TXON is read only. TXON is cleared by H_RESET or S_RESET or setting the STOP bit.

3 TDMD Transmit Demand, when set, causes the Buffer Management Unit to access the Transmit Descriptor Ring without waiting for the poll-time counter to elapse. If TXON is not enabled, TDMD bit will be reset and no Transmit Descriptor Ring access will occur.

TDMD is required to be set if the DPOLL bit in CSR4 is set. Setting TDMD while DPOLL = 0 merely hastens the PCnet-32 controller’s response to a Transmit Descriptor Ring Entry.

TDMD is set by writing a “1”. Writing a “0” has no effect. TDMD will be cleared by the Buffer Management Unit when it fetches a Transmit Descriptor. TDMD is cleared by H_RESET or S_RESET or setting the STOP bit.

2 STOP STOP assertion disables the chip from all DMA activity. The chip remains inactive until either STRT or INIT are set. If STOP, STRT and INIT are all set together, STOP will override STRT and INIT.

STOP is set by writing a “1” or by H_RESET or S_RESET. Writing a “0” has no effect. STOP is cleared by setting either STRT or INIT.

1 STRT STRT assertion enables PCnet-32 controller to send and receive frames, and perform buffer management operations. Setting STRT clears the STOP bit. If STRT and INIT are set together, PCnet-32 controller initialization will be performed first.

STRT is set by writing a “1”. Writing a “0” has no effect. STRT is cleared by H_RESET or S_RESET or by setting the STOP bit.

0 INIT INIT assertion enables PCnet-32 controller to begin the initialization procedure which reads in the initialization block from memory. Setting INIT clears the STOP bit. If STRT and INIT are set together, PCnet-32 controller initialization will be performed first.

INIT is not cleared when the initialization sequence has completed.

INIT is set by writing a “1”. Writing a “0” has no effect. INIT is cleared by H_RESET or S_RESET or by setting the STOP bit.

CSR1: IADR[15:0]

Bit	Name	Description
31-16	RES	Reserved locations. Written as zeros and read as undefined.
15-0	IADR[15:0]	Lower 16 bits of the address of the Initialization Block. Regardless of the value of SSIZE32(BCR20/CSR58, bit 8) IADR[1:0] must be zero. This register is aliased with CSR16. Read/Write accessible only when the STOP bit in CSR0 is set. Unaffected by H_RESET or S_RESET or by setting the STOP bit.

CSR2: IADR[31:16]

Bit	Name	Description
31-16	RES	Reserved locations. Written as zeros and read as undefined.
15-8	IADR[31:24]	If SSIZE32 is set (BCR20[8]), then the IADR[31:24] bits will be used strictly as the upper 8 bits of the initialization block address. However, if SSIZE32 is reset, then the IADR[31:24] bits will be used to generate the upper 8 bits of all bus

		mastering addresses, as required for a 32 bit address bus. Note that the 16-bit software structures specified by the SSIZE32 = 0 setting will yield only 24 bits of address for PCnet-32 controller bus master accesses, while the 32-bit hardware for which the PCnet-32 controller is intended will require 32 bits of address. Therefore, whenever SSIZE32 = 0, the IADR[31:24] bits will be appended to the 24-bit initialization address, to each 24-bit descriptor base address and to each beginning 24-bit buffer address in order to form complete 32-bit addresses. The upper 8 bits that exist in the descriptor address registers and the buffer address registers which are stored on board the PCnet-32 controller will be overwritten with the IADR[31:24] value, so that CSR accesses to these registers will show the 32 bit address that includes the appended field. If SSIZE32 = 1, then software will provide 32-bit pointer values for all of the shared software structures (i.e. descriptor bases and buffer addresses), and therefore, IADR[31:24] will not be written to the upper 8 bits of any of these resources, but it will be used as the upper 8 bits of the initialization address. Read/Write accessible only when the STOP bit in CSR0 is set. Unaffected by H_RESET or S_RESET or by setting the STOP bit.
7-0	IADR[23:16]	Bits 23 through 16 of the address of the Initialization Block. Whenever this register is written, CSR17 is updated with CSR2's contents. Read/Write accessible only when the STOP bit in CSR0 is set. Unaffected by H_RESET or S_RESET or by setting the STOP bit.

CSR3: Interrupt Masks and Deferral Control

Bit	Name	Description
31-16	RES	Reserved locations. Written as zeros and read as undefined.
15	RES	Reserved location. Read and written as zero.
14	BABLM	Babble Mask. If BABLM is set, the BABL bit in CSR0 will be masked and unable to set INTR flag in CSR0. Read/Write accessible always. BABLM is cleared by H_RESET or S_RESET and is not affected by STOP.
13	RES	Reserved location. Read and written as zero.
12	MISSM	Missed Frame Mask. If MISSM is set, the MISS bit in CSR0 will be masked and unable to set INTR flag in CSR0. Read/Write accessible always. MISSM is cleared by H_RESET or S_RESET and is not affected by STOP.
11	MERRM	Memory Error Mask. If MERRM is set, the MERR bit in CSR0 will be masked and unable to set INTR flag in CSR0. Read/Write accessible always. MERRM is cleared by H_RESET or S_RESET and is not affected by STOP.
10	RINTM	Receive Interrupt Mask. If RINTM is set, the RINT bit in CSR0 will be masked and unable to set INTR flag in CSR0. Read/Write accessible always. RINTM is cleared by H_RESET or S_RESET and is not affected by STOP.
9	TINTM	Transmit interrupt Mask. If TINTM is set, the TINT bit in CSR0 will be masked and unable to set INTR flag in CSR0. Read/Write accessible always. TINTM is cleared by H_RESET or S_RESET and is not affected by STOP.
8	IDONM	Initialization Done Mask. If IDONM is set, the IDON bit in CSR0 will be

masked and unable to set INTR flag in CSR0.

Read/Write accessible always. IDONM is cleared by H_RESET or S_RESET and is not affected by STOP.

7 RES Reserved location. Read and written as zeroes.

6 DXSUFLO Disable Transmit Stop on Underflow error.

When DXSUFLO (CSR3, bit 6) is set to ZERO, the transmitter is turned off when an UFLO error occurs (CSR0, TXON = 0).

When DXSUFLO is set to ONE, the PCnet-32 controller gracefully recovers from an UFLO error. It scans the transmit descriptor ring until it finds the start of a new frame and starts a new transmission.

Read/Write accessible always. DXSUFLO is cleared by H_RESET or S_RESET and is not affected by STOP.

5 LAPPEN Look-Ahead Packet Processing Enable. When set to a ONE, the LAPPEN bit will cause the PCnet-32 controller to generate an interrupt following the descriptor write operation to the *first* buffer of a receive frame. This interrupt will be generated *in addition* to the interrupt that is generated following the descriptor write operation to the last buffer of a receive frame. The interrupt will be signaled through the RINT bit of CSR0.

Setting LAPPEN to a ONE also enables the PCnet-32 controller to read the STP bit of receive descriptors. The PCnet-32 controller will use the STP information to determine where it should begin writing a receive frame's data. Note that while in this mode, the PCnet-32 controller can write intermediate frame data to buffers whose descriptors do not contain STP bits set to ONE. Following the write to the last descriptor used by a frame, the PCnet-32 controller will scan through the next descriptor entries to locate the next STP bit that is set to a ONE. The PCnet-32 controller will begin writing the next frame's data to the buffer pointed to by that descriptor.

Note that because several descriptors may be allocated by the host for each frame, and not all messages may need all of the descriptors that are allocated between descriptors that contain STP = ONE, then some descriptors/buffers may be skipped in the ring. While performing the search for the next STP bit that is set to ONE, the PCnet-32 controller will advance through the receive descriptor ring regardless of the state of ownership bits. If any of the entries that are examined during this search indicate PCnet-32 controller ownership of the descriptor but also indicate STP = "0", then the PCnet-32 controller will RESET the OWN bit to ZERO in these entries. If a scanned entry indicates host ownership with STP = "0" then the PCnet-32 controller will not alter the entry, but will advance to the next entry.

When the STP bit is found to be true, but the descriptor that contains this setting is not owned by the PCnet-32 controller, then the PCnet-32 controller will stop advancing through the ring entries and begin periodic polling of this entry. When the STP bit is found to be true, and the descriptor that contains this setting is owned by the PCnet-32 controller, then the PCnet-32 controller will stop advancing through the ring entries, store the descriptor information that it has just read, and wait for the next receive to arrive.

This behavior allows the host software to preassign buffer space in such a manner that the "header" portion of a receive frame will always be written to a particular memory area, and the "data" portion of a receive frame will always be written to a separate memory area. The interrupt is generated when the "header" bytes have been written to the "header" memory area.

Read/Write accessible always. The LAPPEN bit will be reset to ZERO by H_RESET or S_RESET and will be unaffected by STOP.

See Appendix D for more information on the LAPP concept.

4 DXMT2PD Disable Transmit Two Part Deferral. If DXMT2PD is set, Transmit Two Part Deferral will be disabled.

Read/Write accessible always. DXMT2PD is cleared by H_RESET or S_RESET and is not affected by STOP.

3 EMBA Enable Modified Back-off Algorithm. If EMBA is set, a modified back-off algorithm is implemented.

Read/Write accessible always. EMBA is cleared by H_RESET or S_RESET and is not affected by STOP.

2 BSWP Byte Swap. This bit is used to choose between big and little Endian modes of operation. When BSWP is set to a ONE, big Endian mode is selected. When BSWP is set to ZERO, little Endian mode is selected.

When big Endian mode is selected, the PCnet-32 controller will swap the order of bytes on the data bus during FIFO transfers only. Specifically, D31–24 becomes Byte0, D23–16 becomes Byte1, D15–8 becomes Byte2 and D7–0 becomes Byte3 when big Endian mode is selected. When little Endian mode is selected, the order of bytes on the data bus is: D31–24 is Byte3, D23–16 is Byte2, D15–8 is Byte1 and D7–0 is Byte0.

Byte swap only affects data transfers that involve the FIFOs. Initialization block transfers are not affected by the setting of the BSWP bit. Descriptor transfers are not affected by the setting of the BSWP bit. RDP, RAP and BDP accesses are not affected by the setting of the BSWP bit. APROM transfers are not affected by the setting of the BSWP bit.

BSWP is write/readable regardless of the state of the STOP bit. BSWP is cleared by H_RESET or S_RESET and is not affected by STOP.

1-0 RES Reserved locations. The default value is ZERO for both locations. Writing a ONE to these bits has no effect on device function; if a ONE is written to these bits, then a ONE will be read back. Existing drivers may write a ONE to these bits for compatibility, but new drivers should write a ZERO to these bits and should treat the read value as undefined.

CSR4: Test and Features Control

Bit	Name	Description
31-16	RES	Reserved locations. Written as zeros and read as undefined.
15	ENTST	Enable Test Mode operation. Setting ENTST to ONE enables internal test functions which are useful only for stand alone integrated circuit testing. In addition, the Runt Packet Accept (RPA) bit (CSR124, bit 3) may be changed only when ENTST is set to ONE. To enable RPA, the user must first write a ONE to the ENTST bit. Next, the user must first write a ONE to the RPA bit (CSR124, bit 3). Finally, the user must write a ZERO to the ENTST bit to take the device out of test mode operation. Once the RPA bit has been set to ONE, the device will remain in the Runt Packet Accept mode until the RPA bit is cleared to ZERO. Read/Write accessible. ENTST is cleared by H_RESET or S_RESET and is unaffected by the STOP bit.
14	DMAPLUS	When DMAPLUS = “1”, disables the burst transaction counter, CSR80. If DMAPLUS = “0”, the burst transaction counter is enabled. Read and Write accessible. DMAPLUS is cleared by H_RESET or S_RESET and is unaffected by the STOP bit.
13	TIMER	Timer Enable Register. If TIMER is set, the Bus Activity Timer Register, CSR82 is enabled. If TIMER is cleared, the Bus Activity Timer Register is disabled.

Read/Write accessible. TIMER is cleared by H_RESET or S_RESET and is unaffected by the STOP bit.

12 DPOLL Disable Transmit Polling. If DPOLL is set, the Buffer Management Unit will disable transmit polling. Likewise, if DPOLL is cleared, automatic transmit polling is enabled. If DPOLL is set, TDMD bit in CSR0 must be periodically set in order to initiate a manual poll of a transmit descriptor. Transmit descriptor polling will not take place if TXON is reset.

Read/Write accessible. DPOLL is cleared by H_RESET or S_RESET and is unaffected by the STOP bit.

11 APAD_XMT Auto Pad Transmit. When set, APAD_XMT enables the automatic padding feature. Transmit frames will be padded to extend them to 64 bytes including FCS. The FCS is calculated for the entire frame including pad, and appended after the pad field. APAD_XMT will override the programming of the DXMTFCS bit.

Read and Write accessible. APAD_XMT is reset by H_RESET or S_RESET and is unaffected by the STOP bit.

10 ASTRP_RCV Auto Strip Receive. When set, ASTRP_RCV enables the automatic pad stripping feature. The pad and FCS fields will be stripped from receive frames and not placed in the FIFO.

Read and Write accessible. ASTRP_RCV is reset by H_RESET or S_RESET and is unaffected by the STOP bit.

9 MFCO Missed Frame Counter Overflow interrupt. Indicates the MPC (CSR112) wrapped around. Can be cleared by writing a 1 to this bit. Also cleared by H_RESET or S_RESET or setting the STOP bit. Writing a 0 has no effect.

When MFCO is set, INTR is asserted if IENA = 1 and the mask bit MFCOM is cleared.

When the SWSTYLE register (BCR20[7:0]) has been programmed to the ILACC compatibility mode, then this bit has no meaning and the PCnet-32 controller will never set the value of this bit to ONE.

8 MFCOM Missed Frame Counter Overflow Mask. If MFCOM is set, MFCO will be unable to set INTR in CSR0. Set to a ONE by H_RESET or S_RESET, unaffected by the STOP bit.

When the SWSTYLE register (BCR20[7:0]) has been programmed to the ILACC compatibility mode, then this bit has no meaning and the PCnet-32 controller will set the value of this bit to a ZERO.

7 RES Reserved location. Written as zero and read as zero.

6 RES Reserved location. This bit may be written to as either a ONE or a ZERO, but will always be read as a ZERO. This bit has no effect on PCnet-32 controller operation.

5 RCVCCO Receive Collision Counter Overflow. Indicates the Receive Collision Counter (CSR114) wrapped around. Can be cleared by writing a 1 to this bit. Also cleared by H_RESET or S_RESET or by setting the STOP bit. Writing a 0 has no effect.

When RCVCCO is set, INTR is asserted if IENA=1 and the mask bit RCVCCOM is cleared.

When the SWSTYLE register (BCR20[7:0]) has been programmed to the ILACC compatibility mode, then this bit has no meaning and PCnet-32 controller will not set the value of this bit to ONE.

4 RCVCCOM Receive Collision Counter Over-flow Mask. If RCVCCOM is set, RCVCCO will be unable to set INTR in CSR0. RCVCCOM is set to a ONE by H_RESET or S_RESET and is not affected by STOP.

When the SWSTYLE register (BCR20[7:0]) has been programmed to the ILACC compatibility mode, then this bit has no meaning and PCnet-32 controller

will set the value of this bit to a ZERO.

3 TXSTRT Transmit Start status is set whenever PCnet-32 controller begins transmission of a frame. When TXSTRT is set, INTR is asserted if IENA = 1 and the mask bit TXSTRTM (CSR4 bit 2) is cleared.

TXSTRT is set by the MAC Unit and cleared by writing a "1", by H_RESET or S_RESET, or setting the STOP bit. Writing a "0" has no effect.

2 TXSTRTM Transmit Start Mask. If TXSTRTM is set, the TXSTRT bit in CSR4 will be masked and unable to set INTR flag in CSR0.

Read/Write accessible. TXSTRTM is set to a ONE by H_RESET or S_RESET and is not affected by the STOP bit.

1 JAB Jabber Error is set when the PCnet-32 controller Twisted-pair MAU function exceeds an allowed transmission limit. Jabber is set by the T-MAU cell and can only be asserted in 10BASE-T mode.

When JAB is set, INTR is asserted if IENA = 1 and the mask bit JABM (CSR4[0]) is cleared.

JAB is set by the T-MAU circuit and cleared by writing a "1". Writing a "0" has no effect. JAB is also cleared by H_RESET or S_RESET or setting the STOP bit.

When the SWSTYLE register (BCR20[7:0]) has been programmed to the ILACC compatibility mode, then this bit has no meaning and PCnet-32 controller will never set the value of this bit to ONE.

0 JABM Jabber Error Mask. If JABM is set, the JAB bit in CSR4 will be masked and unable to set INTR flag in CSR0.

Read/Write accessible. JABM is set to a ONE by H_RESET or S_RESET and is not affected by STOP.

When the SWSTYLE register (BCR20[7:0]) has been programmed to the ILACC compatibility mode, then this bit has no meaning and PCnet-32 controller will set the value of this bit to a ZERO.

CSR6: RX/TX Descriptor Table Length

Bit	Name	Description
31-16	RES	Reserved locations. Written as zeros and read as undefined.
15-12	TLEN	Contains a copy of the transmit encoded ring length (TLEN) field read from the initialization block during PCnet-32 controller initialization. This field is written during the PCnet-32 controller initialization routine. Read accessible only when STOP bit is set. Write operations have no effect and should not be performed. TLEN is only defined after initialization. These bits are unaffected by H_RESET, S_RESET, or STOP.
11-8	RLEN	Contains a copy of the receive encoded ring length (RLEN) read from the initialization block during PCnet-32 controller initialization. This field is written during the PCnet-32 controller initialization routine. Read accessible only when STOP bit is set. Write operations have no effect and should not be performed. RLEN is only defined after initialization. These bits are unaffected by H_RESET, S_RESET, or STOP.
7-0	RES	Reserved locations. Read as zero. Write operations should not be performed.

CSR8: Logical Address Filter, LADRF[15:0]

Bit	Name	Description
31-16	RES	Reserved locations. Written as zeros and read as undefined.
15-0	LADRF[15:0]	Logical Address Filter, LADRF[15:0]. Defined only after the initialization block has been successfully read or a direct I/O write has been performed on this register.

Read/write accessible only when STOP bit is set. These bits are unaffected by H_RESET, S_RESET, or STOP.

CSR9: Logical Address Filter, LADRF[31:16]

Bit	Name	Description
31-16	RES	Reserved locations. Written as zeros and read as undefined.
15-0	LADRF[31:16]	Logical Address Filter, LADRF[31:16]. Defined only after the initialization block has been successfully read or a direct I/O write has been performed on this register. Read/write accessible only when STOP bit is set. These bits are unaffected by H_RESET, S_RESET, or STOP.

CSR10: Logical Address Filter, LADRF[47:32]

Bit	Name	Description
31-16	RES	Reserved locations. Written as zeros and read as undefined.
15-0	LADRF[47:32]	Logical Address Filter, LADRF[47:32]. Defined only after the initialization block has been successfully read or a direct I/O write has been performed on this register. Read/write accessible only when STOP bit is set. These bits are unaffected by H_RESET, S_RESET, or STOP.

CSR11: Logical Address Filter, LADRF[63:48]

Bit	Name	Description
31-16	RES	Reserved locations. Written as zeros and read as undefined.
15-0	LADRF[63:48]	Logical Address Filter, LADRF[63:48]. Defined only after the initialization block has been successfully read or a direct I/O write has been performed on this register. Read/write accessible only when STOP bit is set. These bits are unaffected by H_RESET, S_RESET, or STOP.

CSR12: Physical Address Register, PADR[15:0]

Bit	Name	Description
31-16	RES	Reserved locations. Written as zeros and read as undefined.
15-0	PADR[15:0]	Physical Address Register, PADR[15:0]. Defined only after the initialization block has been successfully read or a direct I/O write has been performed on this register. The PADR bits are transmitted PADR[0] first and PADR[47] last. Read/write accessible only when STOP bit is set. These bits are unaffected by H_RESET, S_RESET, or STOP.

CSR13: Physical Address Register, PADR[31:16]

Bit	Name	Description
31-16	RES	Reserved locations. Written as zeros and read as undefined.
15-0	PADR[31:16]	Physical Address Register, PADR[31:16]. Defined only after the initialization block has been successfully read or a direct I/O write has been performed on this register. The PADR bits are transmitted PADR[0] first and PADR[47] last. Read/write accessible only when STOP bit is set. These bits are unaffected by H_RESET, S_RESET, or STOP.

CSR14: Physical Address Register, PADR[47:32]

Bit	Name	Description
31-16	RES	Reserved locations. Written as zeros and read as undefined.
15-0	PADR[47:32]	Physical Address Register, PADR[47:32]. Defined only after the initialization block has been successfully read or a direct I/O write has been performed on this register. The PADR bits are transmitted PADR[0] first and PADR[47] last. Read/write accessible only when STOP bit is set. These bits are unaffected by H_RESET, S_RESET, or STOP.

CSR15: Mode Register

Bit	Name	Description
		This register's fields are loaded during the PCnet-32 controller initialization routine with the corresponding Initialization Block values or a direct I/O write has been performed to this register.
31-16	RES	Reserved locations. Written as zeros and read as undefined.
15	PROM	Promiscuous Mode. When PROM = "1", all incoming receive frames are accepted. Read/write accessible only when STOP bit is set.
14	DRCVBC	Disable Receive Broadcast. When set, disables the PCnet-32 controller from receiving broadcast messages. Used for protocols that do not support broadcast addressing, except as a function of multicast. DRCVBC is cleared by H_RESET or S_RESET (broadcast messages will be received) and is unaffected by STOP. Read/write accessible only when STOP bit is set.
13	DRCVPA	Disable Receive Physical Address. When set, the physical address detection (Station or node ID) of the PCnet-32 controller will be disabled. Frames addressed to the nodes individual physical address will not be recognized (although the frame may be accepted by the EADI mechanism). Read/write accessible only when STOP bit is set.
12	DLNKTST	Disable Link Status. When DLNKTST = "1", monitoring of Link Pulses is disabled. When DLNKTST = "0", monitoring of Link Pulses is enabled. This bit only has meaning when the 10BASE-T network interface is selected. Read/write accessible only when STOP bit is set.
11	DAPC	Disable Automatic Polarity Correction. When DAPC = "1", the 10BASE-T receive polarity reversal algorithm is disabled. Likewise, when DAPC = "0", the polarity reversal algorithm is enabled. This bit only has meaning when the 10BASE-T network interface is selected. Read/write accessible only when STOP bit is set.
10	MENDECL	MENDEC Loopback Mode. See the description of the LOOP bit in CSR15. Read/write accessible only when STOP bit is set.
9	LRT/TSEL	Low Receive Threshold (T-MAU Mode only) Transmit Mode Select (AUI Mode only)
	LRT	Low Receive Threshold. When LRT = "1", the internal twisted pair receive thresholds are reduced by 4.5 dB below the standard 10BASE-T value (approximately 3/5) and the unsquelch threshold for the RXD circuit will be 180–312 mV peak. When LRT = "0", the unsquelch threshold for the RXD circuit will be the standard 10BASE-T value, 300 – 520 mV peak. In either case, the RXD circuit post squelch threshold will be one half of the unsquelch threshold. This bit only has meaning when the 10BASE-T network interface is selected. Read/write accessible only when STOP bit is set. Cleared by H_RESET or S_RESET and is unaffected by STOP.
	TSEL	TSEL Transmit Mode Select. TSEL controls the levels at which the AUI drivers rest when the AUI transmit port is idle. When TSEL = 0, DO+ and DO- yield "zero" differential to operate transformer coupled loads (Ethernet 2 and 802.3). When TSEL = 1, the DO+ idles at a higher value with respect to DO-, yielding a logical HIGH state (Ethernet 1). This bit only has meaning when the AUI network interface is selected.

Read/write accessible only when STOP bit is set. Cleared by H_RESET or S_RESET and is unaffected by STOP.

8-7 PORTSEL[1:0] Port Select bits allow for software controlled selection of the network medium.

PORTSEL settings of AUI and 10BASE-T are ignored when the ASEL bit of BCR2 (bit 1) has been set to ONE.

The network port configuration is shown in Table 41.

Table 41. Network Port Configuration.

PORTSEL[1:0]	ASEL (BCR2[1])	Link Status (of 10BASE-T)	Network Port
0X	1	Fail	AUI
0X	1	Pass	10BASE-T
0 0	0	X	AUI
0 1	0	X	10BASE-T
1 0	X	X	GPSI
1 1	X	X	Reserved

Refer to the section on General Purpose Serial Interface for detailed information on accessing GPSI.

Read/write accessible only when STOP bit is set. Cleared by H_RESET or S_RESET and is unaffected by STOP.

6 INTL Internal Loopback. See the description of LOOP, CSR15-2.

Read/write accessible only when STOP bit is set.

5 DRTY Disable Retry. When DRTY = “1”, PCnet-32 controller will attempt only one transmission. If DRTY = “0”, PCnet-32 controller will attempt 16 retry attempts before signaling a retry error. DRTY is defined when the initialization block is read.

Read/write accessible only when STOP bit is set.

4 FCOLL Force Collision. This bit allows the collision logic to be tested. PCnet-32 controller must be in internal loopback for FCOLL to be valid. If FCOLL = “1”, a collision will be forced during loop-back transmission attempts. A Retry Error will ultimately result. If FCOLL = “0”, the Force Collision logic will be disabled.

Read/write accessible only when STOP bit is set.

3 DXMTFCS Disable Transmit CRC (FCS). When DXMTFCS = 0, the transmitter will generate and append a FCS to the transmitted frame. When DXMTFCS = 1, the FCS logic is allocated to the receiver and no FCS is generated or sent with the transmitted frame. DXMTFCS is overridden when ADD_FCS is set in TMD1.

See also the ADD_FCS bit in TMD1. If DXMTFCS is set and ADD_FCS is clear for a particular frame, no FCS will be generated. The value of ADD_FCS is valid only when STP is set. If ADD_FCS is set for a particular frame, the state of DXMTFCS is ignored and a FCS will be appended on that frame by the transmit circuitry.

In loopback mode, this bit determines if the transmitter appends FCS or if the receiver checks the FCS.

This bit was called DTCR in the LANCE (Am7990).

Read/write accessible only when STOP bit is set.

2 LOOP Loopback Enable allows PCnet-32 controller to operate in full duplex mode for test purposes. When LOOP = “1”, loop-back is enabled. In combination with INTL and MENDECL, various loopback modes are defined in Table 42.

Table 42. Loopback Modes.

LOOP	INTL	MENDECL	Loopback Mode
0	X	X	Non-Loopback
1	0	X	External Loopback
1	1	0	Internal Loopback Include MENDEC
1	1	1	Internal Loopback Exclude MENDEC

Read/write accessible only when STOP bit is set. LOOP is cleared by

H_RESET or S_RESET and is unaffected by the STOP bit.

1	DTX	Disable Transmit results in PCnet-32 controller not accessing the Transmit Descriptor Ring and therefore no transmissions are attempted. DTX = “0”, will set TXON bit (CSR0 bit 4) if STRT (CSR0 bit 1) is asserted. Read/write accessible only when STOP bit is set.
0	DRX	Disable Receiver results in PCnet-32 controller not accessing the Receive Descriptor Ring and therefore all receive frame data are ignored. DRX = “0”, will set RXON bit (CSR0 bit 5) if STRT (CSR0 bit 1) is asserted. Read/write accessible only when STOP bit is set.

CSR16: Initialization Block Address Lower

Bit	Name	Description
		This register is an alias of the location CSR1. Accesses to/from this register are equivalent to access to CSR1.
31-16	RES	Reserved locations. Written as zeros and read as undefined.
15-0	IADR	Lower 16 bits of the address of the Initialization Block. This register is an alias of CSR1. Whenever this register is written, CSR1 is updated with CSR16’s contents. Read/Write accessible only when the STOP bit in CSR0 is set. Unaffected by RESET.

CSR17: Initialization Block Address Upper

Bit	Name	Description
		This register is an alias of the location CSR2. Accesses to/from this register are equivalent to access to CSR2.
31-16	RES	Reserved locations. Written as zeros and read as undefined.
15-0	IADR	Upper 16 bits of the address of the Initialization Block. This register is an alias of CSR2. Whenever this register is written, CSR2 is updated with CSR17’s contents. Read/Write accessible only when the STOP bit in CSR0 is set. Unaffected by RESET.

CSR18: Current Receive Buffer Address Lower

Bit	Name	Description
31-16	RES	Reserved locations. Written as zeros and read as undefined.
15-0	CRBA	Contains the lower 16 bits of the current receive buffer address at which the PCnet-32 controller will store incoming frame data. Read/write accessible only when STOP bit is set. These bits are unaffected by H_RESET, S_RESET, or STOP.

CSR19: Current Receive Buffer Address Upper

Bit	Name	Description
31-16	RES	Reserved locations. Written as zeros and read as undefined.
15-0	CRBA	Contains the upper 16 bits of the current receive buffer address at which the PCnet-32 controller will store incoming frame data. Read/write accessible only when STOP bit is set. These bits are unaffected by H_RESET, S_RESET, or STOP.

CSR20: Current Transmit Buffer Address Lower

Bit	Name	Description
31-16	RES	Reserved locations. Written as zeros and read as undefined.
15-0	CXBA	Contains the lower 16 bits of the current transmit buffer address from which the PCnet-32 controller is transmitting. Read/write accessible only when STOP bit is set.

CSR21: Current Transmit Buffer Address Upper

Bit	Name	Description
31-16	RES	Reserved locations. Written as zeros and read as undefined.
15-0	CXBA	Contains the upper 16 bits of the current transmit buffer address from which the PCnet-32 controller is transmitting. Read/write accessible only when STOP bit is set.

CSR22: Next Receive Buffer Address Lower

Bit	Name	Description
31-16	RES	Reserved locations. Written as zeros and read as undefined.
15-0	NRBA	Contains the lower 16 bits of the next receive buffer address to which the PCnet-32 controller will store incoming frame data. Read/write accessible only when STOP bit is set.

CSR23: Next Receive Buffer Address Upper

Bit	Name	Description
31-16	RES	Reserved locations. Written as zeros and read as undefined.
15-0	NRBA	Contains the upper 16 bits of the next receive buffer address to which the PCnet-32 controller will store incoming frame data. Read/write accessible only when STOP bit is set.

CSR24: Base Address of Receive Ring Lower

Bit	Name	Description
31-16	RES	Reserved locations. Written as zeros and read as undefined.
15-0	BADR	Contains the lower 16 bits of the base address of the Receive Ring. Read/write accessible only when STOP bit is set.

CSR25: Base Address of Receive Ring Upper

Bit	Name	Description
31-16	RES	Reserved locations. Written as zeros and read as undefined.
15-0	BADR	Contains the upper 16 bits of the base address of the Receive Ring. Read/write accessible only when STOP bit is set.

CSR26: Next Receive Descriptor Address Lower

Bit	Name	Description
31-16	RES	Reserved locations. Written as zeros and read as undefined.
15-0	NRDA	Contains the lower 16 bits of the next RDRE address pointer. Read/write accessible only when STOP bit is set.

CSR27: Next Receive Descriptor Address Upper

Bit	Name	Description
31-16	RES	Reserved locations. Written as zeros and read as undefined.
15-0	NRDA	Contains the upper 16 bits of the next RDRE address pointer. Read/write accessible only when STOP bit is set.

CSR28: Current Receive Descriptor Address Lower

Bit	Name	Description

31-16 RES Reserved locations. Written as zeros and read as undefined.

15-0	CRDA	Contains the lower 16 bits of the current RDRE address pointer. Read/write accessible only when STOP bit is set.

CSR29: Current Receive Descriptor Address Upper

Bit	Name	Description
31-16	RES	Reserved locations. Written as zeros and read as undefined.
15-0	CRDA	Contains the upper 16 bits of the current RDRE address pointer. Read/write accessible only when STOP bit is set.

CSR30: Base Address of Transmit Ring Lower

Bit	Name	Description
31-16	RES	Reserved locations. Written as zeros and read as undefined.
15-0	BADX	Contains the lower 16 bits of the base address of the Transmit Ring. Read/write accessible only when STOP bit is set.

CSR31: Base Address of Transmit Ring Upper

Bit	Name	Description
31-16	RES	Reserved locations. Read and written as zero.
15-0	BADX	Contains the upper 16 bits of the base address of the Transmit Ring. Read/write accessible only when STOP bit is set.

CSR32: Next Transmit Descriptor Address Lower

Bit	Name	Description
31-16	RES	Reserved locations. Written as zeros and read as undefined.

15-0	NXDA	Contains the lower 16 bits of the next TDRE address pointer. Read/write accessible only when STOP bit is set.

CSR33: Next Transmit Descriptor Address Upper

Bit	Name	Description
31-16	RES	Reserved locations. Written as zeros and read as undefined.
15-0	NXDA	Contains the upper 16 bits of the next TDRE address pointer. Read/write accessible only when STOP bit is set.

CSR34: Current Transmit Descriptor Address Lower

Bit	Name	Description
31-16	RES	Reserved locations. Read and written as zero.
15-0	CXDA	Contains the lower 16 bits of the current TDRE address pointer. Read/write accessible only when STOP bit is set.

CSR35: Current Transmit Descriptor Address Upper

Bit	Name	Description
31-16	RES	Reserved locations. Written as zeros and read as undefined.
15-0	CXDA	Contains the upper 16 bits of the current TDRE address pointer. Read/write accessible only when STOP bit is set.

CSR36: Next Next Receive Descriptor Address Lower

Bit	Name	Description
31-16	RES	Reserved locations. Written as zeros and read as undefined.
15-0	NNRDA	Contains the lower 16 bits of the next next receive descriptor address pointer. Read/write accessible only when STOP bit is set.

CSR37: Next Next Receive Descriptor Address Upper

Bit	Name	Description
31-16	RES	Reserved locations. Written as zeros and read as undefined.
15-0	NNRDA	Contains the upper 16 bits of the next next receive descriptor address pointer. Read/write accessible only when STOP bit is set.

CSR38: Next Next Transmit Descriptor Address Lower

Bit	Name	Description
31-16	RES	Reserved locations. Written as zeros and read as undefined.
15-0	NNXDA	Contains the lower 16 bits of the next next transmit descriptor address pointer. Read/write accessible only when STOP bit is set.

CSR39: Next Next Transmit Descriptor Address Upper

Bit	Name	Description
31-16	RES	Reserved locations. Written as zeros and read as undefined.
15-0	NNXDA	Contains the upper 16 bits of the next next transmit descriptor address pointer. Read/write accessible only when STOP bit is set.

CSR40: Current Receive Status and Byte Count Lower

Bit	Name	Description
31-16	RES	Reserved locations. Written as zeros and read as undefined.
15-12	RES	Reserved locations. Read and written as zero.
11-0	CRBC	Current Receive Byte Count. This field is a copy of the BCNT field of RMD2 of the current receive descriptor. Read/write accessible only when STOP bit is set.

CSR41: Current Receive Status and Byte Count Upper

Bit	Name	Description
31-16	RES	Reserved locations. Written as zeros and read as undefined.
15-8	CRST	Current Receive Status. This field is a copy of bits 15:8 of RMD1 of the current receive descriptor.

		Read/write accessible only when STOP bit is set.
7-0	RES	Reserved locations. Read and written as zero.

CSR42: Current Transmit Status and Byte Count Lower

Bit	Name	Description
31-16	RES	Reserved locations. Written as zeros and read as undefined.
15-12	RES	Reserved locations. Read and written as zero.
11-0	CXBC	Current Transmit Byte Count. This field is a copy of the BCNT field of TMD2 of the current transmit descriptor. Read/write accessible only when STOP bit is set.

CSR43: Current Transmit Status and Byte Count Upper

Bit	Name	Description
31-16	RES	Reserved locations. Written as zeros and read as undefined.
15-8	CXST	Current Transmit Status. This field is a copy of bits 15:8 of TMD1 of the current transmit descriptor. Read/write accessible only when STOP bit is set.
7-0		RES Reserved locations. Read and written as zero.

CSR44: Next Receive Status and Byte Count Lower

Bit	Name	Description
31-16	RES	Reserved locations. Written as zeros and read as undefined.
15-12	RES	Reserved locations. Read and written as zero.
11-0	NRBC	Next Receive Byte Count. This field is a copy of the BCNT field of RMD2 of the next receive descriptor. Read/write accessible only when STOP bit is set.

CSR45: Next Receive Status and Byte Count Upper

Bit	Name	Description
31-16	RES	Reserved locations. Written as zeros and read as undefined.
15-8	NRST	Next Receive Status. This field is a copy of bits 15:8 of RMD1 of the next receive descriptor. Read/write accessible only when STOP bit is set.
7-0	RES	Reserved locations. Read and written as zero.

CSR46: Poll Time Counter

Bit	Name	Description
31-16	RES	Reserved locations. Written as zeros and read as undefined.
15-0	POLL	Poll Time Counter. This counter is incremented by the PCnet-32 controller microcode and is used to trigger the descriptor ring polling operation of the PCnet-32 controller. Read/write accessible only when STOP bit is set.

CSR47: Polling Interval

Bit	Name	Description
31-16	RES	Reserved locations. Written as zeros and read as undefined.
15-0	POLLINT	Polling Interval. This register contains the time that the PCnet-32 controller will wait between successive polling operations. The POLLINT value is expressed as the two's complement of the desired interval, where each bit of POLLINT represents 1 BCLK period of time (486 and VL-Bus modes; 2 BCLK 386 mode). POLLINT[3:0] are ignored. (POLLINT[16] is implied to be a one, so POLLINT[15] is significant, and does not represent the sign of the two's complement POLLINT value.) The default value of this register is 0000. This corresponds to a polling interval of 65,536 BCLK periods (486 and VL-Bus modes; 131,072 BCLK 386 mode). The POLLINT value of 0000 is created during the microcode initialization routine, and therefore might not be seen when reading CSR47 after H_RESET or S_RESET. If the user desires to program a value for POLLINT other than the default, then the correct procedure is to first set INIT only in CSR0. Then, when the initialization

sequence is complete, the user must set STOP in CSR0. Then the user may write to CSR47 and then set STRT in CSR0. In this way, the default value of 0000 in CSR47 will be overwritten with the desired user value.

If the user does NOT use the standard initialization procedure (standard implies use of an initialization block in memory and setting the INIT bit of CSR0), but instead, chooses to write directly to each of the registers that are involved in the INIT operation, then it is imperative that the user also write 0000 0000 to CSR47 as part of the alternative initialization sequence.

Read/write accessible only when STOP bit is set.

CSR48: Temporary Storage 2 Lower

Bit	Name	Description
31-16	RES	Reserved locations. Written as zeros and read as undefined.
15-0	TMP2	Lower 16 bits of a Temporary Storage location. Read/write accessible only when STOP bit is set.

CSR49: Temporary Storage 2 Upper

Bit	Name	Description
31-16	RES	Reserved locations. Written as zeros and read as undefined.
15-0	TMP2	Upper 16 bits of a Temporary Storage location. Read/write accessible only when STOP bit is set.

CSR50: Temporary Storage 3 Lower

Bit	Name	Description
31-16	RES	Reserved locations. Written as zeros and read as undefined.
15-0	TMP3	Lower 16 bits of a Temporary Storage location. Read/write accessible only when STOP bit is set.

CSR51: Temporary Storage 3 Upper

Bit	Name	Description
31-16	RES	Reserved locations. Written as zeros and read as undefined.
15-0	TMP3	Upper 16 bits of a Temporary Storage location. Read/write accessible only when STOP bit is set.

CSR52: Temporary Storage 4 Lower

Bit	Name	Description
31-16	RES	Reserved locations. Written as zeros and read as undefined.
15-0	TMP4	Lower 16 bits of a Temporary Storage location. Read/write accessible only when STOP bit is set.

CSR53: Temporary Storage 4 Upper

Bit	Name	Description
31-16	RES	Reserved locations. Written as zeros and read as undefined.
15-0	TMP4	Upper 16 bits of a Temporary Storage location. Read/write accessible only when STOP bit is set.

CSR54: Temporary Storage 5 Lower

Bit	Name	Description
31-16	RES	Reserved locations. Written as zeros and read as undefined.
15-0	TMP5	Lower 16 bits of a Temporary Storage location. Read/write accessible only when STOP bit is set.

CSR55: Temporary Storage 5 Upper

Bit	Name	Description
31-16	RES	Reserved locations. Written as zeros and read as undefined.
15-0	TMP5	Upper 16 bits of a Temporary Storage location. Read/write accessible only when STOP bit is set.

CSR56: Temporary Storage 6 Lower

Bit	Name	Description
31-16	RES	Reserved locations. Written as zeros and read as undefined.
15-0	TMP6	Lower 16 bits of a Temporary Storage location. Read/write accessible only when STOP bit is set.

CSR57: Temporary Storage 6 Upper

Bit	Name	Description
31-16	RES	Reserved locations. Written as zeros and read as undefined.
15-0	TMP6	Upper 16 bits of a Temporary Storage location. Read/write accessible only when STOP bit is set.

CSR58: Software Style

Bit	Name	Description
		This register is an alias of the location BCR20. Accesses to/from this register are equivalent to accesses to BCR20.
31-10	RES	Reserved locations. Written as zeros and read as undefined.
9	CSRPCNET	CSR PCnet-ISA configuration bit. When set, this bit indicates that the PCnet-32 controller register bits of CSR4 and CSR3 will map directly to the CSR4 and CSR3 bits of the PCnet-ISA (Am79C960) device. When cleared, this bit indicates that PCnet-32 controller register bits of CSR4 and CSR3 will map directly to the CSR4 and CSR3 bits of the ILACC (Am79C900) device. The value of CSRPCNET is determined by the PCnet-32 controller. CSRPCNET is read only by the host. The PCnet-32 controller uses the setting of the Software Style register (BCR20[7:0]/CSR58[7:0]) to determine the value for this bit. CSRPCNET is set to a ONE by H_RESET or S_RESET and is not affected by STOP.
8	SSIZE32	Software Size 32 bits. When set, this bit indicates that the PCnet-32 controller utilizes AMD 79C900 (ILACC) software structures. In particular, Initialization Block and Transmit and Receive descriptor bit maps are affected. When cleared, this bit indicates that the PCnet-32 controller utilizes AMD PCnet-ISA software structures. ***Note***: Regardless of the setting of SSIZE32, the Initialization Block must always begin on a double-word boundary. The value of SSIZE32 is determined by the PCnet-32 controller. SSIZE32 is read only by the host. The PCnet-32 controller uses the setting of the Software Style register (BCR20, bits 7-0) to determine the value for this bit. SSIZE32 is cleared by H_RESET or S_RESET and is not affected by STOP. If SSIZE32 is reset, then bits IADR[31:24] of CSR2 will be used to generate values for the upper 8 bits of the 32 bit address bus during master accesses initiated by the PCnet-32 controller. This action is required, since the 16-bit software structures specified by the SSIZE32=0 setting will yield only 24 bits of address for PCnet-32 controller bus master accesses. If SSIZE32 is set, then the software structures that are common to the PCnet-32 controller and the host system will supply a full 32 bits for each address pointer that is needed by the PCnet-32 controller for performing master accesses. The value of the SSIZE32 bit has no effect on the drive of the upper 8 address pins. The upper 8 address pins are always driven, regardless of the state of the SSIZE32 bit. Note that the setting of the SSIZE32 bit has no effect on the defined width for I/O resources. I/O resource width is determined by the state of the DWIO bit.
7-0	SWSTYLE	Software Style register. The value in this register determines the “style” of I/O and memory resources that are used by the PCnet-32 controller. The S/W resource style selection will affect the interpretation of a few bits within the CSR space and the width of the descriptors and initialization block. See Table 43. All PCnet-32 controller CSR bits and BCR bits and all descriptor, buffer and initialization block entries not cited in the table above are unaffected by the Software Style selection and are therefore always fully functional as specified in the BCR and CSR sections.

Read/write accessible only when STOP bit is set.

The SWSTYLE register will contain the value 00h following H_RESET or S_RESET and will be unaffected by STOP.

Table 43. Software Resource Style Selection

SWSTYLE[7:0] (Hex)	Style Name	CSRPCNET	SSIZE32	Altered Bit Interpretations
00	LANCE/ PCnet-ISA	1	0	ALL CSR4 bits will function as defined in the CSR4 section. TMD1[29] functions as ADD_FCS
01	ILACC	0	1	CSR4[9:8], CSR4[5:4] and CSR4[1:0] will have *no function*, but will be writeable and readable. CSR4[15:10], CSR4[7:6] and CSR4[3:2] will function as defined in the CSR4 section. TMD1[29] becomes NO_FCS.
02	PCnet-32	1	1	ALL CSR4 bits will function as defined in the CSR4 section. TMD1[29] functions as ADD_FCS
All other combinations	Reserved	Undefined	Undefined	Undefined

CSR59: IR Register

Bit	Name	Description
31-16	RES	Reserved locations. Written as zeros and read as undefined.
15-0	IRREG	Contains the value 0105. This register always contains the same value. It is not writable. Read accessible only when STOP bit is set.

CSR60: Previous Transmit Descriptor Address Lower

Bit	Name	Description
31-16	RES	Reserved locations. Written as zeros and read as undefined.
15-0	PXDA	Contains the lower 16 bits of the previous TDRE address pointer. PCnet-32 controller has the capability to stack multiple transmit frames. Read/write accessible only when STOP bit is set.

CSR61: Previous Transmit Descriptor Address Upper

Bit	Name	Description
31-16	RES	Reserved locations. Written as zeros and read as undefined.
15-0	PXDA	Contains the upper 16 bits of the previous TDRE address pointer. PCnet-32 controller has the capability to stack multiple transmit frames. Read/write accessible only when STOP bit is set.

CSR62: Previous Transmit Status and Byte Count Lower

Bit	Name	Description
31-16	RES	Reserved locations. Written as zeros and read as undefined.
15-12	RES	Reserved locations. Read and written as zero. Accessible only when STOP bit is set.
11-0	PXBC	Previous Transmit Byte Count. This field is a copy of the BCNT field of TMD2 of the previous transmit descriptor. Read/write accessible only when STOP bit is set.

CSR63: Previous Transmit Status and Byte Count Upper

Bit	Name	Description
31-16	RES	Reserved locations. Written as zeros and read as undefined.
15-8	PXST	Previous Transmit Status. This field is a copy of bits 15:8 of TMD1 of the previous transmit descriptor. Read/write accessible only when STOP bit is set.
7-0	RES	Reserved locations. Read and written as zero. Accessible only when STOP bit is set.

CSR64: Next Transmit Buffer Address Lower

Bit	Name	Description
31-16	RES	Reserved locations. Written as zeros and read as undefined.
15-0	NXBA	Contains the lower 16 bits of the next transmit buffer address from which the PCnet-32 controller will transmit an outgoing frame. Read/write accessible only when STOP bit is set.

CSR65: Next Transmit Buffer Address Upper

Bit	Name	Description
31-16	RES	Reserved locations. Written as zeros and read as undefined.
15-0	NXBA	Contains the upper 16 bits of the next transmit buffer address from which the PCnet-32 controller will transmit an outgoing frame. Read/write accessible only when STOP bit is set.

CSR66: Next Transmit Status and Byte Count Lower

Bit	Name	Description
31-16	RES	Reserved locations. Written as zeros and read as undefined.
15-12	RES	Reserved locations. Read and written as zero. Accessible only when STOP bit is set.
11-0	NXBC	Next Transmit Byte Count. This field is a copy of the BCNT field of TMD2 of the next transmit descriptor.

Read/write accessible only when STOP bit is set.

CSR67: Next Transmit Status and Byte Counter Upper

Bit	Name	Description
31-16	RES	Reserved locations. Written as zeros and read as undefined.
15-8	NXST	Next Transmit Status. This field is a copy of bits 15:8 of TMD1 of the next transmit descriptor. Read/write accessible only when STOP bit is set.
7-0	RES	Reserved locations. Read and written as zero. Accessible only when STOP bit is set.

CSR68: Transmit Status Temporary Storage Lower

Bit	Name	Description
31-16	RES	Reserved locations. Written as zeros and read as undefined.
15-0	XSTMP	Lower 16 bits of a Transmit Status Temporary Storage location. Read/write accessible only when STOP bit is set.

CSR69: Transmit Status Temporary Storage Upper

Bit	Name	Description
31-16	RES	Reserved locations. Written as zeros and read as undefined.
15-0	XSTMP	Transmit Status Temporary Storage location. Read/write accessible only when STOP bit is set.

CSR70: Temporary Storage Lower

Bit	Name	Description
31-16	RES	Reserved locations. Written as zeros and read as undefined.
15-0	TMP8	Lower 16 bits of a Temporary Storage location. Read/write accessible only when STOP bit is set.

CSR71: Temporary Storage Upper

Bit	Name	Description
31-16	RES	Reserved locations. Written as zeros and read as undefined.
15-0	TMP8	Upper 16 bits of a Temporary Storage location. Read/write accessible only when STOP bit is set.

CSR72: Receive Ring Counter

Bit	Name	Description
31-16	RES	Reserved locations. Written as zeros and read as undefined.
15-0	RCVRC	Receive Ring Counter location. Contains a Two's complement binary number used to number the current receive descriptor. This counter interprets the value in CSR76 as pointing to the first descriptor. A counter value of zero corresponds to the last descriptor in the ring. Read/write accessible only when STOP bit is set.

CSR74: Transmit Ring Counter

Bit	Name	Description
31-16	RES	Reserved locations. Written as zeros and read as undefined.
15-0	XMTRC	Transmit Ring Counter location. Contains a Two's complement binary number used to number the current transmit descriptor. This counter interprets the value in CSR78 as pointing to the first descriptor. A counter value of zero corresponds to the last descriptor in the ring. Read/write accessible only when STOP bit is set.

CSR76: Receive Ring Length

Bit	Name	Description
31-16	RES	Reserved locations. Written as zeros and read as undefined.
15-0	RCVRL	Receive Ring Length. Contains the two's complement of the receive descriptor ring length. This register is initialized during the PCnet-32 controller initialization routine based on the value in the RLEN field of the initialization block. However, this register can be manually altered. the actual receive ring length is defined by the current value in this register. The ring length can be defined as any value from 1 to 65535. Read/write accessible only when STOP bit is set.

CSR78: Transmit Ring Length

Bit	Name	Description
31-16	RES	Reserved locations. Written as zeros and read as undefined.
15-0	XMTRL	Transmit Ring Length. Contains the two's complement of the transmit descriptor ring length. This register is initialized during the PCnet-32 controller initialization routine based on the value in the TLEN field of the initialization block. However, this register can be manually altered. The actual transmit ring length is defined by the current value in this register. The ring length can be defined as any value from 1 to 65535. Read/write accessible only when STOP bit is set.

CSR80: Burst and FIFO Threshold Control

Bit	Name	Description
31-16	RES	Reserved locations. Written as zeros and read as undefined.
15-14	RES	Reserved locations. Read as ones and written as zero.
13-12	RCVFW[1:0]	Receive FIFO Watermark. RCVFW controls the point at which receive DMA is requested in relation to the number of received bytes in the receive FIFO. RCVFW specifies the number of bytes which must be present (once the frame has been verified as a non-runt) before receive DMA is requested. Note however that in order for receive DMA to be performed for a new frame, at least 64 bytes must have been received. This effectively avoids having to react to receive frames which are runts or suffer a collision during the slot time (512 bit times). If the Runt Packet Accept feature is enabled, receive DMA will be requested as soon as either the RCVFW threshold is reached, or a complete valid receive frame is

detected (regardless of length). RCVFW is set to a value of 10 (64 bytes) after H_RESET or S_RESET and is unaffected by STOP.

RCVFW[1:0]	Bytes Received
00	16
01	32
10	64
11	Reserved

Read/write accessible only when STOP bit is set.

Certain combinations of watermark programming and LINBC (BCR18[2-0]) programming may create situations where no linear bursting is possible, or where the FIFO may be excessively read or excessively written. Such combinations are declared as illegal.

Combinations of watermark settings and LINBC settings must obey the following relationship:

watermark (in bytes) Š LINBC (in bytes)

Combinations of watermark and LINBC settings that violate this rule may cause unexpected behavior.

11-10 XMTSP[1:0]Transmit Start Point. XMTSP controls the point at which preamble transmission attempts commence in relation to the number of bytes written to the transmit FIFO for the current transmit frame. When the entire frame is in the FIFO, transmission will start regardless of the value in XMTSP. XMTSP is given a value of 10 (64 bytes) after H_RESET or S_RESET and is unaffected by STOP. Regardless of XMTSP, the FIFO will not internally over write its data until at least 64 bytes (or the entire frame if <64 bytes) have been transmitted onto the network. This ensures that for collisions within the slot time window, transmit data need not be rewritten to the transmit FIFO, and retries will be handled autonomously by the MAC. This bit is read/write accessible only when the STOP bit is set.

XMTSP[1:0]	Bytes Written
00	4
01	16
10	64
11	112

9-8 XMTFW[1:0] Transmit FIFO Watermark. XMTFW specifies the point at which transmit DMA stops, based upon the number of write cycles that could be performed to the transmit FIFO without FIFO overflow. Transmit DMA is allowed at any time when the number of write cycles specified by SMTFW could be executed without causing transmit FIFO overflow. XMTFW is set to a value of 00b (8 cycles) after H_RESET or S_RESET and is unaffected by STOP. Read/write accessible only when STOP bit is set.

XMTFW[1:0]	Write Cycles
00	8
01	16
10	32
11	Reserved

Certain combinations of watermark programming and LINBC programming may create situations where no linear bursting is possible, or where the FIFO may be excessively read or excessively written. Such combinations are declared as illegal.

Combinations of watermark settings and LINBC settings must obey the following relationship:

watermark (in bytes) Š LINBC (in bytes)

Combinations of watermark and LINBC settings that violate this rule may cause unexpected behavior.

7-0 DMACR[7:0] DMA Cycle Register. This register contains the maximum allowable number of transfers to system memory that the Bus Interface will perform during a single DMA cycle. The Cycle Register is not used to limit the number of transfers during Descriptor transfers. A value of zero will be interpreted as one transfer.

During H_RESET or S_RESET a value of 16 is loaded in the BURST register. If the DMAPLUS bit in CSR4 is set, the DMA Cycle Register is disabled. When the ENTST bit in CSR4 is set, all writes to this register will automatically perform a decrement cycle.

When the Cycle Register times out in the middle of a linear burst, the linear burst will continue until a legal starting address is reached, and then the PCnet-32 controller will relinquish the bus.

Therefore, if linear bursting is enabled, and the user wishes the PCnet-32 controller to limit bus activity to desired_max transfers, then the Cycle Register should be programmed to a value of:

Burst count setting = (desired_ max DIV (length of linear burst in transfers) x length of linear burst in transfers where DIV is the operation that yields the INTEGER portion of the ^{3} operation.

***Note*:** If either Linear Burst Write is enabled, the value has to be greater than or equal to 4.

Read/write accessible only when the STOP bit is set.

CSR82: Bus Activity Timer

Bit	Name	Description
31-16	RES	Reserved locations. Written as zeros and read as undefined.
15-0	DMABAT	Bus Activity Timer Register. If the TIMER bit in CSR4 is set, this register contains the maximum allowable time that PCnet-32 controller will take up on the system bus during FIFO data transfers for a single DMA cycle. The Bus Activity Timer Register does not limit the number of transfers during Descriptor transfers. The DMABAT value is interpreted as an unsigned number with a resolution of 0.1 µs. For instance, a value of 51 µs would be programmed with a value of 510. If the TIMER bit in CSR4 is set, DMABAT is enabled and must be initialized by the user. The DMABAT register is undefined until written. When the ENTST bit in CSR4 is set, all writes to this register will automatically perform a decrement cycle. If the user has NOT enabled the Linear Burst function and wishes the PCnet-32 controller to limit bus activity to MAX_TIME µs, then the Burst Timer should be programmed to a value of: *MAX_TIME- [(11 + 4w) x (BCLK period)],* where w = wait states. If the user has enabled the Linear Burst function and wishes the PCnet-32 controller to limit bus activity to MAX_TIME µs, then the Burst Timer should be programmed to a value of: *MAX_TIME- [((3+lbs) x w + 10 + lbs) x (BCLK period)],* where w = wait states and lbs = linear burst size in number of transfers per sequence. This is because the PCnet-32 controller may use as much as one “linear burst size” plus three transfers in order to complete the linear burst before releasing the bus. As an example, if the linear burst size is four transfers, and the number of wait states for the system memory is two, and the BCLK period is 30 ns and the MAX time allowed on the bus is 3 µs, then the Burst Timer should be programmed for: *MAX_TIME- [((3+lbs) x w + 10 +* *lbs) x (BCLK period)], 3 mS - [(3 + 4) x 2 +10 + 4) x (30 ns)] = 3 mS - (28 x 30 ns) = 3 - 0.84 mS = 2.16 mS.* Then, if the PCnet-32 controller’s Bus ActivityTimer times out after 2.16 µs when the PCnet-32 controller has completed all but the last three transfers of a linear burst, the PCnet-32 controller may take as much as 0.84 µs to complete the bursts and release the bus. The bus release will occur at 2.16 + 0.84 = 3 ms.

A value of zero will in the DMABAT register with the TIMER bit in CSR4 set to ONE will produce single linear burst sequences per bus master period when programmed for linear burst mode, and will yield sets of three transfers when not programmed for linear burst mode.

The Bus Activity Timer is set to a value of 00h after H_RESET or S_RESET and is unaffected by STOP.

Read/write accessible only when STOP bit is set.

CSR84: DMA Address Lower

Bit	Name	Description
31-16	RES	Reserved locations. Written as zeros and read as undefined.
15-0	DMABA	DMA Address Register. This register contains the lower 16 bits of the address of system memory for the current DMA cycle. The Bus Interface Unit controls the Address Register by issuing increment commands to increment the memory address for sequential operations. The DMABA register is undefined until the first PCnet-32 controller DMA operation. When the ENTST bit in CSR4 is set, all writes to this register will automatically perform an increment cycle. Read/write accessible only when STOP bit is set.

CSR85: DMA Address Upper

Bit	Name	Description
31-16	RES	Reserved locations. Written as zeros and read as undefined.
15-0	DMABA	DMA Address Register. This register contains the upper 16 bits of the address of system memory for the current DMA cycle. The Bus Interface Unit controls the Address Register by issuing increment commands to increment the memory address for sequential operations. The DMABA register is undefined until the first PCnet-32 controller DMA operation. When the ENTST bit in CSR4 is set, all writes to this register will automatically perform an increment cycle. Read/write accessible only when STOP bit is set.

CSR86: Buffer Byte Counter

Bit	Name	Description
31-16	RES	Reserved locations. Written as zeros and read as undefined.
15-12	RES	Reserved, Read and written with ones.
11-0	DMABC	DMA Byte Count Register. Contains a Two's complement binary number of the current size of the remaining transmit or receive buffer in bytes. This register is incremented by the Bus Interface Unit. The DMABC register is undefined until written. When ENTST (CSR4.15) is asserted, all writes to this register will automatically perform an increment cycle. Read/write accessible only when STOP bit is set.

CSR88: Chip ID Lower

Bit	Name	Description
		This register is exactly the same as the Chip ID register in the JTAG description.
31 - 28	Version.	This 4-bit pattern is silicon-revision dependent.
27 - 12	Part number.	The 16-bit code for the PCnet-32 controller is 0010 0100 0011 0000b.
11 - 1		Manufacturer ID. The 11-bit manufacturer code for AMD is 00000000001b. This code is per the JEDEC Publication 106-A.
0		Always a logic 1.

CSR89: Chip ID Upper

Bit	Name	Description
		The lower 16 bits of this register are exactly the same as the upper 16 bits of the Chip ID register in the JTAG description, which are exactly the same as the upper 16 bits of CSR88.
31 - 16		Reserved locations. Read as undefined.
15 - 12		Version. This 4-bit pattern is silicon-revision dependent.

11 - 0		Upper 12 bits of the PCnet-32 controller part number, i.e. 0010 0100 0011b.

CSR92: Ring Length Conversion

Bit	Name	Description
31-16	RES	Reserved locations. Written as zeros and read as undefined.
15-0	RCON	Ring Length Conversion Register. This register performs a ring length conversion from an encoded value as found in the initialization block to a Two's complement value used for internal counting. By writing bits 15-12 with an encoded ring length, a Two's complemented value is read. The RCON register is undefined until written. Read/write accessible only when STOP bit is set.

CSR94: Transmit Time Domain Reflectometry Count

Bit	Name	Description
31-16	RES	Reserved locations. Written as zeros and read as undefined.
15-10	RES	Reserved locations. Read and written as zero.
9-0	XMTTDR	Time Domain Reflectometry reflects the state of an internal counter that counts from the start of transmission to the occurrence of loss of carrier. TDR is incremented at a rate of 10 MHz. Read accessible only when STOP bit is set. Write operations are ignored. XMTTDR is cleared by H_RESET or S_RESET.

CSR96: Bus Interface Scratch Register 0 Lower

Bit	Name	Description
31-16	RES	Reserved locations. Written as zeros and read as undefined.
15-0	SCR0	This register is shared between the Buffer Management Unit and the Bus Interface Unit. All Descriptor Data communications between the BIU and BMU are written and read through SCR0 and SCR1 registers. The SCR0 register is undefined until written. Read/write accessible only when STOP bit is set.

CSR97: Bus Interface Scratch Register 0 Upper

Bit	Name	Description
31-16	RES	Reserved locations. Written as zeros and read as undefined.
15-0 S	CR0	This register is shared between the Buffer Management Unit and the Bus Interface Unit. All Descriptor Data communications between the BIU and BMU are written and read through SCR0 and SCR1 registers. The SCR0 register is undefined until written. Read/write accessible only when STOP bit is set.

CSR98: Bus Interface Scratch Register 1 Lower

Bit	Name	Description
31-16	RES	Reserved locations. Written as zeros and read as undefined.
15-0	SCR1	This register is shared between the Buffer Management Unit and the Bus Interface Unit. All Descriptor Data communications between the BIU and BMU are written and read through SCR0 and SCR1 registers. Read/write accessible only when STOP bit is set.

CSR99: Bus Interface Scratch Register 1 Upper

Bit	Name	Description
31-16	RES	Reserved locations. Written as zeros and read as undefined.
15-0	SCR1	This register is shared between the Buffer Management Unit and the Bus Interface Unit. All Descriptor Data communications between the BIU and BMU are written and read through SCR0 and SCR1 registers. Read/write accessible only when STOP bit is set.

CSR100: Bus Time-Out

Bit	Name	Description
31-16	RES	Reserved locations. Written as zeros and read as undefined.
15-0	MERRTO	This register contains the value of the longest allowable bus latency (interval between assertion of HOLD and assertion of HLDA) that a slave device may insert into a PCnet-32 controller master transfer. If this value of bus latency is exceeded,

then a MERR will be indicated in CSR0, bit 11, and an interrupt may be generated, depending upon the setting of the MERRM bit (CSR3, bit 11) and IENA bit (CSR0[6]).

The value in this register is interpreted as a number of XTAL1³2 clock periods. (i.e. the value in this register is given in 0.1 ms increments.) For example, the value 0200h (512 decimal) will cause a MERR to be indicated after 51.2 µs of bus latency.

A value of zero will allow an infinitely long bus latency. i.e. a value of zero will never give a bus time-out error. A non-zero value is interpreted as an unsigned number of BCLK cycles.

This register is set to 0200 by H_RESET or S_RESET and is unaffected by STOP.

Read/write accessible only when STOP bit is set.

CSR104: SWAP Register Lower

Bit	Name	Description
31-16	RES	Reserved locations. Written as zeros and read as undefined.
15-0	SWAP	This register performs word and byte swapping depending upon if 32-bit or 16-bit internal write operations are performed. This register is used internally by the BIU/BMU as a word or byte swapper. The register is externally accessible for test reasons only. CSR104 holds the lower 16 bits and CSR105 holds the upper 16 bits. Read/write accessible only when STOP bit is set.

CSR105: SWAP Register Upper

Bit	Name	Description
31-16	RES	Reserved locations. Written as zeros and read as undefined.
15-0	SWAP	This register performs word and byte swapping depending upon if 32-bit or 16-bit internal write operations are performed. This register is used internally by the BIU/BMU as a word or byte swapper. The register is externally accessible for test reasons only. CSR104 holds the lower 16 bits and CSR105 holds the upper 16 bits. Read/write accessible only when STOP bit is set.

CSR108: Buffer Management Scratch Lower

Bit	Name	Description
31-16	RES	Reserved locations. Written as zeros and read as undefined.
15-0	BMSCR	The Buffer Management Scratch register is used for assembling Receive and Transmit Status. This register is also used as the primary scan register for Buffer Management Test Modes. BMSCR register is undefined until written. Read/write accessible only when STOP bit is set.

CSR109: Buffer Management Scratch Upper

Bit	Name	Description
31-16	RES	Reserved locations. Written as zeros and read as undefined.
15-0	BMSCR	The Buffer Management Scratch register is used for assembling Receive and Transmit Status. This register is also used as the primary scan register for Buffer Management Test Modes. BMSCR register is undefined until written. Read/write accessible only when STOP bit is set.

CSR112: Missed Frame Count

Bit	Name	Description
31-16	RES	Reserved locations. Written as zeros and read as undefined.
15-0	MFC	Missed Frame Count. Indicates the number of missed frames. MFC will roll over to a count of zero from the value 65535. The MFCO bit of CSR4 (bit 8) will be set each time that this occurs. This register is always readable and is cleared by H_RESET or S_RESET or STOP. A write to this register performs an increment when the ENTST bit in CSR4 is set.

CSR114: Receive Collision Count

Bit	Name	Description

31-16	RES	Reserved locations. Written as zeros and read as undefined.
15-0	RCC	Receive Collision Count. Indicates the total number of collisions encountered by the receiver since the last reset of the counter. RCC will roll over to a count of zero from the value 65535. The RCVCCO bit of CSR4 (bit 5) will be set each time that this occurs. The RCC value is read accessible at all times, regardless of the value of the STOP bit. Write operations are ignored. RCC is cleared by H_RESET or S_RESET or by setting the STOP bit. A write to this register performs an increment when the ENTST bit in CSR4 is set.

CSR122: Receive Frame Alignment Control

Bit	Name	Description
31-16	RES	Reserved locations. Written as zeros and read as undefined.
15-1	RES	Reserved locations, written as zeros and read as undefined.
0	RCVALGN	Receive Frame Align. When set, this bit forces the data field of ISO 8802-3 (IEEE/ANSI 802.3) frames to align to 0 MOD 4 address boundaries (i.e. double word aligned addresses). It is important to note that this feature will only function correctly if all receive buffer boundaries are double-word aligned and all receive buffers have 0 MOD 4 lengths. In order to accomplish the data alignment, the PCnet-32 controller simply inserts two bytes of random data at the beginning of the receive packet (i.e. before the ISO 8802-3 (IEEE/ ANSI 802.3) destination address field). The MCNT field reported to the receive descriptor will not include the extra two bytes. RCVALGN is cleared by H_RESET or S_RESET and is not affected by STOP. Read/write accessible only when STOP bit is set.

CSR124: Buffer Management Test

Bit	Name	Description
		This register is used to place the BMU/BIU into various test mode to support Test/Debug. This register is writable only when the ENTST bit in CSR4 and the STOP bit of CSR0 are both set. The functions controlled by this register are enabled only if the ENTST bit is set. ENTST should be set before anything in CSR124 can be programmed, including RUNTACC. ENTST must be reset after writing to CSR124 before writing to any other register. If it is done, the PCnet-32 controller will not run. All bits in this register or cleared by H_RESET or S_RESET and are not affected by STOP.
31-16	RES	Reserved locations. Written as zeros and read as undefined.
15-5	RES	Reserved locations. Read and written as zero.
4	GPSIEN	This bit places the PCnet-32 controller in the GPSI mode. This mode will reconfigure the System Interface Address Pins so that the GPSI port is exposed. This allows bypassing the MENDEC T-MAU logic. The GPSI mode may also be enabled by test shadow bits setting of BCR18, bits 4 and 3 as in Table 44. See Table 45 for pin reconfiguration in GPSI mode. Note that when the GPSI mode is invoked, only the lower 24 bits of the address bus are available. During Software Relocatable Mode the LED2 pin must be pulled LOW. IOAW24 (BCR21[8]) must be set to allow slave operations. During master accesses in GPSI mode, the PCnet-32 controller will not drive the upper 8 bits of the address bus with address information.
3	RPA	Runt Packet Accept. This bit forces the receive logic to accept runt packets (packets shorter than 64 bytes). The state of the RPA bit can be changed only when the device is

in the test mode (when the ENTST bit in CSR4 is set to ONE). To enable RPA, software must first write a ONE to the ENTST bit. Next, software must write a ONE to the RPA bit. Finally, software must write a ZERO to the ENTST bit to take the device out of test mode operation. Once the RPA bit has been set to ONE, the device will remain in the Runt Packet Accept mode until the RPA bit is cleared to ZERO.

2-0 RES Reserved locations. Written as zeros and read as undefined.

Table 44. GPSI Mode Selection

TSTSHDW Value (BCR18[4:3])	PVALID (BCR19[15])	GPSIEN	Operating Mode
00	X	0	Normal Operating Mode
10	1	X	GPSI Mode
01	1	0	Reserved
11	1	X	Reserved
XX	0	0	Normal Operating Mode
XX	0	1	GPSI Mode

Table 45. GPSI Pin Configurations

GPSI Function	GPSI I/O Type	LANCE GPSI Pin	ILACC GPSI Pin	PCnet-32/ PCnet-ISA GPSI Pin	PCnet-32 Pin Number	PCnet-32 Normal Pin Function
Transmit Data	O	TX	TXD	TXDAT	132	A31
Transmit Enable	O	TENA	RTS	TXEN	133	A30
Transmit Clock	I	TCLK	TXC	STDCLK	134	A29
Collision	I	CLSN	CDT	CLSN	137	A28
Receive Carrier Sense	I	RENA	CRS	RXCRS	138	A27
Receive Clock	I	RCLK	RXC	SRDCLK	140	A26
Receive Data	I	RX	RXD	RXDAT	141	A25

Bus Configuration Registers

The Bus Configuration Registers (BCR) are used to program the configuration of the bus interface and other special features of the PCnet-32 controller that are not related to the IEEE 8802-3 MAC functions. The BCRs are accessed by first setting the appropriate RAP value, and then by performing a slave access to the BDP.

All BCR registers are 16 bits in width in WIO mode and 32 bits in width in DWIO mode. The upper 16 bits of all BCR registers is undefined when in DWIO mode. These bits should be written as ZEROs and should be treated as undefined when read. The "Default" value given for any BCR is the value in the register after H_RESET, and is hexadecimal unless otherwise stated. BCR register values are unaffected by S_RESET and are unaffected by the assertion of the STOP bit.

Note that several registers have no default value. BCR3 and BCR8-BCR15 are reserved and have undefined values. BCR2, BCR16, BCR17 and BCR21 are not observable without first being programmed, either through the EEPROM read operation or through the Software Relocatable Mode. Therefore, the only observable values for these registers are those that have been programmed and a default value is not applicable. See Table 46.

Writes to those registers marked as "Reserved" will have no effect. Reads from these locations will produce undefined values.

Table 46. Bus Configuration Registers

RAP Addr.	Mnemonic	Default (Hex)	Name	Programmability		
				User	EEPROM	SRM
1	MSWRA	0005	Master Mode Write Active	No	No	No
2	MC	N/A*	Miscellaneous Configuration	Yes	Yes	Yes
3	Reserved	N/A		No	No	No
4	LNKST	00C0	Link Status (Default)	Yes	No	No
5	LED1	0084	Receive (Default)	Yes	No	No
6	LED2	0088	Receive Polarity (Default)	Yes	No	No
7	LED3	0090	Transmit (Default)	Yes	No	No
8–15	Reserved	N/A		No	No	No
16	IOBASEL	N/A*	I/O Base Address Lower	Yes	Yes	Yes
17	IOBASEU	N/A*	I/O Base Address Upper	Yes	Yes	Yes
18	BSBC	2101	Burst Size and Bus Control	Yes	Yes	No
19	EECAS	0002	EEPROM Control and Status	Yes	No	No
20	SWSTYLE	0000	Software Style	Yes	No	No
21	INTCON	N/A*	Interrupt Control	Yes	Yes	Yes

*Key: **SRM** = Software Relocatable Mode*

** Registers marked with an asterisk (*) have no default value, since they are not observable without first being programmed, either through the EEPROM read operation or through the Software Relocatable Mode. Therefore, the only observable values for these registers are those that have been programmed and a default value is not applicable.*

BCR0: Master Mode Read Active

Bit	Name	Description
31-16	RES	Reserved locations. Written as zeros and read as undefined.
15-0	MSRDA	Reserved locations. After H_RESET, the value in this register will be 0005. The settings of this register will have no effect on any PCnet-32 controller function. Writes to this register have no effect on the operation of the PCnet-32 controller and will not alter the value that is read.

BCR1: Master Mode Write Active

Bit	Name	Description
31-16	RES	Reserved locations. Written as zeros and read as undefined.

15-0	MSWRA	Reserved locations. After H_RESET, the value in this register will be 0005. The settings of this register will have no effect on any PCnet-32 controller function. Writes to this register have no effect on the operation of the PCnet-32 controller and will not alter the value that is read.

BCR2: Miscellaneous Configuration

Bit	Name	Description
		Note that all bits in this register are programmable through the EEPROM PREAD operation and through the Software Relocatable Mode operation.
31-16	RES	Reserved locations. Written as zeros and read as undefined.
15	RES	Reserved location. Written and read as zero.
14	T-MAULOOP	When set, this bit allows external loopback packets to pass onto the network through the T-MAU interface, if the T-MAU interface has been selected. If the T-MAU interface has not been selected, then this bit has no effect. This bit is reset to ZERO by H_RESET and is unaffected by S_RESET or STOP.
13-9	RES	Reserved locations. Written and read as zero.
8	IESRWE	IEEE Shadow Ram Write Enable. The PCnet-32 controller contains a shadow RAM on board for storage of the IEEE address following the serial EEPROM read operation. Accesses to APROM I/O Resources will be directed toward this RAM. When IESRWE is set to a ONE, then 32-bit and 16-bit write access to the shadow RAM will be enabled. When IESRWE is set to a ZERO, then 32-bit and 16-bit write access to the shadow RAM will be disabled. At no time are 8-bit write accesses to the shadow RAM allowed. This bit is reset to ZERO by H_RESET and is unaffected by S_RESET or STOP.
7	INTLEVEL	Interrupt Level. This bit allows the interrupt output signals to be programmed for edge or level-sensitive applications. When INTLEVEL is set to a ZERO, the selected interrupt pin is configured for edge sensitive operation. In this mode, an interrupt request is signaled by a high level driven on the selected interrupt pin by the PCnet-32 controller. When the interrupt is cleared, the selected interrupt pin is driven to a low level by the PCnet-32 controller. This mode is intended for systems that do not allow interrupt channels to be shared by multiple devices. When INTLEVEL is set to a ONE, the selected interrupt pin is configured for level sensitive operation. In this mode, an interrupt request is signaled by a low level driven on the selected interrupt pin by the PCnet-32 controller. When the interrupt is cleared, the selected interrupt pin is floated by the PCnet-32 controller and allowed to be pulled to a high level by an external pull-up device. This mode is intended for systems which allow the interrupt signal to be shared by multiple devices. This bit is reset to ZERO by H_RESET and is unaffected by R_RESET or STOP.
6-4	RES	Reserved locations. Written and read as zero.
3	EADISEL	EADI Select. When set, this bit configures three of the four LED outputs to function as the outputs of an EADI interface. LED1 becomes SFBD, LED2 becomes SRDCLK and LEDPRE3 becomes SRD. $\overline{\text{LNKST}}$ continues to function as an LED output. In addition to these reassignments, the INTR2 pin will be reassigned to function as the $\overline{\text{EAR}}$ pin. This bit is reset to ZERO by H_RESET and is unaffected by S_RESET or STOP.
2	AWAKE	Auto-Wake. If LNKST is set and AWAKE = “1”, the 10BASE-T receive circuitry is active during sleep

and listens for Link Pulses. $\overline{\text{LNKST}}$ indicates Link Status and goes active if the 10BASE-T port comes of out of "link fail" state. This $\overline{\text{LNKST}}$ pin can be used by external circuitry to re-enable the PCnet-32 controller and/or other devices.

When AWAKE = "0", the Auto-Wake circuitry is disabled. This bit only has meaning when the 10BASE-T network interface is selected.

This bit is reset to ZERO by H_RESET and is unaffected by S_RESET or STOP.

1 ASEL Auto Select. When set, the PCnet-32 controller will automatically select the operating media interface port, unless the user has selected GPSI mode through appropriate programming of the PORTSEL bits of the Mode Register (CSR15). If GPSI mode has not been selected and ASEL has been set to a ONE, then when the 10BASE-T transceiver is in the link pass state (due to receiving valid frame data and/or Link Test pulses or the DLNKTST bit is set), the 10BASE-T port will be used. If GPSI mode has not been selected and ASEL has been set to a ONE, then when the 10BASE-T port is port will be used. Switching between the ports will not occur during transmission, to avoid any type of fragment generation.

When ASEL is set to ONE, Link Beat Pulses will be transmitted on the 10BASE-T port, regardless of the state of Link Status. When ASEL is reset to ZERO, Link Beat Pulses will only be transmitted on the 10BASE-T port when the PORTSEL bits of the Mode Register (CSR15) have selected 10BASE-T as the active port.

When ASEL is set to a ZERO, then the selected network port will be determined by the settings of the PORTSEL bits of CSR15.

The ASEL bit is reset to ONE by H_RESET and is unaffected by S_RESET or STOP.

The network port configuration are as follows:

PORTSEL(1:0)	ASEL (BCR2[1])	Link Status (of 10BASE-T)	Network Port
0X	1	0	AUI
0X	1	1	10BASE-T
0 0	0	X	AUI
0 1	0	X	10BASE-T
1 0	X	X	GPSI
1 1	X	X	Reserved

0 RES Reserved location. The default value of this bit is a ZERO. Writing a ONE to this bit has no effect on device function. Existing drivers may write a ONE to this bit, but new drivers should write a ZERO to this bit.

BCR4: Link Status LED

Bit Name Description

BCR4 controls the function(s) that the $\overline{\text{LNKST}}$ pin displays. Multiple functions can be simultaneously enabled on this LED pin. The LED display will indicate the logical OR of the enabled functions. BCR4 defaults to Link Status (LNKST) with pulse stretcher enabled (PSE = 1) and is fully programmable.

The default setting after H_RESET for the $\overline{\text{LNKST}}$ register is 00C0h. The $\overline{\text{LNKST}}$ register value is unaffected by S_RESET or STOP.

31-16 RES Reserved locations. Written as zeros and read as undefined.

15 LEDOUT This bit indicates the current (non-stretched) value of the LED output pin. A value of ONE in this bit indicates that the OR of the enabled signals is true.

The logical value of the LEDOUT status signal is determined by the settings of the individual Status Enable bits of the LED register (Bits 6-0).

This bit is READ only by the host, and is unaffected by H_RESET or S_RESET or STOP.

This bit is valid only if the network link status is PASS.

14	LEDPOL	LED Polarity. When this bit has the value ZERO, then the LED pin will be driven to a LOW level whenever the OR of the enabled signals is true and the LED pin will be floated and allowed to float high whenever the OR of the enabled signals is false (i.e. the LED output will be an Open Drain output and the output value will be the inverse of the LEDOUT status bit). When this bit has the value ONE, then the LED pin will be driven to a HIGH level whenever the OR of the enabled signals is true and the LED pin will be driven to a LOW level whenever the OR of the enabled signals is false (i.e. the LED output will be a Totem Pole output and the output value will be the same polarity as the LEDOUT status bit). The setting of this bit will not affect the polarity of the LEDOUT bit for this register.
13	LEDDIS	LED Disable. This bit is used to disable the LED output. When LEDDIS has the value ONE, then the LED output will always be floated. When LEDDIS has the value ZERO, then the LED output value will be governed by the LEDOUT and LEDPOL values.
12-8	RES	Reserved locations. Written as ZEROs, read as undefined.
7	PSE	Pulse Stretcher Enable. Extends the LED illumination time for each new occurrence of the enabled function for this LED output. A value of 0 disables the function. A value of 1 enables the function.
6	LNKSTE	Link Status Enable. Indicates the current link status on the Twisted Pair interface. When this bit is set, a value of ONE will be passed to the LEDOUT signal to indicate that the link status state is PASS. A value of ZERO will be passed to the LEDOUT signal to indicate that the link status state is FAIL. A value of 0 disables the signal. A value of 1 enables the signal.
5	RCVME	Receive Match status Enable. Indicates receive activity on the network that has passed the address match function for this node. All address matching modes are included: Physical, Logical filtering, Promiscuous, Broadcast, and EADI. A value of 0 disables the signal. A value of 1 enables the signal.
4	XMTE	Transmit status Enable. Indicates PCnet-32 controller transmit activity. A value of 0 disables the signal. A value of 1 enables the signal.
3	RXPOLE	Receive Polarity status Enable. Indicates the current Receive Polarity condition on the Twisted Pair interface. A value of ONE indicates that the polarity of the RXD± pair has been reversed. A value of ZERO indicates that the polarity of the RXD± pair has not been reversed. Receive polarity indication is valid only if the LNKST bit of BCR4 indicates link PASS status. A value of 0 disables the signal. A value of 1 enables the signal.
2	RCVE	Receive status Enable. Indicates receive activity on the network. A value of 0 disables the signal. A value of 1 enables the signal.
1	JABE	Jabber status Enable. Indicates that the PCnet-32 controller is jabbering on the network. A value of 0 disables the signal. A value of 1 enables the signal.
0	COLE	Collision status Enable. Indicates collision activity on the network. When the AUI port is selected, collision activity during the 4.0 ms internal following a transmit completion (SQE internal) will not activate the LEDOUT bit. A value of 0 disables the signal. A value of 1 enables the signal.

BCR5: LED1 Status

Bit	Name	Description
		BCR5 controls the function(s) that the LED1 pin displays. Multiple functions can be simultaneously enabled on this LED pin. The LED display will indicate the logical OR of

the enabled functions. BCR5 defaults to Receive Status (RCV) with pulse stretcher enabled (PSE = 1) and is fully programmable.

The default setting after H_RESET for the LED1 register is 0084h. The LED1 register value is unaffected by S_RESET or STOP.

31-16 RES — Reserved locations. Written as zeros and read as undefined.

15 LEDOUT — This bit indicates the current (non-stretched) value of the LED output pin. A value of ONE in this bit indicates that the OR of the enabled signals is true.

The logical value of the LEDOUT status signal is determined by the settings of the individual Status Enable bits of the LED register (Bits 6-0).

This bit is READ only by the host, and is unaffected by H_RESET S_RESET or STOP.

This bit is valid only if the network link status is PASS.

14 LEDPOL — LED Polarity. When this bit has the value ZERO, then the LED pin will be driven to a LOW level whenever the OR of the enabled signals is true and the LED pin will be floated and allowed to float high whenever the OR of the enabled signals is false (i.e. the LED output will be an Open Drain output and the output value will be the inverse of the LEDOUT status bit).

When this bit has the value ONE, then the LED pin will be driven to a HIGH level whenever the OR of the enabled signals is true and the LED pin will be driven to a LOW level whenever the OR of the enabled signals is false (i.e. the LED output will be a Totem Pole output and the output value will be the same polarity as the LEDOUT status bit).

The setting of this bit will not affect the polarity of the LEDOUT bit for this register.

13 LEDDIS — LED Disable. This bit is used to disable the LED output. When LEDDIS has the value ONE, then the LED output will always be floated. When LEDDIS has the value ZERO, then the LED output value will be governed by the LEDOUT and LEDPOL values.

12-8 RES — Reserved locations. Write as ZEROs, read as undefined.

7 PSE — Pulse Stretcher Enable. Extends the LED illumination time for each new occurrence of the enabled function for this LED output.

A value of 0 disables the function. A value of 1 enables the function.

6 LNKSTE — Link Status Enable. Indicates the current link status on the Twisted Pair interface. When this bit is set, a value of ONE will be passed to the LEDOUT signal to indicate that the link status state is PASS. A value of ZERO will be passed to the LEDOUT signal to indicate that the link status state is FAIL.

A value of 0 disables the signal. A value of 1 enables the signal.

5 RCVME — Receive Match status Enable. Indicates receive activity on the network that has passed the address match function for this node. All address matching modes are included: Physical, Logical filtering, Promiscuous, Broadcast, and EADI.

A value of 0 disables the signal. A value of 1 enables the signal.

4 XMTE — Transmit status Enable. Indicates PCnet-32 controller transmit activity.

A value of 0 disables the signal. A value of 1 enables the signal.

3 RXPOLE — Receive Polarity status Enable. Indicates the current Receive Polarity condition on the Twisted Pair interface. A value of ONE indicates that the polarity of the RXD± pair has been reversed. A value of ZERO indicates that the polarity of the RXD± pair has not been reversed.

Receive polarity indication is valid only if the LNKST bit of BCR4 indicates link PASS status.

A value of 0 disables the signal. A value of 1 enables the signal.

2	RCVE	Receive status Enable. Indicates receive activity on the network. A value of 0 disables the signal. A value of 1 enables the signal.
1	JABE	Jabber status Enable. Indicates that the PCnet-32 controller is jabbering on the network. A value of 0 disables the signal. A value of 1 enables the signal.
0	COLE	Collision status Enable. Indicates collision activity on the network. When the AUI port is selected, collision activity during the 4.0 ms internal following a transmit completion (SQE internal) will not activate the LEDOUT bit. A value of 0 disables the signal. A value of 1 enables the signal.

BCR6: LED2 Status

Bit	Name	Description
		BCR6 controls the function(s) that the LED2 pin displays. Multiple functions can be simultaneously enabled on this LED pin. The LED display will indicate the logical OR of the enabled functions. BCR6 defaults to twisted pair MAU Receive Polarity (RCVPOL) with pulse stretcher enabled (PSE = 1) and is fully programmable. The default setting after H_RESET for the LED2 register is 0088h. The LED2 register value is unaffected by S_RESET or STOP.
31-16	RES	Reserved locations. Written as zeros and read as undefined.
15	LEDOUT	This bit indicates the current (non-stretched) value of the LED output pin. A value of ONE in this bit indicates that the OR of the enabled signals is true. The logical value of the LEDOUT status signal is determined by the settings of the individual Status Enable bits of the LED register (Bits 11-8 and 5-0). This bit is READ only by the host, and is unaffected by H_RESET S_RESET or STOP. This bit is valid only if the network link status is PASS.
14	LEDPOL	LED Polarity. When this bit has the value ZERO, then the LED pin will be driven to a LOW level whenever the OR of the enabled signals is true and the LED pin will be floated and allowed to float high whenever the OR of the enabled signals is false (i.e. the LED output will be an Open Drain output and the output value will be the inverse of the LEDOUT status bit). When this bit has the value ONE, then the LED pin will be driven to a HIGH level whenever the OR of the enabled signals is true and the LED pin will be driven to a LOW level whenever the OR of the enabled signals is false (i.e. the LED output will be a Totem Pole output and the output value will be the same polarity as the LEDOUT status bit). The setting of this bit will not affect the polarity of the LEDOUT bit for this register.
13	LEDDIS	LED Disable. This bit is used to disable the LED output. When LEDDIS has the value ONE, then the LED output will always be floated. When LEDDIS has the value ZERO, then the LED output value will be governed by the LEDOUT and LEDPOL values.
12-8	RES	Reserved locations. Write as ZEROs, read as undefined.
7	PSE	Pulse Stretcher Enable. Extends the LED illumination time for each new occurrence of the enabled function for this LED output. A value of 0 disables the function. A value of 1 enables the function.
6	LNKSTE	Link Status Enable. Indicates the current link status on the Twisted Pair interface. When this bit is set, a value of ONE will be passed to the LEDOUT signal to indicate that the link status state is PASS. A value of ZERO will be passed to the LEDOUT signal to indicate that the link status state is FAIL. A value of 0 disables the signal. A value of 1 enables the signal.
5	RCVME	Receive Match status Enable. Indicates receive activity on the

network that has passed the address match function for this node. All address matching modes are included: Physical, Logical filtering, Promiscuous, Broadcast, and EADI.

A value of 0 disables the signal. A value of 1 enables the signal.

4 XMTE Transmit status Enable. Indicates PCnet-32 controller transmit activity.

A value of 0 disables the signal. A value of 1 enables the signal.

3 RXPOLE Receive Polarity status Enable. Indicates the current Receive Polarity condition on the Twisted

Pair interface. A value of ONE indicates that the polarity of the RXD± pair has been reversed. A value of ZERO indicates that the polarity of the RXD± pair has not been reversed.

Receive polarity indication is valid only if the LNKST bit of BCR4 indicates link PASS status.

A value of 0 disables the signal. A value of 1 enables the signal.

2 RCVE Receive status Enable. Indicates receive activity on the network.

A value of 0 disables the signal. A value of 1 enables the signal.

1 JABE Jabber status Enable. Indicates that the PCnet-32 controller is jabbering on the network.

A value of 0 disables the signal. A value of 1 enables the signal.

0 COLE Collision status Enable. Indicates collision activity on the network. When the AUI port is selected, collision activity during the 4.0 ms internal following a transmit completion (SQE internal) will not activate the LEDOUT bit.

A value of 0 disables the signal. A value of 1 enables the signal.

BCR7: LED3 Status

Bit Name Description

BCR7 controls the function(s) that the LEDPRE3 pin displays. Multiple functions can be simultaneously enabled on this LED pin. The LED display will indicate the logical OR of the enabled functions. BCR7 defaults to Transmit Status (XMT) with pulse stretcher enabled (PSE = 1) and is fully programmable.

The default setting after H_RESET for the LED3 register is 0090h. The LED3 register value is unaffected by S_RESET or STOP.

31-16 RES Reserved locations. Written as zeros and read as undefined.

15 LEDOUT This bit indicates the current (non-stretched) value of the LED output pin. A value of ONE in this bit indicates that the OR of the enabled signals is true.

The logical value of the LEDOUT status signal is determined by the settings of the individual Status Enable bits of the LED register (Bits 11-8 and 5-0).

This bit is READ only by the host, and is unaffected by H_RESET S_RESET or STOP.

This bit is valid only if the network link status is PASS.

14 LEDPOL LED Polarity. When this bit has the value ZERO, then the LED pin will be driven to a LOW level whenever the OR of the enabled signals is true and the LED pin will be floated and allowed to float high whenever the OR of the enabled signals is false (i.e. the LED output will be an Open Drain output and the output value will be the inverse of the LEDOUT status bit).

When this bit has the value ONE, then the LED pin will be driven to a HIGH level whenever the OR of the enabled signals is true and the LED pin will be driven to a LOW level whenever the OR of the enabled signals is false (i.e. the LED output will be a Totem Pole output and the output value will be the same polarity as the LEDOUT status bit).

The setting of this bit will not affect the polarity of the LEDOUT bit for this register.

13 LEDDIS LED Disable. This bit is used to disable the LED output. When LEDDIS has the value ONE, then the LED output will always be

floated. When LEDDIS has the value ZERO, then the LED output value will be governed by the LEDOUT and LEDPOL values.

12-8	RES	Reserved locations. Write as ZEROs, read as undefined.
7	PSE	Pulse Stretcher Enable. Extends the LED illumination time for each new occurrence of the enabled function for this LED output. A value of 0 disables the function. A value of 1 enables the function.
6	LNKSTE	Link Status Enable. Indicates the current link status on the Twisted Pair interface. When this bit is set, a value of ONE will be passed to the LEDOUT signal to indicate that the link status state is PASS. A value of ZERO will be passed to the LEDOUT signal to indicate that the link status state is FAIL. A value of 0 disables the signal. A value of 1 enables the signal.
5	RCVME	Receive Match status Enable. Indicates receive activity on the network that has passed the address match function for this node. All address matching modes are included: Physical, Logical filtering, Promiscuous, Broadcast, and EADI. A value of 0 disables the signal. A value of 1 enables the signal.
4	XMTE	Transmit status Enable. Indicates PCnet-32 controller transmit activity. A value of 0 disables the signal. A value of 1 enables the signal.
3	RXPOLE	Receive Polarity status Enable. Indicates the current Receive Polarity condition on the Twisted Pair interface. A value of ONE indicates that the polarity of the RXD± pair has been reversed. A value of ZERO indicates that the polarity of the RXD± pair has not been reversed. Receive polarity indication is valid only if the LNKST bit of BCR4 indicates link PASS status. A value of 0 disables the signal. A value of 1 enables the signal.
2	RCVE	Receive status Enable. Indicates receive activity on the network. A value of 0 disables the signal. A value of 1 enables the signal.
1	JABE	Jabber status Enable. Indicates that the PCnet-32 controller is jabbering on the network. A value of 0 disables the signal. A value of 1 enables the signal.
0	COLE	Collision status Enable. Indicates collision activity on the network. When the AUI port is selected, collision activity during the 4.0 ms internal following a transmit completion (SQE internal) will not activate the LEDOUT bit. A value of 0 disables the signal. A value of 1 enables the signal.

BCR16: I/O Base Address Lower

Bit	Name	Description
		Note that all bits in this register are programmable through the EEPROM PREAD operation and through the Software Relocatable Mode operation.
31-16	RES	Reserved locations. Written as zeros and read as undefined.
15-5	IOBASEL	I/O Base Address Lower 16 bits. These bits are used to determine the location of the PCnet-32 controller in all of I/O space. They function as bits 15 through 5 of the I/O address of the PCnet-32 controller. Note that the lowest five bits of the PCnet-32 controller I/O space are not programmable. These bits are assumed to be ZEROs. This means that it is not possible to locate the PCnet-32 controller on a space that does not begin on a 32-byte block boundary. The value of IOBASEL is determined in either of two ways: 1. The ISOBASEL value may be set during the EEPROM read. 2. If no EEPROM exists, or if there is an error detected in the EEPROM data, then the PCnet-32 controller will enter Software Relocatable Mode, and a specific sequence of write accesses to I/O address 378h will cause the

IOBASEL value to be updated. Refer to the Software Relocatable Mode section of this document for more details.

A direct write access to the I/O Base Address Lower register may be performed.

IOBASEL is not affected by S_RESET or STOP.

4-0	RES	Reserved locations. Written as ZEROs, read as undefined.

BCR17: I/O Base Address Upper

Bit	Name	Description
		Note that all bits in this register are programmable through the EEPROM PREAD operation and software relocatable mode.
31-16	RES	Reserved locations. Written as zeros and read as undefined.
15-0	IOBASEU	I/O Base Address Upper 16 bits. These bits are used to determine the location of the PCnet-32 controller in all of I/O space. They function as bits 31 through 16 of the I/O address of the PCnet-32 controller. The value of IOBASEU is determined in either of two ways: 1. The IOBASEU value may be set during the EEPROM read. 2. If no EEPROM exists, or if there is an error detected in the EEPROM data, then the PCnet-32 controller will enter Software Relocatable Mode, and a specific sequence of write accesses to I/O address 378h will cause the IOBASEU value to be updated. Refer to the Software Relocatable Mode section of this document for more details. A direct write access to the I/O Base Address Upper register may be performed. IOBASEU is not affected by S_RESET or STOP.

BCR18: Burst Size and Bus Control

Bit	Name	Description
		Note that all bits in this register are programmable through the EEPROM PREAD operation.
31-16	RES	Reserved locations. Written as zeros and read as undefined.
15-11	CLL	Cache Line Length. These bits are used to determine how often a cache invalidation cycle needs to be performed when Generate Cache Invalidation Cycles has been activated by setting the GCIC bit (bit 10 of BCR18) to a one. CLL values are interpreted as multiples of 4 bytes. For example, a CLL value of 00001b means the cache line length is 4 bytes and a cache invalidation cycle (assertion of $\overline{\text{EADS}}$) will be performed every 4 bytes. A CLL value of 00010b means the cache line length is 8 bytes, and a cache invalidation cycle will be performed every 8 bytes. A CLL value of 00100 means the cache line length is 16 bytes. A value of 00000 means that the cache line size is “infinite”. In other words, a single $\overline{\text{EADS}}$ assertion will be performed on the first access at the beginning of each bus mastership period (write accesses only) and no subsequent $\overline{\text{EADS}}$ assertions will be made during this bus mastership period. Cache invalidation cycles are performed only during PCnet-32 controller bus master write accesses. Some CLL values are reserved (see chart below). The portion of the Address Bus that will be floated at the time of an address hold operation (AHOLD asserted) will be determined by the value of the Cache Line Length register. The following chart lists all of the legal values of CLL showing the portion of the Address Bus that will become floated during an address hold operation:

CLL Value	Portion of Address Bus Floated During AHOLD
00000	None
00001	A31-A2
00010	A31-A3
00011	Reserved CLL Value
00100	A31-A4
00101–00111	Reserved CLL Values
01000	A31-A5
01001 –01111	Reserved CLL Values
10000	A31-A6
10001– 11111	Reserved CLL Values

Note that the default value of CLL after RESET is 00100b. All timing diagrams in this document are drawn with the assumption that this is the value of CLL.

When VESA VL-Bus mode is selected, then the $\overline{\text{LEADS}}$ pin functions as if CLL = 00001b regardless of the actual CLL setting.

CLL is set to 00100b by H_RESET and is not affected by S_RESET or STOP.

10 GCIC/IWBACK Generate Cache Invalidation Cycles. Ignore $\overline{\text{WBACK}}$ signal.

GCIC Generate Cache Invalidation Cycles. When this bit is set and the Am486 mode has been selected, then the PCnet-32 controller assumes that the host system contains a cache, but the host system does not snoop the local bus master accesses of the PCnet-32 controller, and therefore, that cache invalidation cycles must be generated by the PCnet-32 controller for PCnet-32 controller master accesses. PCnet-32 controller will not perform snoops to invalidate cache lines for local bus accesses that other local bus masters may have executed. When GCIC is a ZERO, the PCnet-32 controller assumes that logic in the host system will create the cache invalidation cycles that may be necessary as a result of PCnet-32 controller local bus master accesses. The cache invalidation logic performs its functions according to the GCIC and CLL bits, regardless of the setting of the $\overline{\text{Am486}}$/Am386 pin.

IWBACK Ignore $\overline{\text{WBACK}}$ signal. When this bit is set and the VESA VL-Bus Mode has been selected, then the PCnet-32 device will operate as though the $\overline{\text{WBACK}}$ input pin is always held HIGH, even though the real value of the $\overline{\text{WBACK}}$ input may change. This function is provided to allow the PCnet-32 device to operate in systems which violate VESA VL-Bus $\overline{\text{WBACK}}$ operation by either floating this pin or by always driving this pin LOW.

$\overline{\text{LEADS}}$ is always asserted with each assertion of $\overline{\text{ADS}}$ when VESA VL-Bus mode has been selected, regardless of the setting of the GCIC/IWBACK bit.

GCIC/IWBACK is cleared by H_RESET and is not affected by S_RESET or STOP.

9 PRPCNET Priority PCnet. This bit is used to set the priority of the daisy chain arbitration logic that resides within the PCnet-32 controller. When PRPCNET = 1, the priority of the daisy chain logic is set to PCnet-32 controller highest priority, External device lowest priority. When PRPCNET = 0, the priority of the daisy chain logic is set to External device highest priority, PCnet-32 controller lowest priority. In the case PRPCNET = 0, where the PCnet-32 controller has lower priority than the external device and the PCnet-32 controller is preempted due to a HOLDI request from the external device, the PCnet-32 controller will complete the current sequence of accesses and then pass the HLDA to the external device by asserting the HLDAO signal. The HOLD output signal from the PCnet-32 controller will not change state during the HLDAO hand-off. If the PCnet-32 controller is performing a linear burst, then the PCnet-32 controller will complete the linear burst and then pass the HLDA to the external device through the HLDAO signal. For more details on exact timing of preemption events, see the

individual descriptions of the various DMA transfers.

PRPCNET is cleared by H_RESET and is not affected by S_RESET or STOP.

The default setting for this bit will be PRPCNET = 0. This default value reflects the nature of the CPU's handling of a HOLD request (i.e. the CPU has lowest priority). By making this the default setting, the PCnet-32 controller response to HOLDI is as close as possible to the timing of the CPU response to HOLD, so that minimal design difficulty will be created by inserting the PCnet-32 controller into the system as the mediating device between the CPU and the extension bus chipset.

8 RES — Reserved bit. Must be written as a ONE. Will be read as a ONE.

This reserved location is SET by H_RESET and is not affected by S_RESET or STOP.

7 DWIO — Double Word I/O. When set, this bit indicates that the PCnet-32 controller is programmed for DWIO mode. When cleared, this bit indicates that the PCnet-32 controller is programmed for Word I/O mode. This bit affects the I/O Resource Offset map and it affects the defined width of the PCnet-32 controller's I/O resources. See the DWIO and WIO sections for more details.

The PCnet-32 controller will set DWIO if it detects a double-word *write* access to offset 10h from the PCnet-32 controller I/O Base Address (corresponding to the RDP resource). A double word write access to offset 10h is the only way that the DWIO bit can be set. DWIO cannot be set by a direct write to BCR18.

Once the DWIO bit has been set to a ONE, only a H_RESET can reset it to a ZERO.

DWIO is ***read only*** by the host.

DWIO is cleared by H_RESET and is not affected by S_RESET or STOP.

6 BREADE — Burst Read Enable. When set, this bit enables Linear Bursting during memory read accesses, where Linear Bursting is defined to mean that only the first transfer in the current bus arbitration will contain an address cycle. Subsequent transfers will consist of data only. However, the entire address bus will still be driven with appropriate values during the subsequent cycles, but $\overline{ADS}$ will not be asserted. When cleared, this bit prevents the part from performing linear bursting during read accesses. In no case will the part linearly burst a descriptor access or an initialization access.

BREADE is cleared by H_RESET and is not affected by S_RESET or STOP.

Burst Read activity is not allowed when the BCLK frequency is >33 MHz. Linear bursting is disabled in VL-Bus systems that operate above this frequency by connecting the VLBEN pin to either ID(3) (for VL-Bus version 1.0 systems) or ID(4) AND ID(3) AND ID(1) AND ID(0) (for VL-Bus version 1.1 or 2.0 systems). In Am486-style systems that have BCLK frequencies above 33 MHz, disabling the linear burst capability is ideally carried out through EEPROM bit programming, since the EEPROM programming can be setup for a particular machine's architecture. When the VLBEN pin has been reset to a ZERO, then the BREADE bit will be forced to a value of ZERO. Any attempt to change this value by writing to the BREADE bit location will have no effect.

5 BWRITE — Burst Write Enable. When set, this bit enables Linear Bursting during memory write accesses, where Linear Bursting is defined to mean that only the first transfer in the current bus arbitration will contain an address cycle. Subsequent transfers will consist of data only. However, the entire address bus will still be driven with appropriate values during the subsequent cycles, but $\overline{ADS}$ will not be asserted. When cleared, this bit prevents the

part from performing linear bursting during write accesses. In no case will the part linearly burst a descriptor access or an initialization access.

BWRITE is cleared by H_RESET and is not affected by S_RESET or STOP.

Burst Write activity is not allowed when the BCLK frequency is >33 MHz. Linear bursting is disabled in VL-Bus systems that operate above this frequency by connecting the VLBEN pin to either ID(3) (for VL-Bus version 1.0 systems) or ID(4) AND ID(3) AND ID(1) AND ID(0) (for VL-Bus version 1.1 or 2.0 systems). In Am486-style systems that have BCLK frequencies above 33 MHz, disabling the linear burst capability is ideally carried out through EEPROM bit programming, since the EEPROM programming can be setup for a particular machine's architecture. When the VLBEN pin has been reset to a ZERO, then the BWRITE bit will be forced to a value of ZERO. Any attempt to change this value by writing to the BWRITE bit location will have no effect.

4-3 TSTSHDW Test Shadow bits. These bits are used to place the PCnet-32 controller into GPSI mode. BCR18[3] must be set to ZERO. The operating modes possible are indicated in Table 47.

See Table 48 for pin reconfiguration in GPSI mode.

TSTSHDW Value (BCR18[4:3])	PVALID (BCR19[15])	GPSIEN (CSR124[4])	Operating Mode
00	X	0	Normal Operating Mode
10	1	X	GPSI Mode
01	1	0	Reserved
11	1	X	Reserved
XX	0	0	Normal Operating Mode
XX	0	1	GPSI Mode

Note that when the GPSI mode is invoked, only the lower 24 bits of the address bus are available. IOAW24 (BCR21[8]) must be set to allow slave operations. During master accesses in GPSI mode, the PCnet-32 controller will not drive the upper 8 bits of the address bus with address information.

These bits are not writable, regardless of the setting of the ENTST bit in CSR4. Values may only be programmed to these bits through the EEPROM read operation.

BCR18[4:3] are set to 0 by H_RESET and are unaffected by S_RESET or STOP.

2-0 LINBC[2:0] Linear Burst Count. The 3-bit value in this register sets the upper limit for the number of transfer cycles in a Linear Burst. This limit determines how often the PCnet-32 controller will assert the $\overline{ADS}$ signal during linear burst transfers. Each time that the interpreted value of LINBC transfers is reached, the PCnet-32 controller will assert the $\overline{ADS}$ signal with a new valid address. The LINBC value should contain only one active bit. LINBC values with more than one active bit may produce predictable results, but such values will not be compatible with future AMD network controllers. The LINBC entry is shifted by two bits before being used by the PCnet-32 controller. For example, the value LINBC[2:0] = 010 is understood by the PCnet-32 con-troller to mean 01000 = 8. Therefore, the value LINBC[2:0] = 010 will cause the PCnet-32 controller to issue a new $\overline{ADS}$ every 01000b = 8 transfers. The PCnet-32 controller may linearly burst fewer than the value represented by LINBC, due to other conditions that cause the burst to end prematurely. Therefore, LINBC should be regarded as an upper limit to the length of linear burst.

Table 48. GPSI Pin Configuration

GPSI Function	GPSI I/O Type	LANCE GPSI Pin	ILACC GPSI Pin	PCnet-32/ PCnet-ISA GPSI Pin	PCnet-32 Pin Number	PCnet-32 Normal Pin Function
Transmit Data	O	TX	TXD	TXDAT	132	A31
Transmit Enable	O	TENA	RTS	TXEN	133	A30
Transmit Clock	I	TCLK	TXC	STDCLK	134	A29
Collision	I	CLSN	CDT	CLSN	137	A28
Receive Carrier Sense	I	RENA	CRS	RXCRS	138	A27
Receive Clock	I	RCLK	RXC	SRDCLK	140	A26
Receive Data	I	RX	RXD	RXDAT	141	A25

Note that linear burst operation will only begin on certain addresses. The general rule for linear burst starting addresses is:

A[31:0] MOD (LINBC x 16) = 0.

Table 49 illustrates all possible starting address values for all legal LINBC values (only 1, 2, and 4 are legal; other values are reserved). Note that A[31:8] are don't care values for all addresses. (A[1:0] do not exist within a 32 bit system, however, they are valid bits within the buffer pointer field of descriptor word 0.)

Table 49. Linear Burst Cycles

LINBC[2:0]	LBS = Linear Burst Size (No. of Transfers)	Size of Burst (Byte)	Linear Burst Beginning Addresses (A[31:6] = Don't Care) A[5:0] =
1	4	16	00, 10, 20, 30
2	8	32	00, 20
4	16	64	00

Due to the beginning address restrictions just given, it can be shown that some portion of the address bus will be held stable throughout each linear burst sequence, while the lowest portion of the address bus will change value with each new cycle. The portion of the address bus that will be held stable during a linear burst access is given in Table 50.

Table 50. Linear Burst Address Bus

LINBC Value	Portion of Address Bus Stable During Linear Burst
001	A[31:4]
010	A[31:5]
100	A[31:6]

The assertion of $\overline{\text{RDYRTN}}$ in the place of $\overline{\text{BRDY}}$ within a linear burst cycle will cause the linear burst to be interrupted. In that case, the PCnet-32 controller will revert to ordinary two-cycle transfers, except that $\overline{\text{BLAST}}$ will remain de-asserted to show that linear bursting is being requested by the PCnet-32 controller. This situation is defined as interrupted linear burst cycles. If $\overline{\text{BRDY}}$ is sampled as asserted (without also sampling $\overline{\text{RDYRTN}}$ asserted during the same access) during interrupted linear burst cycles, then linear bursting will resume.

There are several events which may cause early termination of linear burst. Among those events are: no more data available for transfer in either a buffer or in the FIFO or if either the Cycle Register (CSR80) or the Bus Activity Timer Register (CSR82) times out. In any of these cases, the PCnet-32 controller will end the Linear Burst by asserting $\overline{\text{BLAST}}$ and then releasing the bus. A Partial Linear Burst may have been sent out before the assertion of $\overline{\text{BLAST}}$, where "Partial Linear Burst" refers to the case where the number of data words transferred between the last asserted $\overline{\text{ADS}}$ and the

assertion of $\overline{\text{BLAST}}$ is less than LINBC.

The value on the address bus will be updated with appropriate values every clock cycle during linear burst operations, even though $\overline{\text{ADS}}$ will not be asserted during every clock cycle.

Certain combinations of watermark programming and LINBC programming may create situations where no linear bursting is possible, or where the FIFO may be excessively read or excessively written. Such combinations are declared as illegal.

Combinations of watermark settings and LINBC settings must obey the following relationship:

watermark (in bytes) ³ LINBC (in bytes)

Combinations of watermark and LINBC settings that violate this rule may cause unexpected behavior.

LINBC is set to the value of 001 by H_RESET and is not affected by S_RESET or STOP. This gives a default linear burst length of 4 transfers = 001 x 4.

BCR19: EEPROM Control and Status

Bit	Name	Description
31-16	RES	Reserved locations. Written as zeros and read as undefined.
15	PVALID	EEPROM Valid status bit. This bit is read only by the host. A value of ONE in this bit indicates that a PREAD operation has occurred, and that 1) there is an EEPROM connected to the PCnet-32 controller microwire interface pins and 2) the contents read from the EEPROM have passed the checksum verification operation. A value of ZERO in this bit indicates that the contents of the EEPROM are different from the contents of the applicable PCnet-32 controller on-board registers and/or that the checksum for the entire 36 bytes of EEPROM is incorrect or that no EEPROM is connected to the microwire interface pins. PVALID is set to ZERO during H_RESET and is unaffected by S_RESET or the STOP bit. However, following the H_RESET operation, an automatic read of the EEPROM will be performed. Just as is true for the normal PREAD command, at the end of this automatic read operation, the PVALID bit may be set to ONE. Therefore, H_RESET will set the PVALID bit to ZERO at first, but the automatic EEPROM read operation may later set PVALID to a ONE. If PVALID becomes ZERO following an EEPROM read operation (either automatically generated after H_RESET, or requested through PREAD), then all EEPROM-programmable BCR locations will be reset to their H_RESET values. If no EEPROM is present at the EESK, EEDI and EEDO pins, then all attempted PREAD commands will terminate early and PVALID will NOT be set. This applies to the automatic read of the EEPROM after H_RESET as well as to host-initiated PREAD commands.
14	PREAD	EEPROM Read command bit. When this bit is set to a ONE by the host, the PVALID bit (BCR19[15]) will immediately be reset to a ZERO and then the PCnet-32 controller will perform a read operation of 36 bytes from the EEPROM through the microwire interface. The EEPROM data that is fetched during the read will be stored in the appropriate internal registers on board the PCnet-32 controller. Upon completion of the EEPROM read operation, the PCnet-32 controller will assert the PVALID bit. EEPROM contents will be indirectly accessible to the host through I/O read accesses to the Address PROM (offsets 0h through Fh) and through I/O read accesses to other EEPROM programmable registers. Note that I/O read accesses from these locations will not actually access the EEPROM itself, but instead will access the PCnet-32 controller's internal copy of the EEPROM contents. I/O write

accesses to these locations may change the PCnet-32 controller register contents, but the EEPROM locations will not be affected. EEPROM locations may be accessed directly through BCR19.

At the end of the read operation, the PREAD bit will automatically be reset to a ZERO by the PCnet-32 controller and PVALID will bet set, provided that an EEPROM existed on the microwire interface pins and that the checksum for the entire 36 bytes of EEPROM was correct.

Note that when PREAD is set to a ONE, then the PCnet-32 controller will no longer respond to I/O accesses directed toward it, until the PREAD operation has completed successfully.

If a PREAD command is given to the PCnet-32 controller but no EEPROM is attached to the microwire interface pins, then the PREAD command will terminate early, the PREAD bit will be cleared to a ZERO and the PVALID bit will remain reset with a value of ZERO. The PCnet-32 controller will then enter Software Relocatable Mode to await further programming. This applies to the automatic read of the EEPROM after H_RESET as well as to host initiated PREAD commands. EEPROM programmable locations on board the PCnet-32 controller will be set to their default values by such an aborted PREAD operation. For example, if the aborted PREAD operation immediately followed the H_RESET operation, then the final state of the EEPROM programmable locations will be equal to the H_RESET programming for those locations.

If a PREAD command is given to the PCnet-32 controller and the autodetection pin (EESK/LED1/SFBD) indicates that no EEPROM is present, then the EEPROM read operation will still be attempted.

Note that at the end of the H_RESET operation, a read of the EEPROM will be performed automatically. This H_RESET-generated EEPROM read function will not proceed if the auto-detection pin (EESK/LED1/SFBD) indicates that no EEPROM is present. Instead, Software Relocatable Mode will be entered immediately.

PREAD is set to ZERO during H_RESET and is unaffected by S_RESET or STOP.

PREAD is only writable when the STOP bit is set to ONE.

13 EEDET EEPROM Detect. This bit indicates the sampled value of the EESK/LED1/SFBD pin at the end of H_RESET. The value of this bit is independent whether or not an EEPROM is actually present at the EEPROM interface. It is only a function of the sampled value of the EESK/LEDI/SFBD pin at the end of H_RESET.

The value of this bit is determined at the end of the H_RESET operation. It is unaffected by S_RESET or STOP.

This bit is not writable. It is read only.

Table 51 indicates the possible combinations of EEDET and the existence of an EEPROM and the resulting operations that are possible on the EEPROM microwire interface.

12-5 RES Reserved locations. Written as ZERO, read as undefined.

4 EEN EEPROM port enable. When this bit is set to a one, it causes the values of EBUSY, ECS, ESK and EDI to be driven onto the SHFBUSY, EECS, EESK and EEDI pins, respectively. When this bit is reset to a zero, then the SHFBUSY pin will be driven with the inverse of the PVALID (bit 15 of BCR19) value. PVALID is set to “ONE” if the EEPROM read was successful. It is set to “ZERO” otherwise.

Table 51. EEDET Effects on EEPROM Operation

EEDET Value (BCR19[3])	EEPROM Connected?	Result if PREAD is set to ONE	Result of Automatic EEPROM Read Operation Following H_RESET
0	No	EEPROM read operation is attempted. Entire read sequence will occur, checksum failure will result, PVALID is reset to ZERO.	First TWO EESK clock cycles are generated, then EEPROM read operation is aborted and PVALID is reset to ZERO.
0	Yes	EEPROM read operation is attempted. Entire read sequence will occur, checksum operation will pass, PVALID is set to ONE.	First TWO EESK clock cycles are generated, then EEPROM read operation is aborted and PVALID is reset to ZERO.
1	No	EEPROM read operation is attempted. Entire read sequence will occur, checksum failure will result, PVALID is reset to ZERO.	EEPROM read operation is attempted. Entire read sequence will occur, checksum failure will result, PVALID is reset to ZERO.
1	Yes	EEPROM read operation is attempted. Entire read sequence will occur, checksum operation will pass, PVALID is set to ONE.	EEPROM read operation is attempted. Entire read sequence will occur, checksum operation will pass, PVALID is set to ONE.

If EEN = 0 and no EEPROM read function is currently active, then EECS will be driven LOW. When EEN = 0 and no EEPROM read function is currently active, EESK and EEDI pins will be driven by the LED registers BCR5 and BCR4, respectively. See Table 52.

EEN is set to ZERO by H_RESET and is unaffected by S_RESET or STOP.

3 EBUSY EEPROM BUSY. This bit controls the value of the SHFBUSY pin of the PCnet-32 controller when the EEN bit is set to ONE and the PREAD bit is set to ZERO. This bit is used to indicate to external EEPROM-programmable logic that an EEPROM access is occurring.

When user programming of the EEPROM is desired through the BCR19 EEPROM Port, then EBUSY should be set to ONE before EEN is set to ONE in systems where EEPROM-programmable external logic exists. At the end of the EEPROM programming operation, EBUSY should either remain set at ONE until after EEN is set to ZERO, or the user may reset EBUSY to ZERO with EEN = 1 immediately following a read of EEPROM byte locations 35 and 36, which should be the last accesses performed during BCR19 accesses to the EEPROM. A programmed PREAD operation following the BCR19 EEPROM programming accesses will cause the SHFBUSY pin to become LOW if the EEPROM checksum is verified.

EBUSY has no effect on the output value of the SHFBUSY pin unless the PREAD bit is set to ZERO and the EEN bit is set to ONE.

EBUSY is set to ZERO by H_RESET and is unaffected by S_RESET or STOP.

2 ECS EEPROM Chip Select. This bit is used to control the value of the EECS pin of the microwire interface when the EEN bit is set to ONE and the PREAD bit is set to ZERO. If EEN = “1” and PREAD = “0” and ECS is set to a ONE, then the EECS pin will be forced to a HIGH level at the rising edge of the next BCLK following bit programming. If EEN = “1” and PREAD = “0” and ECS is set to a ZERO, then the EECS pin will be forced to a LOW level at the rising edge of the next BCLK following bit programming.

ECS has no effect on the output value of the EECS pin unless the PREAD bit is set to ZERO and the EEN bit is set to ONE.

ECS is set to ZERO by H_RESET and is unaffected by S_RESET or STOP.

1 ESK EEPROM Serial Clock. This bit and the EDI/EDO bit are used to control host access to the EEPROM. Values programmed to this bit are placed onto the EESK pin at the rising edge of the next BCLK following bit programming, except when the PREAD bit is set to ONE or the EEN bit is set to ZERO. If both the ESK bit and the EDI/EDO bit values are changed during one BCR19 write operation, while EEN = 1, then setup and hold times of the EEDI pin value with respect to the EESK signal edge are not guaranteed.

Table 52. EEPROM Enable

Reset Pin	PREAD or Auto Read in Progress	EEN	EECS	SHFBUSY	EESK	EEDI
High	X	X	0	1	Z	Z
Low	1	X	Active	1	Active	Active
Low	0	1	From ECS Bit of BCR19	From EBUSY Bit of BCR19	From ESK Bit of BCR19	From EEDI Bit of BCR19
Low	0	0	0	$\overline{\text{PVALID}}$	LED1	$\overline{\text{LNKST}}$

ESK has no effect on the EESK pin unless the PREAD bit is set to ZERO and the EEN bit is set to ONE.

ESK is reset to ONE by H_RESET and is unaffected by S_RESET or STOP.

0 EDI/EDO EEPROM Data In/EEPROM Data Out. Data that is written to this bit will appear on the EEDI output of the microwire interface, except when the PREAD bit is set to ONE or the EEN bit is set to ZERO. Data that is read from this bit reflects the value of the EEDO input of the microwire interface.

EDI/EDO has no effect on the EEDI pin unless the PREAD bit is set to ZERO and the EEN bit is set to ONE.

EDI/EDO is reset to ZERO by H_RESET and is unaffected by S_RESET or STOP.

BCR20: Software Style

Bit Name Description

This register is an alias of the location CSR58. Accesses to/from this register are equivalent to accesses to CSR58.

31-10 RES Reserved locations. Written as ZEROs and read as undefined.

9 CSRPCNET CSR PCnet-ISA configuration bit. When set, this bit indicates that the PCnet-32 controller register bits of CSR4 and CSR3 will map directly to the CSR4 and CSR3 bits of the PCnet-ISA (Am79C960) device. When cleared, this bit indicates that PCnet-32 controller register bits of CSR4 and CSR3 will map directly to the CSR4 and CSR3 bits of the ILACC (Am79C900) device.

The value of CSRPCNET is determined by the PCnet-32 controller. CSRPCNET is read only by the host.

The PCnet-32 controller uses the setting of the Software Style register (BCR20[7:0]) to determine the value for this bit.

CSRPCNET is set by H_RESET and is not affected by S_RESET or STOP.

8 SSIZE32 Software Size 32 bits. When set, this bit indicates that the PCnet-32 controller utilizes AMD79C900 (ILACC) software structures. In particular, Initialization Block and Transmit and Receive descriptor bit maps are affected. When cleared, this bit indicates that the PCnet-32 controller utilizes AMD PCnet-ISA software structures. ***Note:*** *Regardless of the setting of SSIZE32, the Initialization Block*

must always begin on a double-word boundary.

The value of SSIZE32 is determined by the PCnet-32 controller. SSIZE32 is read only by the host.

The PCnet-32 controller uses the setting of the Software Style register(BCR20[7:0]/CSR58[7:0]) to determine the value for this bit. SSIZE32 is cleared by H_RESET and is not affected by S_RESET or STOP.

If SSIZE32 is reset, then bits IADR[31:24] of CSR2 will be used to generate values for the upper 8 bits of the 32 bit address bus during master accesses initiated by the PCnet-32 controller. This action is required, since the 16-bit software structures specified by the SSIZE32=0 setting will yield only 24 bits of address for PCnet-32 controller bus master accesses.

If SSIZE32 is set, then the software structures that are common to the PCnet-32 controller and the host system will supply a full 32 bits for each address pointer that is needed by the PCnet-32 controller for performing master accesses.

The value of the SSIZE32 bit has no effect on the drive of the upper 8 address pins. The upper 8 address pins are always driven, regardless of the state of the SSIZE32 bit.

Note:*The setting of the SSIZE32 bit has no effect on the defined width for I/O resources.I/O resource width is determined by the state of the DWIO bit.*

7-0 SWSTYLE Software Style register. The value in this register determines the “style” of I/O and memory resources that shall be used by the PCnet-32 controller. The Software Style selection will affect the interpretation of a few bits within the CSR space and the width of the descriptors and initialization block. See Table 53.

All PCnet-32 controller CSR bits and BCR bits and all descriptor, buffer and initialization block entries not cited in the table above are unaffected by the Software Style selection and are therefore always fully functional as specified in the CSR and BCR and descriptor sections.

Read/write accessible only when the STOP bit is set.

The SWSTLYE register will contain the value 00h following H_RESET and is not affected by S_RESET or STOP.

BCR21: Interrupt Control

Bit Name Description

Note that all bits in this register are programmable through the EEPROM PREAD operation and software relocatable mode.

31-9 RES Reserved locations. Written as ZEROs and read as undefined.

8 IOAW24 I/O Address Width 24 bits. When set to a ONE, the IOAW24 bit will cause the PCnet-32 controller to ignore the upper 8 bits of the address bus when determining whether an I/O address matches PCnet-32 controller I/O space. When IOAW24 is set to a ZERO, then the PCnet-32 controller will examine all 32 bits of the address bus when determining whether an I/O address matches PCnet-32 controller I/O space.

Read/write accessible only when the STOP bit is set.

The IOAW24 bit will be reset to ZERO by H_RESET and is not affected by S_RESET or STOP.

7 REJECTDIS Reject Disable. When set to a ONE, the REJECTDIS bit will cause the $\overline{\text{EAR}}$ function of the EADI interface to be disabled. Specifically, the INTR2 pin will retain its function as INTR2 and will not function as $\overline{\text{EAR}}$, regardless of the setting of the BCR2 EADISEL bit. When reset to a ZERO, the REJECTDIS bit will allow the INTR2 pin to be redefined to function as $\overline{\text{EAR}}$ of the EADI interface when the EADISEL bit of BCR2 has been set to a ONE.

Read/write accessible only when STOP bit is set.

The REJECTDIS bit will be reset to ZERO by H_RESET and is not affected by S_RESET or STOP.

6-2 RES Reserved locations. Written as ZEROs and read as undefined.

1-0 INTSEL Interrupt Select. The value of these bits determines which of the interrupt pins will be the active interrupt. Table 55 indicates which interrupt will be selected for each combination of INTSEL and JTAGSEL values.

Interrupt pins that are not selected will be floated.

The INTSEL register will contain the value 00 following H_RESET and is not affected by S_RESET or STOP.

Table 53. Software Style Selection

SWSTYLE[7:0] (Hex)	Style Name	CSRPCNET	SSIZE32	Altered Bit Interpretations
00	LANCE/ PCnet-ISA	1	0	ALL CSR4 bits will function as defined in the CSR4 section. TMD1[29] functions as ADD_FCS
01	ILACC	0	1	CSR4[9:8], CSR4[5:4] and CSR4[1:0] will have *no function*, but will be writeable and readable. CSR4[15:10], CSR4[7:6] and CSR4[3:2] will function as defined in the CSR4 section. TMD1[29] becomes NO_FCS.
02	PCnet-32	1	1	ALL CSR4 bits will function as defined in the CSR4 section. TMD1[29] functions as ADD_FCS
All other combinations	Reserved			Undefined

Table 54. Interrupt Select

INTSEL[1:0]	JTAGSEL	Selected interrupt Pin
00	0	INTR1
01	0	INTR2
10	0	INTR3
11	0	INTR4
00	1	INTR1
01	1	INTR2
10	1	INTR1
11	1	INTR2

Initialization Block

When SSIZE32=0 (BCR20/CSR58, bit 8) the software structures are defined to be 16 bits wide and the initialization block looks as shown in Table 55. When SSIZE32=1, the software structures are defined to be 32 bits wide and the initialization block looks as shown in Table 56. Regardless of the value of SSIZE32, the initialization block must be aligned to a double word boundary, i.e. CSR1[1:0] and CSR16[1:0] must be set to ZERO.

Note that the PCnet-32 device performs double-word accesses to read the initialization block. This statement is always true, regardless of the setting of the SSIZE32 bit.

Table 55. Initialization Block (when SSIZE = 0)

Address	Bits 15-13	Bit 12	Bits 11-8	Bits 7-4	Bits 3-0
IADR+00	MODE 15-00				
IADR+02	PADR 15-00				
IADR+04	PADR 31-16				
IADR+06	PADR 47-32				
IADR+08	LADRF 15-00				
IADR+0A	LADRF 31-16				
IADR+0C	LADRF 47-32				
IADR+0E	LADRF 63-48				
IADR+10	RDRA 15-00				
IADR+12	RLEN	0	RES	RDRA 23-16	
IADR+14	TDRA 15-00				
IADR+16	TLEN	0	RES	TDRA 23-16	

Table 56. Initialization Block (when SSIZE = 1)

Address	Bits 31-28	Bits 27-24	Bits 23-20	Bits 19-16	Bits 15-12	Bits 11-8	Bits 7-4	Bits 3-0
IADR+00	TLEN	RES	RLEN	RES	MODE			
IADR+04	PADR 31–00							
IADR+08	RES				PADR 47-32			
IADR+0C	LADR 31–00							
IADR+10	LADR 63–32							
IADR+14	RDRA 31–00							
IADR+18	TDRA 31-00							

RLEN and TLEN

When SSIZE32 = 0 (BCR20[8]), then the software structures are defined to be 16 bits wide, and the RLEN and TLEN fields in the initialization block are 3 bits wide, occupying bits 15,14, and 13 and the value in these fields determines the number of Transmit and Receive Descriptor Ring Entries (DRE) which are used in the descriptor rings. Their meaning is shown in Table 57.

Table 57. Descriptor Ring Entries (SSIZE32 = 0)

R/TLEN	No. of DREs
000	1
001	2
010	4
011	8
100	16
101	32
110	64
111	128

If a value other than those listed in the above table is desired, CSR76 and CSR78 can be written after initialization is complete. See the description of the appropriate CSRs.

When SSIZE32=1 (BCR20[8]), then the software structures are defined to be 32 bits wide, and the RLEN and TLEN fields in the initialization block are 4 bits wide, occupying bits 15,14, 13, and 12, and the value in these fields determines the number of Transmit and Receive Descriptor Ring Entries (DRE) which are used in the descriptor rings. Their meaning is shown in Table 58.

If a value other than those listed in the above table is desired, CSR76 and CSR78 can be written after initialization is complete. See the description of the appropriate CSRs.

Table 58. Descriptor Ring Entries (SSIZE = 1)

R/TLEN	No. of DREs
0000	1
0001	2
0010	4
0011	8
0100	16
0101	32
0110	64
0111	128
1000	256
1001	512
11XX	512
1X1X	512

RDRA and TDRA

TDRA and RDRA indicate where the transmit and receive descriptor rings, respectively, begin. Each DRE must be located at 0 MOD 16 address values when SSIZE32 = 1 (BCR20[8]). Each DRE must be located at 0 MOD 8 address values when SSIZE32 = 0 (BCR20[8]).

LADRF

The Logical Address Filter (LADRF) is a 64-bit mask that is used to accept incoming Logical Addresses. If the first bit in the incoming address (as transmitted on the wire) is a “1”, the address is deemed logical. If the first bit is a “0”, it is a physical address and is compared against the physical address that was loaded through the initialization block.

A logical address is passed through the CRC generator, producing a 32 bit result. The high order 6 bits of the CRC is used to select one of the 64 bit positions in the Logical Address Filter. If the selected filter bit is set, the address is accepted and the frame is placed into memory. See Figure 38.

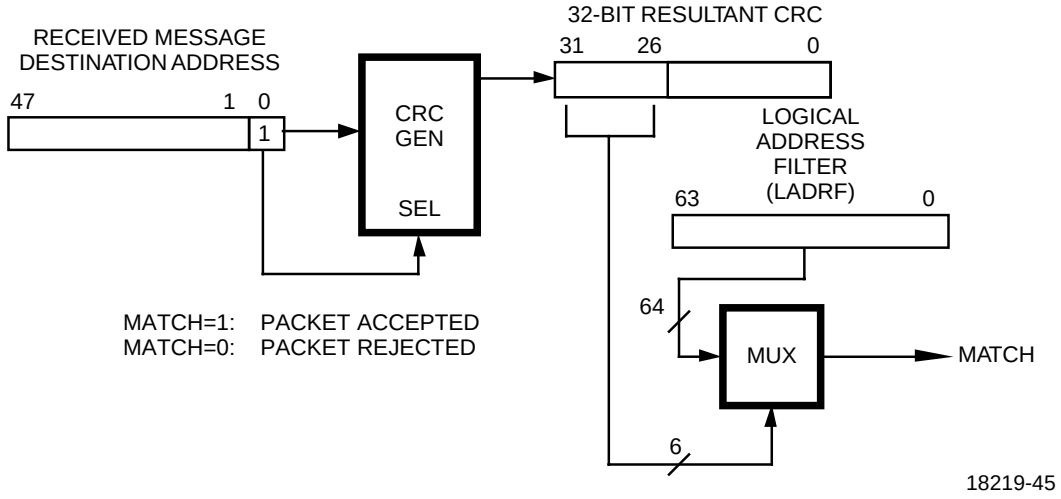


Figure 38. Address Match Logic

The Logical Address Filter is used in multicast addressing schemes. The acceptance of the incoming frame based on the filter value indicates that the message may be intended for the node. It is the node’s responsibility to determine if the message is actually intended for the node by comparing the destination address of the stored message with a list of acceptable logical addresses.

If the Logical Address Filter is loaded with all zeroes and promiscuous mode is disabled, all incoming logical addresses except broadcast will be rejected.

The Broadcast address, which is all ones, does not go through the Logical Address Filter and is handled as follows:

- If the Disable Broadcast Bit is cleared, the broadcast address is accepted
- If the Disable Broadcast Bit is set and promiscuous mode is enabled, the broadcast address is accepted.
- If the Disable Broadcast Bit is set and promiscuous mode is disabled, the broadcast address is rejected.

If external loopback is used, the FCS logic must be allocated to the receiver (by setting the DXMTFCS bit in CSR15, and clearing the ADD_FCS bit in TMD1) when using multicast addressing.

PADR

This 48-bit value represents the unique node address assigned by the ISO 8802-3 (IEEE/ANSI 802.3) and used for internal address comparison. PADR[0] is the first address bit transmitted on the wire, and must be zero. The six hex-digit nomenclature used by the ISO 8802-3 (IEEE/ANSI 802.3) maps to the PCnet-32 controller PADR register as follows: the first byte comprises PADR[7:0], with PADR[0] being the least significant bit of the byte. The second ISO 8802-3 (IEEE/ANSI 802.3) byte maps to PADR[15:8], again from LS bit to MS bit, and so on. The sixth byte maps to PADR[47:40], the LS bit being PADR[40].

MODE

The mode register in the initialization block is copied into CSR15 and interpreted according to the description of CSR15.

Receive Descriptors

When SSIZE32 = 0 (BCR20[8]), then the software structures are defined to be 16 bits wide, and receive descriptors look as shown in Table 59.

Table 59. Receive Descriptors (SSIZE32 = 0)

	LANCE/ PCnet-ISA Descriptor Designation	PCnet-32 Descriptor Designation	
Address	**Bits 15-0**	**Bits 15-8**	**Bits 7-0**
CRDA+00	RMD0	RMD0[15:0]	
CRDA+02	RMD1	RMD1[31:24]	RMD0[23:16]
CRDA+04	RMD2	RMD1[15:0]	
CRDA+06	RMD3	RMD2[15:0]	

PCnet-32 reference names within the table above refer to the descriptor definitions given in text below. Since the text descriptions are for 32-bit descriptors, the table above shows the mapping of the 32-bit descriptors into the 16-bit descriptor space. Since 16-bit descriptors are a subset of the 32-bit descriptors, some portions of the 32-bit descriptors may not appear in Table 59.

When SSIZE32 = 1 (BCR 20[8]), then the software structures are defined to be 32 bits wide, and receive descriptors look as shown in Table 60.

Table 60. Receive Descriptors (SSIZE 32 = 1)

	LANCE/PCnet-ISA Descriptor Designation				PCnet-32 Descriptor Designation
Address	**Bits 31-24**	**Bits 23-16**	**Bits 15-8**	**Bits 7-0**	**Bits 31-0**
CRDA+00	*NA	RMD1[7:0]	RMD0[15:8]	RMD0[7:0]	RMD0
CRDA+04	RMD1[15:8]	*NA	RMD2[15:8]	RMD2[7:0]	RMD1
CRDA+08	*NA	*NA	RMD3[15:8]	RMD3[7:0]	RMD2
CRDA+0C	*NA	*NA	*NA	*NA	RMD3

** NA = These 8 bits do not exist in any LANCE descriptor.*

The Receive Descriptor Ring Entries (RDREs) are composed of four receive message descriptors (RMD0– RMD3). Together they contain the following information:

- The address of the actual message data buffer in user (host) memory.
- The length of that message buffer.
- Status information indicating the condition of the buffer. The eight most significant bits of RMD1 (RMD1[31:24]) are collectively termed the STATUS of the receive descriptor.

RMD0

Bit	Name	Description
31-0	RBADR	RECEIVE BUFFER ADDRESS. This field contains the address of the receive buffer that is associated with this descriptor.

RMD1

Bit	Name	Description
31	OWN	This bit indicates that the descriptor entry is owned by the host (OWN=0) or by the PCnet-32 controller (OWN=1). The PCnet-32

controller clears the OWN bit after filling the buffer pointed to by the descriptor entry. The host sets the OWN bit after emptying the buffer. Once the PCnet-32 controller or host has relinquished ownership of a buffer, it must not change any field in the descriptor entry.

30	ERR	ERR is the OR of FRAM, OFLO, CRC, or BUFF. ERR is set by the PCnet-32 controller and cleared by the host.
29	FRAM	FRAMING ERROR indicates that the incoming frame contained a non-integer multiple of eight bits and there was an FCS error. If there was no FCS error on the incoming frame, then FRAM will not be set even if there was a non integer multiple of eight bits in the frame. FRAM is not valid in internal loopback mode. FRAM is valid only when ENP is set and OFLO is not. FRAM is set by the PCnet-32 controller and cleared by the host.
28	OFLO	OVERFLOW error indicates that the receiver has lost all or part of the incoming frame, due to an inability to store the frame in a memory buffer before the internal FIFO overflowed. OFLO is valid only when ENP is not set. OFLO is set by the PCnet-32 controller and cleared by the host.
27	CRC	CRC indicates that the receiver has detected a CRC (FCS) error on the incoming frame. CRC is valid only when ENP is set and OFLO is not. CRC is set by the PCnet-32 controller and cleared by the host.
26	BUFF	BUFFER ERROR is set any time the PCnet-32 controller does not own the next buffer while data chaining a received frame. This can occur in either of two ways: 1. The OWN bit of the next buffer is zero. 2. FIFO overflow occurred before the PCnet-32 controller received the STATUS (RMD1[31:24]) of the next descriptor. If a Buffer Error occurs, an Overflow Error may also occur internally in the FIFO, but will not be reported in the descriptor status entry unless both BUFF and OFLO errors occur at the same time. BUFF is set by the PCnet-32 controller and cleared by the host.
25	STP	START OF PACKET indicates that this is the first buffer used by the PCnet-32 controller for this frame. It is used for data chaining buffers. STP is set by the PCnet-32 controller and cleared by the host.
24	ENP	END OF PACKET indicates that this is the last buffer used by the PCnet-32 controller for this frame. It is used for data chaining buffers. If both STP and ENP are set, the frame fits not one buffer and there is no data chaining. ENP is set by the PCnet-32 controller and cleared by the host.
23-16	RES	Reserved locations. These locations should be read and written as ZEROs.
15-12	ONES	These four bits must be written as ONES. They are written by the host and unchanged by the PCnet-32 controller.
11-0	BCNT	BUFFER BYTE COUNT is the length of the buffer pointed to by this descriptor, expressed as the two's complement of the length of the buffer. This field is written by the host and unchanged by the PCnet-32 controller.

RMD2

Bit	Name	Description
31-24	RCC	Receive Collision Count. Indicates the accumulated number of collisions on the network since the last frame was successfully received, excluding collisions that occurred during transmissions from this node. The PCnet-32 controller implementation of this counter may not be compatible with the ILACC RCC definition. If network statistics are to be monitored, then CSR114 should be used for the purpose of monitoring Receive collisions instead of these bits.

23-16	RPC	Runt Packet Count. Indicates the accumulated number of runts that were addressed to this node since the last time that a receive packet was successfully received and its corresponding RMD2 ring entry was written to by the PCnet-32 controller. In order to be included in the RPC value, a runt must be long enough to meet the minimum requirement of the internal address matching logic. The minimum requirement for a runt to pass the internal address matching mechanism is: 18 bits of valid preamble plus a valid SFD detected, followed by 7 bytes of frame data. This requirement is unvarying, regardless of the address matching mechanisms in force at the time of reception (i.e. physical, logical, broadcast or promiscuous). The PCnet-32 controller implementation of this counter may not be compatible with the ILACC RPC definition.
15-12	ZEROs	This field is reserved. PCnet-32 controller will write ZEROs to these locations.
11-0	MCNT	MESSAGE Byte COUNT is the length in bytes of the received message, expressed as an unsigned binary integer. MCNT is valid only when ERR is clear and ENP is set. MCNT is written by the PCnet-32 controller and cleared by the host.

RMD3

Bit	Name	Description
31-0	RES	Reserved locations. All bits must be ZEROs.

Transmit Descriptors

When SSIZE32 = 0 (BCR 20[8]), then the software structures are defined to be 16 bits wide, and transmit descriptors look as shown in Table 61.

PCnet-32 reference names within the table above refer to the descriptor definitions given in text below. Since the text descriptions are for 32-bit descriptors, the table above shows the mapping of the 32-bit descriptors into the 16-bit descriptor space. Since 16-bit descriptors are a subset of the 32-bit descriptors, some portions of the 32-bit descriptors may not appear in Table 61.

When SSIZE32 = 1 (BCR 20[8]), then the software structures are defined to be 32 bits wide, and transmit descriptors look as shown in Table 62.

Table 61. Transmit Descriptor (SSIZE32 = 0)

	LANCE Descriptor Designation	PCnet-32 Descriptor Designation	
Address	**Bits 15-0**	**Bits 15-8**	**Bits 7-0**
CRDA+00	TMD0	TMD0[15:0]	
CRDA+02	TMD1	TMD1[31:24]	TMD0[23:16]
CRDA+04	TMD2	TMD1[15:0]	
CRDA+06	TMD3	TMD2[15:0]	

Table 62. Transmit Descriptor (SSIZE32 = 1)

	LANCE Descriptor Designation				PCnet-32 Descriptor Designation
Address	**Bits 31-24**	**Bits 23-16**	**Bits 15-8**	**Bits 7-0**	**Bits 31-0**
CTDA+00	*NA	TMD1[7:0]	TMD0[15:8]	TMD0[7:0]	TMD0
CTDA+04	TMD1[15:8]	*NA	TMD2[15:8]	TMD2[7:0]	TMD1
CTDA+08	TMD3[15:8]	TMD3[7:0]	*NA	*NA	TMD2
CTDA+0C	*NA	*NA	*NA	*NA	TMD3

** NA = These 8 bits do not exist in any LANCE descriptor.*

The Transmit Descriptor Ring Entries (TDREs) are composed of 4 transmit message descriptors (TMD0-TMD3). Together they contain the following information:

- The address of the actual message data buffer in user or host memory.
- The length of the message buffer.
- Status information indicating the condition of the buffer. The eight most significant bits of TMD1 (TMD1[31:24]) are collectively termed the STATUS of the transmit descriptor.

TMD0

Bit	Name	Description
31-0	TBADR	Transmit Buffer address. This field contains the address of the transmit buffer that is associated with this descriptor.

TMD1

Bit	Name	Description
31	OWN	This bit indicates that the descriptor entry is owned by the host (OWN=0) or by the PCnet-32 controller (OWN=1). The host sets the OWN bit after filling the buffer pointed to by the descriptor entry. The PCnet-32 controller clears the OWN bit after transmitting the contents of the buffer. Both the PCnet-32 controller and the host must not alter a descriptor entry after it has relinquished ownership.
30	ERR	ER is the OR of UFLO, LCOL, LCAR, or RTRY. ERR is set by the PCnet-32 controller and cleared by the host. This bit is set in the current descriptor when the error occurs, and therefore may be set in any descriptor of a chained buffer transmission.
29	ADD_FCS/NO_FCS	Bit 29 functions as ADD_FCS when programmed for the default I/O style of PCnet-ISA and when programmed for the I/O style PCnet-32 controller. Bit 29 functions as NO_FCS when programmed for the I/O style ILACC.
	ADD_FCS	ADD_FCS dynamically controls the generation of FCS on a frame by frame basis. It is valid only if the STP bit is set. When ADD_FCS is set, the state of DXMTFCS is ignored and transmitter FCS generation is activated. When ADD_FCS = 0, FCS generation is controlled by DXMTFCS. ADD_FCS is set by the host, and unchanged by the PCnet-32 controller. This was a reserved bit in the LANCE (Am7990). *This function differs from the ILACC function for this bit.*
	NO_FCS	NO_FCS dynamically controls the generation of FCS on a frame by frame basis. It is valid only if the ENP bit is set. When NO_FCS is set, the state of DXMTFCS is ignored and transmitter FCS generation is deactivated. When NO_FCS = 0, FCS generation is controlled by DXMTFCS. NO_FCS is set by the host, and unchanged by the PCnet-32 controller. This was a reserved bit in the LANCE (Am7990). *This function is identical to the ILACC function for this bit.*
28	MORE	MORE indicates that more than one retry was needed to transmit a frame. The value of MORE is written by the PCnet-32 controller. This bit has meaning only if the ENP bit is set.
27	ONE	ONE indicates that exactly one retry was needed to transmit a frame. ONE flag is not valid when LCOL is set. The value of the ONE bit is written by the PCnet-32 controller. This bit has meaning only if the ENP bit is set.
26	DEF	DEFERED indicates that the PCnet-32 controller had to defer while trying to transmit a frame. This condition occurs if the channel is busy when the PCnet-32 controller is ready to transmit. DEF is set by the PCnet-32 controller and cleared by the host.
25	STP	START OF PACKET indicates that this is the first buffer to be used by the PCnet-32 controller for this frame. It is used for data chaining buffers. The STP bit must be set in the first buffer of the frame, or the PCnet-32 controller will skip over the descriptor and poll the next descriptor(s) until the OWN and STP bits are set. STP is set by the host and unchanged by the PCnet-32 controller.
24	ENP	END OF PACKET indicates that this is the last buffer to be used by the PCnet-32 controller for this frame. It is used for data chaining buffers. If both STP and ENP are set, the frame fits into one buffer

and there is no data chaining. ENP is set by the host and unchanged by the PCnet-32 controller.

23-16 RES — Reserved locations.

15-12 ONES — Must be Ones. This field is written by the host and unchanged by the PCnet-32 controller.

11-0 BCNT — BUFFER BYTE COUNT is the usable length of the buffer pointed to by this descriptor, expressed as the two's complement of the length of the buffer. This is the number of bytes from this buffer that will be transmitted by the PCnet-32 controller. This field is written by the host and unchanged by the PCnet-32 controller. There are no minimum buffer size restrictions.

TMD2

Bit	Name	Description

31 BUFF — BUFFER ERROR is set by the PCnet-32 controller during transmission when the PCnet-32 controller does not find the ENP flag in the current buffer and does not own the next buffer. This can occur in either of two ways:

1. The OWN bit of the next buffer is zero.
2. FIFO underflow occurred before the PCnet-32 controller obtained the STATUS byte (TMD1[31:24]) of the next descriptor. BUFF is set by the PCnet-32 controller and cleared by the host. BUFF error will turn off the transmitter (CSR0, TXON = 0).

If a Buffer Error occurs, an Underflow Error will also occur. BUFF is not valid when LCOL or RTRY error is set during transmit data chaining. BUFF is set by the PCnet-32 controller and cleared by the host.

30 UFLO — UNDERFLOW ERROR indicates that the transmitter has truncated a message due to data late from memory. UFLO indicates that the FIFO has emptied before the end of the frame was reached. Upon UFLO error, the transmitter is turned off (CSR0, TXON = 0). UFLO is set by the PCnet-32 controller and cleared by the host.

29 EXDEF — EXCESSIVE DEFERRAL. Indicates that the transmitter has experienced Excessive Deferral on this transmit frame, where Excessive Deferral is defined in ISO 8802-3 (IEEE/ANSI 802.3).

28 LCOL LATE COLLISION indicates that a collision has occurred after the slot time of the channel has elapsed. The PCnet-32 controller does not retry on late collisions. LCOL is set by the PCnet-32 controller and cleared by the host.

27 LCAR — LOSS OF CARRIER is set in AUI mode when the carrier is lost during an PCnet-32 controller-initiated transmission. The PCnet-32 controller does not re-try upon loss of carrier. It will continue to transmit the whole frame until done. In 10BASE-T mode LCAR will be set when the T-MAU is in Link Fail state. LCAR is not valid in Internal Loopback Mode. LCAR is set by the PCnet-32 controller and cleared by the host.

26 RTRY — RETRY ERROR indicates that the transmitter has failed after 16 attempts to successfully transmit a message, due to repeated collisions on the medium. If DRTY = 1 in the MODE register, RTRY will set after 1 failed transmission attempt. RTRY is set by the PCnet-32 controller and cleared by the host.

25-16 TDR — TIME DOMAIN REFLECTOMETRY reflects the state of an internal PCnet-32 controller counter that counts at a 10 MHz rate from the start of a transmission to the occurrence of a collision or loss of carrier. This value is useful in determining the approximate distance to a cable fault. The TDR value is written by the PCnet-32 controller and is valid only if RTRY is set.

Note that 10 MHz gives very low resolution and in general has not been found to be particularly useful. This feature is here primarily to maintain full

		compatibility with the LANCE (Am7990).
15-4	RES	Reserved locations.
3-0	TRC	TRANSMIT RETRY COUNT. Indicates the number of transmit retries of the associated packet. The maximum count is 15. However, if a RETRY error occurs, the count will roll over to zero. In this case only, the Transmit Retry Count value of zero should be interpreted as meaning 16. TRC is written by the PCnet-32 controller into the last transmit descriptor of a frame, or when an error terminates a frame. Valid only when OWN = 0.

TMD3

Bit	Name	Description
31-0	RES	Reserved locations. All bits must be ZEROs.

REGISTER SUMMARY

CSRs — Control and Status Registers

RAP Addr	Symbol	Default Value After H_RESET (Hex)	Comments	Use
00	CSR0	xxxx 0004	PCnet-32 Controller Status	R
01	CSR1	xxxx xxxx	Lower IADR: maps to location 16	S
02	CSR2	xxxx xxxx	Upper IADR: maps to location 17	S
03	CSR3	xxxx 0000	Interrupt Masks and Deferral Control	S
04	CSR4	xxxx 0115	Test and Features Control	R
05	CSR5	xxxx 0000	Reserved	T
06	CSR6	xxxx xxxx	RXTX: RX/TX Descriptor Table Length	T
07	CSR7	xxxx 0000	Reserved	T
08	CSR8	xxxx xxxx	LADR0: Logical Address Filter — LADRF[15:0]	T
09	CSR9	xxxx xxxx	LADR1: Logical Address Filter — LADRF[31:16]	T
10	CSR10	xxxx xxxx	LADR2: Logical Address Filter — LADRF[47:32]	T
11	CSR11	xxxx xxxx	LADR3: Logical Address Filter — LADRF[63:48]	T
12	CSR12	xxxx xxxx	PADR0: Physical Address Register — PADR[15:0][	T
13	CSR13	xxxx xxxx	PADR1: Physical Address Register — PADR[31:16]	T
14	CSR14	xxxx xxxx	PADR2: Physical Address Register — PADR[47:32]	T
15	CSR15	See Register Desc.	MODE: Mode Register	S
16	CSR16	xxxx xxxx	IADR: Base Address of INIT Block Lower (Copy)	T
17	CSR17	xxxx xxxx	IADR: Base Address of INIT Block Upper (Copy)	T
18	CSR18	xxxx xxxx	CRBA: Current RCV Buffer Address Lower	T
19	CSR22	xxxx xxxx	CRBA: Current RCV Buffer Address Upper	T
20	CSR20	xxxx xxxx	CXBA: Current XMT Buffer Address Lower	T
21	CSR21	xxxx xxxx	CXBA: Current XMT Buffer Address Upper	T
22	CSR22	xxxx xxxx	NRBA: Next RCV Buffer Address Lower	T
23	CSR23	xxxx xxxx	NRBA: Next RCV Buffer Address Upper	T
24	CSR24	xxxx xxxx	BADR: Base Address of RCV Ring Lower	S
25	CSR25	xxxx xxxx	BADR: Base Address of RCV Ring Upper	S
27	CSR27	xxxx xxxx	NRDA: Next RCV Descriptor Address Upper	T
28	CSR28	xxxx xxxx	CRDA: Current RCV Descriptor Address Lower	T
29	CSR29	xxxx xxxx	CRDA: Current RCV Descriptor Address Upper	T
30	CSR30	xxxx xxxx	BADX: Base Address of XMT Ring Lower	S
31	CSR31	xxxx xxxx	BADX: Base Address of XMT Ring Upper	S
32	CSR32	xxxx xxxx	NXDA: Next XMT Descriptor Address Lower	T
33	CSR33	xxxx xxxx	NXDA: Next XMT Descriptor Address Upper	T
34	CSR34	xxxx xxxx	CXDA: Current XMT Descriptor Address Lower	T
35	CSR35	xxxx xxxx	CXDA: Current XMT Descriptor Address Upper	T
36	CSR36	xxxx xxxx	NNRDA: Next Next Receive Descriptor Address Lower	T

Key: ***x*** *= undefined* ***R*** *= Running Register* ***S*** *= Setup Register* ***T*** *= Test Register*

REGISTER SUMMARY (continued)

CSRs — Control and Status Registers

RAP Addr	Symbol	Default Value After H_RESET (Hex)	Comments	Use
37	CSR37	xxxx xxxx	NNRDA: Next Next Receive Descriptor Address Upper	T
38	CSR38	xxxx xxxx	NNXDA: Next Next Transmit Descriptor Address Lower	T
39	CSR39	xxxx xxxx	NNXDA: Next Next Transmit Descriptor Address Upper	T
40	CSR40	xxxx xxxx	CRBC: Current RCV Status and Byte Count Lower	T
41	CSR41	xxxx xxxx	CRBC: Current RCV Status and Byte Count Upper	T
42	CSR42	xxxx xxxx	CXBC: Current XMT Status and Byte Count Lower	T
43	CSR43	xxxx xxxx	CXBC: Current XMT Status and Byte Count Upper	T
44	CSR44	xxxx xxxx	NRBC: Next RCV Status and Byte Count Lower	T
45	CSR45	xxxx xxxx	NRBC: Next RCV Status and Byte Count Upper	T
46	CSR46	xxxx xxxx	POLL: Poll Time Counter	T
47	CSR47	xxxx xxxx	PI: Polling Interval	S
48	CSR48	xxxx xxxx	TMP2: Temporary Storage Lower	T
49	CSR48	xxxx xxxx	TMP2: Temporary Storage Upper	T
50	CSR50	xxxx xxxx	TMP3: Temporary Storage Lower	T
51	CSR50	xxxx xxxx	TMP3: Temporary Storage Upper	T
52	CSR52	xxxx xxxx	TMP4: Temporary Storage Lower	T
53	CSR52	xxxx xxxx	TMP4: Temporary Storage Upper	T
54	CSR54	xxxx xxxx	TMP5: Temporary Storage Lower	T
55	CSR54	xxxx xxxx	TMP5: Temporary Storage Upper	T
56	CSR56	xxxx xxxx	TMP6: Temporary Storage Lower	T
57	CSR56	xxxx xxxx	TMP6: Temporary Storage Upper	T
58	CSR58	See Register Desc.	SWS: Software Style	S
59	CSR59	xxxx 0105	IR: IR Register	T
60	CSR60	xxxx xxxx	PXDA: Previous XMT Descriptor Address Lower	T
61	CSR61	xxxx xxxx	PXDA: Previous XMT Descriptor Address Upper	T
62	CSR62	xxxx xxxx	PXBC: Previous XMT Status and Byte Count Lower	T
63	CSR63	xxxx xxxx	PXBC: Previous XMT Status and Byte Count Upper	T
64	CSR64	xxxx xxxx	NXBA: Next XMT Buffer Address Lower	T
65	CSR65	xxxx xxxx	NXBA: Next XMT Buffer Address Upper	T
66	CSR66	xxxx xxxx	NXBC: Next XMT Status and Byte Count Lower	T
67	CSR67	xxxx xxxx	NXBC: Next XMT Status and Byte Count Upper	T
68	CSR68	xxxx xxxx	XSTMP: XMT Status Temporary Storage Lower	T
69	CSR69	xxxx xxxx	XSTMP: XMT Status Temporary Storage Upper	T
70	CSR70	xxxx xxxx	RSTMP: RCV Status Temporary Storage Lower	T
71	CSR71	xxxx xxxx	RSTMP: RCV Status Temporary Storage Upper	T
72	CSR72	xxxx xxxx	RCVRC: RCV Ring Counter	T

Key: ***x*** *= undefined* ***R*** *= Running Register* ***S*** *= Setup Register* ***T*** *= Test Register*

REGISTER SUMMARY

CSRs — Control and Status Registers (continued)

RAP Addr	Symbol	Default Value After H_RESET (Hex)	Comments	Use
73	CSR73	xxxx xxxx	Reserved	T
74	CSR74	xxxx xxxx	XMTRC: XMT Ring Counter	T
75	CSR75	xxxx xxxx	Reserved	T
76	CSR76	xxxx xxxx	RCVRL: RCV Ring Length	S
77	CSR77	xxxx xxxx	Reserved	T
78	CSR78	xxxx xxxx	XMTRL: XMT Ring Length	S
79	CSR79	xxxx xxxx	Reserved	T
80	CSR80	xxxx E810	Burst and FIFO Threshold Control	S
81	CSR81	xxxx xxxx	Reserved	T
82	CSR82	xxxx 0000	DMABAT: Bus Acitivty Timer	S
83	CSR83	xxxx xxxx	Reserved	T
84	CSR84	xxxx xxxx	DMABA: DMA Address Lower	T
85	CSR85	xxxx xxxx	DMABA: DMA Address Upper	T
86	CSR86	xxxx xxxx	DMABC: Buffer Byte Counter	T
87	CSR87	xxxx xxxx	Reserved	T
88	CSR88	0243 0003	Chip ID Lower	T
89	CSR89	xxxx 0243	Chip ID Upper	T
90	CSR90	xxxx xxxx	Reserved	T
91	CSR91	xxxx xxxx	Reserved	T
92	CSR92	xxxx xxxx	RCON: Ring Length Conversion	T
93	CSR93	xxxx xxxx	Reserved	T
94	CSR94	xxxx 0000	XMTTDR: Transmit Time Domain Reflectometry Count	T
95	CSR95	xxxx xxxx	Reserved	T
96	CSR96	xxxx xxxx	SCR0: BIU Scratch Register 0 Lower	T
97	CSR97	xxxx xxxx	SCR0: BIU Scratch Register 0 Upper	T
98	CSR98	xxxx xxxx	SCR1: BIU Scratch Register 1 Lower	T
99	CSR99	xxxx xxxx	SCR1: BIU Scratch Register 1 Upper	T
100	CSR100	xxxx 0200	Bus Time-Out	S
101	CSR101	xxxx xxxx	Reserved	T
102	CSR102	xxxx xxxx	Reserved	T
103	CSR103	xxxx 0105	Reserved	T
104	CSR104	xxxx xxxx	SWAP: Swap Register Lower	T
105	CSR105	xxxx xxxx	SWAP: Swap Register Upper	T
106	CSR106	xxxx xxxx	Reserved	T
107	CSR107	xxxx xxxx	Reserved	T
108	CSR108	xxxx xxxx	BMSCR: BMU Scratch Register Lower	T
109	CSR109	xxxx xxxx	BMSCR: BMU Scratch Register Upper	T

Key: ***x*** *= undefined* ***R*** *= Running Register* ***S*** *= Setup Register* ***T*** *= Test Register*

REGISTER SUMMARY (continued)

CSRs — Control and Status Registers

RAP Addr	Symbol	Default Value After H_RESET (Hex)	Comments	Use
110	CSR110	xxxx xxxx	Reserved	T
111	CSR111	xxxx xxxx	Reserved	T
112	CSR112	xxxx 0000	Missed Frame Count	R
113	CSR113	xxxx xxxx	Reserved	T
114	CSR114	xxxx 0000	Receive Collision Count	R
115	CSR115	xxxx xxxx	Reserved	T
116	CSR116	xxxx xxxx	Reserved	T
117	CSR117	xxxx xxxx	Reserved	T
118	CSR118	xxxx xxxx	Reserved	T
119	CSR119	xxxx xxxx	Reserved	T
120	CSR120	xxxx xxxx	Reserved	T
121	CSR121	xxxx xxxx	Reserved	T
122	CSR122	See Register Desc.	Receive Frame Alignment Control	S
123	CSR123	xxxx xxxx	Reserved	T
124	CSR124	See Register Desc.	BMU Test	T
125	CSR125	xxxx xxxx	Reserved	T
126	CSR126	xxxx xxxx	Reserved	T
127	CSR127	xxxx xxxx	Reserved	T

Key: ***x*** *= undefined* ***R*** *= Running Register* ***S*** *= Setup Register* ***T*** *= Test Register*

REGISTER SUMMARY

BCR — Bus Configuration Registers

RAP Addr.	Mnemonic	Default (Hex)	Name	Programmability		
				User	EEPROM	SRM
0	MSRDA	0005	Master Mode Read Active	No	No	No
1	MSWRA	0005	Master Mode Write Active	No	No	No
2	MC	N/A*	Miscellaneous Configuration	Yes	Yes	Yes
3	Reserved	N/A		No	No	No
4	$\overline{\text{LNKST}}$	00C0	Link Status (Default)	Yes	No	No
5	LED1	0084	Receive (Default)	Yes	No	No
6	LED2	0088	Receive Polarity (Default)	Yes	No	No
7	LED3	0090	Transmit (Default)	Yes	No	No
8–15	Reserved	N/A		No	No	No
16	IOBASEL	N/A*	I/O Base Address Lower	Yes	Yes	Yes
17	IOBASEU	N/A*	I/O Base Address Upper	Yes	Yes	Yes
18	BSBC	2101	Burst Size and Bus Control	Yes	Yes	No
19	EECAS	0002	EEPROM Control and Status	Yes	No	No
20	SWSTYLE	0000	Software Style	Yes	No	No
21	INTCON	N/A*	Interrupt Control	Yes	Yes	Yes

Key: SRM = Software Relocatable Mode

* *Registers marked with an asterisk (*) have no default value, since they are not observable without first being programmed, either through the EEPROM read operation or through the Software Relocatable Mode. Therefore, the only observable values for these registers are those that have been programmed and a default value is not applicable.*

ABSOLUTE MAXIMUM RATINGS

Storage Temperature ..... –65°C to +150°C

Ambient Temperature
Under Bias ..... –65°C to +125°C

Supply Voltage to AV_{SS}
or DV_{SS} (AV_{DD}, DV_{DD})..... –0.3 V to +6.0 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. Programming conditions may differ.

OPERATING RANGES

Commercial (C) Devices

Temperature (T_A) ..... 0°C to +70°C

Supply Voltages

(AV_{DD}, DV_{DD}) .....5 V ± 5%

All inputs within the range: ..... AV_{SS} – 0.5 V ≥ Vin ≥ AV_{DD} + 0.5 V, or DV_{SS} – 0.5 V ≥ Vin ≥ DV_{DD} + 0.5 V

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

DC CHARACTERISTICS over COMMERCIAL operating ranges unless otherwise specified

Parameter Symbol	Parameter Description	Test Conditions		Min	Max	Unit
Digital Input Voltage						
V_{IL}	Input LOW Voltage				0.8	V
V_{IH}	Input HIGH Voltage			2.0		V
Digital Ouput Voltage						
V_{OL}	Output LOW Voltage	I_{OL1} = 8 mA, I_{OL2} = 4 mA (Notes 1 and 5)			0.45	V
V_{OH}	Output HIGH Voltage (Note 2)	I_{OH} = –4 mA (Note 5)		2.4		V
Digital Input Leakage Current						
I_{IX}	Input Leakage Current (Note 3)	V_{DD} = 5 V, V_{IN} = 0 V		–10	10	µA
Digital Ouput Leakage Current						
I_{OZL}	Output Low Leakage Current (Note 4)	V_{OUT} = 0 V		–10		µA
I_{OZH}	Output High Leakage Current (Note 4)	V_{OUT} = V_{DD}			10	µA
Crystal Input						
V_{ILX}	XTAL1 Input LOW Threshold Voltage	V_{IN} = External Clock		–0.5	0.8	V
V_{ILHX}	XTAL1 Input HIGH Threshold Voltage	V_{IN} = External Clock		V_{DD}–0.8	V_{DD} + 0.5	V
I_{ILX}	XTAL1 Input LOW Current	V_{IN} = 0 V	Active	–120	0	µA
			Sleep	–10	+10	µA
I_{IHX}	XTAL1 Input HIGH Current	V_{IN} = V_{DD}	Active	0	120	µA
			Sleep		400	µA
Attachment Unit Interface						
I_{IAXD}	Input Current at DI+ and DI–	–1 V < V_{IN} < AV_{DD} +0.5 V		–500	+500	µA
I_{IAXC}	Input Current at CI+ and CI–	–1 V < V_{IN} < AV_{DD} +0.5 V		–500	+500	µA
V_{AOD}	Differential Output Voltage |(DO+)–(DO–)|	R_L = 78 Ω		630	1200	mV
V_{AODOFF}	Transmit Differential Output Idle Voltage	R_L = 78 Ω (Note 9)		–40	+40	mV

DC CHARACTERISTICS over COMMERCIAL operating ranges unless otherwise specified (continued)

Parameter Symbol	Parameter Description	Test Conditions	Min	Max	Unit
Attachment Unit Interface (Continued)					
I_{AODOFF}	Transmit Differential Output Idle Current	R_L = 78 Ω (Note 8)	–1	+1	mA
V_{CMT}	Transmit Output Common Mode Voltage	R_L = 78 Ω	2.5	AV_{DD}	V
V_{ODI}	DO± Transmit Differential Output Voltage Imbalance	R_L = 78 Ω (Note 7)		25	mV
V_{ATH}	Receive Data Differential Input Threshold		–35	35	mV
V_{ASQ}	DI± and CI± Differential Input Threshold (Squelch)		–275	–160	mV
V_{IRDVD}	DI± and CI± Differential Mode Input Voltage Range		–1.5	+1.5	V
V_{ICM}	DI± and CI± Input Bias Voltage	I_{IN} = 0 mA	AV_{DD}–3.0	AV_{DD}–1.0	V
V_{OPD}	DO± Undershoot Voltage at Zero Differential on Transmit Return to Zero (ETD)	(Note 9)		–100	mV
Twisted Pair Interface					
I_{IRXD}	Input Current at RXD±	$AV_{SS} < V_{IN} < AV_{DD}$	–500	500	µA
R_{RXD}	RXD± Differential Input Resistance		10		KΩ
V_{TIVB}	RXD+, RXD– Open Circuit Input Voltage (Bias)	I_{IN} = 0 mA	AV_{DD} – 3.0	AV_{DD} – 1.5	V
V_{TIDV}	Differential Mode Input Voltage Range (RXD±)	AV_{DD} = 5 V	–3.1	+3.1	V
V_{TSQ+}	RXD Positive Squelch Threshold (Peak)	Sinusoid 5 MHz ≤ f ≤10 MHz	300	520	mV
$V_{TSQ–}$	RXD Negative Squelch Threshold (Peak)	Sinusoid 5 MHz ≤ f ≤10 MHz	–520	–300	mV
V_{THS+}	RXD Post-Squelch Positive Threshold (Peak)	Sinusoid 5 MHz ≤ f ≤10 MHz	150	293	mV
$V_{THS–}$	RXD Post-Squelch Negative Threshold (Peak)	Sinusoid 5 MHz ≤ f ≤10 MHz	–293	–150	mV
V_{LTSQ+}	RXD Positive Squelch Threshold (Peak)	$\overline{LRT}$ = LOW	180	312	mV
$V_{LTSQ–}$	RXD Negative Squelch Threshold (Peak)	$\overline{LRT}$ = LOW	–312	–180	mV
V_{LTHS+}	RXD Post-Squelch Positive Threshold (Peak)	$\overline{LRT}$ = LOW	90	176	mV
$V_{LTHS–}$	RXD Post-Squelch Negative Threshold (Peak)	$\overline{LRT}$ = LOW	–176	–90	mV

DC CHARACTERISTICS over COMMERCIAL operating ranges unless otherwise specified (continued)

Parameter Symbol	Parameter Description	Test Conditions	Min	Max	Unit
Twisted Pair Interface (continued)					
V_{RXDTH}	RXD Switching Threshold	(Note 4)	–35	35	mV
V_{TXH}	TXD± and TXP± Output HIGH Voltage	DV_{SS} = 0 V	DV_{DD} – 0.6	DV_{DD}	V
V_{TXL}	TXD± and TXP± Output LOW Voltage	DV_{DD} = +5 V	DV_{SS}	DV_{SS} + 0.6	V
V_{TXI}	TXD± and TXP± Differential Output Voltage Imbalance		–40	+40	mV
V_{TXOFF}	TXD± and TXP± Idle Output Voltage			40	mV
R_{TX}	TXD± Differential Driver Output Impedance	(Note 4)		40	Ω
	TXP± Differential Driver Output Impedance	(Note 4)		80	Ω
IEEE 1149.1 (JTAG) Test Port					
V_{IL}	TCK, TMS, TDI			0.8	V
V_{IH}	TCK, TMS, TDI		2.0		V
V_{OL}	TDO	I_{OL} = 2.0 mA		0.4	V
V_{OH}	TDO	I_{OH} = –0.4 mA	2.4		V
I_{IL}	TCK, TMS, TDI	V_{DD} = 5.5 V, V_I = 0.5 V		–200	µA
I_{IH}	TCK, TMS, TDI	V_{DD} =5.5 V, V_I = 2.7V		–100	µA
I_{OZ}	TDO	V_{OUT} = 0 V, V_{OUT} = V_{DD}	–10	+10	µA
Power Supply Current					
I_{DD}	Active Power Supply Current	XTAL1 = 20 MHz		100	mA
I_{DDCOMA}	Coma Mode Power Supply Current	$\overline{SLEEP}$ active		200	µA
$I_{DDSNOOZE}$	Snooze Mode Power Supply Current	AWAKE bit set active		10	mA
Pin Capacitance					
C_{IN}	Input Pin Capacitance	F_C = 1 MHz (Note 6)		10	pF
C_O	I/O or Output Pin Capacitance	F_C = 1 MHz (Note 6)		10	pF
C_{CLK}	BCLK Pin Capacitance	F_C = 1 MHz (Note 6)		10	pF

Notes:

1. IOL1 = 8 mA applies to all output and I/O pins except HOLD and $\overline{LDEV}$. IOL2 = 4 mA applies to HOLD and $\overline{LDEV}$ only.

2. VOH does not apply to open-drain output pins.

3. IIX applies to all input only pins except DI±, CI±, and XTAL1.

4. IOZL and IOZH apply to all three-state output pins and bi-directional pins.

5. Outputs are CMOS and will be driven to rail if the load is not resistive.

6. Parameter not tested; value determined by characterization.

7. Tested, but to values in excess of limits. Test accuracy not sufficient to allow screening guard bands.

8. Correlated to other tested parameters – not tested directly.

9. Test not implemented to data sheet specification.

SWITCHING CHARACTERISTICS: BUS INTERFACE

Parameter Symbol	Parameter Description	Test Conditions (C_L = 50 pF Unless Otherwise Noted)	Min	Max	Unit
Clock Timing					
	CLK Frequency		1	33	MHz
t_1	CLK Period		30	1000	ns
t_2	CLK High time	@ 2.0 V	14		ns
t_3	CLK Low Time	@ 0.8 V	14		ns
t_4	CLK Fall Time	2.0 V to 0.8 V (Note 1)		3	ns
t_5	CLK Rise Time	0.8 V to 2.0 V (Note 1)		3	ns
Output and Float Delay Timing					
t_6	A2–A31, $\overline{BE0}$–$\overline{BE3}$, M/$\overline{IO}$, D/$\overline{C}$, W/$\overline{R}$, $\overline{ADS}$, HOLD, HLDAO, INTR, $\overline{EADS}$, $\overline{RDY}$ Valid Delay		3	12	ns
t_7	A2–A31, $\overline{BE0}$–$\overline{BE3}$, M/$\overline{IO}$, D/$\overline{C}$, W/$\overline{R}$, $\overline{ADS}$, $\overline{RDY}$ Float Delay			18	ns
t_8					ns
t_{8a}	$\overline{BLAST}$ Valid Delay		3	12	ns
t_9	$\overline{BLAST}$ Float Delay			18	ns
t_{10}	D0–D31 Data Valid Delay		3	12	ns
t_{11}	D0–D31 Data Float Delay			18	ns
t_{12}	(This symbol is not used)				
t_{13}	(This symbol is not used)				
Setup and Hold Timing					
t_{14}	$\overline{LBS16}$ Setup time		5		ns
t_{15}	$\overline{LBS16}$ Hold Time		2		ns
t_{16}	$\overline{BRDY}$, $\overline{RDYRTN}$ Setup Time		5		ns
t_{17}	$\overline{BRDY}$, $\overline{RDYRTN}$ Hold Time		2		ns
t_{18}	HOLDI, AHOLD, HLDA, $\overline{SLEEP}$ Setup Time		6		ns
t_{18a}	$\overline{BOFF}$ Setup Time		8		ns
t_{19}	HOLDI, $\overline{BOFF}$, AHOLD, HLDA, $\overline{SLEEP}$ Hold Time		2		ns
t_{20}	RESET, CLKRESET Setup Time		5		ns
t_{21}	RESET, CLKRESET Hold Time		2		ns
t_{22}	D0–D31, A2–A31, $\overline{BE0}$–$\overline{BE3}$, $\overline{ADS}$, M/$\overline{IO}$, D/$\overline{C}$, W/$\overline{R}$ Setup Time		4		ns
t_{23}	D0–D31, A2–A31, $\overline{BE0}$–$\overline{BE3}$, $\overline{ADS}$, M/$\overline{IO}$, D/$\overline{C}$, W/$\overline{R}$ Hold Time		2		ns

SWITCHING CHARACTERISTICS: BUS INTERFACE (CONTINUED)

Parameter Symbol	Parameter Description	Test Conditions (C_L = 50 pF Unless Otherwise Noted)	Min	Max	Unit
JTAG (IEEE 1149.1) Test Signal Timing					
t_{24}	TCK Frequency			10	MHz
t_{25}	TCK Period		100		ns
t_{26}	TCK High Time	@ 2.0 V	45		ns
t_{27}	TCK Low Time	@ 0.8 V	45		ns
t_{28}	TCK Rise Time			4	ns
t_{29}	TCK Fall Time			4	ns
t_{30}	TDI, TMS Setup Time		16		ns
t_{31}	TDI, TMS Hold Time		10		ns
t_{32}	TDO Valid Delay		3	60	ns
t_{33}	TDO Float Delay			50	ns
t_{34}	All Outputs (Non-Test) Valid Delay		3	25	ns
t_{35}	All Outputs (Non-Test) Float Delay			36	ns
t_{36}	All Inputs (Non-Test)) Setup Time		8		ns
t_{37}	All Inputs (Non-Test) Hold Time		7		ns
$\overline{\text{LDEV}}$ Timing					
t_{38}	$\overline{\text{ADS}}$ Asserted to $\overline{\text{LDEV}}$ Asserted			20	ns
t_{39}	$\overline{\text{LDEV}}$ Valid Pulse Width		t_{BCLK}		
Data Bus Activation Timing (Slave)					
t_{40}	Data Bus Driven After $\overline{\text{ADS}}$ Sampled Asserted		1		BCLK
HOLD Inactive Timing					
t_{41}	HOLD Deasserted to HOLD Asserted		2 BCLK –15 ns		
EEPROM Timing					
t_{42}	EESK Frequency	(Note 2)		650	kHz
t_{43}	EESK HIGH Time	(Note 2)	780		ns
t_{44}	EESK LOW Time	(Note 2)	780		ns
t_{45}	EECS, EEDI, SHFBUSY Valid Output Delay from EESK	(Note 2)	–15	+15	ns
t_{46}	EECS LOW Time	(Note 2)	1550		ns
t_{47}	EEDO Setup Time to EESK		50		ns
t_{48}	EEDO Hold Time to EESK		0		ns

Notes:

1. *Not 100% tested; guaranteed by design characterization.*
2. *Parameter value is given for automatic EEPROM read operation. When EEPROM port (BCR19) is used to access the EEPROM, software is responsible for meeting EEPROM timing requirements.*

SWITCHING CHARACTERISTICS: 10BASE-T INTERFACE

Parameter Symbol	Parameter Description	Test Conditions	Min	Max	Unit
Transmit Timing					
t_{TETD}	Transmit Start of Idle		250	350	ns
t_{TR}	Transmitter Rise Time	(10% to 90%)		5.5	ns
t_{TF}	Transmitter Fall Time	(90% to 10%)		5.5	ns
t_{TM}	Transmitter Rise and Fall Time Mismatch			1	ns
t_{PERLP}	Idle Signal Period		8	24	ms
t_{PWLP}	Idle Link Pulse Width	(Note 1)	75	120	ns
t_{PWPLP}	Predistortion Idle Link Pulse Width	(Note 1)	45	55	ns
t_{JA}	Transmit Jabber Activation Time		20	150	ms
t_{JR}	Transmit Jabber Reset Time		250	750	ms
t_{JREC}	Transmit Jabber Recovery Time (minimum time gap between transmitted frames to prevent jabber activation)		1.0		µs
Receive Timing					
t_{PWNRD}	RXD Pulse Width Not to Turn Off Internal Carrier Sense	VIN > VTHS (min)	136	—	ns
t_{PWROFF}	RXD Pulse Width to Turn Off	VIN > VTHS (min)		200	ns
t_{RETD}	Receive Start of Idle		200		ns
t_{RCVON}	$\overline{RCV}$ Asserted Delay		T_{RON} - 50	T_{RON} + 100	ns
t_{RCVOFF}	$\overline{RCV}$ De-Asserted Delay		20	62	ms
Collision Detection and SQE Test					
t_{COLON}	$\overline{COL}$ Asserted Delay		750	900	ns
t_{COLOFF}	$\overline{COL}$ De-Asserted Delay		20	62	ms

Note:

1. Not tested; parameter guaranteed by characterization.

SWITCHING CHARACTERISTICS: AUI

Parameter Symbol	Parameter Description	Test Conditions	Min	Max	Unit
AUI Port					
t_{DOTR}	DO+,DO- rise time (10% to 90%)		2.5	5.0	ns
t_{DOTF}	DO+,DO- fall time (90% to 10%)		2.5	5.0	ns
t_{DORM}	DO+,DO- rise and fall time mismatch		—	1.0	ns
t_{DOETD}	DO+/- End of Transmission		200	375	ns
t_{PWODI}	DI pulse width accept/reject threshold	$\|V_{IN}\| > \|V_{ASQ}\|$ (Note 1)	15	45	ns
t_{PWKDI}	DI pulse width maintain/turn-off threshold	$\|V_{IN}\| > \|V_{ASQ}\|$ (Note 2)	136	200	ns
t_{PWOCI}	CI pulse width accept/reject threshold	$\|V_{IN}\| > \|V_{ASQ}\|$ (Note 3)	10	26	ns
t_{PWKCI}	CI pulse width maintain/turn-off threshold	$\|V_{IN}\| > \|V_{ASQ}\|$ (Note 4)	90	160	ns
Internal MENDEC Clock Timing					
t_{X1}	XTAL1 period	V_{IN} = External Clock	49.995	50.005	ns
t_{X1H}	XTAL1 HIGH pulse width	V_{IN} = External Clock	20		ns
t_{X1L}	XTAL1 LOW pulse width	V_{IN} = External Clock	20		ns
t_{X1R}	XTAL1 rise time	V_{IN} = External Clock		5	ns
t_{X1F}	XTAL1 fall time	V_{IN} = External Clock		5	ns

Notes:

1. *DI pulses narrower than tPWODI (min) will be rejected; pulses wider than tPWODI (max) will turn internal DI carrier sense on.*
2. *DI pulses narrower than tPWKDI (min) will maintain internal DI carrier sense on; pulses wider than tPWKDI (max) will turn internal DI carrier sense off.*
3. *CI pulses narrower than tPWOCI (min) will be rejected; pulses wider than tPWOCI (max) will turn internal CI carrier sense on.*
4. *CI pulses narrower than tPWKCI (min) will maintain internal CI carrier sense on; pulses wider than tPWKCI (max) will turn internal CI carrier sense off.*

SWITCHING CHARACTERISTICS: GPSI

Parameter Symbol	Parameter Description	Test Conditions	Min	Max	Unit
Transmit Timing					
t_{GPT1}	STDCLK period (802.3 Compliant)		99.99	100.01	ns
t_{GPT2}	STDCLK High Time		40	60	ns
t_{GPT3}	TXDAT and TXEN delay from ↑ STDCLK		0	70	ns
t_{GPT4}	RXCRS setup before ↑ STDCLK (Last Bit)		210		ns
t_{GPT5}	RXCRS hold after ↓ TXEN		0		ns
t_{GPT6}	CLSN active time to trigger collision	(Note 1)	110		ns
t_{GPT7}	CLSN active to ↓ RXCRS to prevent LCAR assertion		0		ns
t_{GPT8}	CLSN active to ↓ RXCRS for SQE Hearbeat Window		0	4.0	µs
t_{GPT9}	CLSN active to ↑ RXCRS for normal collision		0	51.2	µs
Receive Timing					
t_{GPR1}	SRDCLK period	(Note 2)	80	120	ns
t_{GPR2}	SRDCLK High Time	(Note 2)	30	80	ns
t_{GPR3}	SRDCLK Low Time	(Note 2)	30	80	ns
t_{GPR4}	RXDAT and RXCRS setup to ↑ SRDCLK		15		ns
t_{GPR5}	RXDAT hold after ↑ SRDCLK		15		ns
t_{GPR6}	RXCRS hold after ↓ SRDCLK		0		ns
t_{GPR7}	CLSN active to first ↑ SRDCLK (Collision Recognition)		0		ns
t_{GPR8}	CLSN active to ↑ SRDCLK for Address Type Designation Bit	(Note 3)	51.2		µs
t_{GPR9}	CLSN setup to last ↑ SRDCLK for Collision Recognition		210		ns
t_{GPR10}	CLSN active		110		ns
t_{GPR11}	CLSN inactive setup to first ↑ SRDCLK		300		ns
t_{GPR12}	CLSN inactive hold to last ↑ SRDCLK		300		ns

Notes:

1. *CLSN must be asserted for a continuous period of 110 nsec or more. Assertion for less than 110 nsec period may or may not result in CLSN recognition.*
2. *RXCRS should meet jitter requirements of IEEE 802.3 specification.*
3. *CLSN assertion before 51.2* msec *will be indicated as a normal collision. CLSN assertion after 51.2* msec *will be considered as a Late Receive Collision.*

SWITCHING CHARACTERISTICS: EADI

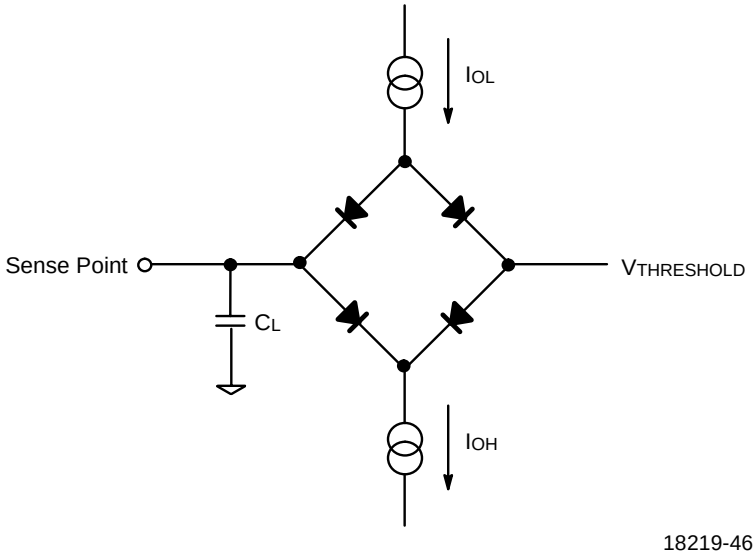
Parameter Symbol	Parameter Description	Test Conditions	Min	Max	Unit
t_{EAD1}	SRD setup to ↑ SRDCLK		40		ns
t_{EAD2}	SRD hold to ↑ SRDCLK		40		ns
t_{EAD3}	$\overline{SF/BD}$ change to ↓ SRDCLK		–15	+15	ns
t_{EAD4}	$\overline{EAR}$ de-assertion to ↑ SRDCLK (first rising edge)		50		ns
t_{EAD5}	$\overline{EAR}$ assertion after SFD event (packet rejection)		0	51,090	ns
t_{EAD6}	$\overline{EAR}$ assertion		110		ns

KEY TO SWITCHING WAVEFORMS

WAVEFORM	INPUTS	OUTPUTS
	Must Be Steady	Will Be Steady
	May Change from H to L	Will Be Changing from H to L
	May Change from L to H	Will Be Changing from L to H
	Don't Care Any Change Permitted	Changing State Unknown
	Does Not Apply	Center Line is High Impedance Off" State

KS000010

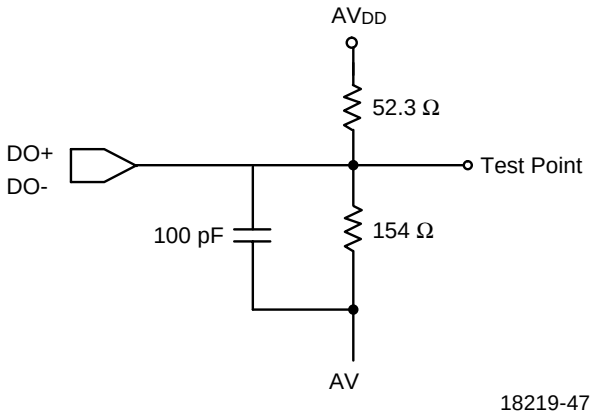
SWITCHING TEST CIRCUITS



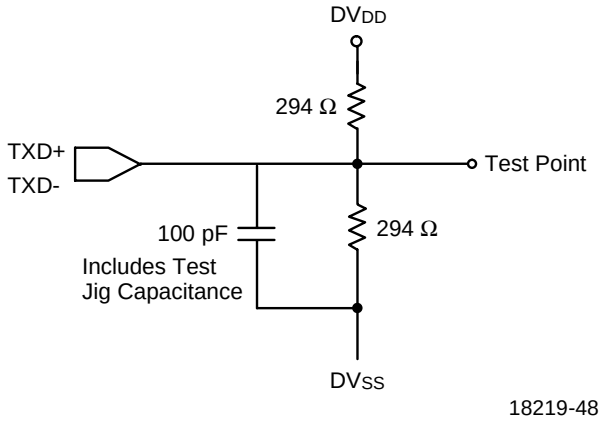
18219-46

Normal and Three-State Outputs

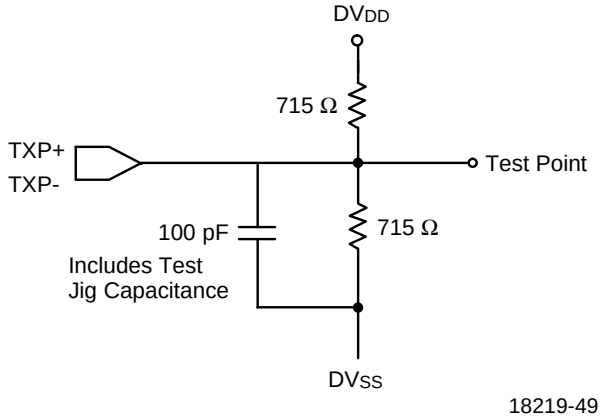
SWITCHING TEST CIRCUITS



AUI DO Switching Test Circuit



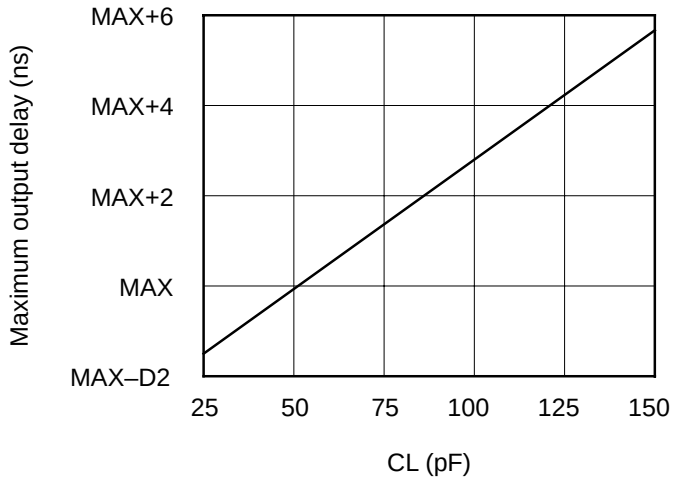
TXD Switching Test Circuit



TXP Outputs Test Circuit

ESTIMATED OUTPUT VALID DELAY VS. LOAD CAPACITANCE

Note that this graph represents change in output delay for estimated conditions.

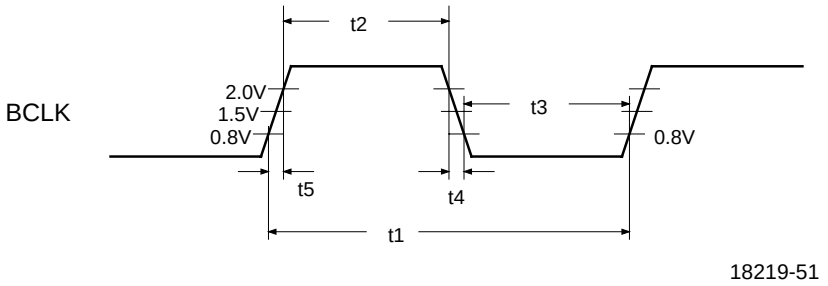


Notes: *This graph will not be linear outside of the CL range shown.*
MAX = maximum value given in A.C. characteristics table.

18219-50

Output Delay Versus Load Capacitance Under Estimated Conditions

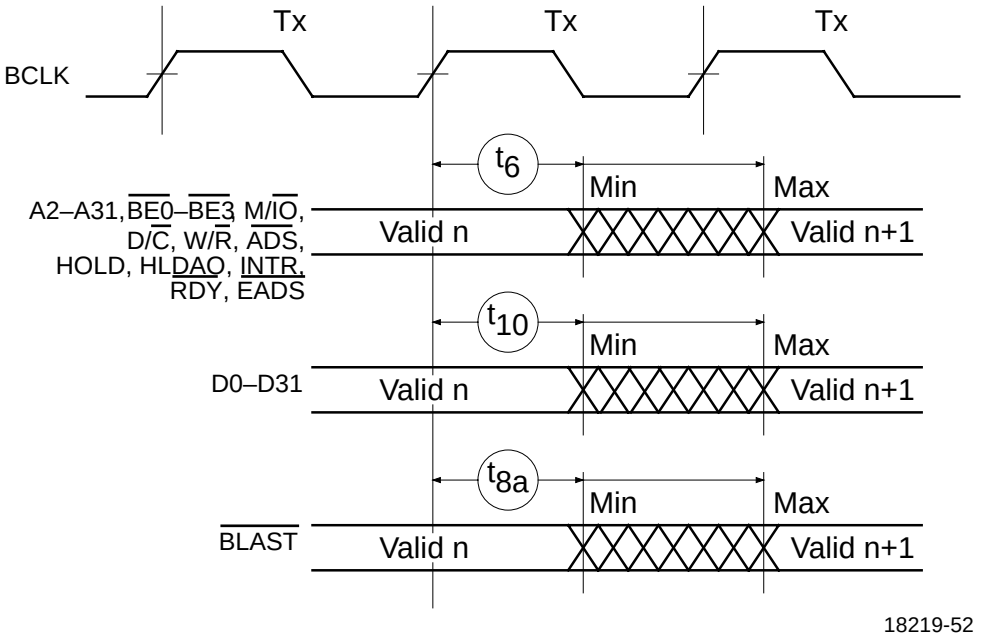
SWITCHING WAVEFORMS: SYSTEM BUS INTERFACE



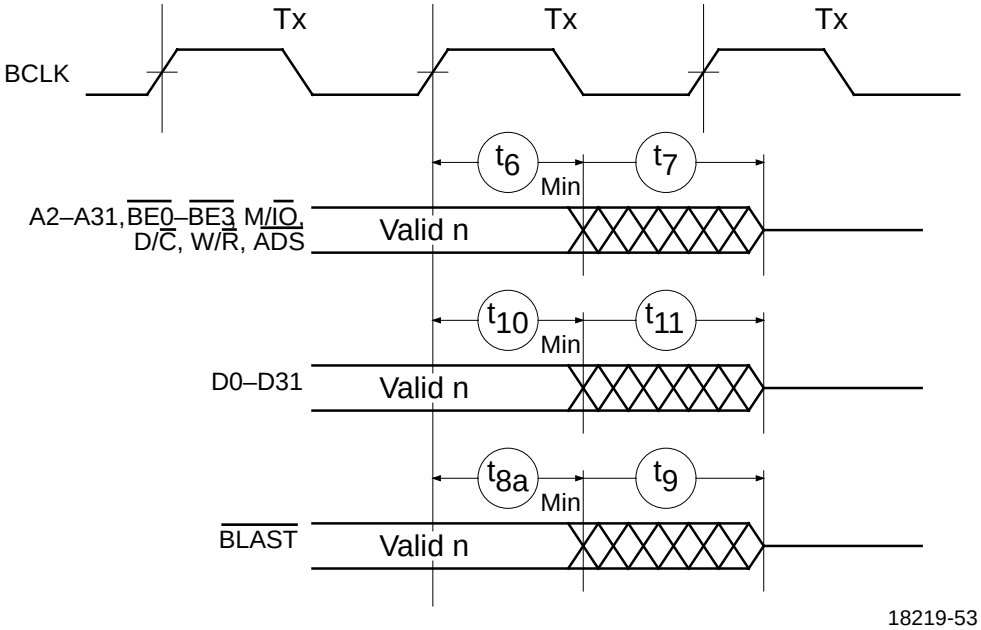
18219-51

BLCK Waveform

SWITCHING WAVEFORMS: SYSTEM BUS INTERFACE

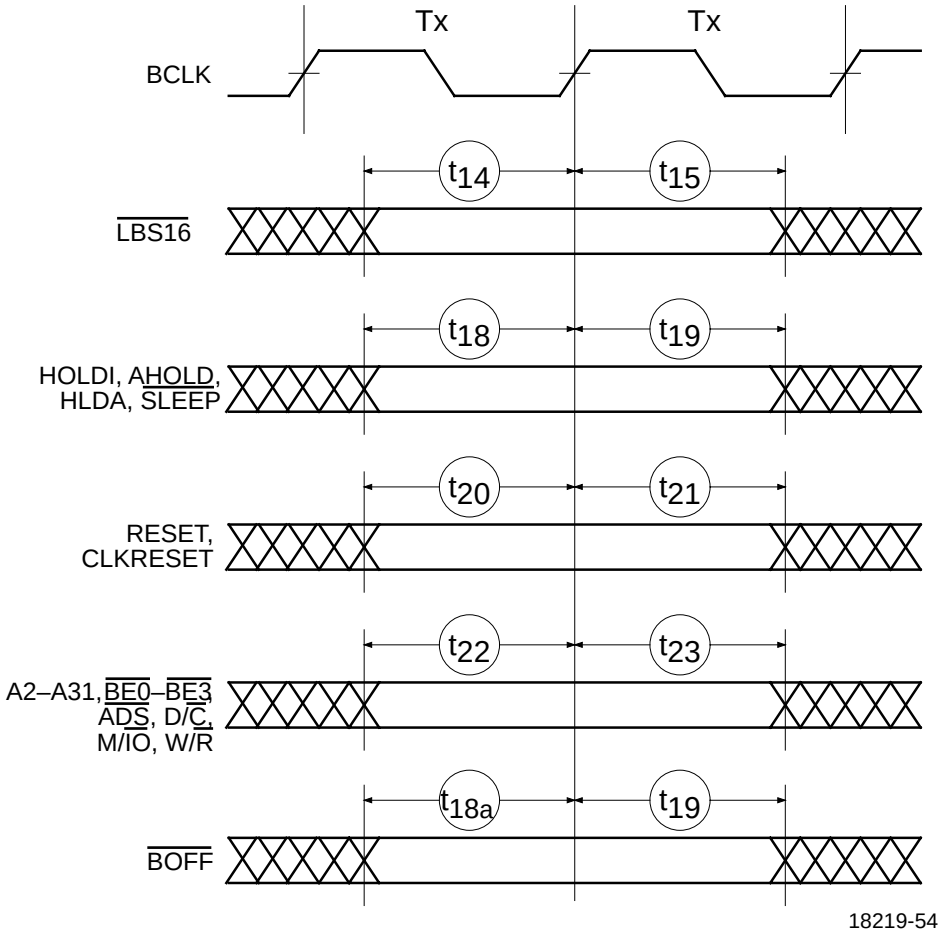


Output Valid Delay Timing

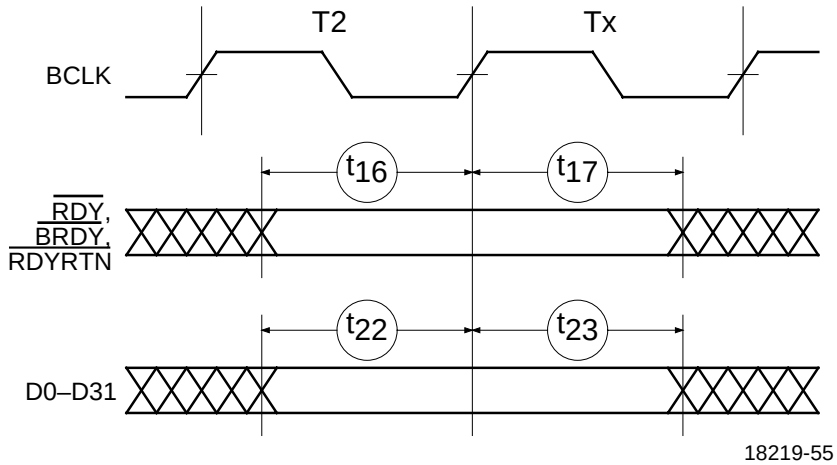


Maximum Float Delay Timing

SWITCHING WAVEFORMS: SYSTEM BUS INTERFACE

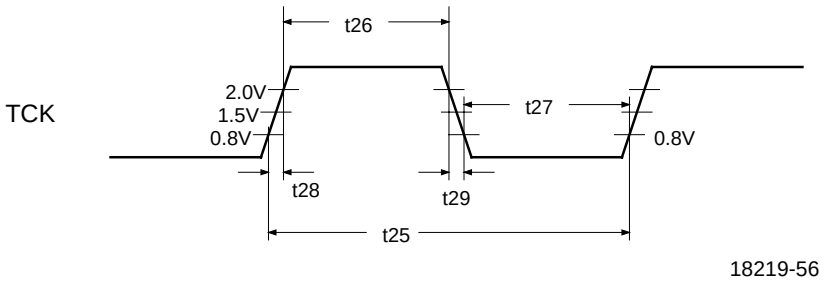


Input Setup and Hold Timing

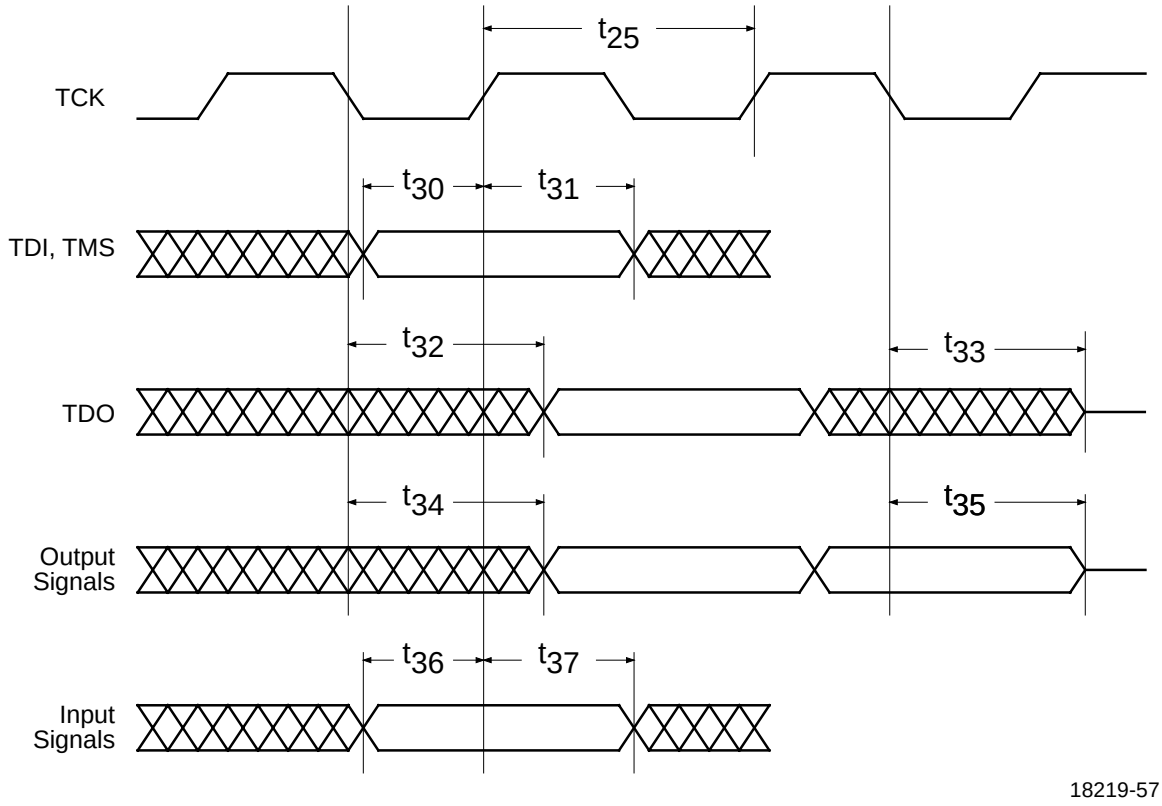


Ready Data Setup and Hold Timing

SWITCHING WAVEFORMS: SYSTEM BUS INTERFACE

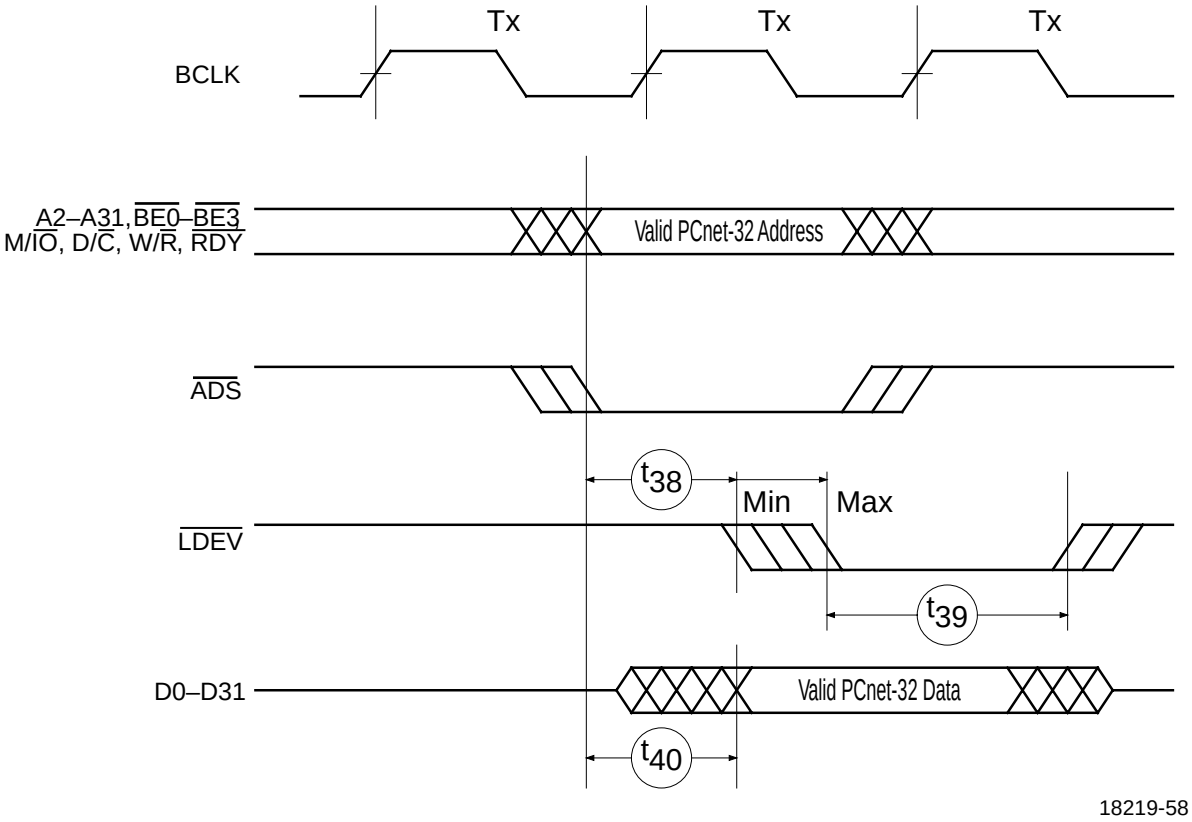


JTAG (IEEE 1149.1) TCK Waveform

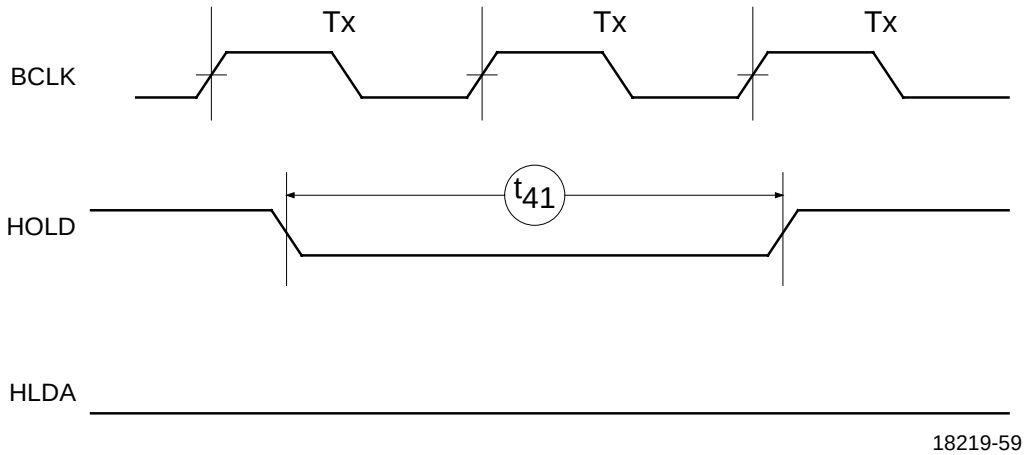


JTAG (IEEE 1149.1) Test Signal Timing

SWITCHING WAVEFORMS: SYSTEM BUS INTERFACE

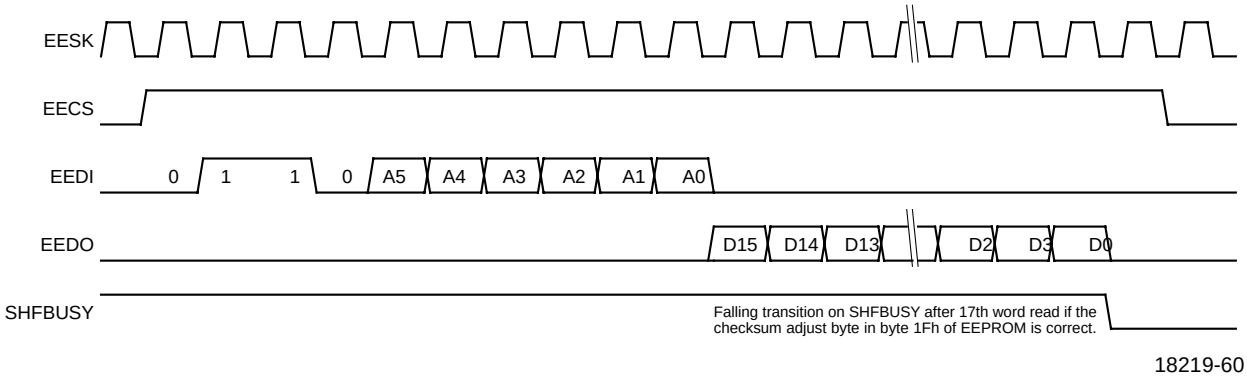


LDEV Timing and Data Bus Activation Timing

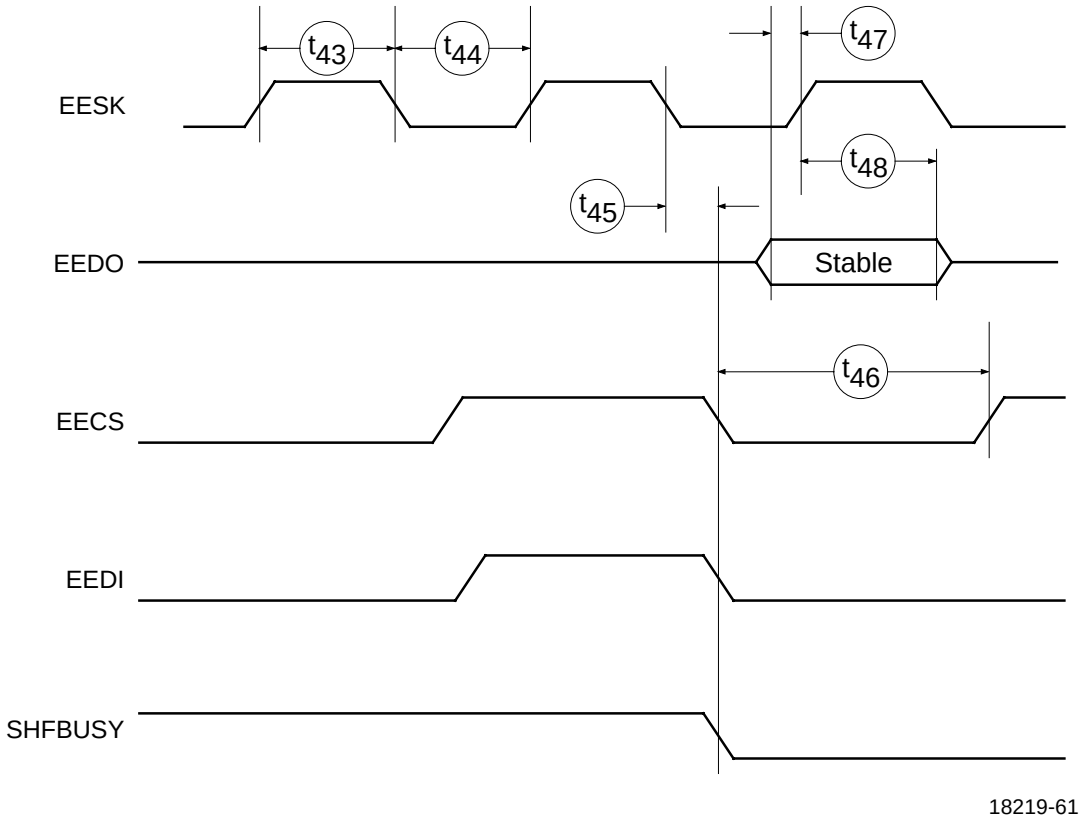


HOLD Inactive Timing

SWITCHING WAVEFORMS: SYSTEM BUS INTERFACE

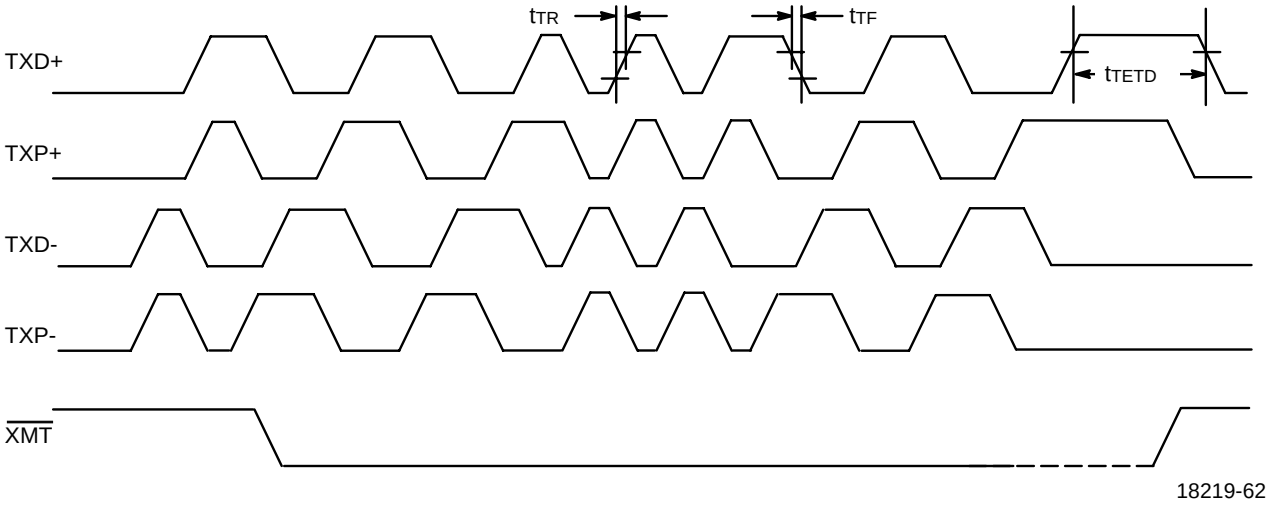


Automatic EEPROM Read Functional Timing

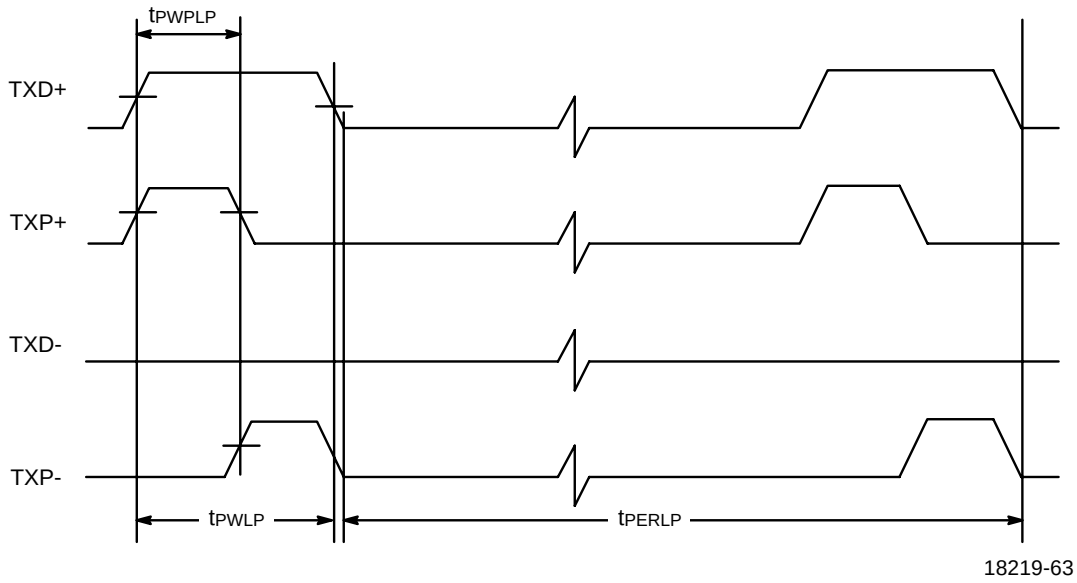


Automatic EEPROM Read Timing

SWITCHING WAVEFORMS: 10BASE-T INTERFACE

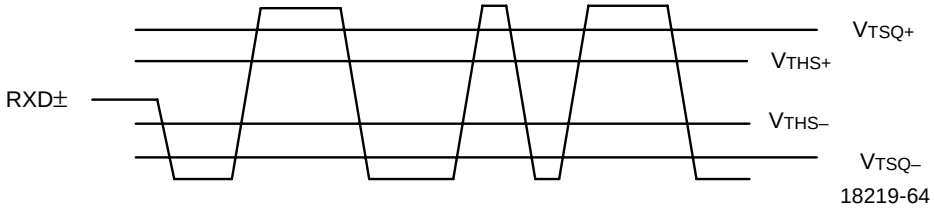


Transmit Timing

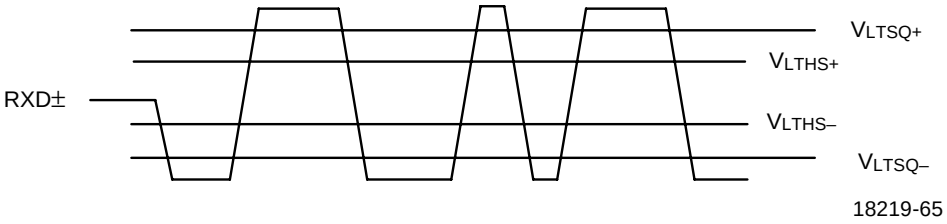


Idle Link Test Pulse

SWITCHING WAVEFORMS: 10BASE-T INTERFACE

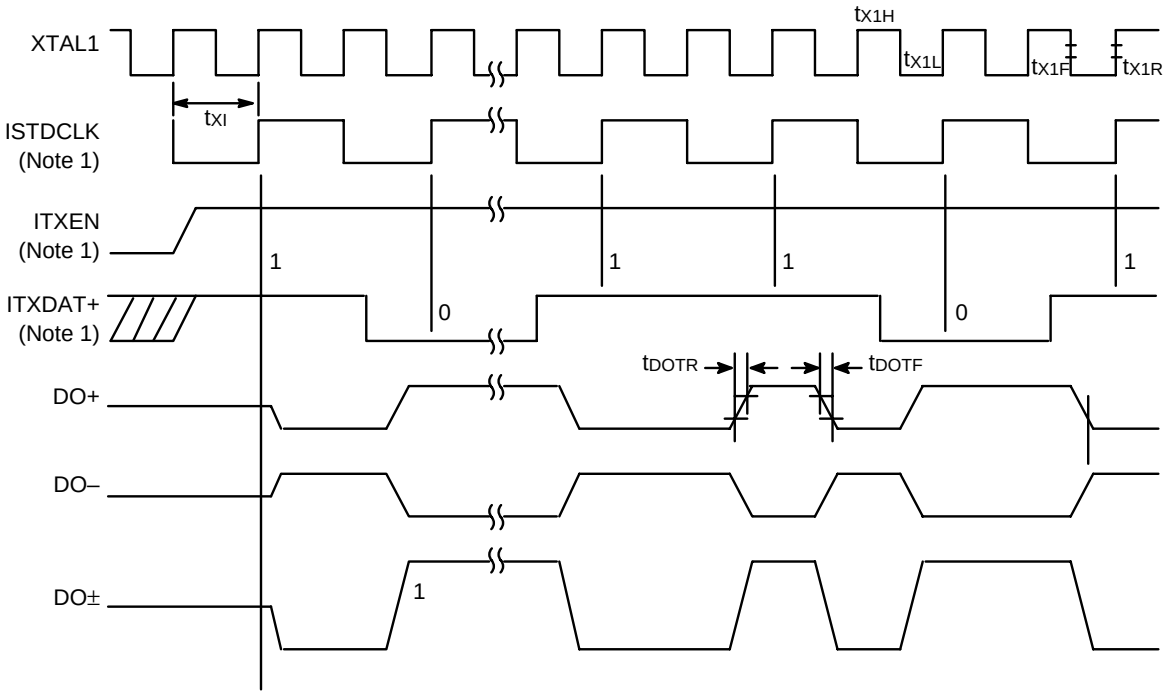


Receive Thresholds (LRT = 0)



Receive Threshold (LRT = 1)

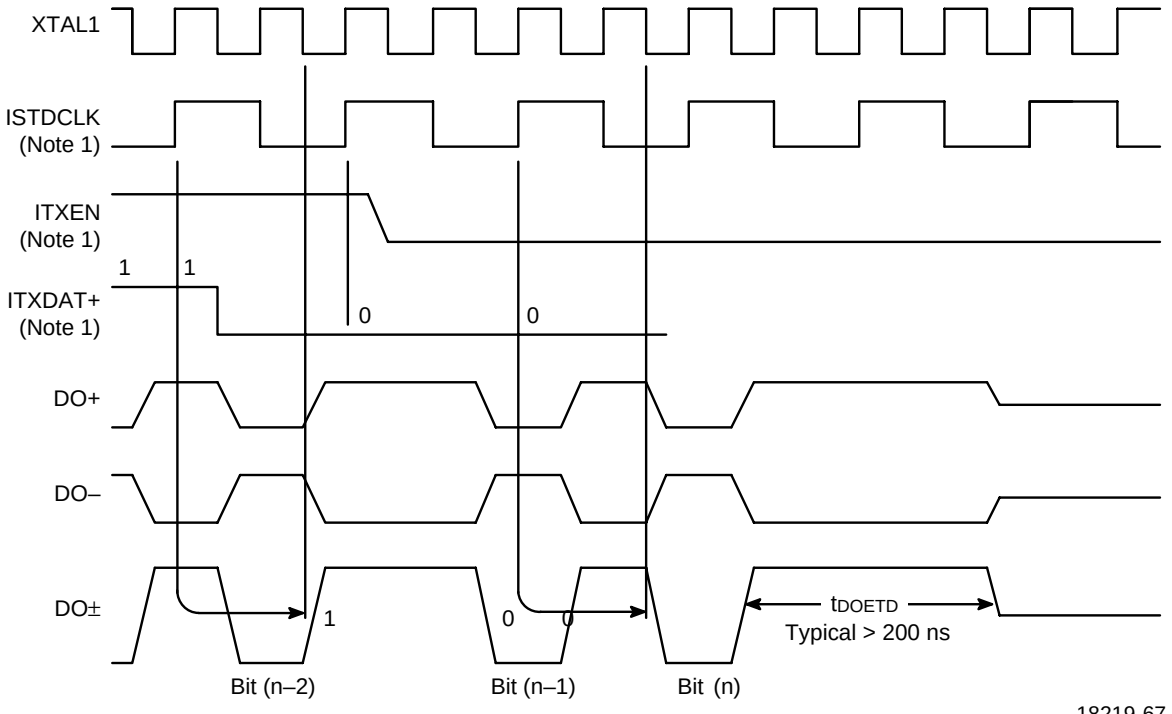
SWITCHING WAVEFORMS: AUI



Note 1: *Internal signal and is shown for clarification only.*

18219-66

Transmit Timing—Start of Packet

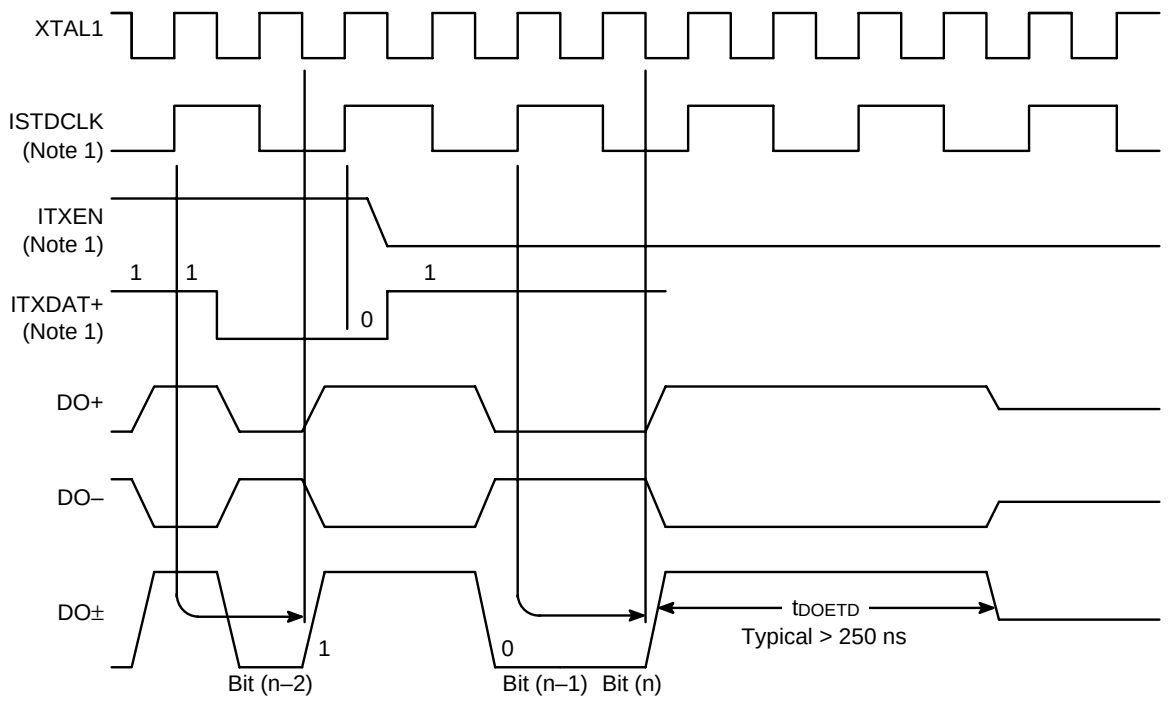


Note 1: *Internal signal and is shown for clarification only.*

18219-67

Transmit Timing—End of Packet (Last Bit = 0)

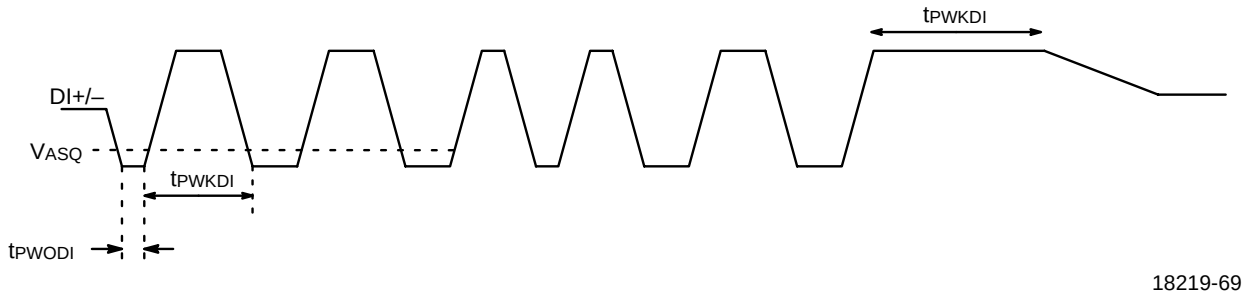
SWITCHING WAVEFORMS: AUI



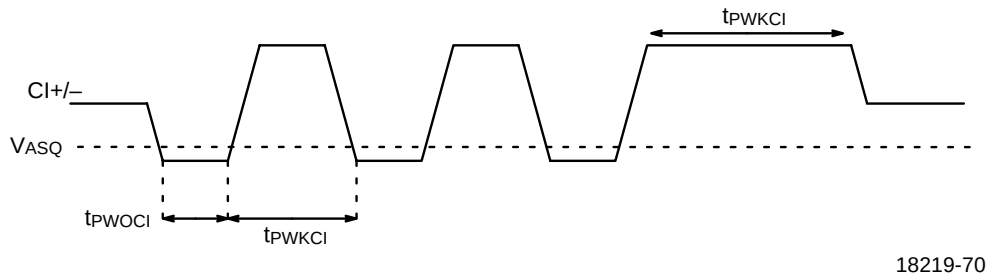
Note 1: *Internal signal and is shown for clarification only.*

Transmit Timing—End of Packet (Last Bit = 1)

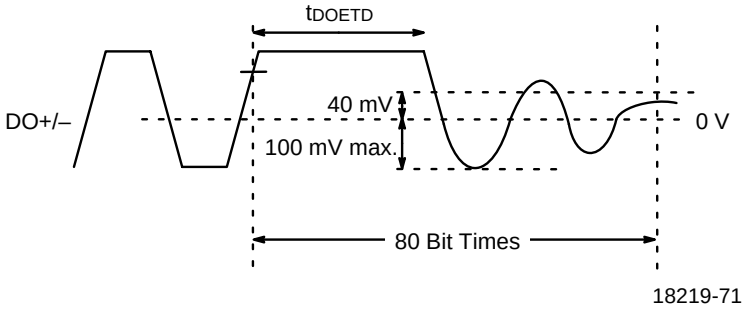
SWITCHING WAVEFORMS: AUI



Receiving Timing Diagram

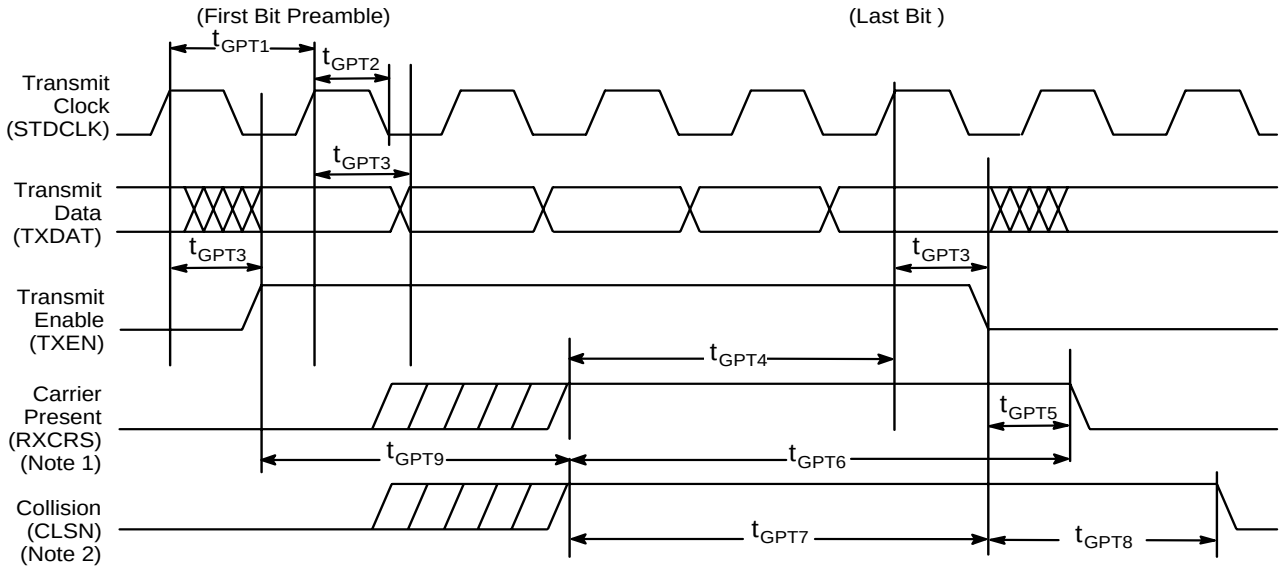


Collision Timing Diagram



Port DO ETD Waveform

SWITCHING WAVEFORMS: GPSI



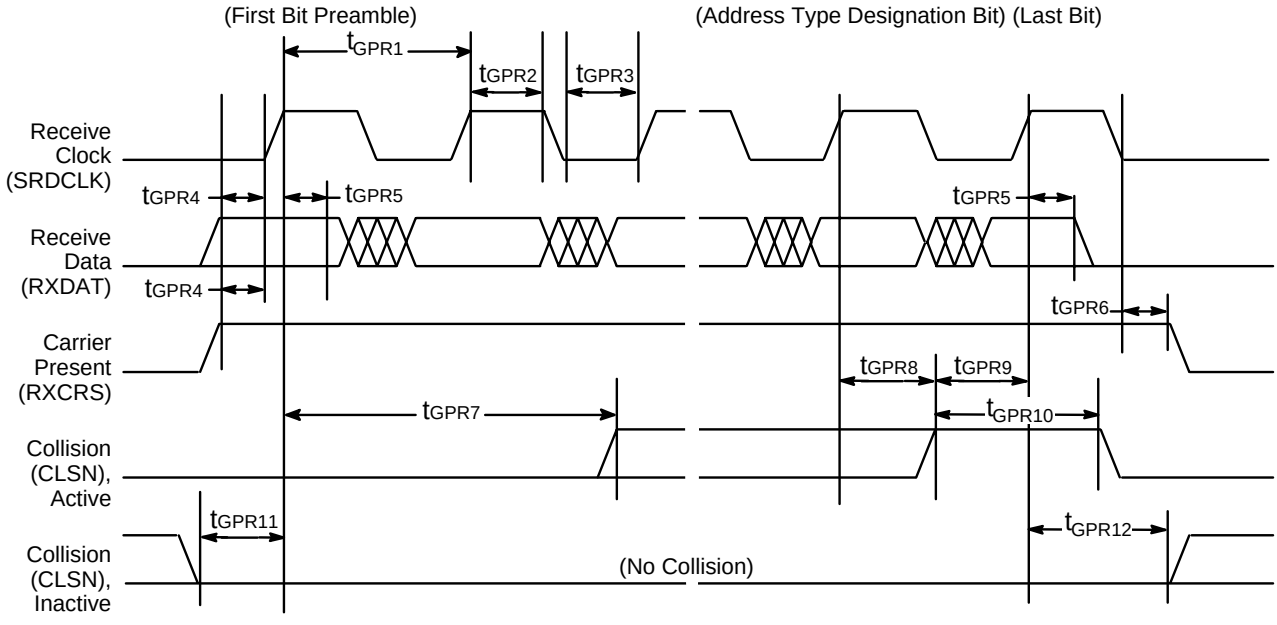
Notes:

1. If RXCRS is not present during transmission, LCAR bit in TMD2 will be set.

2. If CLSN is not present during or shortly after transmission, CERR in CSR0 will be set.

18219-72

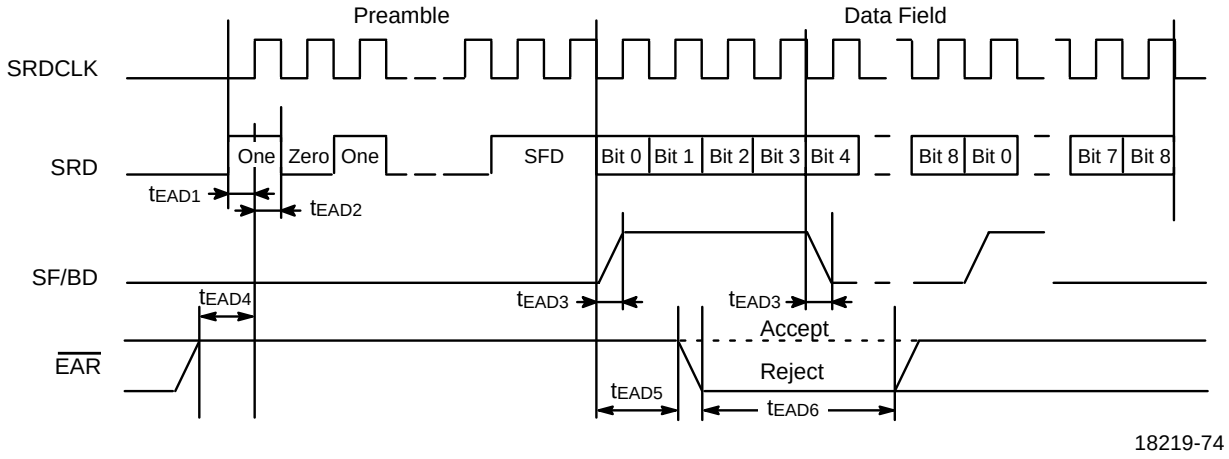
Transmit Timing



18219-73

Receive Timing

SWITCHING WAVEFORMS: EADI



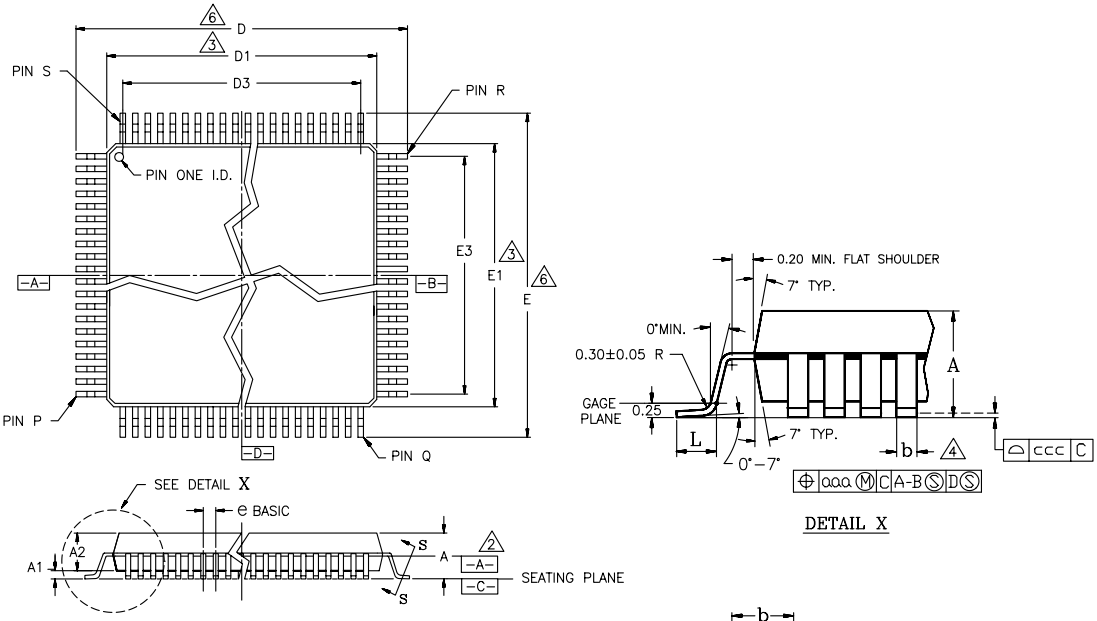
EADI Reject Timing

PHYSICAL DIMENSIONS*

PQR160

Plastic Quad Flat Pack Trimmed and Formed

PQR 160



PACKAGE	PQR 160			NOTES
JEDEC	MO-108(B)DD-1			
ASYMBOL	MIN	NOM	MAX	
A	—	—	3.95	
A1	0.25	—	—	
A2	3.20	3.40	3.60	
b	0.22	—	0.38	4
c	0.13	—	0.23	
D	31.00	31.20	31.40	
D1	27.90	28.00	28.10	3
D3	—	25.35 REF	—	
e	—	0.65 BASIC	—	7
E	31.00	31.20	31.40	
E1	27.90	28.00	28.10	3
E3	—	25.35 REF	—	
aaa	—	0.13	—	
ccc	0.10			
L	0.73	0.88	1.03	
P	40			
Q	80			
R	120			
S	160			

Dwg Rev AA; 08/00

NOTES:

1. ALL DIMENSIONS AND TOLERANCES CONFORM TO ANSI Y14.5M–1982.
2. DATUM PLANE -A- IS LOCATED AT THE MOLD PARTING LINE AND IS COINCIDENT WITH THE BOTTOM OF THE LEAD WHERE THE LEAD EXITS THE PLASTIC BODY.
3. DIMENSIONS "D1" AND "E1" DO NOT INCLUDE MOLD PROTRUSION. ALLOWABLE PROTRUSION IS 0.25mm PER SIDE. DIMENSIONS "D1" AND "E1" INCLUDE MOLD MISMATCH AND ARE DETERMINED AT DATUM PLANE -A-
4. DIMENSION "B" DOES NOT INCLUDE DAMBAR PROTRUSION.
5. CONTROLLING DIMENSIONS: MILLIMETER.
6. DIMENSIONS "D" AND "E" ARE MEASURED FROM BOTH INNERMOST AND OUTERMOST POINTS.
7. DEVIATION FROM LEAD–TIP TRUE POSITION SHALL BE WITHIN ±0.04 FOR PITCH ≤ 0.5 mm.
8. LEAD COPLANARITY SHALL BE WITHIN: 0.076 mm FOR DEVICES WITH LEAD PITCH OF 0.50 mm. COPLANARITY IS MEASURED PER SPECIFICATION 06–500.
9. HALF SPAN (CENTER OF PACKAGE TO LEAD TIP) SHALL BE WITHIN ±0.0085"

PCnet-32 Compatible Media Interface Modules

PCNET-32 COMPATIBLE 10BASE-T FILTERS AND TRANSFORMERS

The table below provides a sample list of PCnet-32 compatible 10BASE-T filter and transformer modules available from various vendors. Contact the respective manufacturer for a complete and updated listing of components.

Manufacturer	Part No.	Package	Filters and Transformers	Filters Transformers and Choke	Filters Transformers Dual Choke	Filters Transformers Dual Chokes
Bel Fuse	A556-2006-DE	16-pin 0.3" DIL	√			
Bel Fuse	0556-2006-00	14-pin SIP	√			
Bel Fuse	0556-2006-01	14-pin SIP			√	
Bel Fuse	0556-6392-00	16-pin 0.5" DIL			√	
Halo Electronics	FD02-101G	16-pin 0.3" DIL	√			
Halo Electronics	FD12-101G	16-pin 0.3" DIL		√		
Halo Electronics	FD22-101G	16-pin 0.3" DIL			√	
PCA Electronics	EPA1990A	16-pin 0.3" DIL	√			
PCA Electronics	EPA2013D	16-pin 0.3" DIL		√		
PCA Electronics	EPA2162	16-pin 0.3" SIP			√	
Pulse Engineering	PE-65421	16-pin 0.3" DIL	√			
Pulse Engineering	PE-65434	16-pin 0.3" SIL			√	
Pulse Engineering	PE-65445	16-pin 0.3" DIL			√	
Pulse Engineering	PE-65467	12-pin 0.5" SMT				√
Valor Electronics	PT3877	16-pin 0.3" DIL	√			
Valor Electronics	FL1043	16-pin 0.3" DIL			√	

PCnet-32 Compatible AUI Isolation Transformers

The table below provides a sample list of PCnet-32 compatible AUI isolation transformers available from various vendors. Contact the respective manufacturer for a complete and updated listing of components.

Manufacturer	Part No.	Package	Description
Bel Fuse	A553-0506-AB	16-pin 0.3" DIL	50 µH
Bel Fuse	S553-0756-AE	16-pin 0.3" SMD	75 µH
Halo Electronics	TD01-0756K	16-pin 0.3" DIL	75 µH
Halo Electronics	TG01-0756W	16-pin 0.3" SMD	75 µH
PCA Electronics	EP9531-4	16-pin 0.3" DIL	50 µH
Pulse Engineering	PE64106	16-pin 0.3" DIL	50 µH
Pulse Engineering	PE65723	16-pin 0.3" SMT	75 µH
Valor Electronics	LT6032	16-pin 0.3" DIL	75 µH
Valor Electronics	ST7032	16-pin 0.3" SMD	75 µH

PCnet-32 Compatible DC/DC Converters

The table below provides a sample list of PCnet-32 compatible DC/DC converters available from various vendors. Contact the respective manufacturer for a complete and updated listing of components.

Manufacturer	Part No.	Package	Voltage	Remote On/Off
Halo Electronics	DCU0-0509D	24-pin DIP	5/-9	No
Halo Electronics	DCU0-0509E	24-pin DIP	5/-9	Yes
PCA Electronics	EPC1007P	24-pin DIP	5/-9	No
PCA Electronics	EPC1054P	24-pin DIP	5/-9	Yes
PCA Electronics	EPC1078	24-pin DIP	5/-9	Yes
Valor Electronics	PM7202	24-pin DIP	5/-9	No
Valor Electronics	PM7222	24-pin DIP	5/-9	Yes

MANUFACTURER CONTACT INFORMATION

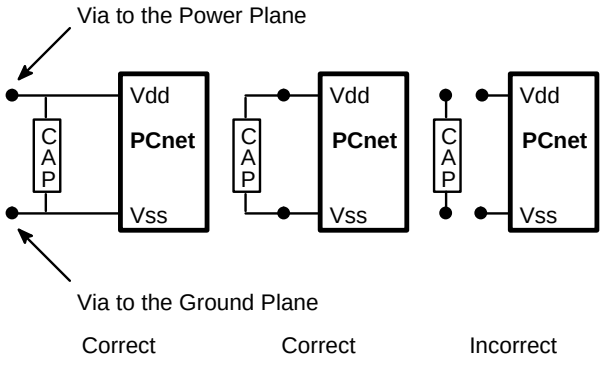
Contact the following companies for further information on their products:

Company		U.S. and Domestic	Asia	Europe
Bel Fuse	Phone:	(201) 432-0463	852-328-5515	33-1-69410402
	FAX:	(201) 432-9542	852-352-3706	33-1-69413320
Halo Electronics	Phone:	(415) 969-7313	65-285-1566	
	FAX:	(415) 367-7158	65-284-9466	
PCA Electronics	Phone:	(818)-892-0761	852-553-0165	33-1-44894800
(HPC in Hong Kong)	FAX:	(818)-894-5791	852-873-1550	33-1-42051579
Pulse Engineering	Phone:	(619) 674-8100	852-425-1651	353-093-24107
	FAX:	(619) 675-8262	852-480-5974	353-093-24459
Valor Electronics	Phone:	(619) 537-2500	852-513-8210	49-89-6923122
	FAX:	(619) 537-2525	852-513-8214	49-89-6926542

Recommendation for Power and Ground Decoupling

The mixed analog/digital circuitry in the PCnet-32 make it imperative to provide noise-free power and ground connections to the device. Without clean power and ground connections, a design may suffer from high bit error rates or may not function at all. Hence, it is highly recommended that the guidelines presented here are followed to ensure a reliable design.

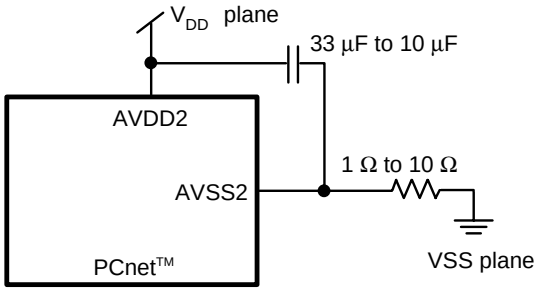
Decoupling/Bypass Capacitors: Adequate decoupling of the power and ground pins and planes is required by all PCnet-32 designs. This includes both low-frequency bulk capacitors and high frequency capacitors. It is recommended that **at least one** low-frequency bulk (e.g. 22 µF) decoupling capacitor be used in the area of the PCnet-32 device. The bulk capacitor(s) should be connected directly to the power and ground planes. In addition, **at least 8** high frequency decoupling capacitors (e.g. 0.1 µF multilayer ceramic capacitors) should be used around the periphery of the PCnet-32 device to prevent power and ground bounce from affecting device operation. To reduce the inductance between the power and ground pins and the capacitors, the pins should be connected directly to the capacitors, rather than through the planes to the capacitors. The suggested connection scheme for the capacitors is shown in the figure below. Note also that the traces connecting these pins to the capacitors should be as wide as possible to reduce inductance (15 mils is desirable).



The most critical pins in the layout of a PCnet-32 design are the 4 analog power and 2 analog ground pins, AVDD[1-4] and AVSS[1-2], respectively. All of these pins are located in one corner of the device, the “analog corner”. Specific functions and layout requirements of the analog power and ground pins are given below.

AVSS1 and AVDD3: These pins provide the power and ground for the Twisted Pair and AUI drivers. In addition AVSS1 serves as the ground for the logic interfaces in the 20 MHz Crystal Oscillator. Hence, these pins can be very noisy. A dedicated 0.1 µF capacitor between these pins is recommended.

AVSS2 and AVDD2: These pins are the **most critical** pins on the PCnet-32 device because they provide the power and ground for the phase-lock loop (PLL) portion of the chip. The voltage-controlled oscillator (VCO) portion of the PLL is sensitive to noise in the 60 kHz – 200 kHz range. To prevent noise in this frequency range from disrupting the VCO, it is **strongly recommended** that the low-pass filter shown below be implemented on these pins.



To determine the value for the resistor and capacitor, the formula is:

$$R * C \geq 88$$

where R is in ohms and C is in microfarads. Some possible combinations are given below. To minimize the voltage drop across the resistor, the R value should not be more than 10 Ω.

R	C
2.7 Ω	33 µF
4.3 Ω	22 µF
6.8 Ω	15 µF
10 Ω	10 µF

AVSS2 and AVDD2/AVDD4: These pins provide power and ground for the AUI and twisted pair receive circuitry. In addition, as mentioned earlier, AVSS2 and AVDD2 provide power and ground for the phase-lock loop portion of the chip. Except for the filter circuit already mentioned, no specific decoupling is necessary on these pins.

AVDD1: AVDD1 provides power for the control and interface logic in the PLL. Ground for this logic is provided by digital ground pins. No specific decoupling is necessary on this pin.

Special Note for Adapter Cards: In adapter card designs, it is important to utilize all available power and ground pins available on the bus edge connector. In addition, the connection from the bus edge connector to the power or ground plane should be made through more than one via and with wide traces (15 mils desirable) wherever possible. Following these recommendations results in minimal inductance in the power and ground paths. By minimizing this inductance, ground bounce is minimized.

Alternative Method for Initialization

The PCnet-32 controller may be initialized by performing I/O writes only. That is, data can be written directly to the appropriate control and status registers (CSR) instead of reading from the Initialization Block in memory. The registers that must be written are shown in the table below. These books are followed by writing the START bit in CSR0.

Control and Status Register	Comment
CSR8	LADRF[15:0]
CSR9	LADRF[31:16]
CSR10	LADRF[47:32]
CSR11	LADRF[63:48]
CSR12	PADR[15:0]
CSR13	PADR[31:16]
CSR14	PADR[47:32]
CSR15	Mode
CSR24-25	BADR
CSR30-31	BADX
CSR47	POLLINT
CSR76	RCVRL
CSR78	XMTRL

Note: *The INIT bit must not be set or the initialization block will be accessed instead.*

AMD

Introduction of the Look-Ahead Packet Processing (LAPP) Concept

A driver for the PCnet-32 controller would normally require that the CPU copy receive frame data from the controller's buffer space to the application's buffer space after the entire frame has been received by the controller. For applications that use a ping-pong windowing style, the traffic on the network will be halted until the current frame has been completely processed by the entire application stack. This means that the time between last byte of a receive frame arriving at the client's Ethernet controller and the client's transmission of the first byte of the next outgoing frame will be separated by:

1. the time that it takes the client's CPU's interrupt procedure to pass software control from the current task to the driver
2. plus the time that it takes the client driver to pass the header data to the application and request an application buffer
3. plus the time that it takes the application to generate the buffer pointer and then return the buffer pointer to the driver
4. plus the time that it takes the client driver to transfer all of the frame data from the controller's buffer space into the application's buffer space and then call the application again to process the complete frame
5. plus the time that it takes the application to process the frame and generate the next outgoing frame
6. plus the time that it takes the client driver to set up the descriptor for the controller and then write a TDMD bit to CSR0

The sum of these times can often be about the same as the time taken to actually transmit the frames on the wire, thereby yielding a network utilization rate of less than 50%.

An important thing to note is that the PCnet-32 controller's data transfers to its buffer space are such that the system bus is needed by the PCnet-32 controller for approximately 4% of the time. This leaves 96% of the sytem bus bandwidth for the CPU to perform some of the inter-frame operations *in advance of the completion of network receive activity*, if possible. The question then becomes: how much of the tasks that need to be performed between reception of a frame and transmission of the next frame can be performed *before* the reception of the frame actually ends at the network, and how can the CPU be instructed to perform these tasks during the network reception time?

The answer depends upon exactly what is happening in the driver and application code, but the steps that can be performed at the same time as the receive data are arriving include as much as the first three steps and part of the fourth step shown in the sequence above. By performing these steps before the entire frame has arrived, the frame throughput can be substantially increased.

A good increase in performance can be expected when the first three steps are performed before the end of the network receive operation. A much more significant performance increase could be realized if the PCnet-32 controller could place the frame data directly into the application's buffer space; (i.e. eliminate the need for step four.) In order to make this work, it is necessary that the application buffer pointer be determined before the frame has completely arrived, then the buffer pointer in the next desriptor for the receive frame would need to be modified in order to direct the PCnet-32 controller to write directly to the application buffer. More details on this operation will be given later.

An alternative modification to the existing system can gain a smaller, but still significant improvement in performance. This alternative leaves step four unchanged in that the CPU is still required to perform the copy operation, but it allows a large portion of the copy operation to be done before the frame has been completely received by the controller, (i.e. the CPU can perform the copy operation of the receive data from the PCnet-32 controller's buffer space into the application buffer space *before* the frame data has completely arrived from the network.) This allows the copy operation of step four to be performed concurrently with the arrival of network data, rather than sequentially, following the end of network receive activity.

Outline of the LAPP Flow:

This section gives a suggested outline for a driver that utilizes the LAPP feature of the PCnet-32 controller.

Note: *The labels in the following text are used as references in the timeline diagram that follows.*

SETUP:

The driver should set up descriptors in groups of 3, with the OWN and STP bits of each set of three descriptors to read as follows: 11b, 10b, 00b.

An option bit (LAPPEN) exists in CSR3, bit position 5. The software should set this bit. When set, the LAPPEN bit directs the PCnet-32 controller to generate an INTERRUPT when STP has been written to a receive descriptor by the PCnet-32 controller.

FLOW:

The PCnet-32 controller polls the current receive descriptor at some point in time before a message arrives. The PCnet-32 controller determines that this receive buffer is OWNed by the PCnet-32 controller and it stores the descriptor information to be used when a message does arrive.

N0: Frame preamble appears on the wire, followed by SFD and destination address.

N1: The 64th byte of frame data arrives from the wire. This causes the PCnet-32 controller to begin frame data DMA operations to the first buffer.

C0: When the 64th byte of the message arrives, the PCnet-32 controller performs a look-ahead operation to the next receive descriptor. This descriptor should be owned by the PCnet-32 controller.

C1: The PCnet-32 controller intermittently requests the bus to transfer frame data to the first buffer as it arrives on the wire.

S0: The driver remains idle.

C2: When the PCnet-32 controller has completely filled the first buffer, it writes status to the first descriptor.

C3: When the first descriptor for the frame has been written, changing ownership from the PCnet-32 controller to the CPU, the PCnet-32 controller will generate an SRP INTERRUPT. (This interrupt appears as a RINT interrupt in CSR0.)

S1: The SRP INTERRUPT causes the CPU to switch tasks to allow the PCnet-32 controller's driver to run.

C4: During the CPU interrupt–generated task switching, the PCnet-32 controller is performing a look-ahead operation to the third descriptor. At this point in time, the third descriptor is owned by the CPU. *[Note: Even though the third buffer is not owned by the PCnet-32 controller, existing AMD Ethernet controllers will continue to perform data DMA into the buffer space that the controller already owns (i.e. buffer number 2). The controller does not know if buffer space in buffer number 2 will be sufficient or not, for this frame, but it has no way to tell except by trying to move the entire message into that space. Only when the message does not fit will it signal a buffer error condition – there is no need to panic at the point that it discovers that it does not yet own descriptor number 3.]*

S2: The first task of the driver's interrupt service routine is to collect the header information from the PCnet-32 controller's first buffer and pass it to the application.

S3: The application will return an application buffer pointer to the driver. The driver will add an offset to the application data buffer pointer, since the PCnet-32 controller will be placing the first portion of the message into the first and second buffers. (The modified application data buffer pointer will only be directly used by the PCnet-32 controller when it reaches the third buffer.) The driver will place the modified data buffer pointer into the final descriptor of the group (#3) and will grant ownership of this descriptor to the PCnet-32 controller.

C5: Interleaved with S2, S3 and S4 driver activity, the PCnet-32 controller will write frame data to buffer number 2.

S4: The driver will next proceed to copy the contents of the PCnet-32 controller's first buffer to the *beginning* of the application space. This copy will be to the exact (unmodified) buffer pointer that was passed by the application.

S5: After copying all of the data from the first buffer into the beginning of the application data buffer, the driver will begin to poll the ownership bit of the second descriptor. The driver is waiting for the PCnet-32 controller to finish filling the second buffer.

C6: At this point, knowing that it had not previously owned the third descriptor, and knowing that the current message has not ended (there is more data in the fifo), the PCnet-32 controller will make a "last ditch look-ahead" to the final (third) descriptor; This time, the ownership will be TRUE (i.e. the descriptor belongs to the controller), because the driver wrote the application pointer into this descriptor and then changed the ownership to give the descriptor to the PCnet-32 controller back at S3. Note that if steps S1, S2 and S3 have not completed at this time, a BUFF error will result.

C7: After filling the second buffer and performing the last chance look-ahead to the next descriptor, the PCnet-32 controller will write the status and change the ownership bit of descriptor number 2.

S6: After the ownership of descriptor number 2 has been changed by the PCnet-32 controller, the *next* **driver** poll of the 2nd descriptor will show ownership granted to the CPU. The driver now copies the data from buffer number 2 into the "middle section"

of the application buffer space. This operation is interleaved with the C7 and C8 operations.

C8: The PCnet-32 controller will perform data DMA to the last buffer, whose pointer is pointing to application space. Data entering the last buffer will not need the infamous “double copy” that is required by existing drivers, since it is being placed directly into the application buffer space.

N2: The message on the wire ends.

S7: When the driver completes the copy of buffer number 2 data to the application buffer space, it begins polling descriptor number 3.

C9: When the PCnet-32 controller has finished all data DMA operations, it writes status and changes ownership of descriptor number 3.

S8: The driver sees that the ownership of descriptor number 3 has changed, and it calls the application to tell the application that a frame has arrived.

S9: The application processes the received frame and generates the next TX frame, placing it into a TX buffer.

S10:The driver sets up the TX descriptor for the PCnet-32 controller.

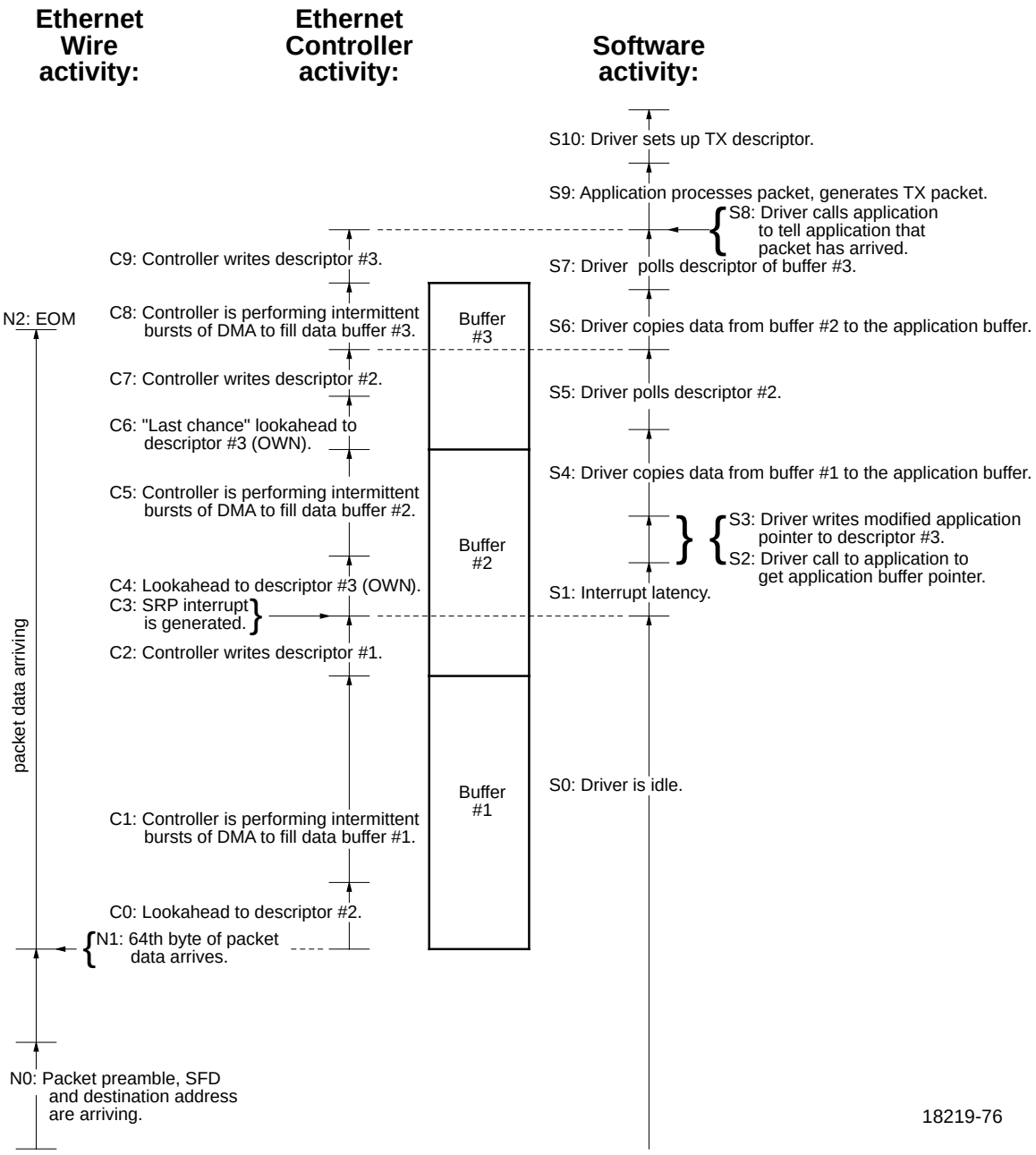


Figure 1. LAPP Timeline

LAPP Software Requirements

Software needs to set up a receive ring with descriptors formed into groups of 3. The first descriptor of each group should have OWN = 1 and STP = 1, the second descriptor of each group should have OWN = 1 and STP = 0. The third descriptor of each group should have OWN = 0 and STP = 0. The size of the first buffer (as indicated in the first descriptor), should be ***at least*** equal to the largest expected header size; However, for maximum efficiency of CPU utilization, the first buffer size should be larger than the header size. It should be equal to the expected number of message bytes, minus the time needed for Interrupt latency and minus the application call latency, minus the time needed for the driver to write to the third descriptor, minus the time needed for the driver to copy data from buffer #1 to the application buffer space, and minus the time needed for the driver to copy data from buffer #2 to the application buffer space. Note that the time needed for the copies performed by the driver depends upon the sizes of the 2nd and 3rd buffers, and that the sizes of the second and third buffers need to be set according to the time needed for the data copy operations! This means that an iterative self-adjusting mechanism needs to be placed into the software to determine the correct buffer sizing for optimal operation. Fixed values for buffer sizes may be used; In such a case, the LAPP method will still provide a significant performance increase, but the performance increase will not be maximized.

The following diagram illustrates this setup for a receive ring size of 9:

LAPP Rules for Parsing of Descriptors

When using the LAPP method, software must use a modified form of descriptor *parsing* as follows:

Software will examine OWN and STP to determine where a RCV frame begins. RCV frames will only begin in buffers that have OWN = 0 and STP = 1.

Software shall assume that a frame continues until it finds *either* ENP = 1 or ERR= 1.

Software must discard all descriptors with OWN = 0 and STP = 0 and move to the next descriptor when searching for the beginning of a new frame; ENP and ERR should be ignored by software during this search.

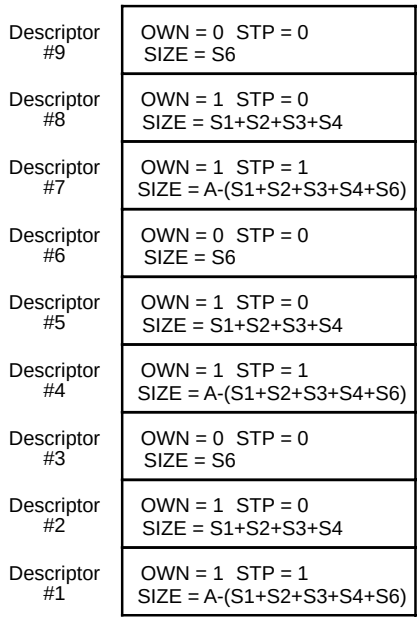
Software cannot change an STP value in the receive descriptor ring after the initial setup of the ring is complete, even if software has ownership of the STP descriptor unless the previous STP descriptor in the ring is also OWNED by the software.

When LAPPEN = 1, then hardware will use a modified form of descriptor parsing as follows:

The controller will examine OWN and STP to determine where to begin placing a RCV frame. A new RCV frame will only begin in a buffer that has OWN = 1 and STP = 1.

The controller will always obey the OWN bit for determining whether or not it may use the next buffer for a chain.

The controller will always mark the end of a frame with *either* ENP = 1 or ERR = 1.



A = Expected message size in bytes
S1 = Interrupt latency
S2 = Application call latency
S3 =Time needed for driver to write to third descriptor
S4 = Time needed for driver to copy data from buffer #1 to application buffer space
S6 = Time needed for driver to copy data from buffer #2 to application buffer space

Note that the times needed for tasks S1, S2, S3, S4, and S6 should be divided by 0.8 microseconds to yield an equivalent number of network byte times before subtracting these quantities from the expected message size A.

18219-77

Figure 2. LAPP 3 Buffering Grouping

The controller will *discard* all descriptors with **OWN = 1** and STP = 0 and move to the next descriptor ***when searching for a place to begin a new frame***. It discards these desciptors by simply changing the ownership bit from OWN=1 to OWN = 0. Such a descriptor is unused for receive purposes by the controller, and the driver must recognize this. (The driver will recognize this if it follows the software rules.)

The controller will *ignore* all descriptors with **OWN = 0** and STP = 0 and move to the next descriptor ***when searching for a place to begin a new frame***. In other words, the controller is allowed to skip entries in the ring that it does not own, but only when it is looking for a place to begin a new frame.

Some Examples of LAPP Descriptor Interaction

Choose an expected frame size of 1060 bytes.

Choose buffer sizes of 800, 200 and 200 bytes.

1. Assume that a 1060 byte frame arrives correctly, and that the timing of the early interrupt and the software is smooth. The descriptors will have changed from:

Descriptor Number	Before the Frame Arrived			After the Frame has Arrived			Comments (After Frame Arrival)
	OWN	STP	ENP*	OWN	STP	ENP*	
1	1	1	X	0	1	0	Bytes 1–800
2	1	0	X	0	0	0	Bytes 801–1000
3	0	0	X	0	0	1	Bytes 1001–1060
4	1	1	X	1	1	X	Controller's current location
5	1	0	X	1	0	X	Not yet used
6	0	0	X	0	0	X	Not yet used
etc.	1	1	X	1	1	X	Not yet used

** ENP or ERR*

2. Assume that instead of the expected 1060 byte frame, a 900 byte frame arrives, either because there was an error in the network, or because this is the last frame in a file transmission sequence.

Descriptor Number	Before the Frame Arrived			After the Frame has Arrived			Comments (After Frame Arrival)
	OWN	STP	ENP*	OWN	STP	ENP*	
1	1	1	X	0	1	0	Bytes 1–800
2	1	0	X	0	0	1	Bytes 801–900
3	0	0	X	0	0	?**	Discarded buffer
4	1	1	X	1	1	X	Controller's current location
5	1	0	X	1	0	X	Not yet used
6	0	0	X	0	0	X	Not yet used
etc.	1	1	X	1	1	X	Not yet used

**ENP or ERR*

*** Note that the PCnet-32 controller* might *write a ZERO to ENP location in the 3rd descriptor. Here are the two possibilities:*

1) If the controller finishes the data transfers into buffer number 2 after *the driver writes the application's modified buffer pointer into the third descriptor, then the controller will write a ZERO to ENP for this buffer and will write a ZERO to OWN and STP.*

2) If the controller finishes the data transfers into buffer number 2 before *the driver writes the application's modified buffer pointer into the third descriptor, then the controller will complete the frame in buffer number two and then skip the then unowned third buffer. In this case, the PCnet-32 controller will not have had the opportunity to RESET the ENP bit in this descriptor, and it is possible that the software left this bit as ENP=1 from the last time through the ring. Therefore, the software*

must *treat the location as a don't care; The rule is, after finding ENP=1 (or ERR=1) in descriptor number 2, the software must ignore ENP bits until it finds the next STP=1.*

3. Assume that instead of the expected 1060 byte frame, a 100 byte frame arrives, because there was an error in the network, or because this is the last frame in a file transmission sequence, or perhaps because it is an acknowledge frame.

Descriptor Number	Before the Frame Arrived			After the Frame has Arrived			Comments (After Frame Arrival)
	OWN	STP	ENP*	OWN	STP	ENP*	
1	1	1	X	0	1	1	Bytes 1–100
2	1	0	X	0	0	0***	Discarded buffer
3	0	0	X	0	0	?**	Discarded buffer
4	1	1	X	1	1	X	Controller's current location
5	1	0	X	1	0	X	Not yet used
6	0	0	X	0	0	X	Not yet used
etc.	1	1	X	1	1	X	Not yet used

** ENP or ERR*

*** Same as note in case 2 above, except that in this case, it is very unlikely that the driver can respond to the interrupt and get the pointer from the application before the PCnet-32 controller has completed its poll of the next descriptors. This means that for almost all occurrences of this case, the PCnet-32 controller will not find the OWN bit set for this descriptor and therefore, the ENP bit will almost always contain the old value, since the PCnet-32 controller will not have had an opportunity to modify it.*

**** Note that even though the PCnet-32 controller will write a ZERO to this ENP location, the software should treat the location as a don't care, since after finding the ENP=1 in descriptor number 2, the software should ignore ENP bits until it finds the next STP=1.*

Buffer Size Tuning

For maximum performance, buffer sizes should be adjusted depending upon the expected frame size and the values of the interrupt latency and application call latency. The best driver code will minimize the CPU utilization while also minimizing the latency from frame end on the network to frame sent to application from driver (frame latency). These objectives are aimed at increasing throughput on the network while decreasing CPU utilization.

Note that the buffer sizes in the ring may be altered at any time that the CPU has ownership of the corresponding descriptor. The best choice for buffer sizes will maximize the time that the driver is swapped out, while minimizing the time from the last byte written by the PCnet-32 controller to the time that the data is passed from the driver to the application. In the diagram, this corresponds to maximizing S0, while minimizing the time between C9 and S8. (The timeline happens to show a minimal time from C9 to S8.)

Note that by increasing the size of buffer number 1, we increase the value of S0. However, when we increase the size of buffer number 1, we also increase the value of S4. If the size of buffer number 1 is too large, then the driver will not have enough time to perform tasks S2, S3, S4, S5 and S6. The result is that there will be delay from the execution of task C9 until the execution of task S8. A perfectly timed system will have the values for S5 and S7 at a minimum.

An average increase in performance can be achieved if the general guidelines of buffer sizes in figure 2 is followed. However, as was noted earlier, the correct sizing for buffers will depend upon the expected message size. There are two problems with relating expected message size with the correct buffer sizing:

1. Message sizes cannot always be accurately predicted, since a single application may expect different message sizes at different times, therefore, the buffer sizes chosen will not always maximize throughput.
2. Within a single application, message sizes might be somewhat predictable, but when the same driver is to be shared with multiple applications, there may not be a common predictable message size.

Additional problems occur when trying to define the correct sizing because the correct size also depends

upon the interrupt latency, which may vary from system to system, depending upon both the hardware and the software installed in each system. In order to deal with the unpredictable nature of the message size, the driver can implement a self-tuning mechanism that examines the amount of time spent in tasks S5 and S7 as such: While the driver is polling for each descriptor, it could count the number of poll operations performed and then adjust the number 1 buffer size to a larger value, by adding "t" bytes to the buffer count, if the number of poll operations was greater than "x". If fewer than "x" poll operations were needed for each of S5 and S7, then the software should adjust the buffer size to a smaller value by, subtracting "y" bytes from the buffer count. Experiments with such a tuning mechanism must be performed to determine the best values for "X" and "y."

Note whenever the size of buffer number 1 is adjusted, buffer sizes for buffer number 2 and buffer 3 should also be adjusted.

In some systems the typical mix of receive frames on a network for a client application consists mostly of large data frames, with very few small frames. In this case, for maximum efficiency of buffer sizing, *when a frame arrives under a certain size limit, the driver should* ***not*** *adjust the buffer sizes in response to the short frame.*

An Alternative LAPP Flow – the TWO Interrupt Method

An alternative to the above suggested flow is to use two interrupts, one at the start of the Receive frame and the other at the end of the receive frame, instead of just looking for the SRP interrupt as was described above. This alternative attempts to reduce the amount of time that the software "wastes" while polling for descriptor own bits. This time would then be available for other CPU tasks. It also minimizes the amount of time the CPU needs for data copying. This savings can be applied to other CPU tasks.

The time from the end of frame arrival on the wire to delivery of the frame to the application is labeled as frame latency. For the one-interrupt method, frame latency is minimized, while CPU utilization increases. For the two-interrupt method, frame latency becomes greater, while CPU utilization decreases.

Note that some of the CPU time that can be applied to non-Ethernet tasks is used for task switching in the CPU. One task switch is required to swap a non-Ethernet task into the CPU (after S7A) and a second task switch is needed to swap the Ethernet driver back in again (at S8A). If the time needed to perform these task switches exceeds the time saved by not polling descriptors, then there is a net loss in performance with this method. Therefore, the SPRINT method implemented should be carefully chosen.

Figure 3 shows the event flow for the two-interrupt method.

Figure 4 shows the buffer sizing for the two-interrupt method. Note that the second buffer size will be about the same for each method.

There is another alternative which is a marriage of the two previous methods. This third possibility would use the buffer sizes set by the two-interrupt method, but would use the polling method of determining frame end. This will give good frame latency but at the price of very high CPU utilization.

And still, there are even more compromise positions that use various fixed buffer sizes and effectively, the flow of the one-interrupt method. All of these compromises will reduce the complexity of the one-interrupt method by removing the heuristic buffer sizing code, but they all become less efficient than heuristic code would allow.

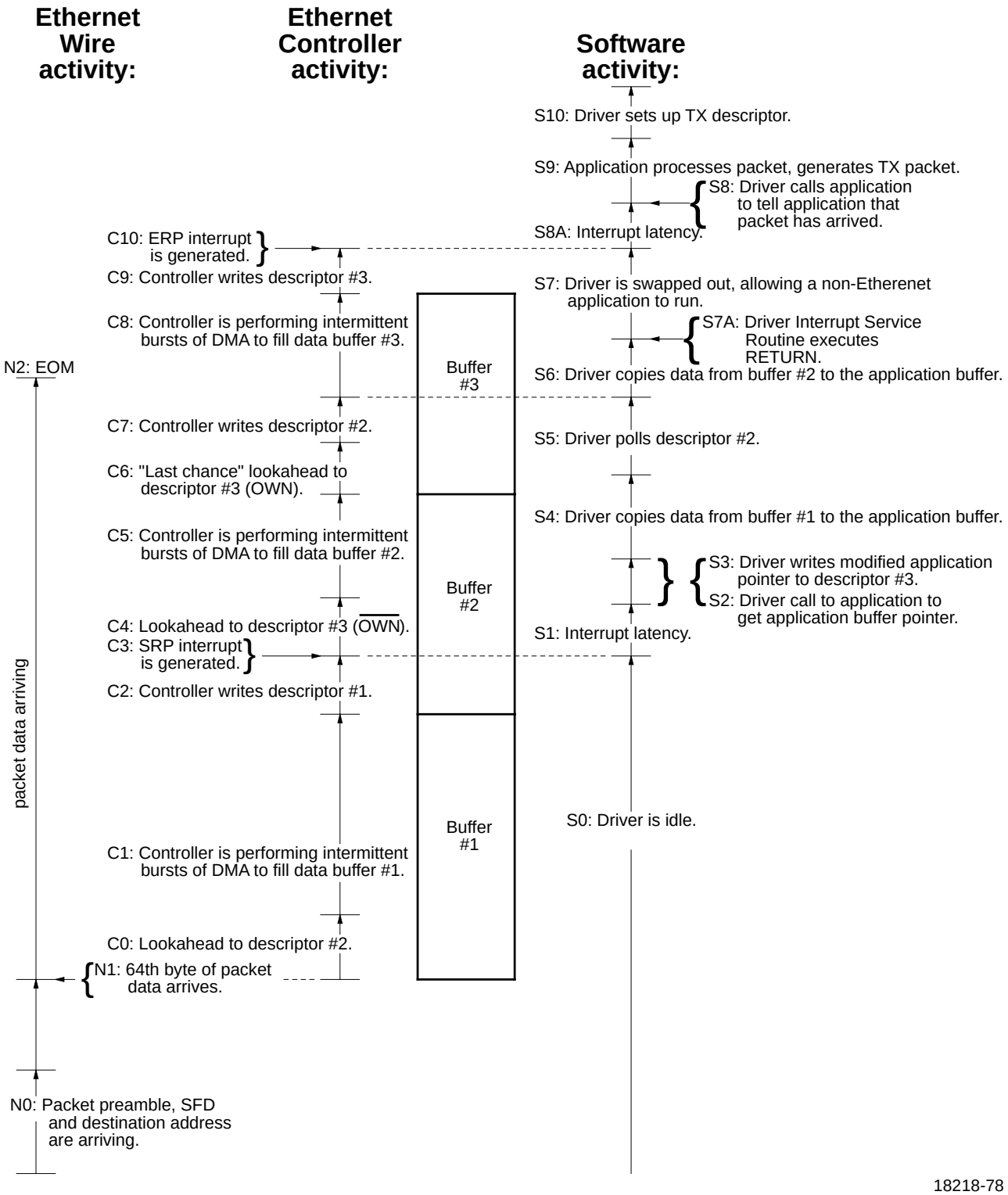


Figure 3. LAPP Timeline for TWO-INTERRUPT Method

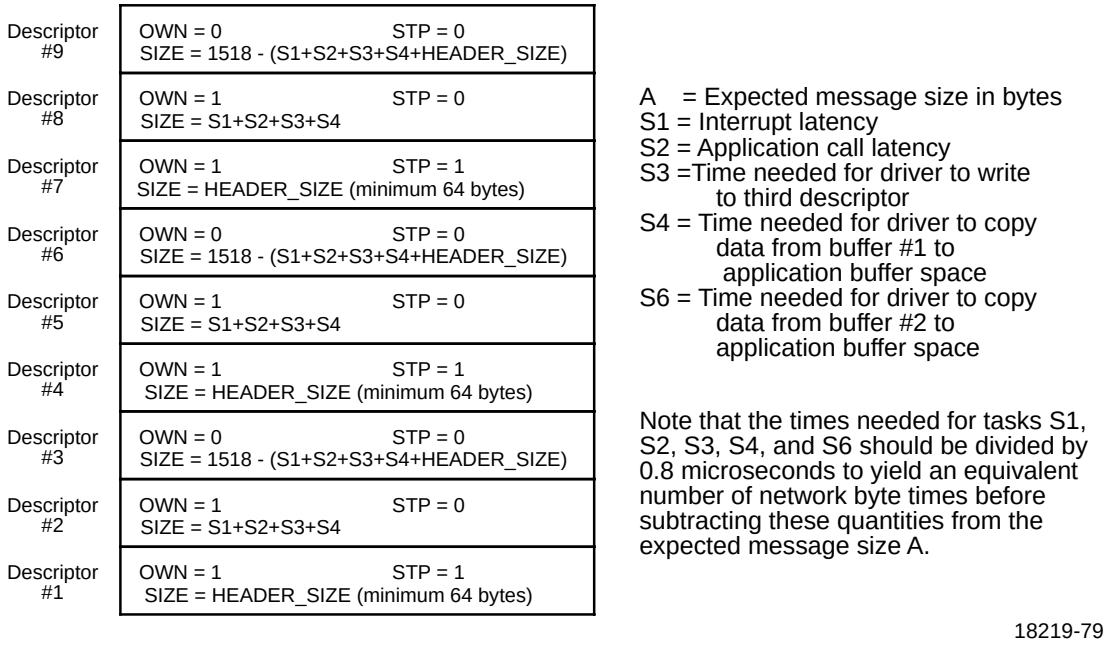


Figure 4. LAPP 3 Buffer Grouping for TW0-INTERRUPT Method

Am79C965A PCnet-32 Silicon Errata Report

AM 79C965A REV B1 SILICON STATUS

The items below are the known errata for Rev B1 silicon. Rev B1 silicon is the production silicon.

Note: *A signal followed by "*" indicates active low; i.e., Master*.*

The **"Description"** section of this document gives an external description of the problem. The **"Implication"** section gives an explanation of how to PCnet-ISA II controller behaves and its impact on the system. The **"Work-around"** section describes a work-around for the problem. The **"Status"** section describes when and how the problem will be corrected.

Current package marking for this revision:

Line 1: <Logo>
Line 2: Am79C965AKC (Assuming package is PQFP)
Line 3: <Date Code> B1
Line 4: (c) 1993 AMD

Value of chip ID registers CSR89+CSR88 [31:0] for this revision = 32430003h. (When read in 16-bit mode.)

1. Incorrect interpretation of descriptor ring length in 32-bit software mode

Description:

In 32-bit software mode, if TLEN and RLEN is set to 8 hex, indicating a descriptor ring length of 256 in the INIT block, the PCnet-32 controller will internally interpret ring length to be 512. Also, if TLEN or RLEN is set to C hex, the PCnet-32 controller converts it to 256 instead of 512. The table below summarizes the corresponding TLEN/RLEN settings and their actual interpretations.

TLEN or RLEN Value	Correct Ring Length	Erratum Ring Length
0	1	1
1	2	2
2	4	4
3	8	8
4	16	16
5	32	32
6	64	64
7	128	128
8	256	***512 *Bad***
9	512	512
A	512	512
B	512	512
C	512	***256 *Bad***
D	512	512
E	512	512
F	512	512

Implication:

There is no jeopardy to customers using AMD drivers, because AMD OS drivers do not use ring lengths that long. For customers developing proprietary drivers, implementation of the following work-around is highly recommended.

Work-around:

1. Program TLEN or RLEN to C hex if ring length of 256 is desired, or
2. – Initialize INIT block with any value for TLEN or RLEN
   – Set INIT, don't set STRT
   – After IDON, set STOP
   – Write to CSR76 (RCVRL) and CSR78 (XMTRL) with correct value
   –Set STRT

Status:

No current plan to revise silicon to fix this erratum.

2. Receive Frame Align feature not functioning correctly

Description:

The Receive Frame Align feature does not function correctly. The data field of a received frame will not necessarily be forced to align to a double-word aligned address boundary when the RCVALGN bit (bit 0) in CSR122 is set. Two extra bytes will not always be inserted at the beginning of the receive packet destination address field.

Implication:

Users are not able to force the data field of 802.3 frames to align to 0 MOD 4 address boundaries, i.e. double-word aligned addresses, by setting this bit.

Work-around:

None. Do not activate this bit. This feature of the device does not function as described in the data sheet.

Status:

No current plan to revise silicon to fix this erratum.

3. False BABL errors generated

Description:

The PCnet-32 device will intermittently give BABL error indications when the network traffic has frames equal to or greater than 1518 bytes.

Implication:

False BABL errors on the receiving station can be passed up to the upper layer software if the PCnet-32 device is just coming out of a deferral and the multi-purpose counter used to count the number of bytes received reaches 1518 at the same time. If the network is heavily loaded with full-size frames, the probability of a false BABL error is high.

Work-around:

There are two possible work-arounds.

1. If the user has no intention to transmit frames larger than 1518 bytes, the BABL bit may be masked to ignore babble errors. In this case, the false babble error will not cause an interrupt, nor will it be passed to the higher-level software.
2. Check to see if the device is transmitting in ISR (Interrupt Service Routine), which is induced by the BABL error. The BCRs that control the LED settings can be programmed to indicate a transmit activity, assuming the interrupt latency is not longer than one minute IFG (Inter-Frame Gap) time.

```
If (ISR_LATENCY < 9.6 us)
   True_bable_err = BABL* (TINT+XMT_LED)
   { i.e.False_bable_err = ~ (BABL* (TINT+XMT_LET))}
else
   Cannot tell if the BABL error is true or false just by reading BABL, TINT,
XMT_LED bits in ISR.
```

Status:

No current plan to fix this item.

Enhancement

The ***Enhancement*** section gives an external description of the enhancement. The ***Implication*** section gives an explanation of how the PCnet-32 ISA behaves and its impact on the system. The ***Status*** section indicates when the enhancement will be implemented.

1. Improved Phase Lock Loop function

Enhancement: The PLL (Phase Lock Loop) circuit in revision B1 silicon is enhanced to provide better phase tracking.

Implication:

With this PLL enhancement, the number of potentially dropped packets is effectively zero.

Status:

Implemented in revision B1 silicon.