MB86965

FUJITSU

ETHERCOUPLER SINGLE-CHIP ETHERNET CONTROLLER

DATA SHEET

APRIL1993

FEATURES

- Provides interface to the I/O bus of PC/XT™/AT® or compatible computers
- Optional, generic host interface to connect to industry-standard microprocessor buses
- Interface to serial EEPROM for Node ID and configuration option storage allows construction of jumperless, electronically configurable adapter cards
- Automatic polarity detection and correction on twisted-pair 10BASE-T cable
- Allows automatic selection of nonconflicting I/O address for self-installing under software control
- High-performance, packet-buffer architecture pipelines data for highest throughput
- On-chip buffer management controls buffer pointers to reduce software overhead and improve performance
- Hash filter for multicast packet reception
- Manchester encoder/decoder tolerates input jitter up to ±18 ns
- Fully compliant with ISO/ANSI/IEEE 8802-3 specifications
- Two network ports, AUI and 10BASE-T, with automatic port selection
- Integrated pulse shape, and transmit and receive filters
- Selectable 150Ω and 100Ω termination for shielded or unshielded twisted-pair cable, respectively
- Powerdown mode to reduce power dissipation in battery-powered equipment
- Low-power CMOS technology
- Single 5-volt power supply
- 160-pin plastic quad flat package (PQFP)

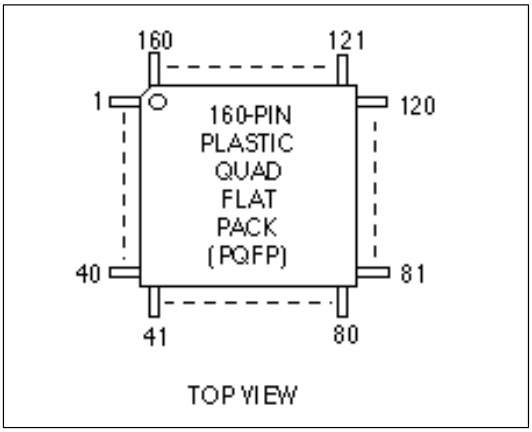
GENERAL DESCRIPTION

The MB86965 EtherCoupler™ Controller is a high-performance, highly integrated monolithic device that incorporates a network controller with buffer management, 10BASE-T transceiver with on-chip transmit and receive filters, Manchester encoder/decoder, and bus interface for a PC/XT/AT/ISA bus. EtherCoupler allows implementation of adapter solutions with as few as four chips. With its optional native bus mode for use directly on a microprocessor bus, it is ideal for use on daughter and motherboards, as well as expansion-bus adapter boards. An EEPROM can be interfaced to the chip for storage of Ethernet ID and configuration settings. EtherCoupler is designed to be configured electronically, thus eliminating the jumpers typically used to configure the system network adapters.

The buffer management architecture of the MB86965 allows packet data to access a buffer memory area simultaneously from the host and from the network media. The network controller updates all receive and transmit pointers automatically to reduce the software overhead needed to control these operations, which results in superior benchmark speed and application performance. EtherCoupler's transmit buffer is programmable as a single 2-kbyte bank or as two banks of 2, 4, or 8 kbytes each. This buffer chains multiple data packets and transmits them to the network from a single transmit command, thereby offering greater design flexibility and throughput. A ring buffer that can be sized from 4 to 62 kbytes, depending on the amount of available memory used for the transmit buffer, captures the receive packets.

The MB86965 performs pulse shaping and filtering internally, which eliminates the need for external filtering components and reduces overall system cost. EtherCoupler is compatible with shielded and unshielded

PIN CONFIGURATION



twisted-pair cables and provides outputs for receive, transmit, collision and link test LEDs. The twisted pair receive threshold can be reduced to allow an extended range between nodes in low-noise environments. Its wide range of features makes EtherCoupler the ideal device for 10BASE-T twisted-pair Ethernet.

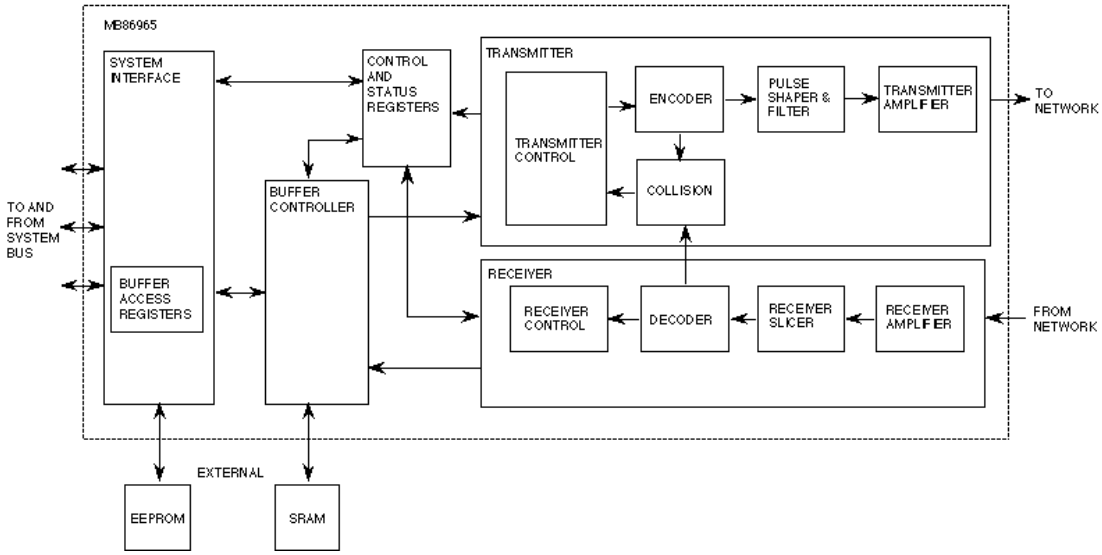
Possible configurations for the system bus interface include I/O mapping, memory mapping and DMA access, or a combination of these. With a 20 Mbyte/s bandwidth, the EtherCoupler system bus interface allows use of full throughput capacity of its packet-buffering architecture. EtherCoupler bus modes are selectable, thereby providing big or little endian byte-ordering, permitting efficient data interface with most microprocessors and higher-level protocols. The Fujitsu high-speed, low-power CMOS process is used to manufacture of the MB86965, which is furnished in a 160-pin plastic quad flat package.

FUNCTIONAL DESCRIPTION

As shown below, the MB86965 comprises five major functional blocks: System Interface, Buffer Controller, Control and Status Registers, Transmitter, and receiver. EtherCoupler's receive and transmit sections fully implement the ISO/ANSI/IEEE 8802-3 CSMA/CD specification for 10-megabit per second Ethernet. The transmitter assembles data packets for transmission and the receiver disassembles received data packets. On-chip Ethernet protocol functions include: automatic generation and stripping of the 64-bit preamble; generation and verification of 32-bit cyclic redundancy check (CRC); collision resolution by binary exponential backoff and retransmission; several modes of address recognition, error detection and reporting; and serial/parallel and parallel/serial conversions.

TYPICAL APPLICATION

Figure 1 is a block diagram illustrating the application of the MB86965 in an add-in adapter card for an ISA bus-based personal computer. The serial EEPROM provides storage for I/O base address, boot PROM address and interrupt channel as well as the Ethernet node ID. One or two 8-bit bidirectional transceivers provide data buffering between the Ethercoupler and the 8- or 16-bit system bus. A single SRAM, typically 8 or 32 kbytes, implements the local packet buffer, while an optional boot PROM allows use of the adapter in diskless environments, where the driver program must be loaded from the network. The EtherCoupler provides interfaces to 100- or 150-ohm twisted pair (10BASE-T) as well as



EtherCoupler Block Diagram

a direct AUI capability (10BASE5). An optional coaxial transceiver interface, such as Fujitsu's MBL8392A, provides additional capability for Thin Ethernet (10BASE2) networks. The MB86965 directly drives several LEDs which provide indication of adapter operational status.

For complete information on a typical adapter design, request the Hardware Reference Manual for the DK86965 EtherCoupler Designer Kit from your authorized Fujitsu Microelectronics sales representative or distributor. Schematics in electronic format, board layouts and Gerber tape for printed circuit card construction are also available for purchase as Hardware Design Kit HD86965.

PIN ASSIGNMENTS AND DESCRIPTIONS

LOGIC CONVENTION

Unless otherwise noted, a positive logic (active high) convention is assumed throughout this document, whereby the presence at a pin of a higher, more-positive voltage (nominally 5 VDC) causes assertion of the signal. An underscore, e.g., RDY indicates that the signal is asserted in the low state (nominally 0 volts). Dual-function pins have their alternate function enclosed in parentheses, e.g., RDY(RDY). Whenever a signal is separated into numbered bits, e.g., BD7, BD6, BD5, BD4, BD3, BD2, BD1 and BD0, the family of bits may also be shown collectively, e.g., as BD<7:0>.

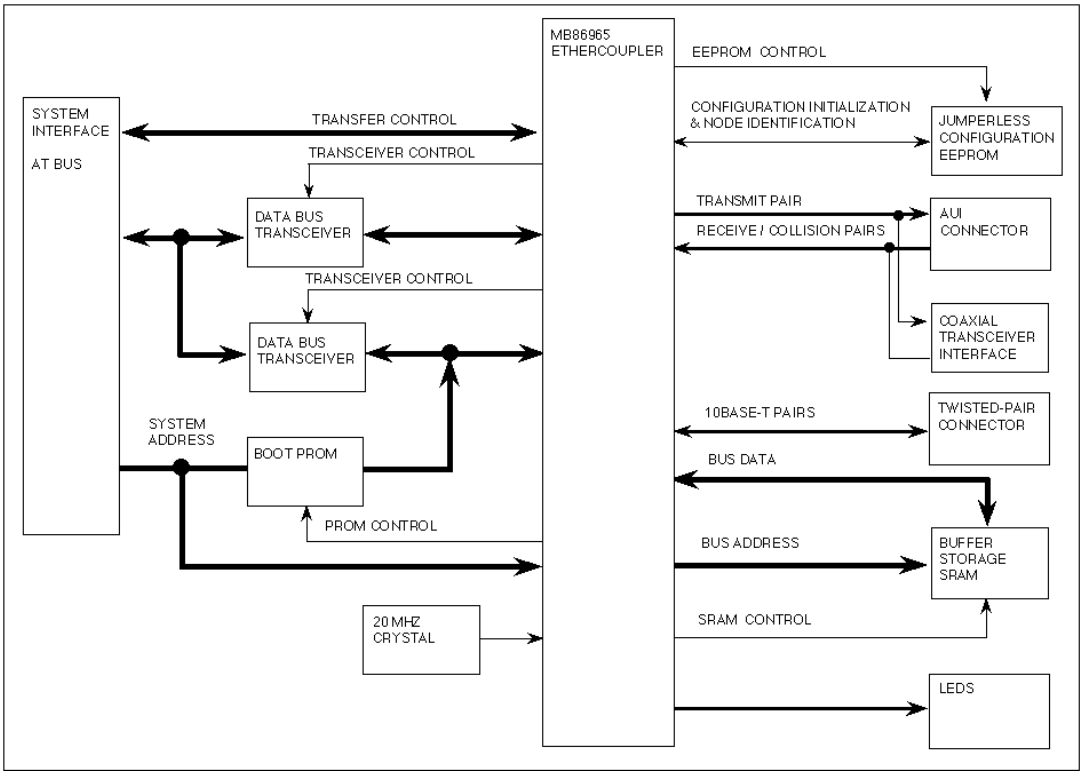


Figure 1. Typical Application, Block Diagram

PIN ASSIGNMENTS

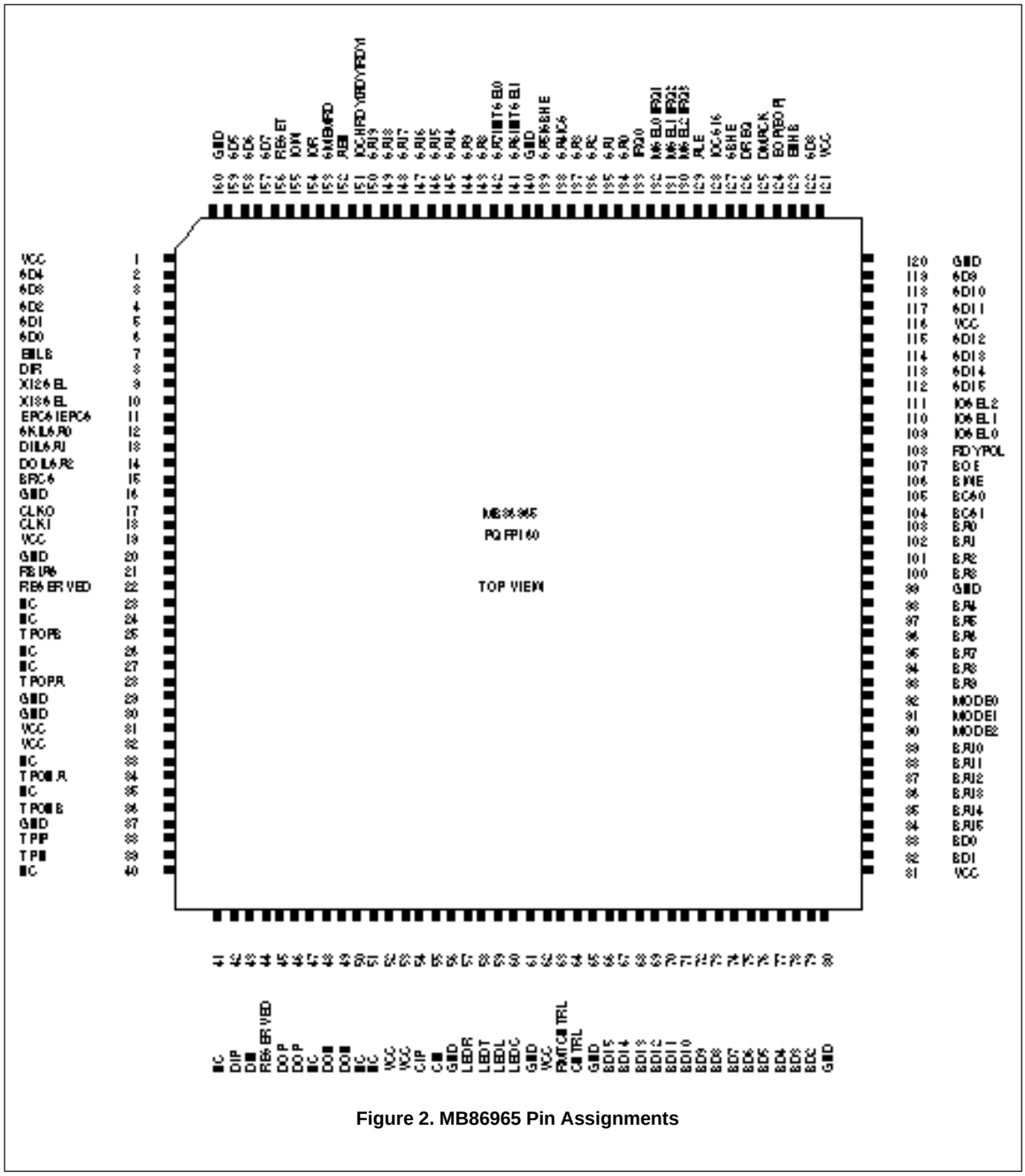


Figure 2. MB86965 Pin Assignments

PIN DESCRIPTIONS

Configuration Mode Selection

The MB86965 can be configured to match specific hardware environments, and pin functions vary according to Mode. Setting pins 90, 91 and 92 allows selection of EtherCoupler for operation in one of four predefined configurations. Operation is selectable as jumperless or requiring jumpers. Initial parameters can be stored in a bit-serial EEPROM or a byte-parallel PROM, depending on Mode. Operation as a NICE chip is also available.

CONFIGURATION PINS

PIN NO.	SYMBOL	MODE	TYPE	DESCRIPTION
90–92	MODE<2:0>	—	I pull-down	**MODE SELECT:** Allows selection of EtherCoupler configuration mode as shown in the **MODE** column in these tables.

No.	Bus	Pin 90	Pin 91	Pin 92	Operation	Initial Parameters
0	ISA	0	0	0	Jumper-less	Stored in bit-serial EEPROM.
1	ISA	0	0	1	Jumpers	Stored in bit-serial EEPROM.
2	ISA	0	1	0	Jumpers	Stored in byte-parallel ID PROM.
3	Non-ISA	0	1	1	Jumpers	Functions identical to MB86960 NICE (Network Interface Controller with Encoder/ decoder) chip with MB86961 10BASE-T Interface.
X	–	1	X	X	–	Reserved.

PIN NO.	SYMBOL	MODE	TYPE	DESCRIPTION
111–109	IOSEL<2:0>	1, 2	I pull-up	**INPUT/OUTPUT BLOCK ADDRESS SELECT:** Sets the base address at which the controller is located, as follows:

IOSEL2	IOSEL1	IOSEL0	I/O BASE ADDRESS
0	0	0	260 – 27F
0	0	1	280 – 29F
0	1	0	2A0 – 2BF
0	1	1	240 – 25F
1	0	0	340 – 35F
1	0	1	320 – 33F
1	1	0	380 – 39F
1	1	1	300 – 31F

PIN NO.	SYMBOL	MODE	TYPE	DESCRIPTION
111–109	RESERVED	0, 3	I	**NOT USED.** These pins should be left floating, internally pulled high.

SYSTEM BUS TO INTERFACE PINS

PIN NO.	SYMBOL	MODE	TYPE	DESCRIPTION
112–115 117–119 122 157–159 2–6	SD<15:12> SD<11:9> SD<8> SD<7:5> SD<4:0>	0, 1, 2, 3	I/O	**SYSTEM DATA:** All data, command, and status transfers take place over this bus.
150–145 144–141 139–134	SA<19:14> SA<9:6> SA<5:0>	0, 1, 2	I	**SYSTEM ADDRESS:** Selects EtherCoupler internal register or port for read/write operations.
137–134	SA<3:0>	3	I	**SYSTEM ADDRESS:** Selects EtherCoupler internal register or port for read/write operations.
142 141	INTSEL0 INTSEL1	3	I	**INTERRUPT SELECT:** Selects the interrupt signal. Read only. Same as BMPR19<6:7>.
139 127	SBHE SBHE	3 0, 1, 2	I I	**SYSTEM BUS HIGH ENABLE:** Active low. This pin is the byte/word control line. It is used only when EtherCoupler is configured for a 16-bit data bus by the SB/SW bit, DLCR6<5>. It allows word, upper byte only or lower byte only transfers. The address select pin SA0 is used with SBHE for byte or word transfers as follows:

SB/SW	SBHE	SA0	FUNCTION
0	0	0	Word transfer
0	0	1	Byte transfer on upper half of data bus (SD15-8)
0	1	0	Byte transfer on lower half of data bus (SD7-0)
0	1	1	Reserved
1	X	X	Byte transfer (SD7-0)

138	CS	3	I	**CHIP SELECT:** Active low signal which, when set to 0, selects the EtherCoupler chip.
156	RESET	0, 1, 2, 3	I	**CHIP RESET:** Resets the chip internal pointers and initializes internal registers and logic.
154	IOR	0, 1, 2, 3	I	**INPUT/OUTPUT READ:** Active low signal from the system bus which indicates that the current bus cycle is a read operation.
155	IOW	0, 1, 2, 3	I	**INPUT/OUTPUT WRITE:** Active low signal from the system bus which indicates that the current bus cycle is a write operation.
153	SMEMRD	0, 1, 2	I	**SYSTEM MEMORY READ:** Active low signal from the system bus that indicates the current bus cycle is in a memory-read operation. Used for Boot ROM Chip Select (BRCS).
152	AEN	0, 1, 2	I	**ADDRESS ENABLE:** When asserted indicates that a DMA transfer is taking place.
124	EOP(EOP)	0, 1, 2, 3	I	**END OF PROCESS:** When asserted by the DMA controller, indicates that an entire packet has been transferred between buffer memory and the host system.
129	ALE	0, 1, 2	I	**ADDRESS LATCH ENABLE:** Provided by the system to latch valid addresses.
125	DMACK	0, 1, 2, 3	I	**DMA ACKNOWLEDGE:** Active low signal which indicates that the DMA controller is ready to transfer data between the host system and the EtherCoupler buffer memory.
15	BRCS	0, 1, 2	O	**BOOT ROM CHIP SELECT:** Active low signal which indicates that the current memory access is to Boot ROM.

PIN NO.	SYMBOL	MODE	TYPE	DESCRIPTION
133	IRQA	0, 1, 2, 3	O 24-ma tristate	**INTERRUPT REQUEST:** One of four interrupt request lines which indicate that EtherCoupler requires host attention after successful transmission or reception of a packet, or if any error conditions occur, including an EOP after the completion of the DMA cycle.
130–132	IRQD, IRQC, IRQB	0, 3	O 24-ma tristate	**INTERRUPT REQUEST:** In jumperless mode: three of four interrupt request lines which indicate that EtherCoupler requires host attention after successful transmission or reception of a packet, or if any error conditions occur, including an EOP after completion of DMA cycle.
130–132	MSEL<2:0>	1, 2	I	**MEMORY SELECT:** In jumper select mode: sets Boot ROM location base address.

MSEL2	MSEL1	MSEL0	ROM ADDRESS RANGE
0	0	0	C4000 – C7FFF
0	0	1	C8000 – CBFFF
0	1	0	CC000 – CFFFF
0	1	1	D0000 – D3FFF
1	0	0	D4000 – D7FFF
1	0	1	D8000 – DBFFF
1	1	0	DC000 – DFFFF
1	1	1	Decode disabled

126	DREQ	0, 1, 2, 3	O	**DMA REQUEST:** Issued to the DMA controller to indicate that EtherCoupler has data available to be read in its receive buffer, or is ready to accept data into its transmit buffer.
151	RDY(RDY)	3	O 24-ma tristate	**INPUT/OUTPUT CHANNEL READY:** This signal indicates to the host that EtherCoupler is ready to complete the requested read or write operation. Polarity of this pin is programmed by RDYPOL, pin 108. High-impedance to READY valid, or READy valid to high impedance (Tables 41 and 43). N-channel open drain.
151	IOCHRDY	0, 1, 2	O 24-ma tristate	**INPUT/OUTPUT CHANNEL READY:** This signal indicates to the host that EtherCoupler is ready to complete the requested read or write operation. RDYPOL pin must be tied to ground. N-channel open drain.
128	IOCS16	0, 1, 2	O 24-ma tristate	**INPUT/OUTPUT 16-BIT CHANNEL SIZE:** Active low signal which indicates that present data transfer is 16-bit I/O cycle. N-channel open drain.
123	ENHB	0, 1, 2	O	**ENABLE DATA HIGH BYTE:** Active low signal which enables the system data transceiver high byte.
7	ENLB	0, 1, 2	O	**ENABLE DATA LOW BYTE:** Active low signal which enables the system data transceiver low byte.
8	DIR	0, 1, 2	O	**TRANSCEIVER DIRECTION:** Active low signal sets the external transceiver direction, from chip to system (read operation). Active high signal sets the external transceiver direction from system to chip (write operation).
9	X12SEL	0, 1, 2	O	**SELECT CONFIGURATION REGISTER 1:** Active low signal pulse to read the jumper configuration register 1, located at address 0X12.

PIN NO.	SYMBOL	MODE	TYPE	DESCRIPTION
10	X13SEL	0, 1, 2	O	**SELECT CONFIGURATION REGISTER 2:** Active low signal pulse to read the jumper configuration register 2, located at address 0X13.
10	SB/SW	3	O	**SYSTEM BYTE / WORD BUS WIDTH:** When high, System Bus operates in 8-bit mode; when low,16-bit mode is selected. Same as DLCR6<5>.
12	SK	0, 1	O	**EEPROM SHIFT CLOCK:** With EEPROM option, this signal is generated from software, which shifts data in and out of the EEPROM.
12	LSA0	2	O	**LATCHED ADDRESS 0:** With ID PROM option, the pin provides one of three latched address lines to the ID PROM.
13	DI	0, 1	O	**EEPROM DATA IN:** With EEPROM option, this is serial data going to the EEPROM.
13	LSA1	2	O	**LATCHED ADDRESS 1:** With ID PROM option, the pin provides one of three latched address lines to the ID PROM.
14	DO	0, 1	I	**EEPROM DATA OUT:** With EEPROM option, this is serial data coming from the EEPROM.
14	LSA2	2	O	**LATCHED ADDRESS 2:** With ID PROM option, the pin provides one of three latched address lines to the ID PROM.
11	EPCS	0, 1	O	**PROM CHIP SELECT:** High signal selects EEPROM.
11	EPCS	2	O	**PROM CHIP SELECT:** Low signal selects ID PROM.
108	RDYPOL	0, 1, 2, 3	I	**READY LINE POLARITY SELECT:** Controls polarity of IOCHRDY signal at pin 151, where 0 is active low ready and 1 is active high ready.
63	RMTCNTRL	0, 1, 2, 3	O	**REMOTE CONTROL:** When RMT RST bit DLCR5<2> is set high, this pin follows RMT 0900H bit DLCR1<4>, which indicates that a complete special packet with type field = 0900H has been received. This is intended for use as a remotely controlled hardware function from other nodes in the network.
64	CNTRL	0, 1, 2, 3	O	**CONTROL:** This pin is intended as a hardware control pin programmable by the system software. Complement of CNTRL, DLCR4<2>.
127, 129, 143–150, 152, 153	RESERVED	3	I	**NOT USED.** These pins should be pulled low to ground.
7–9, 11–15, 123, 128	RESERVED	3	O	**NOT USED.** These pins should be left floating, internally pulled high.

BUFFER MEMORY INTERFACE PINS

PIN NO.	SYMBOL	MODE	TYPE	DESCRIPTION
66–79, 82–83	BD<15:2>, BD<1:0>	0, 1, 2, 3	I/O	**BUFFER MEMORY DATA BUS:** Data lines between SRAM buffer memory and EtherCoupler. This data bus configurable for 8-bit or 16-bit data size by BB/BW bit DLCR6<4>.
84–89, 93–98, 100-103	BA<15:10> BA<9:4> BA<3:0>	0, 1, 2, 3	O	**BUFFER MEMORY ADDRESS BUS:** These lines address up to 64 kbytes of buffer memory.
104	BCS1	0, 1, 2, 3	O	**BUFFER RAM CHIP SELECT 1:** Active low signal that is chip select for most significant byte of buffer memory.
105	BCS0	0, 1, 2, 3	O	**BUFFER RAM CHIP SELECT 0:** Active low signal that is chip select for least significant byte of buffer memory.
106	BWE	0, 1, 2, 3	O	**BUFFER WRITE ENABLE:** Active low, write-enable-to-buffer -memory signal output during memory write cycles.
107	BOE	0, 1, 2, 3	O	**BUFFER OUTPUT ENABLE:** Active low, output-enable-to-buffer memory signal output during memory read cycles.

POWER AND TRANSCEIVER INTERFACE PINS

PIN NO.	SYMBOL	MODE	TYPE	
1, 19, 31, 32, 52, 53, 62, 81, 116, 121	V_{CC}	0, 1, 2, 3	I	**POWER SUPPLY:** +5 Volts ± 5%, for analog and digital circuits.
16, 20, 29, 30, 37, 56, 61, 65, 80, 99, 120, 140, 160	GND	0, 1, 2, 3	I	**GROUND:** digital and analog ground.
45, 46 48, 49	DOP DON	0, 1, 2, 3	O O	**AUI TRANSMIT PAIR:** Differential output driver pair for AUI transceiver DO circuits; output is Manchester-encoded.
42 43	DIP DIN	0, 1, 2, 3	I I	**AUI RECEIVE PAIR:** A differential input driver pair for the AUI transceiver DI circuits; input is Manchester-encoded.
54 55	CIP CIN	0, 1, 2, 3	I I	**AUI COLLISION PAIR:** A differential input driver pair for the AUI transceiver CI circuits. The input is collision signalling or signal-quality error (SQE).
28, 34 25, 36	TPOPA,TPONA TPOBP,TPONB		O	**TWISTED-PAIR OUTPUT:** Differential driver pair outputs to the twisted-pair cable. The output is pre-equalized so that no external filter is required.
38, 39	TPIP TPIN	0, 1, 2, 3	I	**TWISTED-PAIR INPUT:** A differential input pair from the twisted-pair cable.
60	LEDC	0, 1, 2, 3	O	**COLLISION LED:** Open-drain driver for the collision indicator. Output is pulled low during collision.
59	LEDL	0, 1, 2, 3	O	**LINK LED:** Open-drain driver for link integrity indicator. Output is pulled low during link test pass.
58	LEDT	0, 1, 2, 3	O	**TRANSMIT LED:** Open-drain driver for transmit indicator. Output is pulled low during transmit.
57	LEDR	0, 1, 2, 3	O	**RECEIVE LED:** Open-drain driver for the receive indicator. Output is pulled low during receive.
21	RBIAS	0, 1, 2, 3	I	**BIAS RESISTOR:** To control bias of operating circuit; pulled to ground with 12.4-kilohm ±1% pulldown resistor.
17 18	CLKO CLKI	0, 1, 2, 3	O	**CRYSTAL OSCILLATOR:** A 20-MHz crystal must be connected between these pins. [See figure 4.]
22–24, 26, 27, 33, 35, 40, 41, 44, 47, 50, 51	No Connection	0, 1, 2, 3	O	**NOT USED.** These pins should be left floating, internally pulled high.

SYSTEM CONFIGURATION

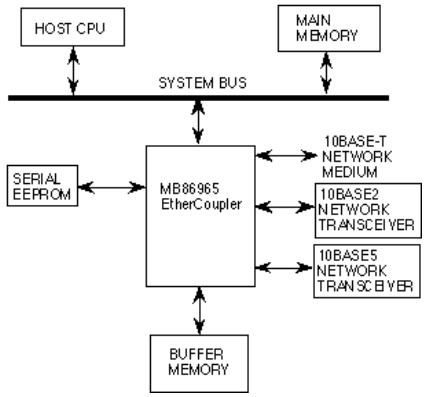


Figure 3. System Configuration Simplified Block Diagram

As shown in Figure 3, EtherCoupler allows a highly integrated system configuration. EtherCoupler's architecture and high level of integration facilitate packet management and storage, and eliminate the need for a local microprocessor. EtherCoupler connects with the host system bus to provide command and status interfaces as well as packet data access. The host processor can directly access command and status registers when mapped into the host I/O or memory space. Data packets to be transmitted to the media initially transfer from host memory through an EtherCoupler port into a dedicated buffer memory where they are temporarily stored until transmitted. Buffer memory initially stores received data packets that are later transferred to the host memory. [Refer to the section on Control and Status Registers for identification of control and status bits of the data link control registers, DLCR0 through DLCR7, and buffer memory port register pair, BMPR8 and BMPR9.]

CONNECTION TO THE LAN MEDIA

Connection to the LAN media can be accomplished by on-board connection to a shielded or unshielded twisted pair, through the on-chip 10BASE-T transceiver.

CRYSTAL OSCILLATOR

The clock rate of 10 Mbits/s specified by the international LAN standard, ISO/ANSI/IEEE 8802-3, derives from an on-chip oscillator that is controlled by a crystal connected across pins 17 and 18 (CLKO and CLKI). Capacitance specified by the crystal manufacturer must

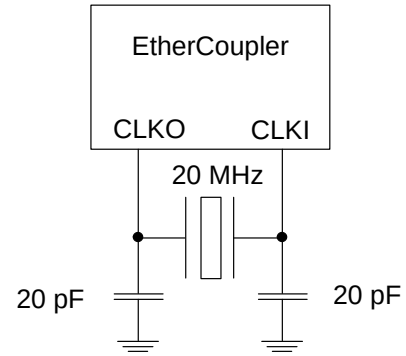


Figure 4. Crystal Oscillator Connection

be connected across pins 17 and 18 to ground, as shown in Figure 4, to stabilize the effects of stray capacitance that may vary crystal frequency. The 20-MHz clock also serves as an internal phase-locked loop (PLL) reference for decoder clock recovery. Internal clocks shut down when the PWRDN bit, DLCR7<5>, is asserted for power down mode. Use a crystal with the following specifications: quartz (AT-cut); 20-MHz; frequency/accuracy of ± 50ppm at 25°C and 100ppm at 0°C to 70°C; parallel resonant with 20 pF-load in fundamental mode. Possible vendors include: Ecliptek Corp. (Costa Mesa, CA) p/n ECSM200-20.000; and M-tron Industries, Inc. (Yankton, SD) p/n MP-1 and MP-2, with 20-MHz, 50ppm over 0°C to 70°C, and 18 pF fundamental load.

SRAM CONFIGURATIONS

Figure 5 shows how to configure industry-standard SRAMs for memory implementation of packet buffers in byte and word modes. The Buffer Byte/Buffer Word (BB/BW) bit, DLCR6<4>, selects the width of the SRAM data path as 8 or 16 bits. Setting this bit to 1 selects byte-wide (8-bit) data; setting it to 0 selects word-wide (16-bit) data. The System Byte/System Word (SB/SW) bit, DLCR6<5>, selects the width of the system data path as 8 or 16 bits. Setting this bit to 1 selects byte-wide (8-bit) data; setting it to 0 selects word-wide (16-bit) data. Do not use the combination of SB/SW bit, DLCR6<5>, set to 1 (byte) and BB/BW bit, DLCR6<4>, set to 0 (word). The Buffer Size (BS1 and BS0) bits, DLCR6<1:0>, select SRAM size as 8, 16, 32 or 64kbytes.

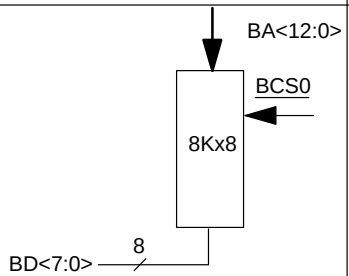
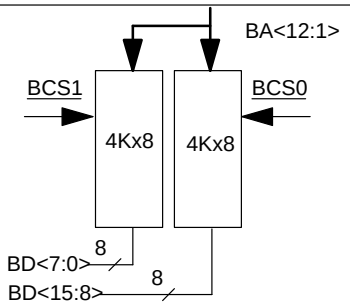
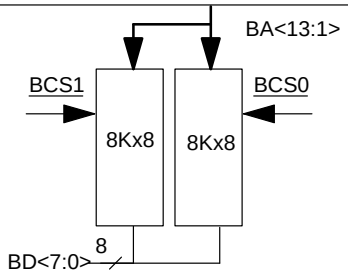
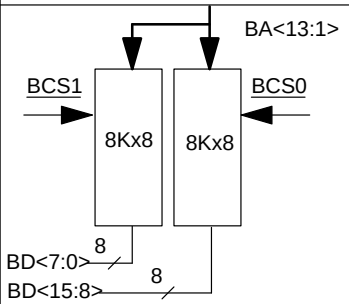
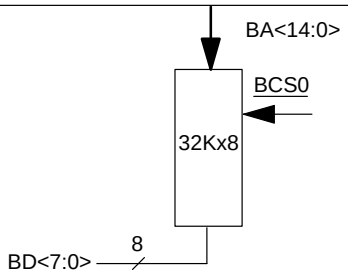
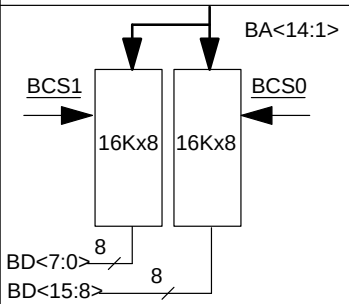
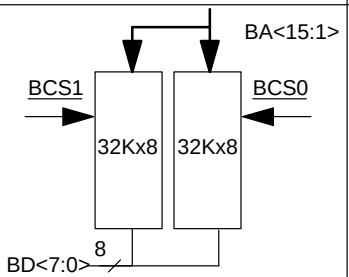
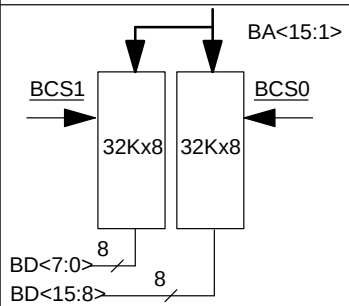
BS1 DLCR6<1>	BS0 DLCR<0>	8–BIT BUFFER BUS	16–BIT BUFFER BUS
0	0		
0	1		
1	0		
1	1		

Figure 5. SRAM Configurations

SYSTEM INTERFACE

GENERAL

The System Interface provides the interface logic necessary to connect to a PC/XT/AT/ISA bus and, via an alternate bus mode, to an x86, RISC or 680X0-type microprocessor bus. The interface supports 8- and 16-bit bus widths and byte or word transfers, as determined by the SB/SW bit, DLCR6<5>. Depending on the type of host CPU, EtherCoupler supplies the data order, MSB or LSB first (big or little endian), according to the setting of the M..LM..L/L..M bit, DLCR7<0>.

EtherCoupler supports I/O-mapping, and burst or single-transfer DMA modes. The user can select the active interrupt pin to inform the CPU of transmit and receive status conditions requiring host processing. EtherCoupler includes three sets of user-accessible registers, all of which are accessible as bytes or words.

Pins 90 to 92 allow selection and configuration of the mode in which the MB86965 operates. Specifically, when MODE<2:0> bits are set to 00H, the bus is in ISA mode with an EEPROM that stores initial parameters and jumperless operation. When MODE<2:0> bits are set to 01H the bus is set for operation with jumpers and EEPROM. When MODE<2:0> bits are set to 02H the bus is set for operation with jumpers and ID PROM. When MODE<2:0> bits are set to 03H the bus is set for generic bus operation.

REGISTER ACCESS

Direct access is available to eight registers in the device's register set, addresses xxx0H through xxx7H. Access to remaining physical addresses is through indirect addressing or bank-switching of three different register banks, for the address set XXX8H through XXXFH. Bank-switching bits reside in register 07H. The bank-switched register group comprises three sets or banks of registers: Node ID (Ethernet Address) and TDR Diagnostics; Hash Table for multicast address filtering; and Buffer Memory Access. During normal operation (excluding initialization or diagnostics), the Buffer Memory Access bank is selected, and access to other registers is not required.

BUFFER ACCESS

The Buffer Memory Port register pair BMPR8 and BMPR9 provide 8 or 16-bit data with access to the receive and transmit buffers through on-chip FIFOs. To eliminate the need for complicated directional control, FIFOs are dedicated to each direction of data transfer. Writing to the transmit buffer interleaves with reading from the receive buffer, with EtherCoupler automatically maintaining buffer memory pointers. The Buffer Memory port register pair is accessible via register address xxx8H, when bank I is selected. When using DMA, the buffer memory port is automatically selected when the DMA Acknowledge input, DMACK is asserted.

Data can transfer from the host memory to the transmit buffer, or from the receive buffer to host memory by using string moves, single-transfer programmed I/O moves, or DMA. Select the method that yields the highest system-level efficiency. A rapid transfer process results in best performance. Slow transfer could result in poor throughput and performance, and cause the receive buffer to overflow and lose packets.

I/O OPERATION

Write transmits data to BMPR10, and I/O Write will address the data to the Transmit Buffer. Read receives data on BMPR10 through I/O Read, which will load the data from the Receive Buffer.

DMA OPERATION

EtherCoupler supports single-cycle and burst DMA operation for data transfer between the host system and the dedicated buffer memory. Handshaking between EtherCoupler and external DMA is accomplished by the DREQ and DMACK signals. The end of process input, pin 124, when asserted by the system DMA controller during a transfer cycle, terminates DMA activity after completion of the current cycle. If a DMA interrupt (DLCR3<5>) is enabled, EtherCoupler generates an interrupt after completion of DMA activity.

DMA Write (Transmit)

Setting the TX DMA Enable bit, BMPR12<0>, enables DMA transfer of data packets from the host memory to the EtherCoupler transmit buffer. When invoked, DMA burst control bits BMPR13<1:0> enable burst transfers. EtherCoupler, when ready to accept data from the host, sets the DMA request output, DREQ, and the host responds by setting DMA acknowledge, DMACK, and write enable, WE, and placing data on the data bus. EtherCoupler sets the ± IOCHRDY output when ready to complete the current data-transfer cycle. (Polarity of the IOCHRDY signal and the EOP input are independently programmable.) EtherCoupler accepts the data byte/word into its Bus Write FIFO and later moves it into buffer memory. At the close of a transfer cycle, the host

negates WE. In burst mode and depending on CNTRL bit, DLCR4<2>, at the next-to-last cycle, EtherCoupler negates DREQ. The host DMA then completes the last two transfer cycles and negates DACK to close the burst. To start another burst, EtherCoupler re-asserts DREQ. The number of DMA write cycles within one burst can be 1, 4, 8, or 12 data transfers (bytes or words), depending on burst control bits BURST 1 and BURST 0, BMPR13<1:0>.

The DMA controller asserts the end of process input, EOP, concurrent with the last data-transfer cycle to indicate completion of the entire transfer process. This action sets the DMA EOP bit, DLCR1<5>, to discontinue further EtherCoupler data requests. Setting the EOP signal will also generate an interrupt. If the DMA EOP INT enable bit, DLCR3<5>, is high. The host can use this interrupt to begin action to close the process. The host should reset EtherCoupler DMA logic and clear the interrupt by resetting the DMA enable bit, BMPR12<0>.

The transmit DMA process must be closed by clearing BMPR12<0> before attempting another DMA process. Clearing the DMA EOP INT EN bit, BMPR3<5>, then automatically clears the EOP status and interrupt, removing the need to clear the interrupt separately. The host initiates packet transmission after loading packets into the buffer by loading the number of packets to be transmitted into the TX PKT CNT bits, BMPR10<6:0>, and asserting TX Start bit, BMPR10<7>.

DMA Read (Receive)

EtherCoupler indicates when it receives packets to be read with status bits or interrupts. Before reading a packet, the host processor can read the RX BUF EMPTY bit, DLCR5<6>, which if 0 indicates one or more packets in the receive buffer to read. After reading each packet, the host again checks this bit for more packets.

Before transferring a packet via DMA from the receive buffer to host memory, the host first reads the buffer four-byte receive packet header to obtain packet status and packet byte length. [Also refer to the discussion of Transmit and Receive Packet Headers.] The host calculates and loads the host DMA controller cycle counter with the number of DMA cycles needed to read the packet. The system memory starting address must also be loaded into the DMA controller. The RX DMA EN bit, BMPR12<1>, is next set to enable the DMA read operation and transfer the packet to host memory. When ready to begin this transfer, EtherCoupler sets the DMA request output DREQ, and the host responds by asserting DMA acknowledge bit, DMACK, and then I/O Read, IOR. EtherCoupler sets the IOCHRDY output when the byte/word is on the data bus, and the data transfer cycle is ready to be completed. After system memory accepts the data, the host negates IOR. EtherCoupler then "pops" the data in the Bus Read FIFO, and points the internal Bus Read pointer to the next buffer byte/word to move that data into the FIFO.

If programmed for burst operation and depending on CNTRL bit, DLCR4<2>, EtherCoupler negates DREQ two cycles before the end of the burst, then re-asserts DREQ to repeat the process, if it can transfer more data after the host negates DMACK. The number of DMA read cycles in a burst can be 1, 4, 8, or 12 data-transfer (byte or word) cycles, depending on burst control bits BURST1 and BURST0, BMPR13<1:0>. The DMA controller asserts the end of process input EOP concurrent with the last byte/word data transfer to indicate completion of the entire process. EtherCoupler then stops requesting more DMA cycles. After the DMA process, RX DMA enable must be cleared then re-asserted when the host wants to begin reading another packet from the Receive Buffer by using DMA.

When the host DMA controller asserts EOP, the DMA EOP bit, DLCR1<5>, is set and generates an interrupt, if so enabled by setting the associated INT EN bit, DLCR3<5>. The host can use this interrupt to close the receive DMA process. Writing 000H to the DMA EN bits, BMPR12<1:0>, clears the interrupt and disables and resets the DMA, removing the need to clear the interrupt separately. The receive DMA process must be terminated before attempting another DMA process.

ETHERCOUPLER INTERFACE TO AT BUS

As shown in Figure 6, EtherCoupler provides jumpered (with ID PROM or serial ID EEPROM) and jumperless design solutions. Designed primarily for AT Bus Systems, EtherCoupler assumes that the program I/O mode is the most effective way to transfer data between controller and system resources.

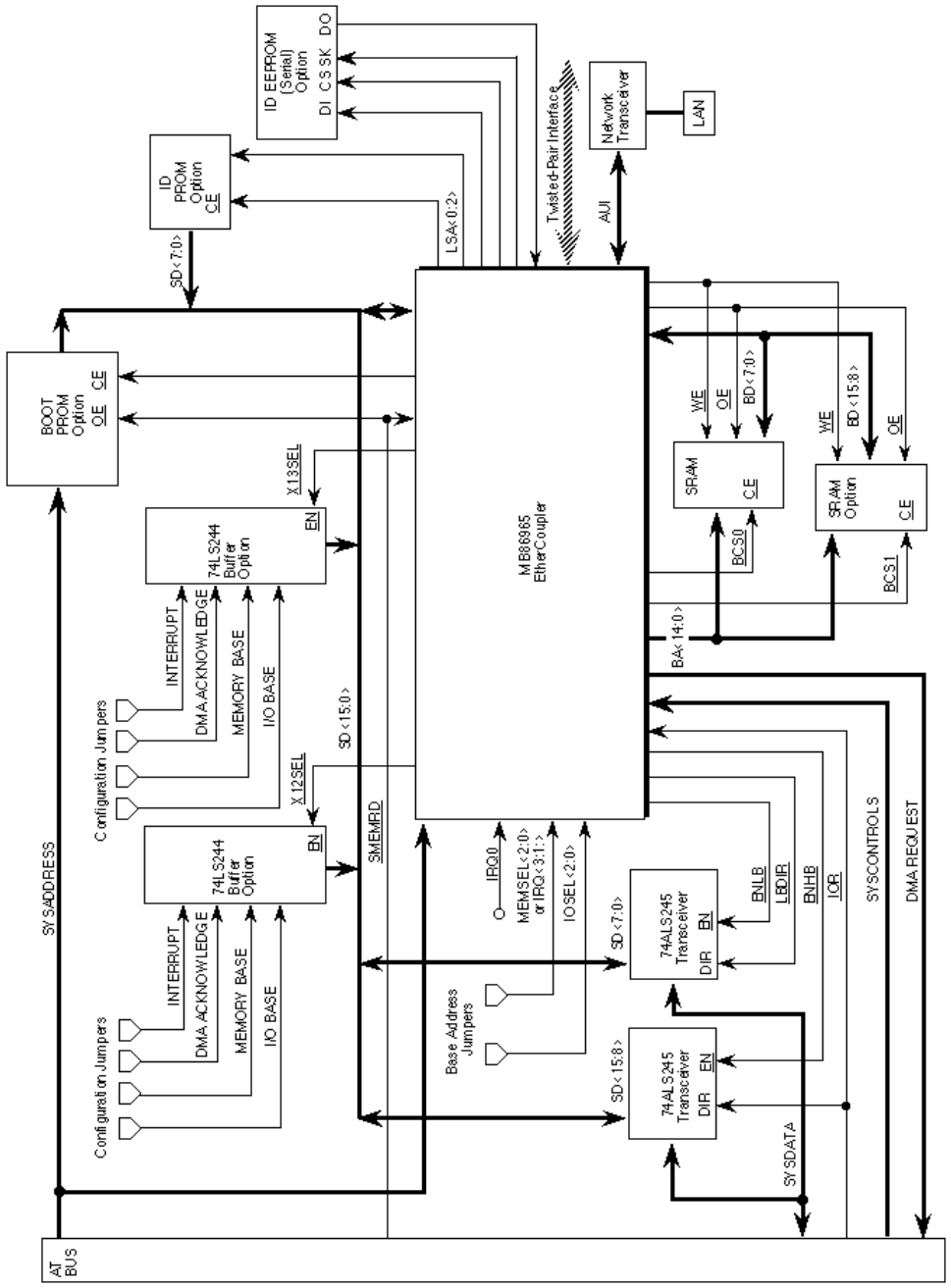


Figure 6. EtherCoupler Bus Operation Block Diagram

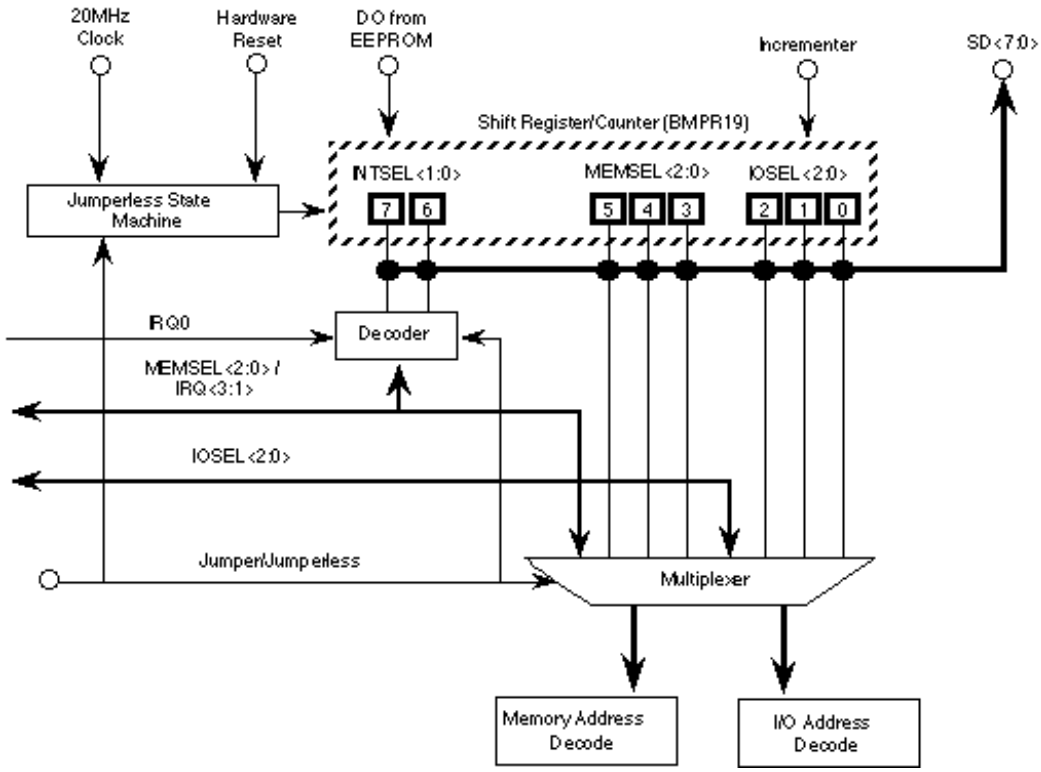


Figure 7. Jumperless/Jumpered Logic

Jumperless And Jumpered Mode Configuration And Control Registers

The jumperless EtherCoupler design approach assumes that an EEPROM is available for storage of I/O and memory base addresses, as well as system interrupt information in the first (Configuration) byte.

Jumperless/Jumpered Operation

The interrupt, I/O base address, and memory base address are set up on powerup and after system hardware reset, by first reading the configuration byte from the EEPROM into a special serial-in/parallel-out shift register/counter, as shown in Figure 7. Tables 1 through 4 provide detailed information about registers used for EtherCoupler Jumperless and Jumpered mode configuration and control.

Abbreviations that describe control and status bits are shown below.

Type	Description
R	Readable
W	Writable
N	Not used, reserved; write 0 when written.
C	Clears associated status bit and interrupt
0/1	Initial state after hardware reset

Table 1. BMPR16 — EPROM Control Register

BIT	SYMBOL	TYPE	DESCRIPTION
7	0	9	**RESERVED:** Write 0
6	ESK	W	**EEPROM SHIFT CLOCK:** Generates a shift clock signal for the EEPROM, by writing 1 or 0 from the software program. Write only.
5	ECS	W	**EEPROM CHIP SELECT:** When set to 1, generates a chip select signal for the EEPROM. Write only.
4-0	0	0	**RESERVED:** Write 0

Table 2. BMPR17 — EEPROM Data Register

BIT	SYMBOL	TYPE	DESCRIPTION
7	EDIO	W/R	**EEPROM INPUT/OUTPUT CLOCK:** When set to 1, indicates data is being input to the EEPROM. When set to 0, indicates that data is being output from the EEPROM
6-0	0	0	**RESERVED:** Write 0

Table 3. BMPR18 — IO Base Address Register

BIT	SYMBOL	TYPE	DESCRIPTION
7-0	IOBA<7:0>	R	**I/O BASE ADDRESS:** When in Jumperless mode, enables storage and incrementing if I/O base address. Read only.
7-0	X12SEL	R	**ADDRESS 12 SELECT:** When set to high in Jumpered mode, enables reading of jumpered information (address 0X12) from the board.

Register BMPR16 controls the selection and operation of the EEPROM and operates in Jumperless mode only.

Register BMPR 17 controls the input or output of the EEPROM and operates in Jumperless mode only.

Register BMPR18 temporarily stores, increments and reads out the I/O base address in Jumperless mode, when in Mode 3 operation with an EEPROM, and generates the X12SEL signal in Jumpered mode to enable reading of jumpered information from the board.

Register BMPR18 temporarily stores, increments and reads out the I/O base address in Jumperless mode, when in Mode 3 operation with an EEPROM, and generates the X12SEL signal in Jumpered mode to enable reading of jumpered information from the board.

Register BMPR19 reads the EtherCoupler jumperless configuration data in Jumperless mode, and generates the X13SEL signal in Jumpered mode to enable reading of jumpered information from the board.

Memory Base Address Configuration

Table 4 shows how BMPR19<5:3> defines the ROM Address read from the EEPROM. The process of reading the byte begins at system powerup, and is in parallel with system BIOS powerup initialization and checkout. This allows the BIOS to scan the ROM location for a valid boot ROM. If there are no memory address conflicts, the boot ROM initialization code then initializes its resources, replaces the required interrupt vectors, and returns control to system BIOS to complete system initialization.

If there are any memory address conflicts, the system may halt during powerup. If so, EtherCoupler should be removed from the system and installed in a system with no memory address conflict. This allows the user to reprogram the EEPROM with a different memory base address. An adapter card reprogrammed with a new base address can then be reinstalled in the original system. In case of I/O conflicts, EtherCoupler is software-configurable. In jumperless mode, removal of EtherCoupler from the system is unnecessary.

Table 4.BMPR19 — Jumperless Configuration Register

BIT	SYMBOL	TYPE	DESCRIPTION
7 6	INTSEL1 INTSEL0	R R	**INTERRUPT SELECT:** In Jumperless mode, enables definition of the interrupt select signal. Ready only.

BIT7	BIT6	DESCRIPTION
0	0	INT0
0	1	INT1
1	0	INT2
1	1	INT3

BIT	SYMBOL	TYPE	DESCRIPTION
5 4 3	MEMSEL2 MEMSEL1 MEMSEL0	R R R	**INPUT/OUTPUT CHANNEL READY:** In Jumperless mode, enables ROM address decoding. Read only.

BIT5	BIT4	BIT3	ROM ADDRESS DECODING
0	0	0	C4000 – C7FFF
0	0	1	C8000 – CBFFF
0	1	0	CC000 – CFFFF
0	1	1	D0000 – D3FFF
1	0	0	D4000 – D7FFF
1	0	1	D8000 – DBFFF
1	1	0	DC000 – DFFFF
1	1	1	Decode disabled

BIT	SYMBOL	TYPE	DESCRIPTION
2 1 0	IOSEL2 IOSEL1 IOSEL0	R R R	**INPUT/OUTPUT SELECT:** In Jumperless mode, enables I/O Base Address decoding. Ready only.
7-0	X13CEL	R	**ADDRESS 13 SELECT:** When set to high in Jumperd mode, enables reading of jumpered information (address 0X13) form the board.

BIT2	BIT1	BIT0	I/O BASE ADDRESS DECODING
0	0	0	260-27F
0	0	1	280-29G
0	1	0	2A0-2BF
0	1	1	240-25F
1	0	0	340-35F
1	0	1	320-33F
1	1	0	380-39F
1	1	1	300-31F

I/O Base Address Configuration

After successful system powerup, the software utility reads and compares the Product Signature or powerup value of the first eight registers to see if the controller is set with a nonconflicting I/O base address. Product Signature is the initial value of the first eight EtherCoupler registers.

The I/O base address is determined by bits 2–0 of the configuration byte read from EEPROM into the shift register/counter at the time of powerup. After reading from the EEPROM, hardware logic sets the shift register/counter to counter mode, and assigns a Counter Enable (Counter/Shift Register Select) bit within EtherCoupler registers to monitor the register and counter modes of this special three-bit block.

Table 4 also shows how BMPR19<2:0> defines the I/O Base Address read from the EEPROM. If the Product Signature read does match expected values, there is no conflict in the I/O address range. If there is a conflict in setting the address, the system may ignore the existence of EtherCoupler, although the software utility should continue reading the 0X12 register. Because the cell is set for counter mode (I/O Base Unlock bit, BMPR13<7>), every time the register is read, bits 2, 1 and 0 increment by one, moving the I/O base address to the next setting. The utility should read the chip for the correct Product Signature each time the register/counter counts up to the next base I/O address. This mechanism allows an adapter board to dynamically reconfigure itself to a nonconflicting I/O base address.

Once the right address is achieved, software locks the register/counter cell to register Address mode by resetting the designated Counter Enable bit. The software utility then reads the new configuration from the 0X13 register, and reprograms the EEPROM configuration byte with correct base addresses.

Interrupt Definition

Additionally, Table 4 indicates how BMPR19<7:6> defines the interrupt read from the EEPROM. Bits 6 and 7, which are used to configure interrupts, can be programmed to select any one of four interrupt lines offered at the system interface side of the EtherCoupler. The user may connect these lines to any four available system interrupt lines. The software utility tests the interrupt configuration for conflict and, if conflicting, reprograms the EEPROM bits with the next available interrupt option, and reboots the system to try again.

REGISTERS

CONTROL AND STATUS REGISTERS

The control and status registers on EtherCoupler are accessed through (direct) register addresses XXX0H through XXXFH, and (indirect) register bank-switching bits RBS1 and RBS0, DLCR7<3:2>.

Table 5. Internal Register Address Map

REGISTER BANK SWITCHING		SYSTEM ADDRESS					REGISTER	
RBS1	RBS0	SA4	SA3	SA2	SA1	SA0	Name	Description
X	X	0	0	0	0	0	DLCR0	Transmit Status
X	X	0	0	0	0	1	DLCR1	Receive Status
X	X	0	0	0	1	0	DLCR2	Transmit Interrupt Enable
X	X	0	0	0	1	1	DLCR3	Receive Interrupt Enable
X	X	0	0	1	0	0	DLCR4	Transmit Mode
X	X	0	0	1	0	1	DLCR5	Receive Mode
X	X	0	0	1	1	0	DLCR6	Configuration 0
X	X	0	0	1	1	1	DLCR7	Configuration 1
0	0	0	1	0	0	0	DLCR8	Node ID 0
0	0	0	1	0	0	1	DLCR9	Node ID 1
0	0	0	1	0	1	0	DLCR10	Node ID 2
0	0	0	1	0	1	1	DLCR11	Node ID 3
0	0	0	1	1	0	0	DLCR12	Node ID 4
0	0	0	1	1	0	1	DLCR13	Node ID 5
0	0	0	1	1	1	0	DLCR14	TDR 0 (LSB)
0	0	0	1	1	1	1	DLCR15	TDR 1 (MSB)
0	1	0	1	0	0	0	HT8	Hash Table 0
0	1	0	1	0	0	1	HT9	Hash Table 1
0	1	0	1	0	1	0	HT10	Hash Table 2
0	1	0	1	0	1	1	HT11	Hash Table 3
0	1	0	1	1	0	0	HT12	Hash Table 4
0	1	0	1	1	0	1	HT13	Hash Table 5
0	1	0	1	1	1	0	HT14	Hash Table 6
0	1	0	1	1	1	1	HT15	Hash Table 7
1	0	0	1	0	0	0	BMPR8	Buffer Memory Port (LSB)
1	0	0	1	0	0	1	BMPR9	Buffer Memory Port (MSB)
1	0	0	1	0	1	0	BMPR10	Transmit Start
1	0	0	1	0	1	1	BMPR11	16 Collisions
1	0	0	1	1	0	0	BMPR12	DMA Enable
1	0	0	1	1	0	1	BMPR13	DMA Burst, Transceiver Mode
1	0	0	1	1	1	0	BMPR14	Filter Self Receive, Transceiver Interrupt Enable
1	0	0	1	1	1	1	BMPR15	Transceiver Status
1	1	0	X	X	X	X	Reserved	Not used.
X	X	1	0	0	0	0	BMPR16	EEPROM Control. Write Only.
X	X	1	0	0	0	1	BMPR17	EEPROM Data. Read/Write.

REGISTER BANK SWITCHING		SYSTEM ADDRESS					REGISTER	
RBS1	RBS0	SA4	SA3	SA2	SA1	SA0	Name	Description
X	X	1	0	0	1	0	BMPR18	I/O Base Address. Read Only.
X	X	1	0	0	1	1	BMPR19	Jumperless Configuration. Read Only.
X	X	1	0	1	0	0	Reserved	Not used.
X	X	1	0	1	0	1	Reserved	Not used.
X	X	1	0	1	1	0	Reserved	Not used.
X	X	1	0	1	1	1	Reserved	Not used.
X	X	1	1	0	0	0	IDRB0	ID ROM Byte 0
X	X	1	1	0	0	1	IDRB1	ID ROM Byte 1
X	X	1	1	0	1	0	IDRB2	ID ROM Byte 2
X	X	1	1	0	1	1	IDRB3	ID ROM Byte 3
X	X	1	1	1	0	0	IDRB4	ID ROM Byte 4
X	X	1	1	1	0	1	IDRB5	ID ROM Byte 5
X	X	1	1	1	1	0	IDRB6	ID ROM Byte 6
X	X	1	1	1	1	1	IDRB7	ID ROM Byte 7

Table 5 summarizes the addressing scheme and provides functional selection criteria for system addressing. System address is relative to base I/O address selected by IOSEL<2:0>. Writing to a location within the optional IDROM space resets EtherCoupler internal registers and state machines. In System Word mode, data transfers 16 bits at a time on the system bus, or 8 bits at a time by using EtherCoupler byte lane controls. In 16-bit mode, even direct addresses select the registers when transferring to and from the registers. For example, address XXX0H accesses the Transmit/Receive Status registers. Transmit Status would be on the low byte and Receive Status on the high byte. Appropriate byte-access processor instructions access high and low bytes separately.

Tables 6–11 describe control and status bits.

Table 6. Control and Status Bits, DLCR0-7

REGISTER	BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
TX STATUS DLDR0	TX DONE	NET BSY	TX PKT RCD	CR LOST	JABBER	COL	16 COL	0
RX STATUS DLCR1	RX PKT	BUS RD ERR	DMA EOP	RMT 0900H	SHORT PKT ERR	ALIGN ERR	CRC ERR	RX BUF OVRFLO
TX INT ENABLE DLCR2	INT EN	0	0	0	INT EN	INT EN	INT EN	0
RX INT ENABLE DLCR3	INT EN	INT EN	INT EN	INT EN	INT EN	INT EN	INT EN	INT EN
TX MODE DLCR4	COL CTR 3	COL CTR 2	COL CTR 1	COL CTR 0	0	CNTRL & DREQ EXTND	LBC	EN TX DEFER
RX MODE DLCR5	0	RX BUF EMPTY	ACPT BAD PKTS	RX SHORT ADDR	ACPT SHORT PKTS	RMT RST	AF1	AF0
CONFIG 0 DLCR6	DLC EN	100NS/ 150NS SRAM	SB/SW	BB/BW	TBS 1	TBS 0	BS 1	BS 0
CONFIG 1 DLCR7	ECID 1	ECID 0	PWRDN	RDYPN SEL	RBS 1	RBS 0	EOPPOL	M..L/L..M

Table 7. Control and Status Bits, BMPR8-15

REGISTER	BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
BUF MEM PORT (LSB) BMPR8	7	6	5	4	3	2	1	0
BUF MEM PORT (MSB) BMPR9	15	14	13	12	11	10	9	8
TX START BMPR10	TX START	TX PKT CNT 6	TX PKT CNT 5	TX PKT CNT 4	TX PKT CNT 3	TX PKT CNT 2	TX PKT CNT 1	TX PKT CNT 0
16 COLLISIONS BMPR11	0	0	0	0	0	16 COL CNTRL 2	16 COL CNTRL 1	16 COL CNTRL 0
DMA ENABLE BMPR13	0	0	0	0	0	0	RX DMA EN	TX DMA EN
DMA BURST/ TXVR MODE BMPR13	I/O BASE UNLOCK	LOWER SQUELCH THRESH	LINK TEST EN	AUI/TP	AUTO PORT SEL	STP/UTP	BURST 1	BURST 0
FILTER SELF RX BMPR14	INT EN	INT EN	INT EN	0	0	SKIP PKT	INT EN	FILTER SELF RX
TXVR STATUS BMPR15	RLD	LLD	RJAB	RMT PORT	RXI PIN REV	0	SQE	0

Table 8. Control and Status Bits, BMPR16-19

REGISTER	BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
EEPROM CONTROL BMPR16	0	ESK	ECS	0	0	0	0	0
EEPROM DATA BMPR17	EDIO	0	0	0	0	0	0	0
IO BASE ADDRESS INCREMENT & X12SEL BMPR18	0	0	0	0	0	0	0	0
JUMPERLESS & X13SEL BMPR19	INTSEL1	INTSEL0	MEMSEL2	MEMSEL1	MEMSEL0	IOSEL2	IOSEL1	IOSEL0

Table 9. Control and Status Bits, DLCR8-15

REGISTER	BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
NODE ID 0 DLCR8	7	6	5	4	3	2	1	0
NODE ID 1 DLCR9	15	14	13	12	11	10	9	8
NODE ID 2 DLCR10	23	22	21	20	19	18	17	16
NODE ID 3 DLCR11	31	30	29	28	27	26	25	24
NODE ID 4 DLCR12	39	38	37	36	35	34	33	32
NODE ID 5 DLCR13	47	46	45	44	43	42	41	40
TDR 0 DLCR14	7	6	5	4	3	2	1	0
TDR 1 DLCR15	N/A (0)	N/A (0)	13	12	11	10	9	8

Table 1–. Control and Status Bits, HTR8-15

REGISTER	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
HASH TABLE 0 HT8	7	6	5	4	3	2	1	0
HASH TABLE 1 HT9	15	14	13	12	11	10	9	8
HASH TABLE 2 HT12	23	22	21	20	19	18	17	16
HASH TABLE 3 HT11	31	30	29	28	27	26	25	24
HASH TABLE 4 HT12	39	38	37	36	35	34	33	32
HASH TABLE 5 HT13	47	46	45	44	43	42	41	40
HASH TABLE 6 HT14	55	54	53	52	51	50	49	48
HASH TABLE 7 HT15	63	62	61	60	59	58	57	56

Table 11. Control and Status Bits, Packet Buffer Headers

REGISTER	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
TRANSMIT PACKET HEADER								
TX LENGTH (LSB) TPH1	7	6	5	4	3	2	1	0
TX LENGTH (MSB) TPH2	N/A (0)	N/A (0)	N/A (0)	N/A (0)	N/A (0)	10	9	8
RECEIVE PACKET HEADER								
PKT STATUS RPH1	N/A (0)	N/A (0)	GOOD PKT	RMT 0900H	SHORT ERR	ALIGN ERR	CRC ERR	N/A (0)
RESERVED RPH2	N/A (0)	N/A (0)	N/A (0)	N/A (0)	N/A (0)	N/A (0)	N/A (0)	N/A (0)
RX LENGTH (LSB) RPH3	7	6	5	4	3	2	1	0
RX LENGTH (MSB) RPH4	N/A (0)	N/A (0)	N/A (0)	N/A (0)	N/A (0)	10	9	8

DATA LINK CONTROL REGISTERS

Status and control bits for EtherCoupler Transmitter and Receiver sections are located in Data Link Control registers DLCR0 through DLCR7.

Transmit Status Register

As shown in Table 12, this register provides transmit status for the host processor. The system can enable interrupts based on setting (active high) of bits 7, 3, 2, or 1 of this register by setting the corresponding interrupt enable bits in the Transmit Interrupt Enable register, DLCR4.

Writing 1 to bits 7, 3, 2, and 1 clears this register, which can generate interrupts. Writing 0 to these bits has no effect; only EtherCoupler control logic can set these bits. Clearing the bit that causes an interrupt clears the bit and the interrupt. Because two or more status conditions may occur simultaneously, the interrupt routine must read and act on all status conditions that are set.

One method to clear interrupts is to read the contents of the status register, then write the same value back to the register, thus clearing all bits that were set. Another technique is to clear each status bit separately, by writing its mask (interrupt enable) to the register. This might be done as the corresponding interrupt service is performed. Note that wholesale clearing of all status bits by writing FFH to the register is not recommended, because this action may clear a just-set status that has not yet been read by the system. Note also that the Transmitter must be idle and TX DONE, DLCR0<7>, must be cleared by writing 1 to it, before starting the Transmitter by writing 1 to TX START bit, BMPR10<7>.

Table 12. DLCR0 — Transmit Status Register

BIT	SYMBOL	TYPE	DESCRIPTION
7	TX DONE	RC0	**TRANSMIT DONE:** This bit is set high when all packets in the active TRANSMIT BUFFER have been successfully transmitted to the LAN media, or skipped due to excessive collisions. Can generate interrupts if enabled by DLCR 2<7>.
6	NET BSY	R	**NET BUSY:** This is a real-time image of the Carrier Sense signal of the Receiver.
5	TX PKT RCD	RC0	**TRANSMIT PACKET RECEIVED:** Indicates that good packet was received by EtherCoupler shortly after transmission was completed. This is used to indicate self-reception of the packet. This bit is cleared as each transmission begins, or by writing 1 to it.
4	CR LOST	R0	**CARRIER LOST:** This bit is set if the receive Carrier Sense input is negated during a packet transmission. This can be caused by collision or a shorted LAN medium. It is automatically cleared as each transmission begins.
3	JABBER	RC0	**JABBER:** When active high, indicates excessive transmit length is detected by Jabber timer. Can generate interrupts if enabled by INT EN bit, DCLR2<3>.
2	COL	RC0	**COLLISIONS:** This bit will assert during transmission of a data packet if a collision occurs on the network. The Buffer Controller will automatically retirements the current packet after collision up to 16 times. The user may read the number of consecutive collisions in collision counter, DCLR4<7:4>. Can generate interrupts if enabled by INT EN bit, DCLR2<2>.
1	16 COL	RC0	**16 COLLISIONS:** This bit is set after the sixteenth unsuccessful transmission of the same packet. Can generate interrupts if enabled by INT EN bit, DCLR2<1>.
0	0	R0	**RESERVED:** Write 0.

Receive Status Register

Table 13. DLCR1 — Receive Status Register

BIT	SYMBOL	TYPE	DESCRIPTION
7	RX PKT	RC0	**RECEIVE PACKET:** Set when a new receive packet is stored on the Receive Buffer. Can generate interrupts if enabled by INT EN bit, DCLR3<7>.
6	BUS TS ERR	RC0	**BUS READ ERROR:** Set when a ready response cannot be issued within 2.4 milliseconds after the —SMEMRD signal is asserted. Occurs when reading an empty buffer. Can generate interrupts if enabled by INT EN bit, DCLR3<6>.
5	DMA EOP	RC0	**DMA END OF PROCESS:** Set when the host DMA asserts the EOP pin indicating that the process is finished. when set, inhibits further assertion of DREQ. Cleared by writing 00H to BMPR12, or by writing 1 to this bit. Can generate interrupts if enabled by INT EN bit, DCLR3<5>.
4	RMT 09900H	RC0	**REMOTE CONTROL PACKET RECEIVED:** This bit is set if a packet is received with 0900H in its Length/Typed field (two bytes following the source address, received MSB first). Can generate interrupts if enabled by INT EN bit, DCLR3<4>.
3	SHORT PKT ERR	RC0	**SHORT PACKET ERROR:** This bit is set when a packet is received with less than 60 bytes, excluding its Preamble and CRC fields. Such a packet usually indicates a collision has truncated its original length, since IEEE 802.3 minimum length is 60 bytes. Can generate interrupts if enabled by INT EN bit, DCLR2<3>.
2	ALIGN ERR	RC0	**ALIGNMENT PACK ERROR:** this bit will assert if a packet is received with an alignment error, meaning there were 1 to 7 extra bits at the end of the packet. Such an occurrence usually indicates a collision, or a faulty transceiver. Can generates interrupts if enabled by INT EN bit, DCLR3<2>.
1	CRC ERR	RC0	**CRC PACK ERROR:** This bit is set if a packet is received with a CRC error. This usually indicates a collision has corrupted the packet. Can generate interrupts if enabled by INT EN bit, DLCR3<1>.
0	RX BUR OVEFLO	RC0	**RECEIVE BUFFER OVERFLOW:** This bit will be set if the Receive Buffer becomes full and must reject a packet for lack space. Can generate interrupts if enabled by INT EN bit, DLCR3<0>.

As shown in Table 13, this register contains eight status bits which can generate interrupts if enabled by the corresponding bit in DLCR3. Five of these bits report the status of the most recently received packet accepted for storage in the Receive Buffer. Bit 7, RX PKT, is set whenever a new packet is successfully received and stored in the buffer. One bit reports reception of a special packet with 0900H in its type field. Other bits in this register report buffer overflow, DMA end of process, and bus read error. Bits 1, 2 and 3 indicate errors, if any, detected in the packet. If ACPT BAD PKTS bit, DLCR5<5> or ACPT SHORT PKTS bit, DLCR5<3> are set allowing acceptance of a bad packet, these error indicators will be stored in the Status Byte of the Receive Packet Header. If DLCR5<5> and/or DLCR5<3> are both 0, all packets with detected errors are automatically discarded and not stored in the buffer.

The bits in this register are cleared by writing 1 to the bit. Writing 0 to these bits has no effect; only EtherCoupler control logic can set these bits. Clearing the bit that causes an interrupt clears the bit and the interrupt. Because two or more status conditions can occur simultaneously, the interrupt routine must read and act on all status conditions that are set. One method to clear interrupts is to read the contents of the status register, then write the same value back to the register, thus clearing all bits that were set. Another technique is to clear each status bit separately, by writing its (interrupt enable) mask to the register. This might be done as the corresponding interrupt service is performed. Note that wholesale clearing of all status bits by writing FFH to the register is not recommended, because this action may clear a just-set status that has not yet been read by the system.

Transmit Interrupt Enable Register

Table 14. DLCR2 — Transmit Interrupt Enable Register

BIT	SYMBOL	TYPE	DESCRIPTION
7	INT EN	RC0	**INTERRUPT ENABLE:** When high, enables TX DON to generate interrupt. Also DCLR0<7>.
6	0	N0	**RESERVED:** Write 0.
5	0	N0	**RESERVED:** Write 0.
4	0	N0	**RESERVED:** Write 0.
3	INT EN	RC0	**INTERRUPT ENABLE:** When high, enables JABBER to generate interrupt. Also DCLR0<3>.
2	INT EN	RC0	**INTERRUPT ENABLE:** When high, enables COL to generate interrupt. Also DCLR0<2>.
1	INT EN	RC0	**INTERRUPT ENABLE:** When high, enables 16 COL to generate interrupt. Also DCLR0<1>.
0	0	N0	**RESERVED:** Write 0.

Receive Interrupt Enable Register

Table 15. DLCR3 — Receive Interrupt Enable Register

BIT	SYMBOL	TYPE	DESCRIPTION
7	INT EN	RC0	**INTERRUPT ENABLE:** When high, enables RX PKT to generate interrupt. Also DCLR1<7>.
6	INT EN	RC0	**INTERRUPT ENABLE:** When high, enables BUS RS ERR to generate interrupt. Also DCLR1<6>.
5	INT EN	RC0	**INTERRUPT ENABLE:** When high, enables DMA EOP to generate interrupt. Also DCLR1<5>.
4	INT EN	RC0	**INTERRUPT ENABLE:** When high, enables RMT 0900H to generate interrupt. Also DCLR1<4>.
3	INT EN	RC0	**INTERRUPT ENABLE:** When high, enables SHORT PKT ERR to generate interrupt. Also DCLR1<3>.
2	INT EN	RC0	**INTERRUPT ENABLE:** When high, enables ALIGN ERR to generate interrupt. Also DCLR1<2>.
1	INT EN	RC0	**INTERRUPT ENABLE:** When high, enables CRC ERR to generate interrupt. Also DCLR1<1>.
0	INT EN	RC0	**INTERRUPT ENABLE:** When high, enables RX BUF OVERFLO to generate interrupt. Also DCLR1<0>.

As shown in Table 14, this register contains the bits that enable the status bits in DLCR0 to generate interrupts. Bit 4 extends the DMA Request from the next-to-last to the last transfer cycle. Only bits 7, 3, 2, and 1 can generate interrupts; the other interrupt enable bits are not used.

As shown in Table 15, this register provides control for enabling interrupts based on the setting of status bits in DLCR1, the Receive Status register.

Table 16. Network Error Monitoring Modes

ACPT BAD PKTS DLCR5<5>	ACPT SHORT PKTS DLCR5<3>	INT EN SHORT PKT ERR DLCR3<3>	INT EN ALIGN ERR DLCR3<2>	INT EN CRC ERR DLCR3<1>	Mode
0	0	0	0	0	Normal, nominator mode.
0	0	X	X	X	Error interrupts only, if enabled.
0	1	0	X	X	Save short packets if otherwise error-free in buffer. Interrupts only for alignment and CRC errors, if enabled. RX PKT bit DLCR1<7> set high if short packet received.
1	0	0	0	0	Save packets with short, alignment or CRC errors in buffer. RX PKT bit DLCR1<7> set high if packet with error received.
All others					Not used.

Transmit Mode Register

Table 17. DLCR4 — Transmit Mode Register

BIT	SYMBOL	TYPE	DESCRIPTION
7	COL CTR 3	R*	**COLLISION COUNT 3 thru 0:** Indicate the number of consecutive collisions encountered by the current transmit packet. Read only. (***** Initial value is not predictable until first packet transmits.)
6	COL CTR 2	R*	
5	COL CTR 1	R*	
4	COL CTR 0	R*	
3	0	N0	**RESERVED.** Write 0.
2	DREQ EXTEND	RW1	**DMA REQUEST EXTEND:** When high, extends DMA Request until last transfer cycle.
	CNTRL	RW0	**CONTROL OUTPUT:** When low, the complement to this bit is for general use as the CNTRL signal, available on EtherCoupler pin 64. When low, DMA Request extends until next-to-last transfer cycle.
1	LBC	RW1	**LOOPBACK CONTROL:** This bit controls encoder/decoder loopback function. A zero on this bit places EtherCoupler in internal loopback mode.
0	EN TX DEFER	RW0	**ENABLE TRANSMIT DEFER:** Program this bit low for normal network operation. When high, the transmitter will not defer to traffic on the network, and will ignore the collision indications.

Table 16 shows details on Network Error Monitoring modes.

As shown in Table 17, this register contains two control bits associated with transmission, a general-purpose control bit that drives an EtherCoupler pin, and extends the DMA request, and a collision counter.

Table 18. Collision Count

COLLISION COUNT	16 COL DLCR0<1>	COL CTR3 DLCR4<7>	COL CTR2 DLCR4<6>	COL CTR1 DLCR4<5>	COL CTR0 DLCR4<4>	COL DLCR0<2>
0	0	0	0	0	0	0
1	0	0	0	0	1	1
2	0	0	0	1	0	1
3	0	0	0	1	1	1
4	0	0	1	0	0	1
5	0	0	1	0	1	1
6	0	0	1	1	0	1
7	0	0	1	1	1	1
8	0	1	0	0	0	1
9	0	1	0	0	1	1
10	0	1	0	1	0	1
11	0	1	0	1	1	1
12	0	1	1	0	0	1
13	0	1	1	0	1	1
14	0	1	1	1	0	1
15	0	1	1	1	1	1
16	1	0	0	0	0	1

Table 18 shows details on collision counting.

Receive Mode Register

As shown in Table 19, this register contains six bits that control receiver function, and one Receive Buffer status bit. Status bit RX BUF EMPTY, DLCR5<6>, is necessary to the software routine, which reads receive packets from the buffer. It tells the host routine whether there are any packets in the Receive Buffer that are complete and ready-to-read. In a multitasking system, this indicator would be used in conjunction with an interrupt when RX PKT asserts, which means a packet has arrived in memory. The interrupt would be used to start the routine that reads packets from the buffer.

As this routine begins, the interrupt on RX PKT can be disabled to prevent unneeded interrupts. After the first packet is read from the buffer, the RX BUF EMPTY bit would be read, to see if more packets have come in (packets may, at times, arrive in bursts). If the buffer is not empty, another packet would be read out, and this procedure repeated until the buffer is empty. After emptying the buffer, the host clears RX PKT, then re-enables interrupts on RX PKT, checks the buffer status one more time (because a packet can arrive at any time), then exits to do other tasks. Note that the packet header can reflect acceptance of a short packet (bytes 3 to 59).

Two of the control bits allow reception of packets with certain types of errors. The ACPT BAD PKTS bit, when set, causes the Receiver to retain and store in the buffer packets with CRC, alignment or short-length errors, provided that there was no indication of collision during reception. Likewise, the ACPT SHORT PKTS bit, when set, allows the retention of short packets down to and including only six bytes in length, excluding Preamble and CRC, provided that there was no indication of collision during reception and no alignment or CRC error.

Under normal operation, packets with less than 60 bytes, the IEEE 802.3 lower limit, would be discarded. These functions are provided for diagnostic purposes. Packets are accepted only if both the address filter and error filter are passed. Packets with no content errors, i.e., short, alignment or CRC, are accepted without regard to collision indications. Normally Node ID is six bytes, except if RX SHORT ADDR bit DLCR5<4> is set high, in which case only the first five bytes of the Node ID are checked.

Table 19. DLCR5 — Receive Mode Register

BIT	SYMBOL	TYPE	DESCRIPTION
7	0	N0	**RESERVED:** Write 0.
6	RX BUF EMPTY	R1	**RECEIVE BUFFER EMPTY:** Status bit which indicates that the Receive Buffer does not have any complete packets to read. (Read only.)
5	ACPT BAD PKTS	RW0	**ACCEPT BAD PACKETS:** When set high, allows packets with CRC, and/or alignment errors or packets that are short, to be saved into the Receive Buffer for analysis. Otherwise such packets would be discarded automatically by the Receiver and removed from the buffer.
4	RX SHORT ADDR	RW0	**RECEIVE SHORT ADDRESS:** When set high, instead of customary 48-bit NODE ID address filter, only first 40 bits of NODE ID are compared.
3	ACCPT SHORT PKTS	RW0	**ACCEPT SHORT PACKETS:** When set high, allows short packets (with less than 60 bytes, excluding Preamble and CRC, i.e. below IEEE minimum length) to be saved into the Receive Buffer. Otherwise such packets would be discarded automatically by the Receiver and removed from the buffer.

ACPT BAD PKTS	ACPT SHORT PKTS	SHORT PKT ERR DLCR1<3>	ALIGN ERR DLCR1<2>	CRC ERR DLCR1<1>	ERROR INTERRUPT
0	0	—	—	—	Accept
0	1	Accept	—	—	Accept
1	X	Accept	Accept	Accept	—

2	RMT RST	RW0	**REMOTE RESET:** When set high, enables remote reset of Receiver.
1	AF1	RW1	**ADDRESS FILTER MODE:** These two bits control the address filtering on incoming packets. Note that self reception of broadcast and multicast packets is prohibited except in Accept all packets and loopback modes. When LBC is low (loopback mode), broadcast packets can be self-received, except in Reject All Packets mode.
0	AF0	RW0	

AF1	AF0	Acceptable Address Description
0	1	NODE ID, Broadcast, Multicast, and 2nd-24th bits of NODE ID.
1	0	NODE ID, Broadcast, Multicast, and Hash Table.
0	0	Reject all packets.
1	1	Accept all packets.

Table 20. Reception Destination Address Filtering

Receiving From Other Nodes								
AF1 DLCR5 <1>	**AF0 DLCR5 <0>**	**LBC DLCR4 <1>**	**FILTER SELF RX BMPR14 <0>**	**Node ID Match***		**Broadcast Address**	**Multicast (bit 0=1) and**	
				Yes	**No**		**Node ID <23:1>**	**Hash Table**
0	0	X	X	—	—	—	—	—
0	1	0	X	—	—	—	—	—
0	1	1	X	Accept	—	Accept	Accept	—
1	0	0	X	—	—	—	—	—
1	0	1	X	Accept	—	Accept	—	Accept
1	1	0	X	—	—	—	—	—
1	1	1	0	Accept	Accept	Accept	Accept	Accept
1	1	1	1	Accept	Accept	Accept	Accept	Accept
Receiving Own Transmission								
AF1 DLCR5 <1>	**AF0 DLCR5 <0>**	**LBC DLCR4 <1>**	**FILTER SELF RX BMPR14 <0>**	**Node ID Match***		**Broadcast Address**	**Multicast (bit 0=1) and**	
				Yes	**No**		**Node ID <23:1>**	**Hash Table**
0	0	X	X	—	—	—	—	—
0	1	0	X	Accept	—	Accept	Accept	—
0	1	1	X	Accept	—	—	—	—
1	0	0	X	Accept	—	Accept	—	Accept
1	0	1	X	Accept	—	—	—	—
1	1	0	X	Accept	Accept	Accept	Accept	Accept
1	1	1	0	Accept	Accept	Accept	Accept	Accept
1	1	1	1	—	—	—	—	—

Note: A NODE ID match occurs when the incoming packet has a destination address whose first bit is a zero, and whose second through forty-eighth bits match bits 1-47 respectively of EtherCoupler's Node ID Registers.

As shown in Table 20, packet destination address filtering depends on the programming of Address Filter bits AF1 and AF0, Loopback Control bit LBC, and self-receive filter bit, Filter Self RX. Received packet addresses can be filtered by Node ID, Broadcast, Multicast plus bits 23 through 1 of Node ID, and/or Multicast plus Hash Table. To be accepted, a packet must not only pass requirements for the address filter but also error filter requirements. [Also refer to Table 30 and the discussion of Filter Self Receive Register BMPR14.]

Configuration Registers 0 and 1

Table 21. DLCR6 — Configuration Register 0

BIT	SYMBOL	TYPE	DESCRIPTION
7	DLC EN	RW1	**DATA LINK CONTROL ENABLE:** When low, enables EtherCoupler Receiver and Transmitter sections. This bit must be set high during initialization, and set low to enable loopback testing and operation on the network. Program Node ID and Hash Table only when this bit is high.
6	100 NS/ 150 NS	RW0	**SRAM CYCLE TIME:** Controls the SRAM time cycle. When high, selects 100 ns; when low, selects 150 ns.
5	SB / SW	RW1	**SYSTEM BYTE/WORD BUS WIDTH:** When high, system bus will operate in 8-bit data mode; when low, 16-bit data mode is selected. Also see BB/BW below.
4	BB / BW	RW1	**BUFFER BYTE/WORD WIDTH:** When high, buffer memory will operate in 8-bit data mode; when low, 16-bit data mode is selected.

SB / SW	BB / BW	System	Buffer
0	0	word	word
0	1	word	byte
1	0	Do not use	Do not use
1	1	byte	byte

BIT	SYMBOL	TYPE	DESCRIPTION
3 2	TBS1 TBS0	RW1 RW0	**TRANSMIT BUFFER SIZE:** Selects size of Transmit buffers.

TBS1	TBS0	Transmit Buffer Quantity	Size, each TX Buffer (kbytes)	Size, total TX Buffer (kbytes)
0	0	1	2	2
0	1	2	2	4
1	0	2	4	8
1	1	2	8	16

BIT	SYMBOL	TYPE	DESCRIPTION
1 0	BS1 BS0	RW1 RW0	**BUFFER SIZE:** Selects physical size of total SRAM buffer memory for transmit and receive functions.

BS1	BS0	SRAM size (kbytes)
0	0	8
0	1	16
1	0	32
1	1	64

Table 22. DLCR7 — Configuration Register 1

BIT	SYMBOL	TYPE	DESCRIPTION
7 6	ECID1 ECID0	R1 R1	**ETHERCOUPLER IDENTIFICATION:** When both set high, indicates to software that EtherCoupler rather than another Fujitsu product is being used as controller.
5	PWRDN	RW1	**POWERDOWN:** When set high, enables power to the chip for all functions; when set low, places chip in powerdown mode for power conservation.
4	RDYPNSEL	R 0/1	**READY PIN SELECT:** Reads the state of RDY POL signal at EtherCoupler pin 108.
3 2	RBS1 RBS0	RW0 RW0	**REGISTER BANK SELECT:** Provides the indirect address for selecting one of three sets of registers to access when the physical register address is xxx8H-xxxF. The lower seven registers are not bank-selectable.
			RBS1 / **RBS0** / **Registers** 0 / 0 / DLCR0 thru DLCR7, DLCR8 thru DLCR15 0 / 1 / word 1 / 0 / Do not use 1 / 1 / byte
1	EOPPOL	RW0	**EOP PIN SIGNAL POLARITY:** When high, the EOP pin is active-high; when low, EOP is active-low.
0	M..L / L.M	RW0	**BYTE ORDER CONTROL:** Selects byte lane ordering for packet data in the buffer (applies only in System Word mode). In both Most.Least and Least.Most modes, the first and second bytes of the packet, as well as its non-transmitted buffer header, will appear in the same word on the system bus. When this bit is high (M..L mode), the first and all odd-numbered bytes of a packet and its header appear on the high byte of the system bus. When low, the first and all odd-numbered bytes of a packet appear on the low byte. Note that header bytes are also swapped.

As shown in Tables 21 and 22, system configuration bits are found in these two registers. Among the configuration controls found here are physical memory size, partitioning between transmit and receive buffers, widths of memory and system busses, byte lane control, and powerdown control. Most configuration parameters are programmed only during initialization, after power start and hardware reset.

Software engineers, who want to use the same node driver for EtherCoupler, NICE and EtherStar controllers from Fujitsu, should note that the driver can determine which chip is being used, by reading DLCR7 or DLCR6 after hardware reset. EtherCoupler reads E0B6H (E0H or F0H for DLCR7 and B6H for DLCR6, when in byte mode); NICE reads 30B6H or 20B6H; and EtherStar reads 0000H.

Powerdown mode saves power when EtherCoupler is not in use. When ready to place the EtherCoupler chip in Powerdown mode, first write 1 to DLC EN, DLCR6<7>, to turn off the Receiver and Transmitter, then write 0 to PWRDN, DLCR7<5>. To exit the Powerdown mode, write 1 to PWRDN. Register contents are preserved, unless hardware is reset. Hardware reset also terminates the Powerdown mode.

Byte-order control provided by Most..Least/Least..Most bit, DLCR7<0>, provides compatibility with various higher-level protocols, such as TCP/IP and XNS. These protocols may have a different order for transmission of the bytes within a word. When M..L/L..M is low, the least-significant byte of the word transmits first, followed by the most-significant. When M..L/L..M is set high, the byte order reverses. This feature applies only when the system bus operates in 16-bit (word) mode.

The byte-order control works by reversing, or not reversing, the bytes of all words as they pass between the buffer memory and the system bus. Thus all data stored in the Transmit Buffer or retrieved from the Receive Buffer is affected, including nontransmitted headers. This control bit does not affect EtherCoupler registers other than Buffer Memory Port registers, BMPR8 and BMPR9. When using this feature, ensure the reversal of header information as well as packet data in the software driver code. See Table 23 for examples of using least..most and most..least byte ordering.

Table 23. Byte Ordering

DATA <15:8>	DATA <7:0>
For Transmit Packet	
Least....Most	
Transmit Length, high byte	Transmit Length, low byte
Destination Address, 2nd byte	Destination Address, 1st byte...
Source Address, 2nd byte	Source Address, 2nd byte...
Length Field, low byte*	Length Field, high byte*
Data Field, 2nd byte	Data Field, 1st byte
Most....Least	
Transmit Length, low byte *	Transmit Length, high byte*
Destination Addr, 1st byte	Destination Addr, 2nd byte...
Source Addr, 1st byte	Source Addr, 2nd byte...
Length Field, high byte	Length Field, low byte...
Data Field, 1st byte	Data Field, 2nd byte...
For Receive Packet	
Least....Most	
Unused, reserved	Receive Packet Status
Receive Length, high byte	Receive Length, low byte...
Destination Address, 2nd byte	Destination Address, 1st byte...
Source Address, 2nd byte	Source Address, 1st byte...
Length Field, low byte*	Length Field, high byte*
Data Field, 2nd byte	Data Field, 1st byte
Most....Least	
Receive Packet Status	Unused; reserved
Receive Length, low byte*	Receive Length, high byte*
Destination Addr, 1st byte	Destination Addr, 2nd byte...
Source Addr, 1st byte	Source Addr, 2nd byte...
Length Field, high byte	Length Field, low byte...
Data Field, 1st byte	Data Field, 2nd byte
Items with asterisk (*) are in numerically reversed byte order.	

NODE ID REGISTERS

The Node ID Registers are accessed in register bank "0" at register addresses xxx8H - xxxDH. During node initialization, the unique Ethernet address assigned to the node loads into these registers. The first register at xxx8H corresponds to the first byte of the Node ID, or the first address byte to be received as a packet arrives from the network. If EtherCoupler is configured to do so via Address Filter bits, DLCR5<1:0>, the destination address field of an incoming packet compares with the Node ID stored in these registers. The packet is accepted, if it passes the error filter and there is a match.

The Node ID registers are readable and writable registers, and should not be accessed while the Receiver is enabled. To avoid interaction with the Receiver, access the Node ID registers only when DLC EN, DLCR6<7>, is 1. It is recommended that the Node ID registers be written and read only during initialization before enabling the Receiver, i.e., before writing 0 to DLC EN. The address contained in the Node ID registers is used only for receive (destination) address filtering, not for the source address of outgoing packets. The system provides outgoing packet addresses as part of the packet data. Within each byte, bits are transmitted and received on the network in a least-significant-bit-first order.

Time Domain Reflectometry (TDR) Counter

The TDR Counter approximately indicates the location of a fault on the network, if one exists. When a node transmits, a short or open on the network causes a reflected signal to the node receiver, which can sometimes be detected. The reflection causes failure of the carrier sense or detection of a false collision. Time domain reflectometry allows estimates of the distance along the network cable from the node to the fault.

The TDR Counter counts the number of transmitted bits before occurrence of a collision or loss of carrier sense, whichever comes first. If neither occurs during packet transmission, the count clears. The elapsed time represents twice the signal delay from node to fault. An open on the network may cause a false collision, whereas a short usually causes loss of carrier sense. The TDR Count comes from DLCR14 (the least-significant byte) and DLCR15 (the most-significant byte). Only the lower 14 bits of the counter are equipped, which is more than is needed for an IEEE or Ethernet LAN. The top two bits, DLCR15<7:6>, are always 0.

To perform the TDR fault test, first enable interrupts for TX DONE, by setting DLCR2<7> high. An alternative to use the interrupt is to poll the TX DONE bit, looking for a high level. Set the 16 Collisions register, BMPR11, to 07H for this test (no halt, skip-failed packet). Clear status bits by writing FF86H to the Receive and Transmit Status registers. Next, try to transmit a packet length of 600 or more bits. Up to 16 attempts may be made automatically, if collisions are indicated. Upon completion of the transmission attempts, TX Done goes high, generating an interrupt, if enabled. When this occurs, read the Transmit Status register and the TDR register.

Interpreting The Results

If the count is zero, no fault was detected. If the count is greater than zero, but smaller than the packet length, a cable fault my exist. If the count is less than 525, a real collision may have occurred during test. Real collisions normally occur within the first 65 bytes of the packet, including preamble. Note the error messages, COL and CR LOST. COL high suggests a cable open, whereas CR LOST suggests a short. Repeat the measurement several times, throw out any anomalous values, and average the rest. A cluster of readings at about the same value is a strong indicator of a valid fault measurement. If such a cluster of readings occurs, multiply the average of the cluster by 39 feet to estimate the distance from the node to the fault. [39 feet = (100 ns x 0.8 x 186,282 miles/second x 5280 feet/mile)/2; this assumes the network is mostly coaxial cable with signal propagation speed of approximately 0.8 x C, the speed of light.]

HASH TABLE

The Hash Table provides a way to filter incoming multicast packets so the host processor need not process packets that are not of interest. The principal behind this filtering process is based on the arrangement of a large number of elements of an array, or database, to facilitate searching for elements associated with a given key or datum. The hash function is a mathematical or logical function that maps all elements in a domain onto a smaller domain called the hash table.

Assume this hashing function as an example: treat the multicast address as a nonnegative 48-bit integer, divide this number by 64 and take the remainder. This function maps all multicast addresses into a 64-element hash table because the remainder must be integer values 0 through 63. Applying this hashing function results in taking the least-significant six bits of the multicast address as an integer. In the hash table, for each element, 0 through 63, a single bit is stored to indicate if the address is accepted (1) or rejected (0). If, for example, the node belongs to three multicast groups, only three or fewer of the hash table elements store ones, and the rest store zeroes. The scheme

allows the acceptance of any number of addresses, including all of them. However, while this filters out most nonspecific addresses, there may be addresses not of interest used on the network that also fall into the accept elements, so filtering may be imperfect.

The actual hashing function used in EtherCoupler is to calculate the CRC on the multicast address and take the most-significant six bits of this calculation. The six bits are used to address the elements of the hash table. If a one is stored in an element of the table, associated packets are accepted. The hash filter criteria are only used on multicast addresses, which all start with a one. Node IDs that start with a zero are not filtered by the hash filter. The broadcast address, a special case of the multicast set wherein all the bits are ones, are accepted anyway unless the Reject All Packets mode is selected.

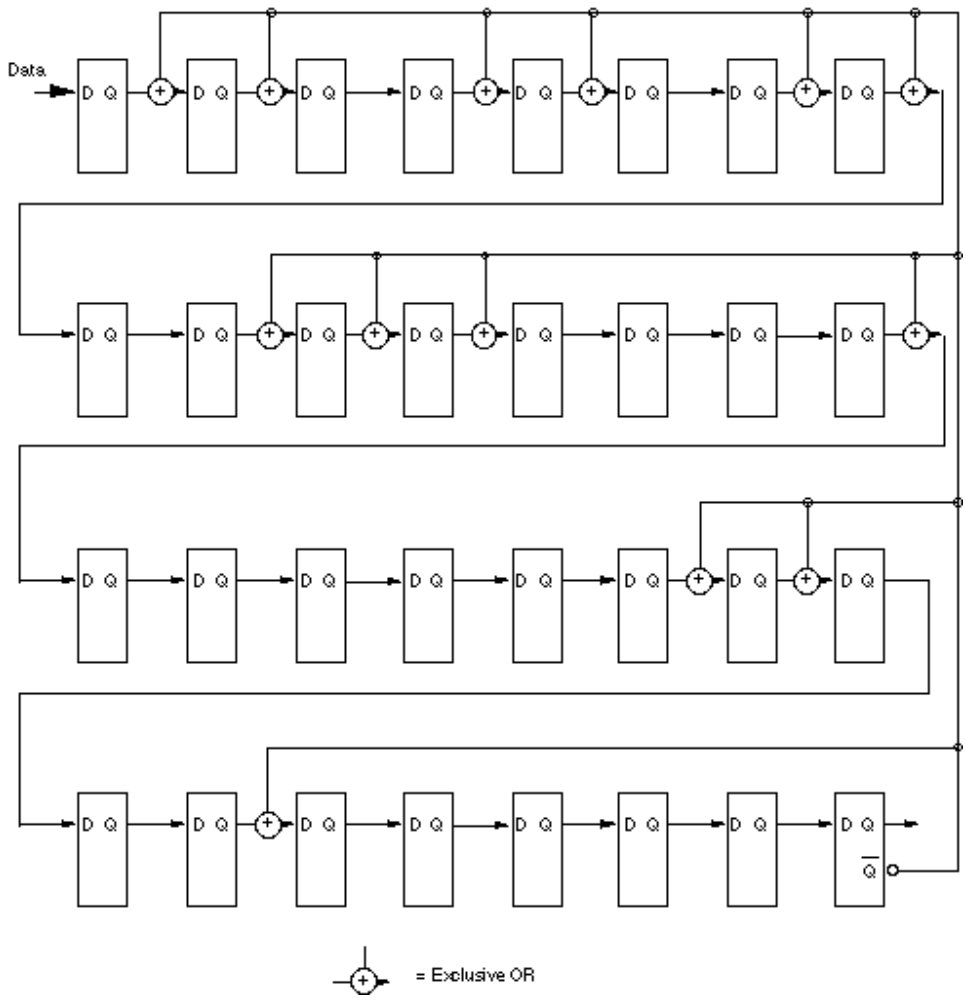


Figure 8. CRC Register Core

Figure 8 shows the Cyclical Redundancy Check (CRC) register core, which is a modified shift-right register used to generate and check CRCs. While some controllers share a single core between transmitter and receiver, EtherCoupler has one core for the generator and one core for the checker, thus allowing concurrent operation during self-receive.

To begin the calculation, the register is first set to all ones. For the generator case, as the packet transmits, data is clocked serially into the left-hand end of the register, starting with 48 bits of the destination address (the Preamble is skipped). After the last bit of the data field clocks into the register, the CRC calculation finishes. The feedback line is then forced low by the NOR gate, and the register becomes a simple shift-right register. Its contents then shift out serially, and are transmitted while appending the CRC to the end of the packet. Calculation for the CRC checker begins similar to CRC generation, by feeding the incoming data into the register. But the CRC field of the packet also feeds the calculation. The result is a fixed constant in the register, if no CRC error occurs.

For the Hash Filter, after the last bit of the destination address clocks into the register, the left-hand six register bits store in another register used to address the Hash Table elements. The left-most bit is most significant. The left-most three bits are used as the Hash Table register address, and the right-most three bits are used as the bit address within a register byte. Such selection of a Hash Table element yields a 1 in the table to indicate that the packet will be accepted, if it is a multicast packet (first bit of destination address equals 1), and passes the error filters.

The hash filter is used only when the Address Filter mode select bit AF1 is 1 and mode select bit AF0 is 0, selecting the Node ID, Broadcast, Multicast + Hash Table mode. Like the Node ID registers, Hash Table registers should only be accessed when the Receiver is disabled, i.e., when DLC EN is high, to avoid interaction with the Receiver. (Software code examples showing how to calculate entries for the Hash Table are available through Fujitsu sales offices.) There are eight bytes of registers in the Hash Table containing 64 one-bit elements, as shown in the Control and Status Table 11.

TRANSMIT AND RECEIVE PACKET HEADERS

Buffer memory is partitioned into Transmit and Receive sections, which store Packet Headers that precede associated outgoing and incoming packet information.

The Transmit Packet Header stored in the first two bytes of the Transmit Buffer reflects length characteristics of the current outgoing packet. Subsequent Transmit Buffer bytes store the contents of Destination ID, Source ID, Length, and Data fields of the outgoing packet.

The Receive Packet Header stored in the first four bytes of the Receive Buffer reflects packet conditions and length characteristics of the current incoming packet. (Refer to the discussion of Receive Status register DLCR1 for received errors.) Subsequent Receive Buffer bytes store the contents of Destination ID, Source ID, Length, and Data fields of the incoming packet.

Table 24. Receive Packet Header Condition

CONDITION	N/A (bit 7)	NA (bit 6)	GOOD PKT (bit 5)	RMT 0900H (bit 4)	SHORT PKT ERR (bit 3)	ALIGN ERR (bit 2)	CRC ERR (bit 1)	NA (bit 0)
GOOD PACKET	0	0	1	X	X	X	X	X
GOOD PACKET WITH TYPE=0900H	0	0	1	1	X	X	X	X
SHORT ERROR PACKET	0	0	0		1			
ALIGNMENT ERROR PACKET	0	0	0			1		
CRC PACKET ERROR	0	0	0				1	
RECEIVE BUFFER OVERFLOW ERROR	0	0	0					1
As indicated by shading in the table, EtherCoupler may detect multiple errors as the occur simultaneously, and then reflect their status in the Receive Packet Header.								

Table 24 identifies possible conditions for incoming packets and describes the contents of the Receive Packet Header byte under such conditions. [Also see Table 11, Control and Status Bits, Packet Buffer Headers.]

Transmit Packet Length

An 11-bit integer indicates the number of bytes in the packet to be transmitted, excluding Preamble and CRC fields generated by EtherCoupler.

Receive Packet Status

The receive packet header comprises one byte of packet status, an unused byte and two bytes (11 bits) for packet length in bytes. Bits 1 through 4 of the status byte are an image of the same bits in Receive Status register, DLCR1, with respect to the packet that follows. Bit 5 is the GOOD PKT bit, which when set to 1 indicates that no errors were detected in the packet. Bits 0, 6 and 7 are unused and are always set to 0.

Receive Packet Length

The third and fourth bytes of the receive packet header indicate the total number of bytes in the stored packet data.

BUFFER MEMORY PORT REGISTERS

Buffer Memory Port registers BMPR8 and BMPR9 provide the host with access to buffer memory. While EtherCoupler is on the network, Register Bank Select bits RBS1 and RBS0, DLCR7<3:2>, may be set to 1 and 0, respectively, to facilitate access to buffer memory.

Writing a byte/word to BMPR8 transfers that data to the currently addressed location in the transmit buffer, and increments the transmit buffer pointer to point to the next byte/word. Reading a byte/word from this port transfers the contents of the currently addressed location in the receive buffer to the host, and increments the receive buffer pointer to point to the next byte/word.

BMPR9 is used only in word mode as the high byte of the word. In word mode, all transfers must be 16-bits wide, as the Buffer Memory Port register does not support byte-wide transfers in this mode. As required, other registers can be accessed word-wide, high-byte-only or low-byte-only.

Transmit Start Register

Table 25 describes Transmit Start register, BMPR10, which contains the TX START bit and the TX PKT CNT bits. Writing a 1 to the TX START bit immediately starts the Transmitter. Transmit Packet Count is a seven-bit binary integer written by the host to indicate the number of packets in the transmit buffer to be transmitted. The Transmit Start register should be written only when the transmitter is idle, but can be read at any time. TX START is always read as a 1.

Table 25. BMPR10 — Transmit Start Register

BIT	SYMBOL	TYPE	DESCRIPTION
7	TX START	W0	**TRANSMITTER START:** Writing 1 to this bit commands the Transmitter to start transmitting packets loaded into the transmit buffer. Before doing so, the Transmitter must be idle (not busy with another buffer).
6 – 0	TX PKT CNT 6 – 0	RW0	**TRANSMIT PACKET COUNT:** A binary integer written by the system to indicate the number of packets contained in the transmit buffer for transmission. This information can be loaded at the same time the TX START bit is set high. As the Transmitter finishes transmitting each packet, this counter is decremented. The value can be read by the system to see how many packets remain to be transmitted.

Table 26. BMPR11 — 16 Collisions Control Register

BIT	SYMBOL	TYPE	DESCRIPTION
7 – 3	0	RW0	**RESERVED:** Write 0.
2	16 COL CNTRL 2	RW0	**16 COLLISIONS CONTROL:** Masks the internal logic for 16 COL CONTRL0 BMPR11<1>. If this bit is set to 0, EtherCoupler halts when the 16th collision occurs. If this bit is set to 1, EtherCoupler does not halt when the 16th collision occurs.
1	16 COL CNTRL 1	RW0	**16 COLLISIONS CONTROL:** Restarts transmission when set to 1. Is set to 0 by EtherCoupler before transmitting a packet, when 16 COL CNTRL 2 set to 0.
0	16 COL CNTRL 0	RW0	**16 COLLISIONS CONTROL:** Retransmit or throw away this transmitted packet, when the 16th collision is met on the transmitted packet. When set to 0, retransmit this packet. When set to 1, throws away this packet.

16 COL CNTRL 2 (BIT 2)	16 COL CNTRL1 (BIT 1)	16 COL CNTRL 0 (BIT 0)	Description
0	0	X	Halt on 16 collisions.
0	1	0	Retransmit the packet, following a halt.
0	1	1	Throw away this packet and resume transmitting, following a halt.
1	X	0	Don't halt; retransmit current packet.
1	X	1	Don't halt; throw away current packet; and go to next packet.

16 Collisions Control Register

Table 26 describes 16 Collisions Control register, BMPR11, which controls action taken when each of 16 consecutive attempts to transmit a packet are met with collision. The 16 Collisions Control register serves as a mode-select register and an action command register. As a mode select register, two functions are selectable: automatic continuation, or halt after 16 collisions. If automatic continuation is selected, there is an option to continue attempting to transmit the same packet or skip to the next packet. If halt is enabled, Transmitter restarts after a halt by writing an action code listed in Table 27.

Table 27. Collision Action Codes Written to BMPR11

ACTION CODE	ACTIONS
02H or 03H	**MODE SETUP:** Halt after 16 collisions.
02H	**COMMAND:** Resume transmitting, repeat failed packet. For use following a halt. Terminates the halt. Instructs Transmitter to resume transmitting by repeating the failed packet. The collision counter is reset, allowing up to 16 additional attempts to be made. Transmitter will again halt after 16 collisions.
03H	**COMMAND:** Resume transmitting, skip failed packet. For use following a halt. Terminates the halt. Instructs Transmitter to skip the failed packet and resume transmitting with the next packet in the buffer. The collision counter is reset, allowing up to 16 additional attempts to be made. If there is no next packet, the Transmitter deactivates, setting TX DONE bit, DLCR0<7>, as it does so. Transmitter halts after 16 collisions.
06H	**MODE SETUP:** Continue automatically after 16 collisions, repeat failed packet. Note that if the network medium disconnects, transmission attempts usually result in false collision detection. Under this condition, this mode causes Transmitter to continue attempting transmission of the same packet indefinitely. Interrupt or periodic polling of the status bits could detect this condition.
07H	**MODE SETUP:** Continue automatically after 16 collisions, skip failed packet. Note that this mode results in failure to transmit some packets, because it skips a packet that has had 16 consecutive collisions. While this condition is rare on a healthy network, it does occur occasionally.

DMA Enable Register

Table 28. BMPR12 — DMA Enable Register

BIT	SYMBOL	TYPE	DESCRIPTION		
7 – 2	0	0	**RESERVED:** Write 0.		
1	RX DMA EN	WR0	**DMA RECEIVE ENABLE:** When set to 0, disables DMA read. When set to 1, enables DMA read.		
0	TX DMA EN	WR0	**DMA TRANSMIT ENABLE:** When set to 0, disables DMA write. When set to 1, enables DMA write.		
			RX DMA EN	**TX DMA EN**	**ACTIONS**
			0	0	Clear or terminate DMA activity, DMA EOP status bit and associated interrupt, if any. Normally used as response to End of Process (DMA EOP) interrupt.
			0	1	Enable Transmit Write DMA.
			1	0	Enable Receive Read DMA.

Table 28 describes DMA Enable register, BMPR12, a write-only register that enables or clears Receive Read DMA or Transmit Write DMA.

Table 29 describes DMA Burst and XVR Mode register, BMPR13, which selects the burst length for DMA operation, and programs the 10BASE-T Transceiver modes. Each burst is one word or one byte, depending on System Byte/System Word mode selected, DLCR6<5>. Writing code 00H, 01H, 02H or 03H to the register selects burst length as 1, 2, 4, or 12 transfers.

DMA Burst and Transceiver Mode Register (BMPR13)

Table 29. BMPR13 — DMA Burst and Transceiver Mode Register

BIT	SYMBOL	TYPE	DESCRIPTION
7	I / O BASE UNLOCK	WR1	**I/O BASE UNLOCK:** When set to 1, enables the I/O Counter for I/O space change. When the bit is set to 1, an I/O read address X12H increases the I/O space 32 bytes at a time. The I/O space change is between 260H and 3FFH. When set to 0, this I/O incremental function is disabled. Any I/O address read as X12H does not change the selected space.
6	LOWER SQLCH THRESH	WR0	**LOWER SQUELCH THRESHOLD:** When set to 1, reduces twisted-pair squelch threshold by 4.5 dB. When set to 0, twisted-pair squelch threshold is normal.
5	LINK TEST EN	WR0	**LINK TEST ENABLE:** When set to 1, disables transmit and receive link integrity test functions. When set to 0, enables link integrity test.
4	AUI / TP	WR0	**AUI /TP PORT SELECT:** When AUTO PORT SELECT bit, BMPR13<3>, is set to 1, the Manual Port Select mode is in effect. When set to 1, the AUI port is selected. When set to 0, the TP port is selected.
3	AUTO PORT SEL	WR0	**AUTOMATIC PORT SELECTION:** When set to 0, Automatic Port Selection mode is in effect. EtherCoupler defaults to the AUI port if twisted-pair link integrity fails. When set to 1, allows manual port selection via AUI/TP bit, BMPR13<4>, which determines the active port.
2	STP / UTP	WR0	**STP / UTP SELECT:** When set to 1, selects 150 ΩΩ termination for shielded twisted pair. When set to 0, selects 100 ΩΩ termination for unshielded twisted pair.
1 0	BURST 1 BURST 0	WR0 WR0	**BURST CONTROL:** Selects the burst length for DMA operation. Each burst is one word or one byte, depending on System Byte/System Word mode selected, DLCR6<5>.

BURST 1	BURST 0	Burst Length (Transfers)
0	0	1
0	1	4
1	0	8
1	1	12

Filter Self Receive Register (BMPR14)

Table 30. BMPR14 — Filter Self Receive Register

BIT	SYMBOL	TYPE	DESCRIPTION
7	INT EN	RC0	**INTERRUPT ENABLE:** When high, enables RLD, DLCR15<7>, to generate an interrupt.
6	INT EN	RC0	**INTERRUPT ENABLE:** When high, enables LLD, DLCR15<6>, to generate an interrupt.
5	INT EN	RC0	**INTERRUPT ENABLE:** When high, enables RJAB, DLCR15<5>, to generate an interrupt.
4	0	0	**RESERVED:** Write 0
3	0	0	**RESERVED:** Write 0
2	SKIP PKT	WR0	**SKIP RECEIVE PACKET:** Flushes one receive packet. EtherCoupler adjusts the system read pointer to the beginning of the next received packer. The new pointer points to an empty buffer if there are no incoming packets, and remains uncharged if it points to an empty packet.
1	INT EN	RC0	**INTERRUPT ENABLE:** When high, enables SQE, DLCR15<1>, to generate an interrupt.
0	FILTER SELF RX	WR1	**FILTER SELF RECEIVE:** When set to 1, disables the Accept All Packets mode Self Receive function. When set to 0, enable the self receive function in Accept All Packets mode.

Table 30 describes Filter Self Receive register, BMPR14. [Also refer to Table 20 and the detailed description of Receive Mode register, DLCR5, for more information about the Filter Self Receive function.]

Writing 04H to this register commands the Buffer Controller to skip the balance of the current receive packet in memory. The bit can then be read to determine completion of the skip process is complete (within 300 ns). If there is another packet, the bit returns to 0 when the chip is ready to read the next packet or, if there is not another packet, to stop reading. Do not use this feature before reading at least four times from the beginning of the packet, nor if there are eight or fewer bytes left of the packet in the buffer; doing so may corrupt the receive buffer pointers.

Regardless of whether EtherCoupler is in byte or word mode, the system has to read four times from the receive buffer so that the SKIP PKT function can operate properly. The SKIP PKT function cannot be used when the receive packet contains only two bytes when the system in byte mode, or two words when the system in word mode. Those bytes or words move into system FIFO, and the System Read Pointer points at the next packet location.

As shown in Table 30, this register provides control for enabling interrupts based on the setting of status bits in BMPR15, the Transceiver Status Register.

Filter Self Receive Register (BMPR14)

Table 30. BMPR14 — Filter Self Receive Register

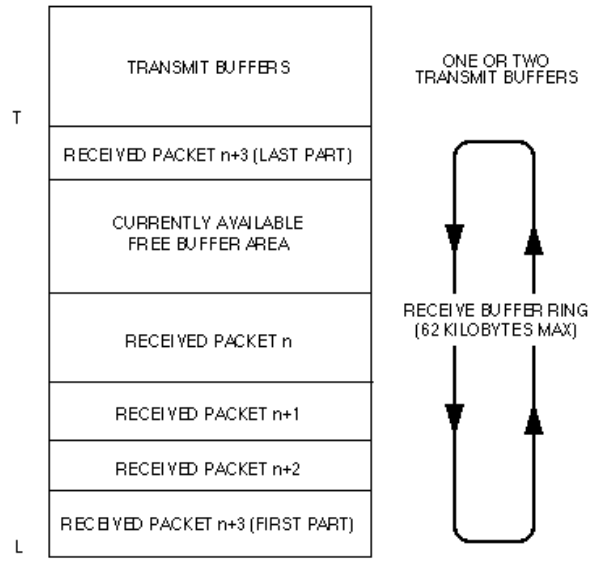
BIT	SYMBOL	TYPE	DESCRIPTION
7	RLD	WR0/1	**REMOTE LINKDOWN:** When set to 1, indicates that port is in linkdown condition. Can generate interrupts if enabled by Interrupt Enable bit BMPR14<7>.
6	LLD	WR0/1	**LOCAL LINKDOWN:** When set to 1, indicates that port is in linkdown condition. Can generate interrupts if enabled by Interrupt Enable bit BMPR14<6>.
5	RJAB	WR0/1	**REMOTE JABBER:** When set to 1, indicates that remote port is in Jabber condition. Can generate interrupts if enabled by Interrupt Enable bit BMPR14<5>.
4	RMT PORT	WR0/1	**REMOTE PORT:** When set to 1, indicates that remote port is compatible with and is using the same kind of TP chip.
3	RXI POL REV	WR0/1	**REMOTE PORT:** When set to 1, indicates reversed polarity.
2	0	0	**RESERVED:** Write 0
1	SQE	WR0	**SIGNAL QUALITY ERROR:** When set to 1, indicates detection of SQE. Can generate interrupts if enabled by Interrupt Enable bit BMPR14<1>.
0	0	0	**RESERVED:** Write 0

Table 31 describes Transceiver Status register, BMPR15. Note that when RMT PORT is set to 0, RLD, DLCR15<7>, and RJAB, DLCR17<5>, are meaningless.

POWERDOWN MODE

When EtherCoupler is not in use, the powerdown feature reduces power consumption by shutting off all internal clocks. This feature is invoked by first setting DLC EN bit, DLCR6<7>, high and disabling the Receiver and Transmitter, and then by setting PWRDN bit, DLCR7<5>, low. The powerdown mode terminates by setting the PWRDN bit high (writing a one to DLCR7<5>) or by implementing a hardware reset.

BUFFER CONTROLLER



1. Length (L) = 8, 16, 32, or 64 kilobytes.
2. Transmit Size (T) = 2 (one bank), 4 (two banks), 8 (two banks), or 16 (two banks) kilobytes.
3. Receive packets are aligned on eight-byte boundary.
4. Each received packet is preceded by a four-byte packet header as follows:

Byte 0	Packet Status
Byte 1	Not Used
Byte 2	Packet Size LSBs
Byte 3	Packet Size MSBs

Figure 9. Buffer Memory Organization

As shown in Figure 9, EtherCoupler uses dedicated buffer memory to store data packets to be transmitted to and received from the network. By direct connection to the EtherCoupler controller rather than a separate, local microprocessor bus, buffer memory eliminates the need for a local microprocessor. The Buffer Controller keeps track of buffer memory partitioning, as well as automatically allocating and updating receive and transmit pointers, eliminating these tasks from software overhead. Benchmark performance tests typically can surpass those of competing controllers because EtherCoupler provides a high level of automation and high-performance packet-buffering.

EtherCoupler on-chip Buffer Controller manages access to buffer memory by updating internal address pointers, which handle transmit, retransmit, receive, rejection of packets with errors, and data transfers to and from the host. The easy-to-operate EtherCoupler access-management feature relieves the host of buffer-management functions and substantially reduces software requirements. EtherCoupler automatically rejects packets with alignment, CRC or short-length errors. Packet rejection occurs unless the host sets the Acpt Bad Pkts bit, which passes packets received with errors to the host processor and sets appropriate error status bits to inform the host of the error. EtherCoupler allows reception of packets as small as six bytes by setting the Acpt short Pkts bit. Normal operation requires IEEE minimum packet length, excluding Preamble and CRC, of 60 bytes.

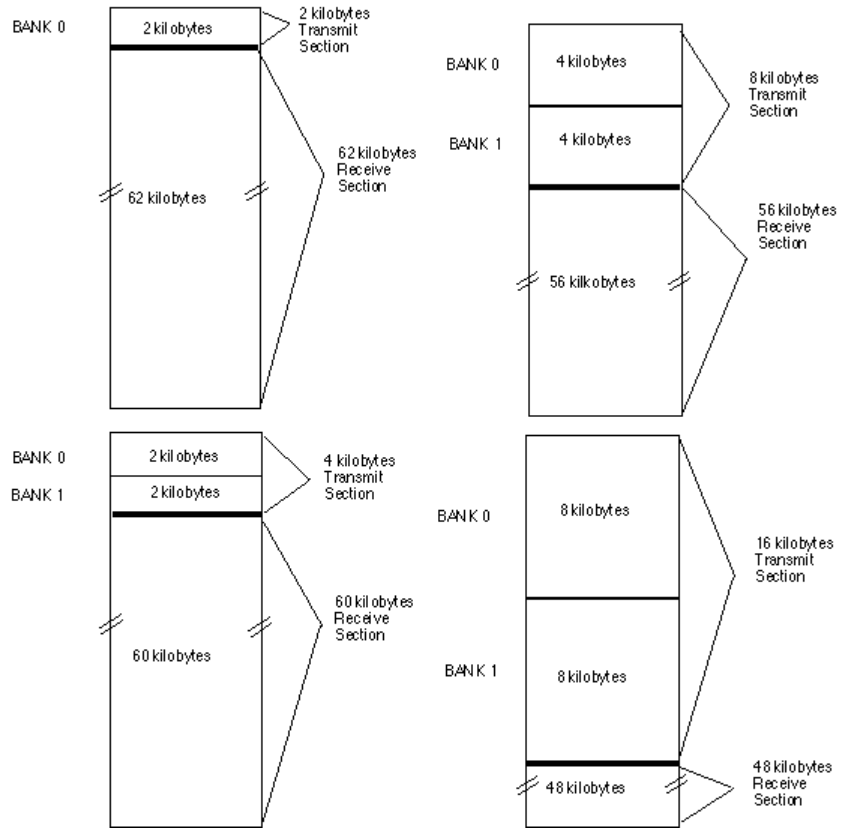


Figure 10. Transmit Buffer

As shown in Figure 10, system software can be used to program and partition buffer memory into transmit and receive buffer areas for optimum operation and for system configurations that accommodate a wide range of application demands. The transmit and receive buffer areas can allocate varying proportions of space to the transmit and receive functions.

Total EtherCoupler buffer memory size is configurable as 8, 16, 32 or 64 kbytes, including transmit and receive spaces. The buffer memory Transmitter section is configurable as a single, 2-kilobyte buffer or as a pair of 2, 4, or 8-kilobyte banks. Therefore, total size of the Transmit buffer space can be either 2, 4, 8 or 16 kbytes. Within each buffer or bank, the system writes one or more packets until available space is too small for another packet. Once begun, the Transmitter automatically transmits buffer packets, then finishes with a status update and interrupt. One bank of a two-bank configuration transmits while the other loads. The use of dual buffers, multiple-packet loading, and chaining produces the highest transmission rate, which boosts performance to accommodate systems that need high-throughput transmission.

EtherCoupler automatically configures memory as a ring buffer and allocates unused Transmitter memory to the Receiver. Like the Transmit buffer, the Receive buffer stores packets head-to-toe, but does so until reaching the end of the linear addressing space, where the EtherCoupler Receive-Write pointer then returns to the top of the receive addressing range, forming an apparently seamless ring with packets aligned on an 8-byte boundary. As packets read out to the system, the

Receive-Read pointer also forms a seamless ring. EtherCoupler automatically provides all necessary buffer-pointer management functions, to relieve host system drivers of this time-consuming task. Accomplished faster in hardware than software, this not only off-loads the host system but also speeds the communication process and increases throughput.

The Buffer Controller automatically prioritizes and services requests for access to memory from Transmitter, Receiver and host system. The Buffer Controller provides complete packet-management functions by updating buffer-memory pointers, allocating memory space for incoming data packets, and controlling pertinent bits within status registers.

The EtherCoupler arbitration mechanism provides packet management, by interleaving packet-data accesses to the buffer memory, so that operation appears to be simultaneous. The host writes data to or reads data from buffer memory via Buffer Memory Port register BMPR8 while the Receiver reads out data packets for transmission or writes in data packets for storage. EtherCoupler appears to independently serve the host system and network access interfaces, whose associated first-in, first-out (FIFO) circuits provide time for buffer interleaving.

As shown in Figure 11, packet data pipelines through the system to increase performance and throughput, with the Buffer Controller supporting all cases of simultaneous access to buffer memory, such as: (1) network data storage in the Receive buffer; (2) host retrieval of packets from the Receive buffer; (3) host loading of packet data into the Transmit buffer; (4) transmitter data acquisition for transmission from the Transmit buffer; and (5) any combination, including all of these, simultaneously.

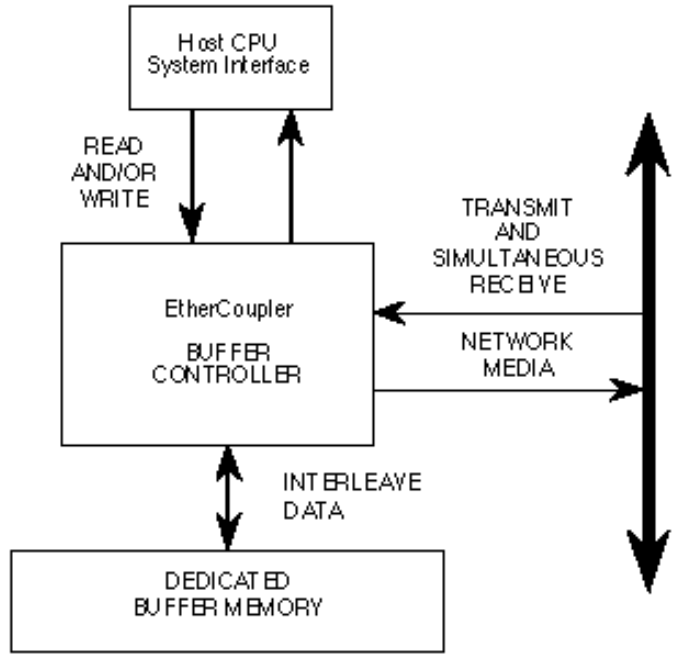


Figure 11. Simultaneous Access to Buffer Memory

System Access To Buffer

EtherCoupler supports programmed-I/O, Single-cycle and Burst mode DMA transfers between buffer memory and host system. The host accesses buffer memory by reading from or writing to EtherCoupler Buffer Memory Port register BMPR8. Data read or written by the system passes through on-chip FIFOs to eliminate effects of real-time interaction among the system, Transmitter and Receiver, as each accesses buffer memory. EtherCoupler also automatically controls read and write operations to external static random-access memory (SRAM).

Transmitter Access To Buffer

Programming Buffer Size bits BS1 and BS0, DLCR6<1:0>, changes the size of each transmit bank. Software allocates transmit buffer size as a single 2-kilobyte Transmit buffer, or as pairs of 2, 4 or 8-kilobyte Transmit buffers. A single packet or multiple packets simultaneously load into the buffer for transmission. The system and Transmitter time-share the buffer with a single Transmit buffer. The system utilizes two Transmit buffer sections to load packets into one of the buffers, while contents of the other buffer section are transmitted.

At reset, pointers initialize to set a beginning for one of the Transmit buffers. Each time the host writes data to the buffer via the Buffer Memory Port register, an internal pointer advances to the next memory location within the Transmit buffer. Once a data byte/word is written, it cannot be read and the internal pointer cannot be reversed.

EtherCoupler manages internal pointers that control access by the host to the two banks and selection of specific bank bytes and words. EtherCoupler switches banks as soon as TX START bit, BMPR10<7>, is set high by the host system, after which EtherCoupler begins transmitting at the earliest opportunity. The Transmit-Read pointer, which is also automatically managed, sequences through the transmitted bank to read packet data into the Transmitter through its FIFO. If a collision occurs, the packet automatically retransmit after a pseudo-random (backoff interval) waiting period. EtherCoupler continues down the list, automatically transmitting all multiple packets in the bank. EtherCoupler can transmit multiple, back-to-back packets of legal Ethernet sizes to the LAN network. Excluding preamble and CRC fields, these packets can be between 60 and 1514 bytes.

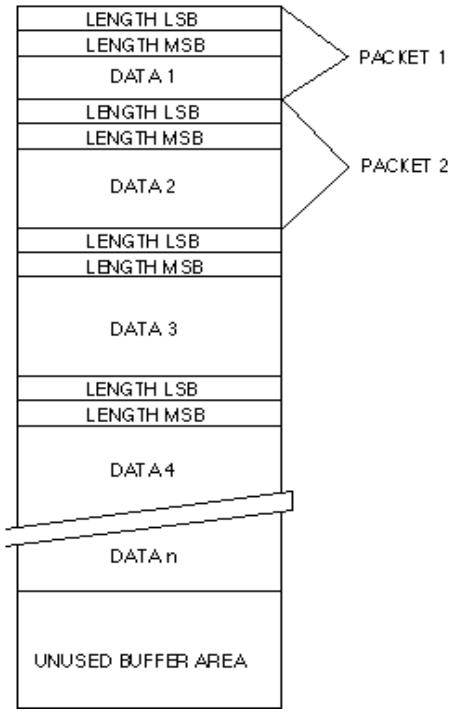


Figure 12. Transmit Buffer Detail

As shown in Figure 12, multiple packets can reside within one transmit bank, separated by a nontransmitted, two-byte header that provides packet byte-length information. At the end of a packet list or chain, the Transmitter stops, updates its status bits and, if enabled, generates an interrupt.

Transmit Packet Data Formats

Packets to be transmitted (minus preamble and CRC fields) first load into the Transmit buffer with a two-byte header indicating packet byte-length. If buffer space permits, multiple packets load simultaneously. After packet-loading, the host initiates transmission by writing the number of packets just loaded into TX PKT CNT bits, BMPR10<6:0>. If the two-bank buffer configuration is selected, the second bank now loads additional packets.

Receiver Access To Buffer

Once initialized and enabled, the Receiver automatically loads the Receive buffer (via address filter and EtherCoupler FIFO) with incoming, error-free packets, while inserting a four-byte packet-status and packet-length header. An interrupt alerts the host processor to the availability of packets in the buffer. As they become available, the host processor reads out receive packets. If an interrupt occurs to indicate when overflow occurs, buffer data writes to free space to resume reception. As soon as space is available in the Receive buffer, the Receiver automatically continues reception.

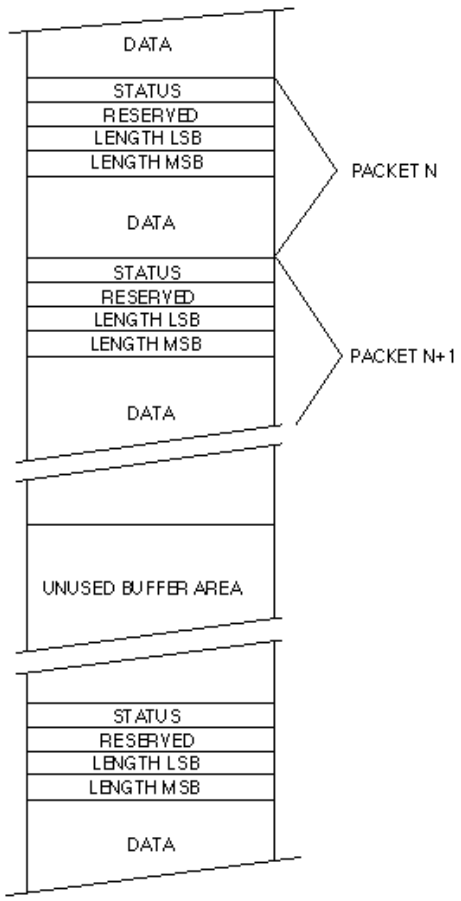


Figure 13. Receiver Buffer Detail

Receive buffer size varies between 62 kilobyte (with 2 kilobytes allocated for the transmit section with maximum memory size of 64 kilobyte) and 4 kbytes, if 4 kbytes are allocated for the Transmit section with minimum memory size of 8 kbytes. The Receive section dynamically allocates space along an eight-byte page boundary for each individual incoming data packet. A four-byte header that shows data packet status and length precedes each received packet. Internal pointers sense data packet length from the header and set a starting address for the next packet to link or chain with the previous packet. Figure 13 shows details of the Receive buffer. In this architecture, EtherCoupler controls a dedicated buffer memory, and FIFO size and depth are not relevant to system timing.

The RX BUF Empty status bit, DLCR5<6>, alerts the host that the Receive buffer holds one or more packets to be read. The host retrieves these packets from buffer memory by successively reading Buffer Memory Port register BMPR8. After a data byte/word is read from buffer memory, internal pointers advance to the next byte/word. As the system sequentially reads data, memory becomes available to receive new packets. EtherCoupler automatically rejects incoming packets if there is insufficient buffer space to fully receive that packet. Therefore, already received packets cannot be overrun by incoming packets.

When ACPT BAD PKTS bit, DLCR5<5>, is disabled (set to 0), a bad incoming packet causes EtherCoupler to release the buffer space containing that packet, and to reset internal pointers so the next incoming packet can use that space. When DLCR5<5> is set to a 1, the packet with a short-length, alignment or CRC error is accepted and appropriate error bits in the header status field are set. Similarly, when ACPT SHORT PKTS bit, DLCR5<3>, is set to a 1, packets shorter than IEEE 802.3 minimum size (less than 60 bytes excluding preamble and CRC) are accepted.

Receive Packet Data Formats

The buffer stores receive packets, less preamble and CRC fields, with a four-byte header. The initial header byte gives status information, such as errors that occurred during packet reception. Normally EtherCoupler automatically eliminates from memory and discards packets with errors. However, EtherCoupler offers mode selection to permit bad-packet reception, and indicates errors in the header status byte. While the second header byte is unused at this time, the last two header bytes provide packet byte count (less preamble and CRC).

TRANSMITTER CIRCUITS

Circuits within the Transmitter include a transmitter state machine, a small FIFO for pipe-lining the packet data, preamble generator, CRC generator, end-of-packet symbol generator, parallel-to-serial converter, Manchester encoder, waveform generator, transmit filter, twisted-pair line driver, backoff generator, inter-packet gap-timer, and time domain reflectometer (TDR) counter.

The transmitter state machine, which sequences Transmitter events (idle, preamble, data, CRC, inter-packet gap, jam and backoff), detects various transmit error conditions and sets appropriate bits within the DLCR registers.

The pipe-line FIFO provides elastic buffering that the Buffer Controller loads with data to be transmitted. The EtherCoupler CRC generator calculates the Ethernet 32-bit CRC on the destination and source address, the length field, and the (ISO/ANSI/IEEE 8802-3 Ethernet specification) data field that appends to the packet.

TRANSMIT ERROR PROCESSING

Transmit error status bits in Transmit Status register DLCR0 report possible transmit errors, the last two of which can be enabled separately to generate interrupts: 1) CR Lost, DLCR0<4>, loss of carrier during transmission, usually a medium fault or collision; 2) Col, DLCR0<2>, collision; and 3) 16 Col, DLCR0<1>, 16 consecutive collisions.

If it detects a collision during transmission, EtherCoupler automatically tries to retransmit the packet until sixteen attempts have been made. collision counter DLCR4<7:4> automatically increments after each Collision up to the sixteenth, at which time it rolls over to zero. (Bit 7 is the most-significant of the four bits.) Appropriate status bits in the Transmit Status register are set when a collision terminates transmission. The occurrence of 16 collisions may indicate a network problem, such as a disconnected cable or terminator, that produces false collisions. While rare, 16 collisions may normally occur.

A pseudo-random number generator, which provides collision backoff clocked at the 10-megahertz bit rate so that distances between stations become part of the randomizing function, is sampled at the time of collision, so as to mask all but the appropriate number of bits specified by the 8802-3 backoff algorithm. This value is then counted down at the slot-time rate (512 bits per second) to generate the backoff interval. Only one bit is used for a first collision, giving a backoff of either 0 or 51.2 microseconds. A second consecutive collision uses two bits, and so forth, up to ten bits. The tenth through 16th collisions use 10 bits to provide a pseudo-random backoff interval from 0 to 52.38 milliseconds that, per 8802-3, is the so-called binary exponential backoff for collisions.

TIME DOMAIN REFLECTOMETRY

The TDR function counts the actual number of bits transmitted for each packet before a collision indication, carrier loss indication or completion of transmission. A complete transmission with no error indications clears the TDR counter. [Refer to register descriptions for Time Domain Reflectometry registers, DLCR14 and DLCR15.]

MEDIA ACCESS CONTROL

The EtherCoupler transmitter state machine implements the 8802-3 networks media-access protocol, CSMA/CD (Carrier Sense, Multiple Access with Collision Detection). carrier sense means that EtherCoupler monitors the network for any other node carrier, and defers transmission (collision avoidance) while other nodes are transmitting. collision detection handles collisions that may still occur when two nodes separated on the network by several microseconds begin transmitting at nearly the same time. All nodes monitor the network for collisions and, when involved in one, transmit a 32-bit Jam to reinforce the collision and then terminate transmission. After waiting a pseudo-random backoff interval, the node automatically retries transmission of the packet.

Packets must be separated by at least 9.6 microseconds, a timing gap during which trunk cabling is idle. The EtherCoupler Transmitter state machine that measures this interval, starting from the end of a packet on the network, does not transmit until the end of this interval. If carrier reappears on the network during the first two-thirds of the interval, EtherCoupler resets the interval timer to re-time the interval from the end of the new transmission.

Superimposition of two packets corrupts data and carrier indications, such as during a collision. During the last one-third of the interval, EtherCoupler ignores the occurrence of a carrier indication, in accordance with 8802-3, and ensures fairness and equality in access to the network. If one station begins transmission slightly ahead of another, there is no advantage to the earlier start. Both nodes transmit, a collision occurs, and backoff interval differentials resolve the media-access contention.

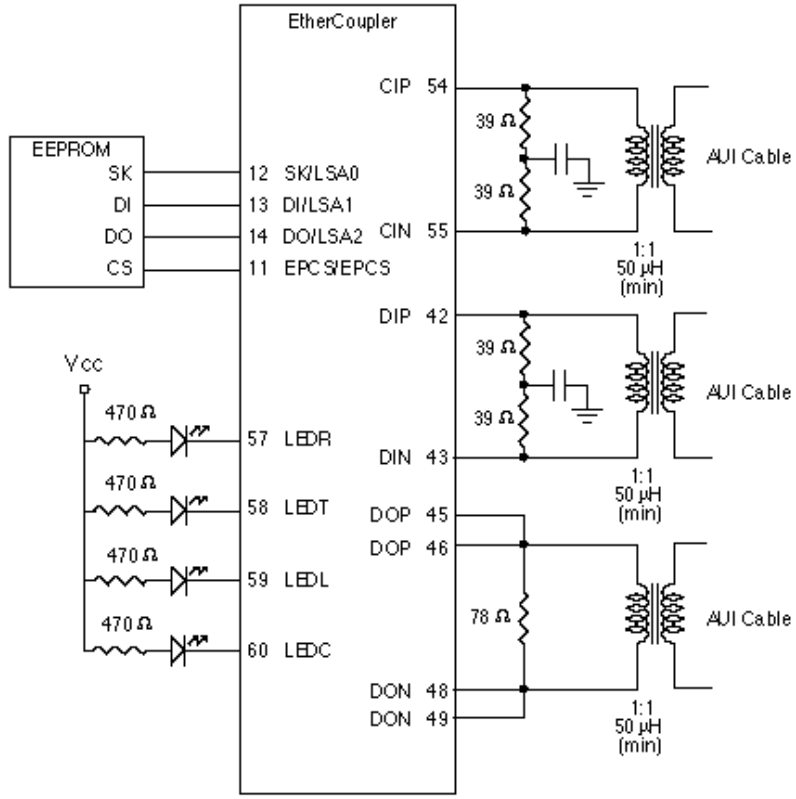


Figure 14. Typical Connections

DATA ENCODER

EtherCoupler serializes the data for transmission, and converts each bit to Manchester Code, the format used on the network media. Manchester Code for a one is a 100-nanosecond interval starting with a low, ending with a high, with a low-to-high transition at the 50-nanosecond point; Manchester Code for a zero is the inverse.

TRANSMIT PACKET PROCESSING

When transmitting one or more packets, the host system first loads the packets into a transmit buffer. The packets are preceded by a two-byte header giving their lengths, and are written to Buffer Memory Port registers BMPR8 and BMPR9. The system loads only packet destination address, source address, length field and data field; EtherCoupler generates the rest. When the packets load, the system turns on the Transmitter to initiate transmission, enabling EtherCoupler to transmit. Observing the media access protocol, EtherCoupler delays transmitting until the absence of carrier from other nodes or intervals for minimum interpacket gap and backoff, and then begins to transmit. EtherCoupler, which serializes and encodes data as Manchester code, generates the Preamble field at the beginning and calculates and appends the CRC field at the end, followed by the non-Manchester End-Of-Packet delimiter.

Figure 14, which shows typical connections to an EEPROM and the LEDs, also indicates typical interfacing to the AUI cable. The Manchester-encoded signals are output to transmitter data pins DOP and DON through a differential driver, which can drive a 50-meter segment of 78Ω transceiver cable as specified in the 8802-3 standard.

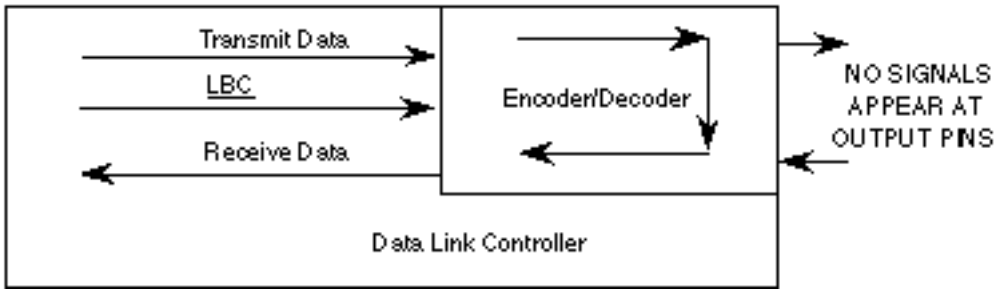


Figure 15. Loopback Operation

The host system activates the Transmitter by writing the packet count to TX PKT CNT register, BMPR10<6:0> and by writing one to the TX START bit, after which the Transmitter transmits each packet in the buffer in the order in which they were loaded. If a collision occurs, the Transmitter automatically retransmits the packet until successful or until 16 consecutive attempts have ended in collision. In the latter case, depending on the mode selection made at initialization time, the Transmitter continues to try to transmit the same packet starting again with a collision count of zero, or skips the current packet and tries to transmit the next packet starting with a collision count of zero, or halts and waits for instruction from the host. In the last case, the host terminates by setting DLC EN, DLCR6<7> to one, or continues to attempt to transmit the same packet (collision counter reset), or skips the current packet and tries to transmit the next packet (collision count is zero).

COLLISION SIGNAL PROCESSING

As collisions are detected on the network media, a 10-megahertz signal is generated on collision differential inputs, CIP and CIN. When EtherCoupler detects this signal, it asserts the internal collision line. The CIP and CIN differential driver inputs require termination similar to that for the DIP and DIN inputs, i.e., 39Ω resistors tied to ground through 0.1-micro farad capacitors.

LOOPBACK

As shown in Figure 15, loopback allows EtherCoupler testing without sending signals onto the LAN media. The loopback function is invoked by clearing LBC bit DLCR4<1>. Data is routed from the Transmit Buffer through a FIFO to the Transmit section of the data link controller, through the internal Manchester encoder, back through the Manchester decoder, through the Receiver section of the data link controller, and is then stored in a Receive Buffer. Software can then read and check the received packet that has traveled through the EtherCoupler Transmit and Receive sections. The Manchester encoder/decoder responds to assertion of the LBC input by internally looping the transmitter output to the receiver input, and blocks the transmit data from appearing at the network output pin.

RECEIVER CIRCUITS

The Receiver includes twisted-pair line receiver, receive squelch, receive filtering, a phase-locked loop (PLL) for clock and data separation (Manchester decoding), a receive state machine, serial-to-parallel conversion, pipe-line FIFO, preamble recognition, bit and byte-framing, address filtering, CRC and other error checking and end-of-packet symbol recognition.

The receiver state machine, which provides sequencing of events for the receiver, including idle, busy, address filtering, and data storage, detects receive error conditions and sets appropriate bits within the data link control registers. A small data FIFO provides elastic buffering for synchronization with Buffer Controller timing, and buffering data while the Buffer Controller is servicing other buffer memory access requests.

All received bytes are delayed by four bytes before storing in the Receive Buffer, so the last four bytes of the packet can be stripped and checked for correct CRC. (The CRC bytes are not transferred to the Receive Buffer.)

During reception, the controller automatically rejects packets if Receive Buffer space is insufficient to hold the entire received packet. Status bits in the receive status register are set to indicate this and other errors. Receive errors are: 1) bus read error, which occurs if the host system attempts to read from an empty receive buffer (this need never occur if the RX BUF EMPTY bit is checked), 2) short packet error, 3) alignment error (incomplete byte fragment at end of packet), 4) CRC error and 5) Buffer Overflow. The software checks Length error, an additional possible receive error. When the length of the packet does not match the value in the Length Field of an 8802-3 packet, a length error occurs. Some protocols use the length field for other purposes, for example, the DIX protocol that uses it for a packet type code. In this case, allowed type codes do not overlap allowed packet length values, providing a means to distinguish which protocol is being used (if length value >1500, it is DIX type code). Length check can be made conditional on protocol type, if necessary to support other protocols such as DIX.

DECODER FUNCTIONS

The data decoder section performs three functions on the data received at the differential receive inputs from the transceiver: clock recovery, carrier detection, and Manchester data decoding. Clock recovery and data separation are accomplished by a phase-locked loop. Use of proprietary techniques in the PLL allows lock-on within 6 - 7 bit times of the beginning of the Preamble, and permits stable operation with input signal jitter of up to ±18 ns. Carrier detection is indicated to the controller by assertion of the internal CRS signal, which occurs shortly after the received data signals appear.

The recovered clock is supplied to the controller, and is also used to convert Manchester-encoded data to NRZ format. Transitions in the state of the NRZ data are synchronous with the falling edge of the receiver clock pulse. During idle periods, the receiver clock pulse is disabled. The DIP and DIN differential inputs are usually terminated with two 39Ω resistors in series and a 0.1-micro farad bypass capacitor to ground at their junction.

MONITORING THE NETWORK

Whenever the data link section is enabled (DLC EN bit, DLCR6<7>, is set to zero), the Receiver constantly monitors the network for carrier. Signals that exceed the AC and DC squelch thresholds of the received data input section cause the internal carrier sense line to assert, which in turn causes the Receiver to attempt to receive a packet. The Transmitter also uses the carrier sense function to defer to transmissions from other nodes, except when EN TX DEFER bit, DLCR4<0>, is high.

After the PLL decoder acquires bit-synchronization with the incoming signal, the Receiver monitors the data stream for the end-of-preamble bit pattern, a four-bit pattern of 1011 ending the Preamble's pattern of alternating ones and zeros. This pattern gives the Receiver byte and field synchronization, because the bit immediately following the four-bit pattern of 1011 ending the Preamble is the first bit of the first byte of the packet's destination address field.

When packet transmission is unflawed, carrier sense remains asserted for the duration of the packet, negating just after the last bit is received. As a packet comes in, the decoder carrier sense function monitors the data stream for the end-of-packet symbol, a special non-Manchester code element at the end of the packet. When detecting this symbol, the carrier sense line is negated. Loss of carrier may also result from negation of the carrier sense line during a collision.

RECEIVE PACKET PROCESSING

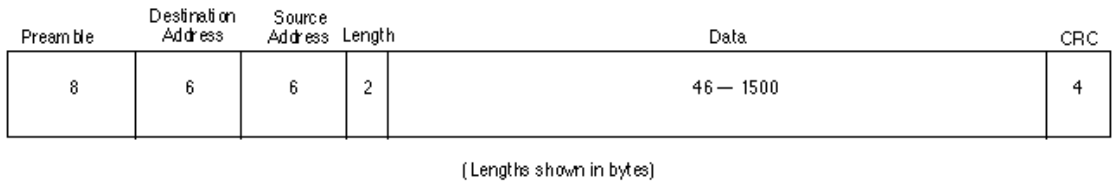


Figure 16. 8802-3 Packet Format

Figure 16 illustrates the 8802-3 packet format. When a received packet enters from the network, its destination address field is tested for address filter criteria selected by the Hash Table and Address Filter Mode bits AF1 and AF0, DLCR5<1:0>. If the address meets filter criteria, the Receive Buffer accepts the packet for storage. The packet must also be error-free unless the reception of flawed packets is allowed, e.g., for diagnostic purposes. If conditions are met, packet reception results in the Buffer storing the packet, updating the four-byte header at the end of reception, clearing the RX BUF EMPTY bit, setting the RX PKT bit high, and (if enabled) generating an interrupt. Otherwise the packet is discarded, and pointers reset to reuse the same portion of memory for the next arriving packet. A flawed packet accepted for storage for diagnostic purposes is reported as an error in the PKT STATUS byte of its header. The Receive Packet Header bit positions for PKT STATUS are shown in Table 24, Receive Packet Header Conditions.

NETWORK MANAGEMENT

Error, traffic and performance statistics may be collected continuously or on a sampled basis. The Receive Status register and Transmit Status register indicate detected errors. Such data can be collected in two ways: interrupts can be used after each packet, and the interrupt service routine reads the status from the status register; or packets are accepted for storage in the Receive Buffer in Accept Bad Packets mode, and content and error status stored in the header are later read in batch mode. Frequent selection of the Accept all packets mode to get maximum statistics for the network and to count all packets including their length, because it maximizes host overhead with user terminal equipment, is discouraged.

Sampling of carrier detection estimates network bandwidth utilization, via Transmit Status register NET BSY bit, DLCR0<6>. Average media-access waiting time is estimated by calculating the elapsed time between starting the Transmitter and TX DONE bit, DLCR0<7>, going high, and subtracting calculated transmit time. Collision counter bits COL CTR3 through COL CTR0, DLCR4<7:4>, determine the number of collisions encountered by the last outgoing packet. The counter resets at the start of transmission of each packet.

RECEIVE ERROR PROCESSING

Interrupts may optionally be generated by these receive error conditions: buffer overflow, CRC error, alignment error, and short packet. None of these errors requires special host processing or intervention, other than optional tallying of the error for network management purposes. EtherCoupler automatically discards any packet with errors. If a buffer overflow occurs, EtherCoupler automatically handles that as well. Packets already stored in the buffer are not lost. The host simply reads out some of the packet data, freeing space in the buffer for more packets. EtherCoupler then automatically resumes reception of packets, as long as there is buffer space. To prevent packet loss although EtherCoupler automatically recovers, avoid overflow by rapidly processing incoming packets.

TRANSCEIVER

The EtherCoupler Transceiver, which is fully compliant with IEEE 802.3 specifications for AUI (Attachment Unit Interface) and 10BASE-T (twisted-pair) interfaces, provides electrical interface to DB15 (AUI) and RJ45 (10BASE-T) connections to an Ethernet local area network. Its functions include Manchester encoding and decoding of serial data streams, level conversion, collision detection, signal quality error (SQE) and link integrity testing, jabber control, loopback, and automatic correction of polarity reversal on the twisted-pair input. Pulse-shaping and filtering functions eliminate the need for external filtering components and thus reduce overall system cost. Also provided are outputs for receive, transmit, collision and link test LEDs, and compatibility with shielded and unshielded twisted-pair cables. Receive threshold can be reduced to allow an extended range between nodes in low-noise environments.

TRANSMIT FUNCTION

The Transceiver Transmitter section receives from the controller section non-return zero (NRZ) data that passes through a Manchester encoder. Encoded data then transfers to either the AUI cable via the DO circuit or the twisted-pair network via the TPO circuit. Advanced integrated pulse-shaping and filtering produces on the TPON and TPOP pins an output signal that is predistorted and prefiltered to meet the 10BASE-T jitter template, which requires no external filters.

During idle periods, EtherCoupler transmits link integrity test pulses on the TPO circuit if Link Test Enable bit, LINK TEST EN, BMPR13<5>, is enabled and the AUI/TP port-select bit, BMPR13<4> is set low for twisted-pair (TP) mode. EtherCoupler is programmed for either shielded (150 Ω) or unshielded (100 Ω) twisted-pair through STP/UTP bit BMBR13<2>.

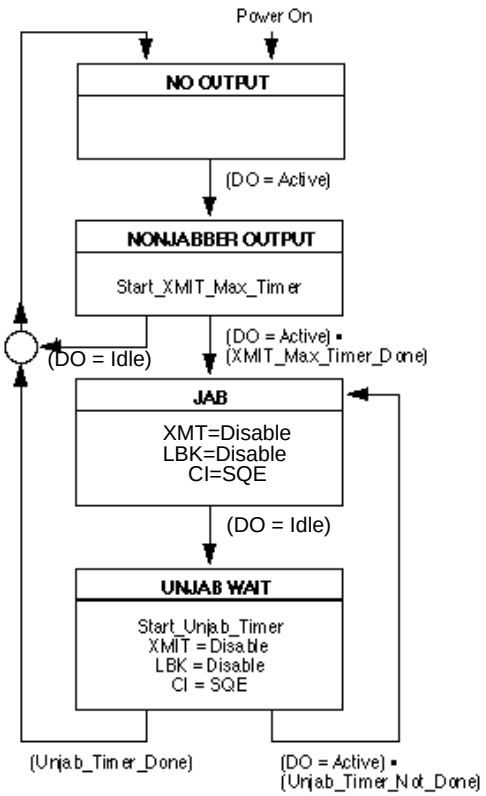


Figure 17. Jabber Control Function

Jabber Control Function

Figure 17 is a state diagram of the jabber control function. An on-chip watchdog timer prevents the data terminal equipment (DTE) from locking into a continuous transmit mode. When a transmission exceeds the time limit, the watchdog timer disables the transmit and loopback functions, and activates the JABBER signal. Before EtherCoupler can exit the jabber state, the transmit data circuit must remain idle for between 0.25 and 0.75 seconds.

SQE TEST FUNCTION

The Transceiver supports the signal quality error (SQE) test function as shown in Figure 18. After every successful transmission on the 10BASE-T network, EtherCoupler transmits the SQE signal to the DTE for 10± 5 bit times over the internal CI circuit. Bit 1 of Transceiver Status register BMPR15 reflects the status of this SQE signal.

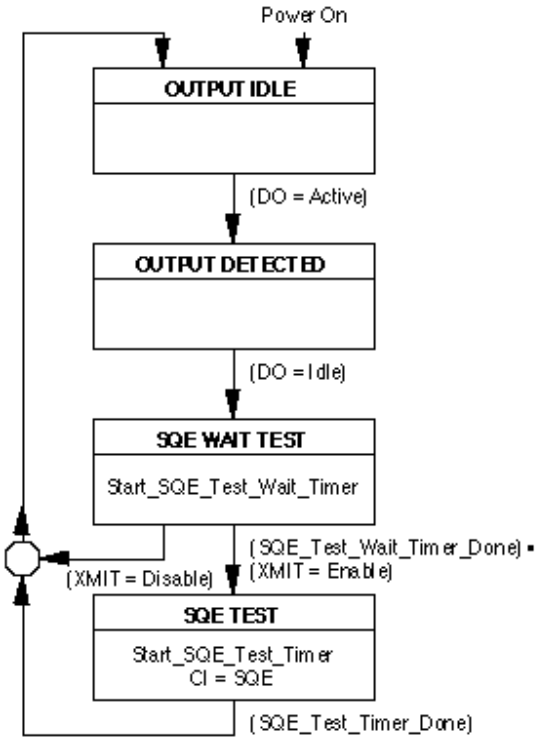


Figure 18. SQE Test Function

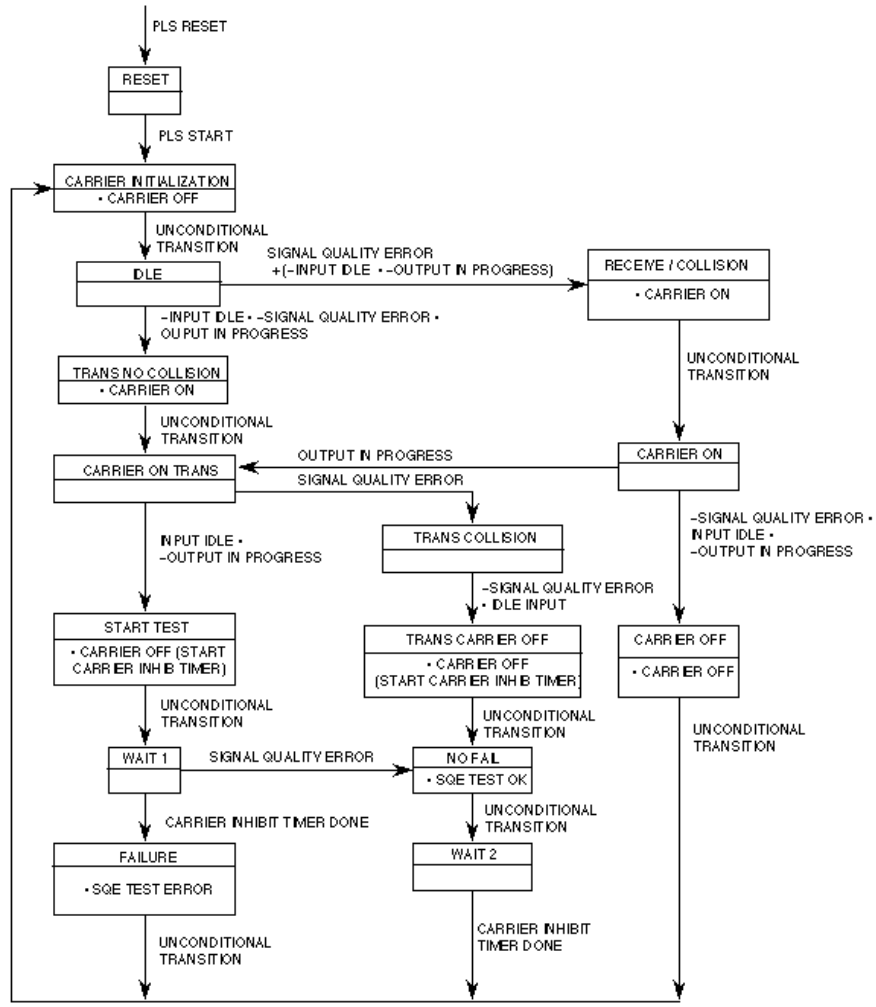


Figure 19. Carrier Sense Function

RECEIVE FUNCTION

The Transceiver receive function acquires timing and data from the twisted-pair network (the TPI circuit) or from the AUI (the DI circuit). Valid received signals pass through the on-chip filters and Manchester decoder, and output as decoded NRZ data and receive timing on the Receiver Data and Receiver Clock to the Receiver State Machine. No external filters are required.

An internal intelligent squelch function discriminates noise from link test pulses and valid data streams. The receive function is activated only by valid data streams above the squelch level and with proper timing. If the differential signal at the TPI or the DI circuit inputs falls below 75% of the threshold level (unsquelched) for eight bit times (typical), the receive function enters the idle state. If polarity of the TPI circuit reverses, the Transceiver detects the polarity reversal and reports it via RXI POL REV bit, BMPR15<3>. The Transceiver automatically corrects reversed polarity. Figure 19 is a state diagram of the carrier sense function.

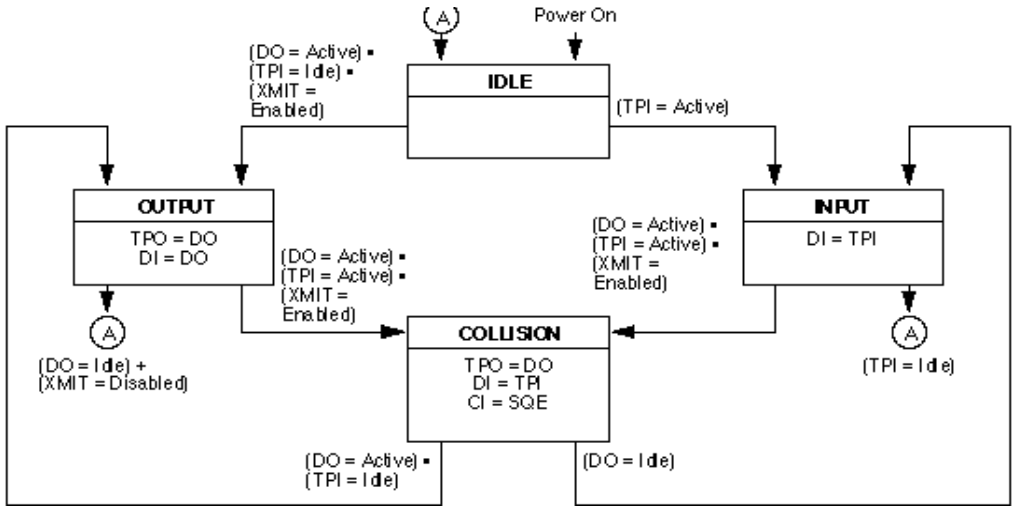


Figure 20. Collision Detection Function

POLARITY REVERSE FUNCTION

The Transceiver polarity reverse function uses link pulses and end-of-frame data to determine the polarity of received signals. A reversed polarity condition is detected when eight opposite receive link pulses are detected, without receiving of a link pulse of the expected polarity. Reversed polarity is also detected if four frames are received with a reversed start-of-idle. Whenever polarity is reversed, these two counters are reset to zero. If the Transceiver enters the link fail state and no valid data or link pulses are received within 96 to 128 milliseconds, polarity resets to the default uninverted condition. If Link Integrity Testing is disabled, polarity detection is based only on received data. Polarity correction is always enabled.

COLLISION DETECTION FUNCTION

The collision detection function operates on the twisted-pair side of the interface. A collision is defined as the simultaneous presence of valid signals on both the TPI circuit and the TPO circuit. The Transceiver reports collisions to the back-end via the COL pin. If the TPI circuit is active while there is activity on the TPO circuit, the TPI data passes to the back-end as received data, disabling normal loopback. Figure 20 is a state diagram of the collision detection function.

TWISTED-PAIR LOOPBACK FUNCTION

EtherCoupler provides the normal loopback function specified by the 10BASE-T standard for the twisted-pair port. The loopback function operates in conjunction with the transmit function. Data transmitted by the Transmit State Machine is internally looped back within EtherCoupler before the TPO drivers to the Manchester decoder and returned to the Receive State Machine. The normal loopback function is disabled when a data collision occurs, clearing the received data circuit in the Transceiver for the twisted-pair input data. The normal loopback is also disabled during the link fail and jabber states.

EtherCoupler provides additional loopback functions controlled by LBC bit, DLCR4<1>. When the twisted-pair port is selected and LBC is set high, the twisted-pair loopback is forced to override collisions on the twisted-pair circuit. Normal loopback is in effect when LBC is set low. When the AUI port is selected and LBC is set high, data transmitted by the back-end internally loops back from the Transmit Data pin through the Manchester encoder/decoder to the Receive Data pin. When LBC is set low, no AUI loopback occurs.

AUTOMATIC PULSE-STRETCHING FUNCTION

EtherCoupler incorporates open-drain drivers to provide signals to each LED that indicates packet reception, transmission, collision, and linking. The chip provides automatic stretching of pulses to allow perception by the human eye of the presence of packets — as they are being processed for each function — despite true rates which would be too fast to visibly indicate state changes.

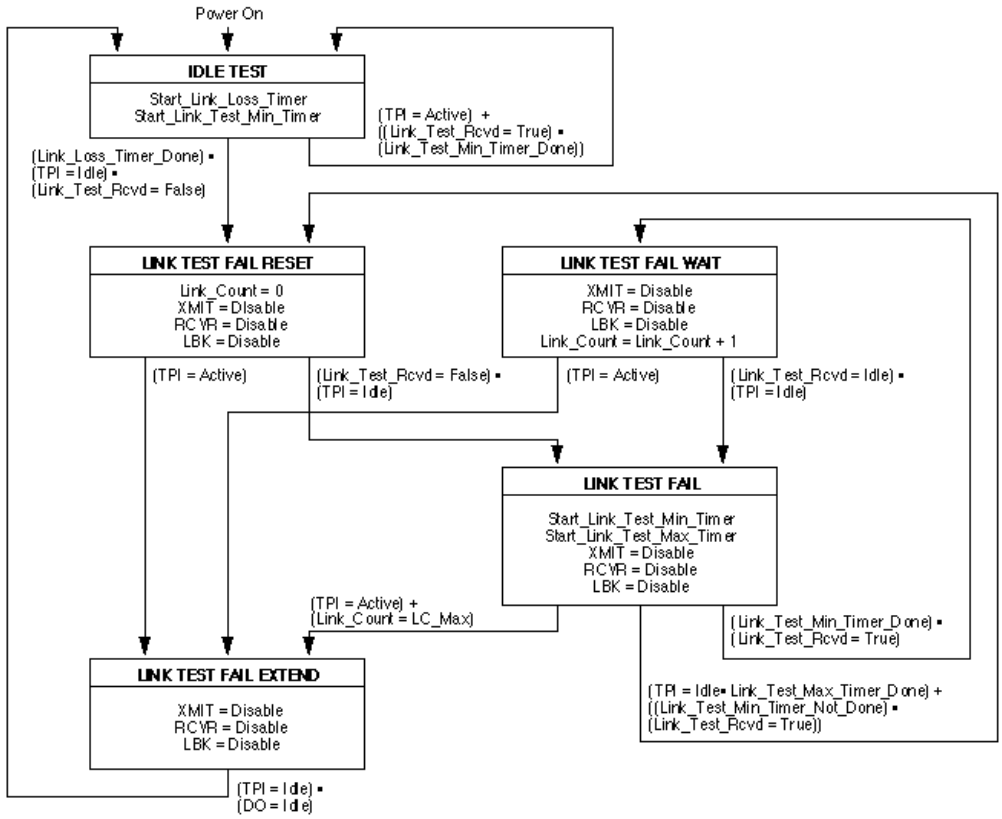


Figure 21. Link Integrity Function

LINK INTEGRITY TEST

Figure 21 is a state diagram of the link integrity test function. The link integrity test determines the status of the receive side twisted-pair cable. Link integrity testing is enabled when Link Test En bit BMPR13<5> is set low. When enabled, the Receiver recognizes the link integrity pulses transmitted in the absence of receive traffic. If no serial data stream or link integrity pulses are detected within 50 to 150 milliseconds, EtherCoupler enters a link-fail state and disables the transmit and normal loopback functions. EtherCoupler ignores any link integrity pulse with an interval less than 2 to 7 milliseconds. EtherCoupler remains in the link-fail state until it detects either a serial data packet or two or more link integrity pulses.

REMOTE SIGNALING

EtherCoupler transmits standard link pulses that meet the 10BASE-T specification. However, EtherCoupler encodes additional status information into the link pulse by varying the link-pulse timing; this is referred to as remote signaling. By using alternate pulse intervals, EtherCoupler signals three binary conditions: Remote Linkdown (RLD), Remote Jabber (RJAB), and Remote Signaling Capability (RFTN). EtherCoupler also recognizes these alternate pulse intervals when received from a remote unit. Remote status conditions are sensed by the controller as RLD bit, BMPR15<7>, RJAB bit, BMPR15<5> and RFTN bit, BMPR15<4>.

SOFTWARE SUPPORT FOR POPULAR NETWORK OPERATING SYSTEMS

LAN NODE DRIVERS

The so-called driver is the hardware-dependent portion of the software complement for a network node. It's purpose is to marry a specific hardware configuration to a more-or-less generic interface provided by the networking software. Network software suppliers provide such generic interfaces for drivers to encourage universal support from a variety of hardware products. By partitioning or layering the software into a stack of components with standardized interfaces between the layers, the job of integrating various hardware and software offerings with a particular network operating system becomes easier. More drivers will become available, and everyone benefits.

As seen in Figure 22, which depicts a model of LAN node components, the driver sits between the generic network software and the hardware, and acts as a bridge between the system and the node hardware. The interface of the driver to the network software and its applications is a generic interface and virtually the same for all drivers running on a given operating system. But the driver comprehends the configuration and nuances of the hardware, and optimizes it's performance in the system. A good, well-written driver is a positive advantage to the system, allowing it to achieve its performance potential; whereas a poor driver will limit the performance and reliability of the system.

The best drivers have a certain intimacy with the hardware, which allows them to take full advantage of its features. High data throughput, data integrity, and reliable operation are the key goals for which every node-driver writer should strive. In addition, the final driver should be efficient, by requiring minimum host execution time. These are all things the end user will and should take for granted; if they are not supplied with the design, there will be no customer satisfaction.

Most network software suppliers offer technical support for third-party driver development. This often comes in the form of a developer's kit that includes a manual and software examples. Some suppliers also offer test suites and certification testing to verify the driver product, because they know that good drivers benefit both users and suppliers.

WHAT'S IN A DRIVER?

Typical node drivers manage the movement of packet data between system memory and the network, and vice versa, as well as providing diagnostic testing, error processing, and error statistics on-demand for the system.

The first thing a driver does when the system is powered up is system check-out, which may include buffer memory testing, and loopback testing of the transmit and receive circuits. If the network supports it, as does Ethernet, for example, the testing may include sending and receiving test packets on the network to verify the ability to communicate.

As shown in Figure 23, the driver provides control of the initialization, interrupt and branch control processes, supporting both the transmit and the receive functions.

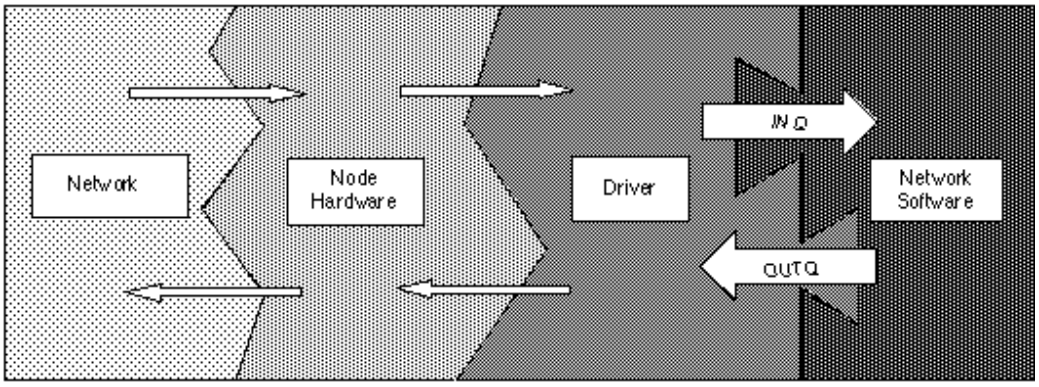


Figure 22. Model of LAN Node

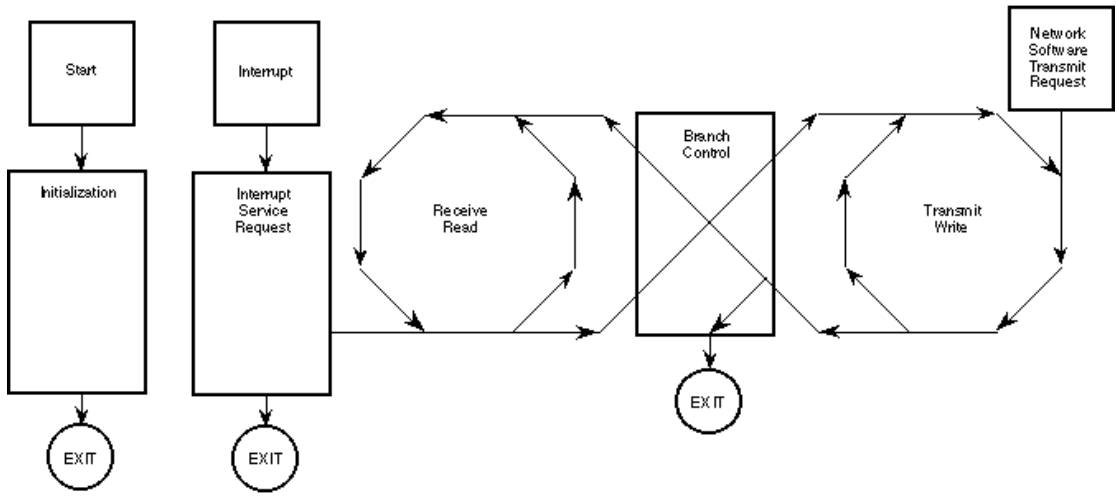


Figure 23. Software Driver Model

Figure 24 shows an example in flow chart form of a check-out routine written for the Fujitsu MB86965 Controller. The driver first initializes the control and status registers in the controller for memory and loopback testing. Loopback and memory testing are conducted simultaneously by transmitting memory test patterns from the transmit buffer area of memory to the receive buffer area using loopback. This test sequence simultaneously exercises and tests the transmitter, the receiver and the buffer memory.

The loopback/memory startup test is performed by first loading a test pattern in the form of several packets into the controller chip's transmit buffer, then transmitting the packets in loopback mode. The loopback transmission path through the chip exercises the data link controller as well as the encoder and decoder circuitry, but does not affect the network.

The EtherCoupler controller has a unique buffer memory architecture which pipelines packets through the system in both directions, optimizing data throughput. EtherCoupler's buffer controller provides all the pointer management for accessing the buffer automatically, greatly reducing the complexity of the driver and minimizing the software overhead. Receive packets with errors are automatically purged by EtherCoupler. When a collision occurs, EtherCoupler automatically re-transmits without host interaction. These features provide high data throughput while minimizing the host and memory overhead. At the successful conclusion of the tests, the driver starts up the EtherCoupler chip for regular service on the network.

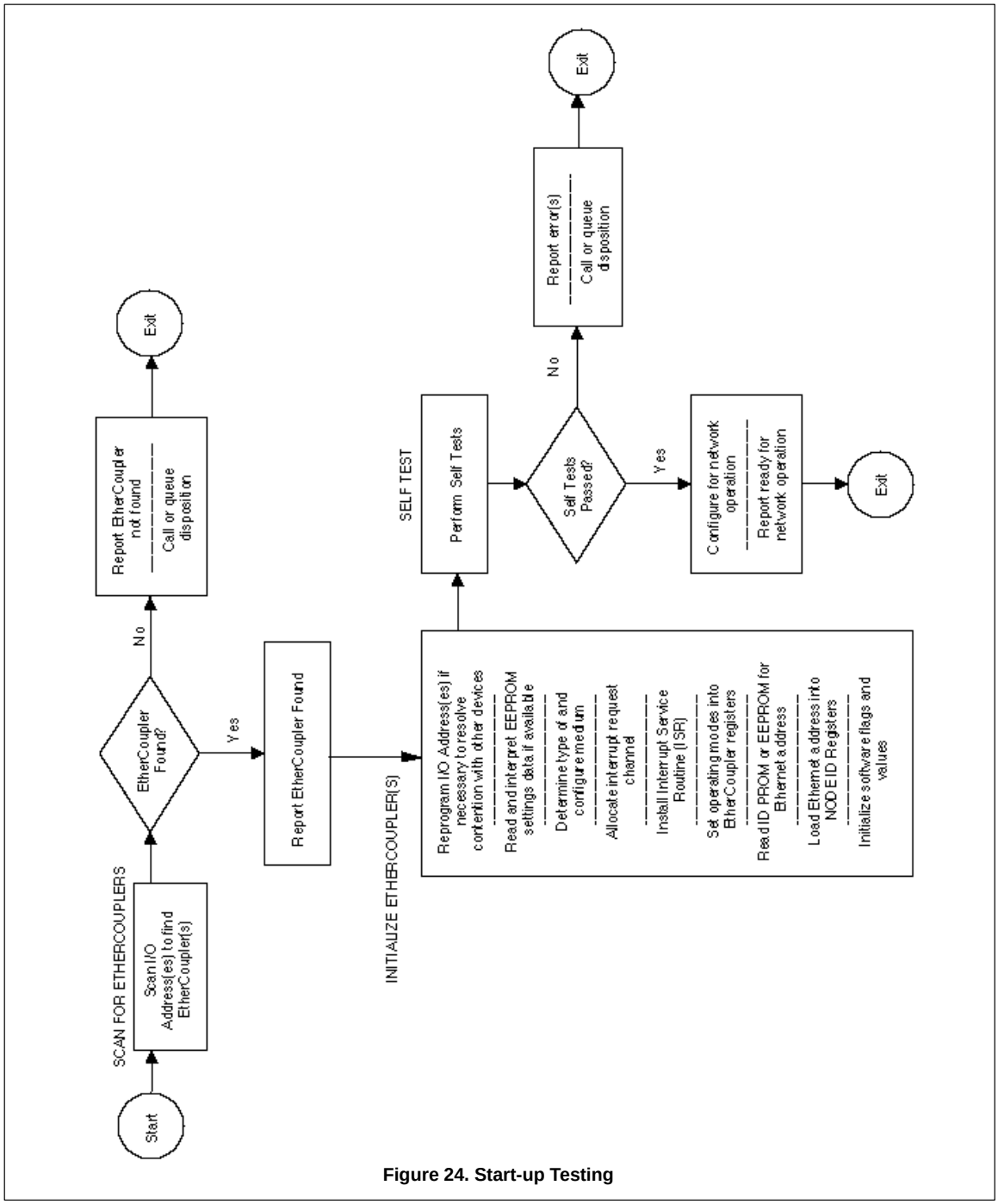


Figure 24. Start-up Testing

OPERATING ON THE NETWORK

Driver code for operating on the network might be partitioned into three main modules as shown in the example for the EtherCoupler controller in this section. The modules, shown in Figures 25, 26, and 27 are Transmit Packet Write, in which packets to be transmitted are moved from host memory to the point marked TX PKT WRITE.

Transmission takes place in two steps. First, packets to be transmitted are loaded into the transmit buffer. Secondly, when the transmitter is not busy, it will be started to transmit the stored packets. Each of these steps may have to wait for resources. The packets cannot be loaded unless there is buffer space available. EtherCoupler provides the option of a single or two independent transmit buffers. With two buffers there is usually no waiting. The transmitter cannot transmit but one buffer full of packets at a time. To manage these resources, two software flags are used, TBUF STAT and TX STAT, the status of the transmit buffer and the transmitter respectively. TBUF STAT refers to the current buffer which might be available to the driver for loading. Its status can be Busy if no buffer is available, Loading when in the process of being loaded, in Standby if ready to transmit, but not full, ready and Full or Empty. The transmitter status can be either Busy or Idle.

Packet length is checked during the loading process to assure that the ISO/ANSI/IEEE 8802-3 length requirements are met. Packets ready to be loaded can be loaded into an Empty or Standby buffer, the latter being a buffer with packets waiting for the transmitter to become idle. If a standby buffer has more room for packets, additional packets can be loaded until it is full. The Empty buffer is available for loading and has no packets. The driver takes ownership of an Empty or Standby buffer by changing its status to Loading.

After the packets are loaded, the transmit status flag is checked for an idle transmitter. If idle, it can be immediately started to transmit the contents of either a Full or Standby buffer. When the transmitter of the EtherCoupler chip is started, buffer status also changes. In single buffer mode, starting the transmitter makes the single buffer unavailable to the system. In dual buffer mode, starting the transmitter re-allocates its previously-transmitted buffer as an empty buffer, available for loading. If the transmitter is busy, the routine will suspend execution at that point pending an idle transmitter.

Two key interrupts used in this example are the receive packet interrupt (RX PKT), indicating that one or more packets has been received since the interrupt was last enabled, and the transmitter done interrupt (TX DONE), indicating that the transmitter has finished transmitting the contents of its current buffer. The interrupt service routine for network operation, illustrated in Figure 26, is short and sweet. If the receive packet interrupt has occurred, it calls or queues the routine for reading packets (RX PKT READ). Further receive interrupts are masked until the driver has emptied the receive buffer. This prevents redundant interrupts which would otherwise occur if packets come in during the read sequence. If the transmitter done interrupt has occurred, the status flags are updated, and appropriate action is taken to satisfy pending activity, if any, with the newly available buffer and/or transmitter resource.

The driver is structured to read all receive packets in the buffer whenever one or more packets arrive. A status bit in EtherCoupler (RBUF EMPTY) indicates whether the receive buffer is empty or not (indicating whole packets only). The packet read routine, shown in Figure 27, starts by masking further receive interrupts until it has emptied the buffer and suspended execution. This prevents redundant interrupts which would otherwise occur if packets come in during the read sequence. As each packet is read, it can be read in parts. If after reading the first part the packet is not of interest, the rest can be discarded without being moved to host memory using EtherCoupler's Skip Packet feature. Reading will continue until the buffer is empty, as indicated by RBUF EMPTY bit.

The Transmit Packet Write Routine, shown in Figure 25, and the Receive Packet Read Routine, shown in Figure 27, together comprise the driver core, which can be called by the network software or from the Network Interrupt Service Routine, shown in Figure 26. Once called, this core routine transfers both transmit and receive packets until there are no more to be transferred, then exits or returns. If packets are transferring, this core routine avoids locking out either the transmitter or the receiver while the other is very busy, by alternating between the two after a fixed number of packets, set by TX MAX and RX MAX. While in operation, the core routine polls key status bits. Interrupts are disabled to prevent unnecessary interrupts while the core is executing.

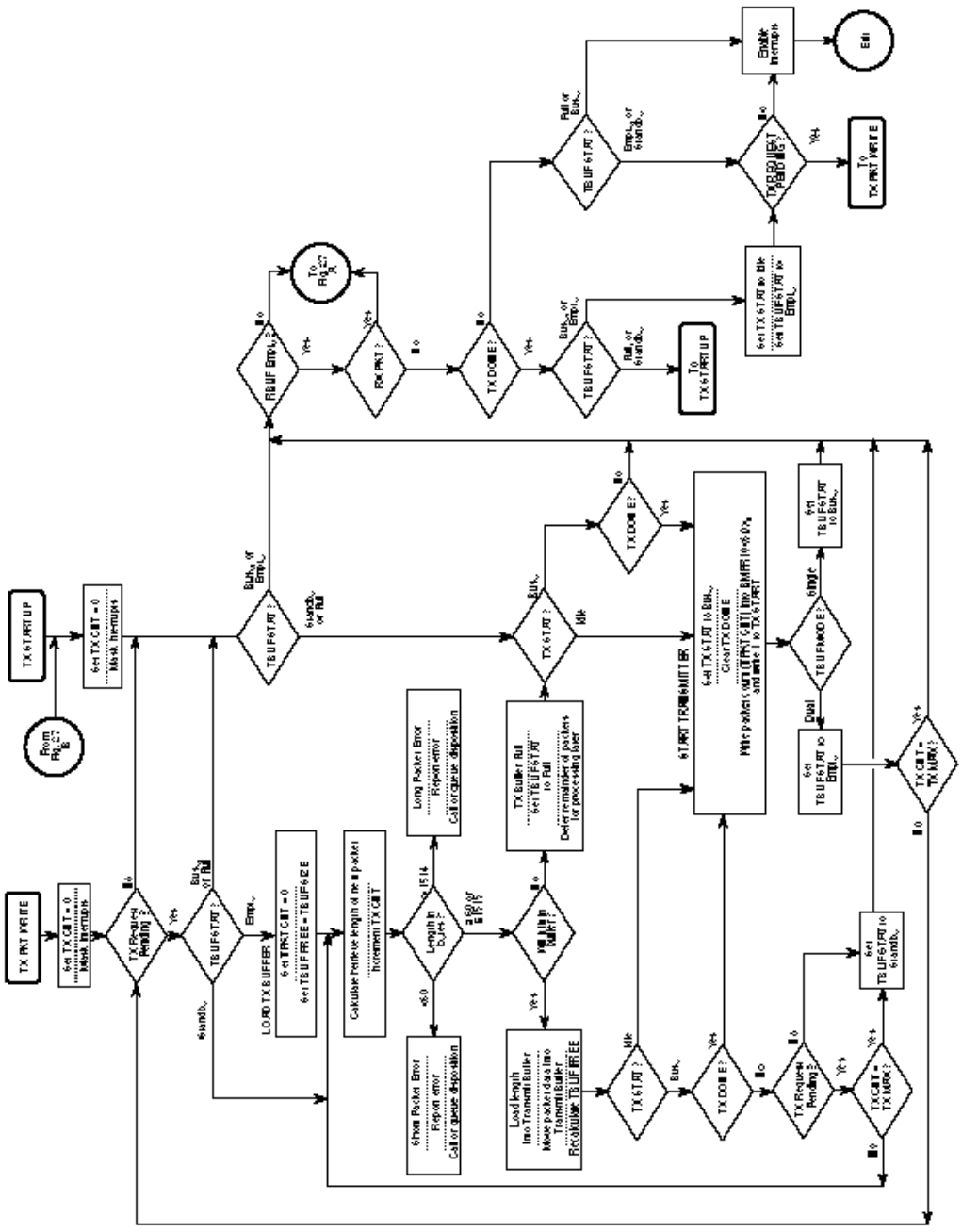


Figure 25. Transmit Packet Write Routine

A driver such as the one illustrated in this section might typically occupy 4 to 6 kilobytes on its distribution diskette. The host-resident portion, when loaded for network operation, might use typically 2 to 3 kilobytes of host memory. A set of quality software drivers bundled with the hardware is well worth providing to LAN equipment customers. By providing better performance and reliability, good drivers will enhance both customer satisfaction and sales, while reducing customer service calls.

Table 32 represents typical control and status parameter used in EtherCoupler Network Drivers.

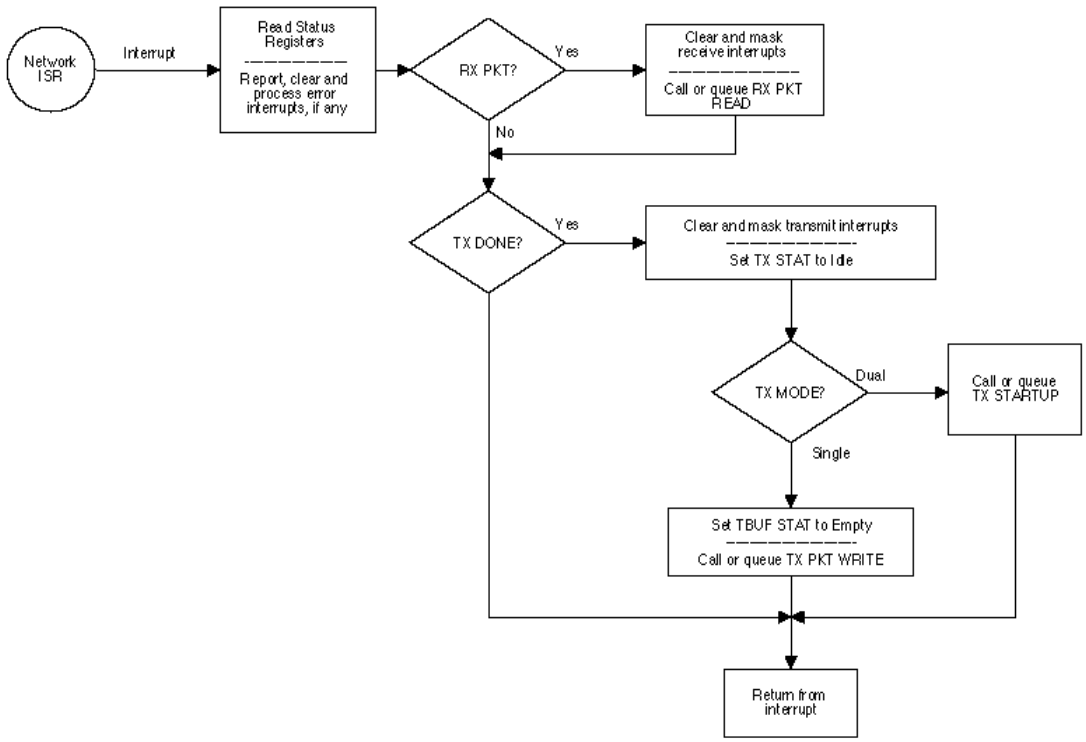


Figure 26. Network Interrupt Service Routine

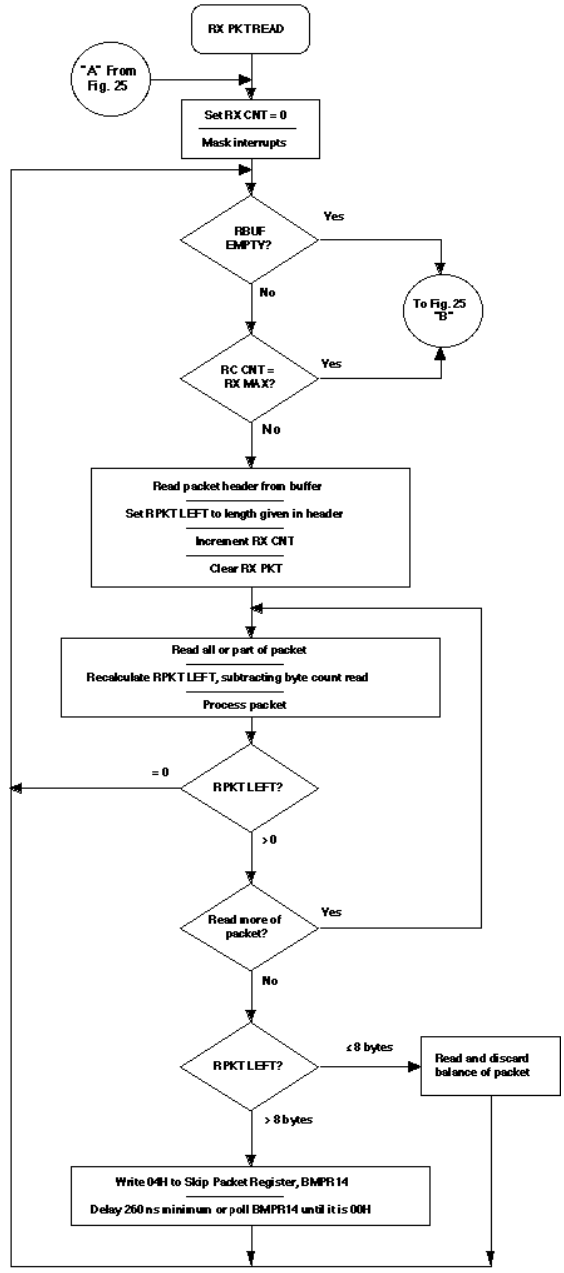


Figure 27. Receive Packet Read Routine

Table 32. Control and Status Parameters

SYMBOL	TYPE	NAME	DESCRIPTION
DLC EN	Register Bit	DATA LINK CONTROL ENABLE	When high, resets all buffer memory pointers and disables both the transmitter and the receiver circuits. When low, enables buffer memory, transmitter and receiver.
LBC	Register Bit	LOOPBACK CONTROL	When set low, places EtherCoupler in internal loopback mode.
RBUF EMPTY	Register Bit	RECEIVE BUFFER EMPTY	When high, indicates that there is at least one complete packet stored in the receive buffer, ready to read. When low, indicates there are no packets ready to read in the receive buffer.
RPKT LEFT	Software Value	RECEIVE PACKET LENGTH LEFT	The number of bytes remaining in the receive packet being read from the buffer. Calculated by the driver from the original length given by EtherCoupler in the 3rd and 4th bytes of the receive packet header, less the number of bytes already read out.
RX CNT	Software Value	RECEIVE TRANSFER COUNT	A running count of the number of packets transferred from the receive buffer to system memory since entering the Receive Packet Read Routine. This parameter can share the same memory location with TX CNT, because both are not used concurrently in the same subroutine.
RX MAX	Software Value	MAXIMUM VALUE FOR RX CNT	The maximum number of packets that may be processed in the Receive Packet Read Routine.before passing control to the Transmit Packet Write Routine. This parameter can be fixed or allowed to vary according to need. Typical range for this parameter is 15 - 32.
TBUF FREE	Software Value	TRANSMIT BUFFER FREE SPACE	The number of available bytes remaining in the transmit buffer being loaded with packets by the driver. Calculated by driver.
TBUF MODE	Register Bits	TRANSMIT BUFFER MODE	The configuration of the transmit buffer space, SINGLE buffer or DUAL buffers.
TBUF SIZE	Software Value	TRANSMIT BUFFER SIZE	The size in bytes of each transmit buffer, which depends on initial configuration parameters for the buffer memory.
TBUF STAT	Software Value	TRANSMIT BUFFER STATUS	Current status of the transmit buffer available for loading packets. Maintained by the driver.

TBUF STAT	VALUE DESCRIPTION
Empty	Current transmit buffer available to system bus is completely empty.
Busy	No buffer is currently available to the system bus because a) EtherCoupler is in single-buffer mode and b) the transmitter is using the buffer.
Standby	The buffer currently available to the system bus has one or more packets in it, but may still have room for additional packet(s).
Full	The buffer currently available to the system bus has one or more packets in it, and does not have room for the next packet presented to the driver from the transmit queue.

TPKT CNT	Software Value	TRANSMIT PACKET COUNT	The number of packets loaded into the current transmit buffer by the driver. Value is counted by driver. This value is written into EtherCoupler register BMPR10<6:0> at the time transmitter is started.
TX CNT	Software Value	TRANSMIT TRANSFER COUNT	A running count of the number of packets transferred from system memory to the transmit buffer since entering the Transmit Packet Write Routine. This parameter can share the same memory location with RX CNT, because both are not used concurrently in the same subroutine.
TX DONE	Register Bit	TRANSMITTER DONE	When transmitter finishes transmitting, EtherCoupler sets this bit high. Normally cleared by driver prior to starting transmitter. Hardware reset or DLC EN being set high also clears this bit.
TX MAX	Software Value	MAXIMUM VALUE FOR TX CNT	The maximum number of packets that may be processed in the Transmit Packet Write Routine before passing control to Receive Packet Read Routine. This parameter can be fixed or allowed to vary according to need. Typical range for this parameter is 15-32.
TX START	Register Bit	START TRANSMITTER	When set high, activates transmitter to transmit all packets in the current transmit buffer. See also TPKT CNT.
TX STAT	Software Value	TRANSMIT BUFFER STATUS	Current status of the- transmitter, maintained by driver.

TX STAT	VALUE DESCRIPTION
Busy	Transmitter has not finished transmitting packets previously given to it to transmit.
Idle	Transmitter has finished transmitting all packets in its buffer.

FEATURES

OPERATIONAL SPECIFICATION

Table 33. Absolute Maximum Ratings

SYMBOL	PARAMETER DESCRIPTION	MINIMUM	MAXIMUM	UNITS
V_{CC}	Supply voltage	– 0.5	6.0	V
V_{IN}	Input voltage	– 0.5	V_{CC}+ 0.5	V
V_{OUT}	Output voltage	– 0.5	V_{CC} + 0.5	V
I_{ODF}	Differential output current on DOP pins		– 4 0	mA
V_{IDC}	Input DC voltage on DIP and CIP	– 0.5	16	V
V_{ODC1}	Output DC voltage on DOP without transformer	– 0.5	14	V
V_{ODC2}	Output DC voltage on DOP with transformer	– 0.5	16	V
T_{BIAS}	Temperature under bias	– 2 5	+85	°C
T_{STG}	Storage temperature	– 40	+125	°C
PWR	Power dissipation		787.5	mW
Permanent device damage may occur if absolute maximum ratings are exceeded. Exposure to absolute maximum rating conditions for extended periods may affect device reliability. No more than one output may be shorted to ground of V_{CC} at a time for a maximum duration of one second.				

Table 34. Recommended Operating Conditions

SYMBOL	PARAMETER DESCRIPTION	MINIMUM	TYPICAL	MAXIMUM	UNITS
V_{DD}	Supply Voltage	4.75		5.25	V
V_{IH}	Logic input high voltage	2.2			V
V_{IL}	Logic input low voltage			0.8	V
R_L	Driver load resistors (across DOP and DON)	77	78	79.5	Ω
R_T	Termination resistors (two in series across DIP and CIP)	38.6	39	39.4	Ω
C_T	Termination bypass capacitors (between junction of termination resistors, ground)		0.1		μF
C_{OSC}	Oscillator load capacitors	12	20	38	pF
f_{XTAL}	Crystal oscillator frequency	19.999	20	20.001	MHz
T_A	Operating temperature	0		70	°C

Table 35. DC Specifications

SYMBOL	PARAMETER DESCRIPTION	CONDITIONS	MINIMUM	TYPICAL	MAXIMUM	UNITS
V_{IL}	Low-level input voltage		0.0		0.8	V
V_{IH}	High-level input voltage		2.2		V_{CC}	V
V_{OL1}	Low-level output voltage, all outputs except DREQ	I_{OL} = 3.2 mA	0.0		0.4	V
V_{OL2}	Low-level output voltage, DREQ only	I_{OL} = 12 mA	0.0		0.4	V
V_{OL3}	Low-level output voltage, pins 130 – 132, 151, and 128	I_{OL} = 24 mA	0.0		0.4	V
V_{OH1}	High-level output voltage, all outputs except DREQ	I_{OH} = –2 mA	4.2		V_{CC}	V
V_{OH2}	High-level output voltage, DREQ only	I_{OH} = –4 mA	4.2		V_{CC}	V
V_{OH3}	High-level output voltage, pins 130 – 132, 151, and 128	I_{OH}= –8 mA	4.2		V_{CC}	V
V_{OP}	DOP peak output	R_L = 270 Ω	±0.5		± 1.3	V
V_{ACCM}	Output AC common mode on DOP	R_T = 78 Ω			± 40	mV
V_{DCCM}	Output DC common mode on DOP	R_L = 270 Ω	2.4	3.4	4.4	V
V_{SQ}	Squelch threshold	AC/DC = 1 AC/DC = 0	–300 –80	–220 0	–140 +80	mV
I_L	Input leakage current	V_I = 0 – V_{CC}	–10		10	µA
I_{PWRDN}	Powerdown V_{CC} current	No output loads		4	10	mA
I_{IDLE}	Idle V_{CC} current	No output loads		83	100	mA
I_{CC}	Operating V_{CC} current	No output loads		110	150	mA

Table 36. General Capacitance

SYMBOL	PARAMETER DESCRIPTION	MINIMUM	MAXIMUM	UNITS
C_{IN}	Input pin capacitance		16	pF
C_{OUT}	Output pin capacitance		16	PF
$C_{I/O}$	I/O pin capacitance		16	pF
T_A = 25°C, V_{DD} = V_I = 0 volts, and f = 1 megahertz				

Table 37. AUI Electrical Characteristics

SYMBOL	PARAMETER	MINIMUM	TYPICAL	MAXIMUM	UNITS
IIL	Input low current			–700	mA
IIH	Input high current Figures for design aid only; not guaranteed and not subject to production testing.			500	µa
VOD	Differential output voltage	±550		±1200	mV
VDS	Differential squelch threshold		220		mV
Typical figures are at 25°C and are used for design aid only; not guaranteed and not subject to production testing. TA = 0°C to +70°C, VCC = 5 V± 5%					

Table 38. Twisted-pair Electrical Characteristics

Z_{OUT}	Transmit output impedance		5		Ω
V_{OD}	Peak differential output voltage	Load = 100 ohms at TPOP and TPON	3.5		V
J_{OUT}	Transmit timing jitter addition Parameter is guaranteed by design; not subject to product testing.	0 line length After line model specified by IEEE 802.3 for 10BASE-T		± 8 ± 3.5	ns
Z_{IN}	Receive input impedance	Between TPIP/TPIN, CIP/CIN and DIP/DIN	20	0	kΩ
V_{DS}	Differential squelch threshold		420		mV
V_{DSL}	Lower squelch threshold		250		mV
Typical figures are at 25°C and are used for design aid only; not guaranteed and not subject to production testing. T_A = 0°C to +70° C, V_{CC} = 5 V ± 5%					

Table 39. Switching Characteristics

SYMBOL	PARAMETER	MINIMUM	TYPICAL	MAXIMUM	UNITS
Jabber Timing	Maximum transmit time	20		150	ms
	Unjab time	250		750	
Link Integrity Timing	Time link loss	55		66	ms
	Time between link integrity pulses	8		24	
	Interval for valid receive link integrity pulses	4.1		65	
General	Receive startup delay	0		500	ns
	Transmit startup delay	0		200	
	Loopback startup delay	0		500	
	(These parameters are guaranteed by design and not subject to product testing.)				
T_A = 0°C to +70°C, V_{CC} = 5 V ± 5%					

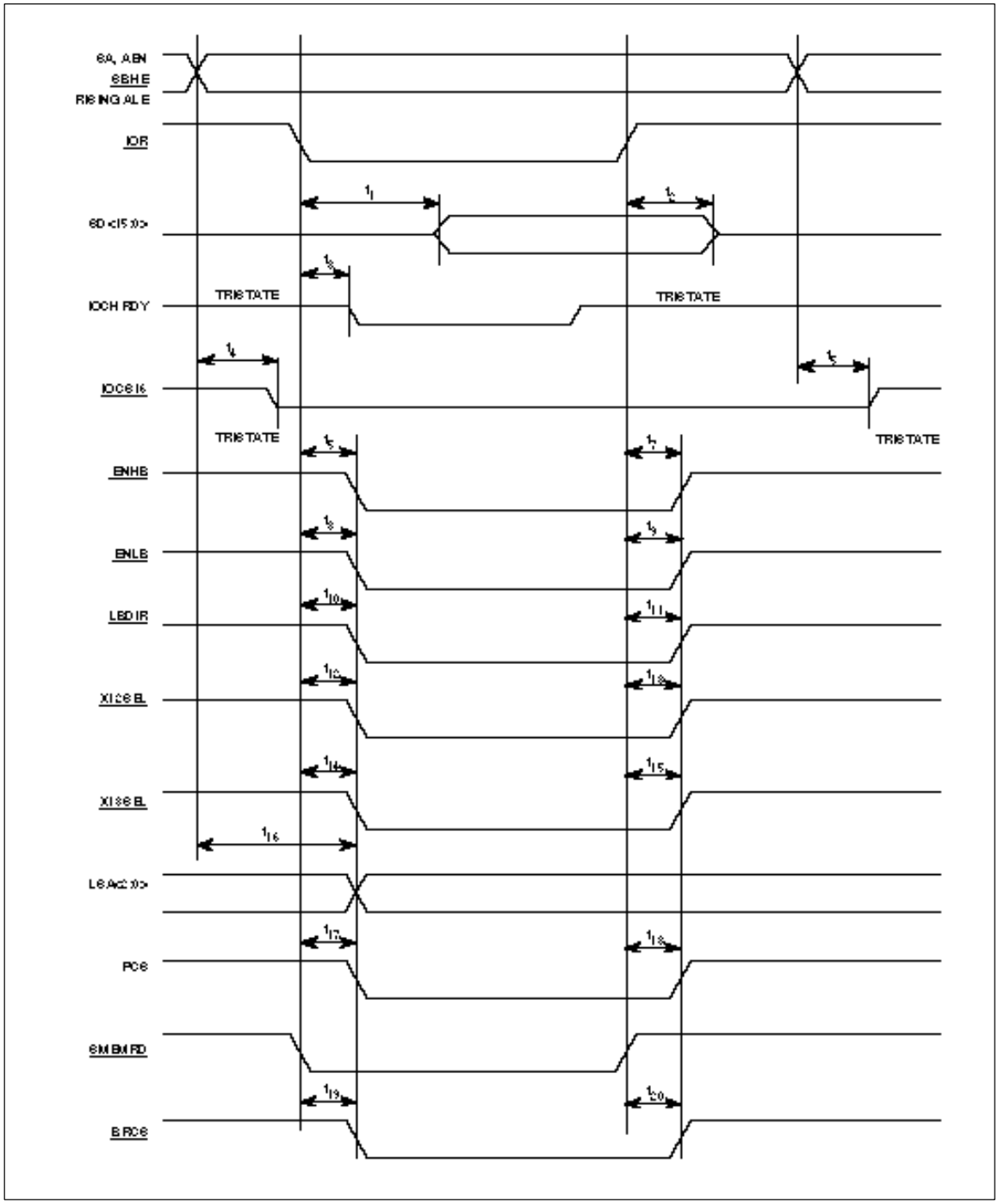
TIMING DIAGRAMS

Table 40. operational modes in which EtherCoupler may operate, and for which timing diagrams are specified in Tables 41-57

Table 40. Operational Modes

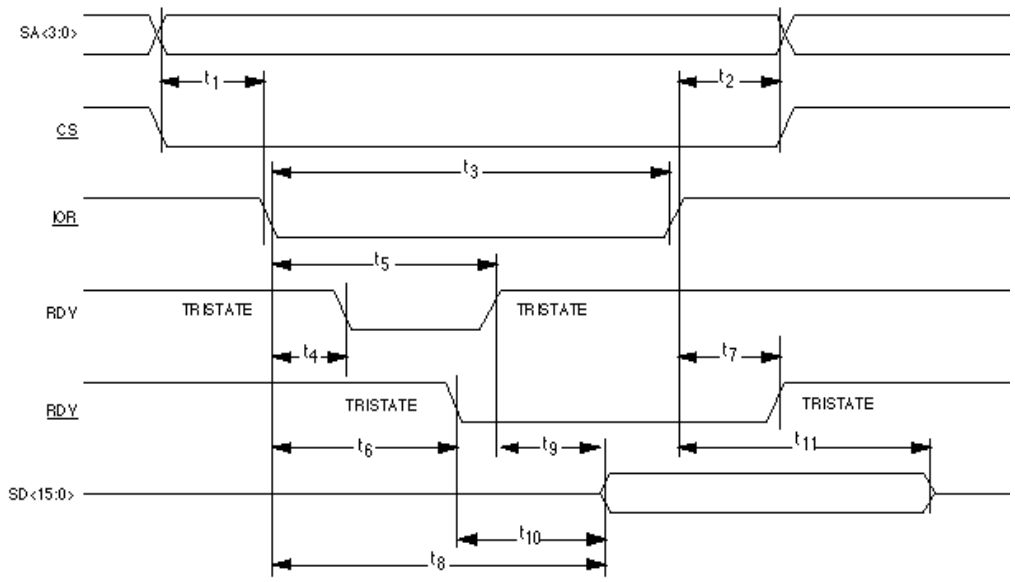
MODE NO.	BUS	PIN 90	PIN 90	PIN 92	OPERATIONAL	INITIAL PARAMETERS
0	ISA	0	0	0	Jumperless	Stored in bit-serial EEPROM
1	ISA	0	0	1	Jumpers	Stored in bit-serial EEPROM
2	ISA	0	1	0	Jumpers	Stored in byte-parallel ID PROM.
3	Non-IS	0	1	1	Jumpers	Functions identical to MB86960 NICE Network Interface Controller with Encoder/Decoder) chip.
X	—	1	X	X	—	Reserved

Table 41. Read Cycle (Mode 0-2)



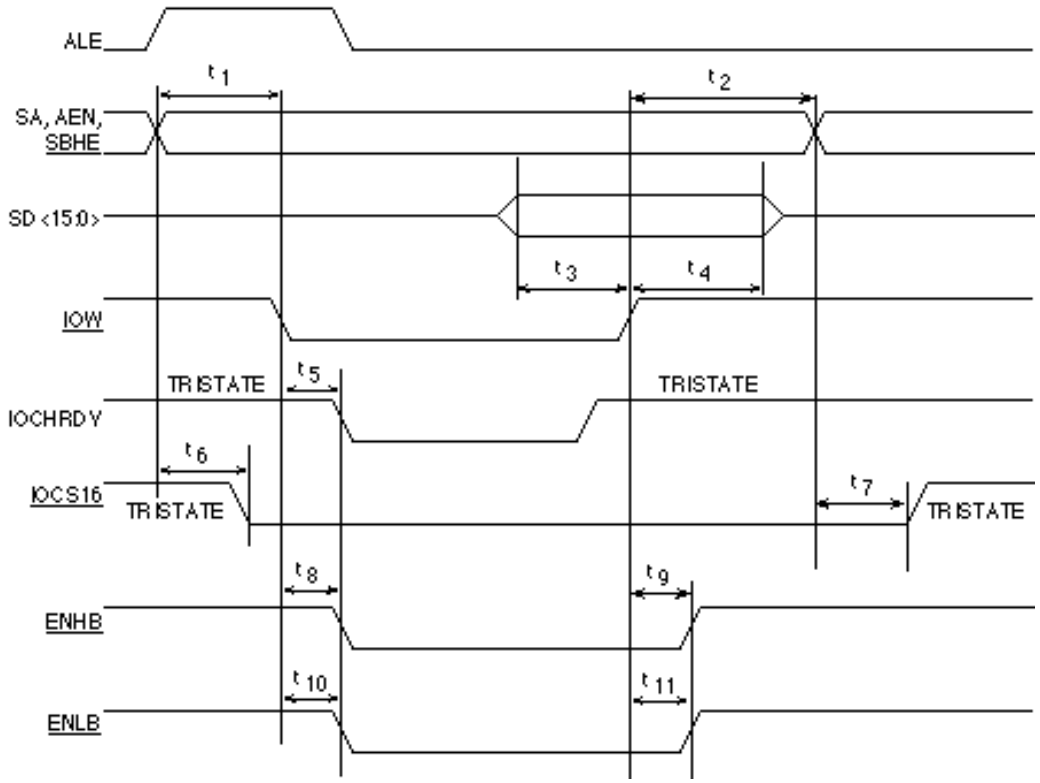
SYMBOL	PARAMETER DESCRIPTION	MINIMUM	TYPICAL	MAXIMUM	UNITS
t_1	IOR to SD<15:0>enable			38	ns
t_2	IOR to SD<15:0>tristate			38	ns
t_3	IOR to IOCHRDY not ready		14	—	ns
t_4	SA, AEN, SBHE to IOCS16 Tristate to low			28	ns
t_5	SA, AEN, to IOCS16 low to Tristate			71	ns
t_6	IOR to ENHB active			46	ns
t_7	IOR to ENLB active			24	ns
t_8	IOR to ENLB inactive			45	ns
t_9	IOR to LBDIR active			24	ns
t_{10}	IOR to ENLB inactive			42	ns
t_{11}	IOR to LBDIR inactive			21	ns
t_{12}	IOR to X12SEL active			44	ns
t_{13}	IOR to X12SEL inactive			23	ns
t_{14}	IOR to X13SEL active			45	ns
t_{15}	IOR to X13SEL inactive			24	ns
t_{16}	SA, AEN, rising ALE to LSA<2:0> stable (Mode 2)			21	ns
t_{17}	IOR to EPCS active (Mode 2)			45	ns
t_{18}	IOR to EPCS inactive (Mode 2)			23	ns
t_{19}	SMEMRD to BRCS active			44	ns
t_{20}	SMEMRD to BRCS inactive			20	ns

Table 42. Read Cycle (Mode 3)



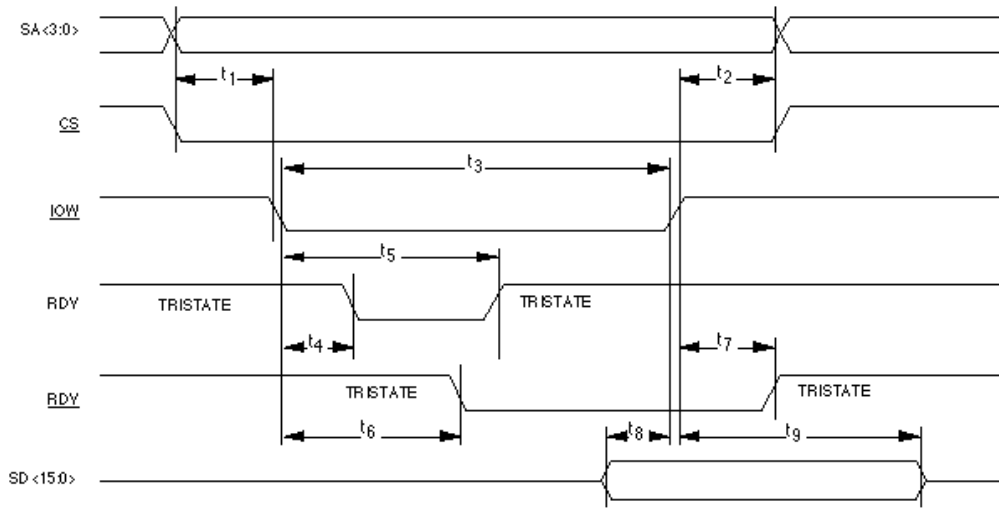
SYMBOL	PARAMETER DESCRIPTION	MINIMUM	MAXIMUM	UNITS
t_1	SA<3:0> valid to IOR low; CS low to IOR Low.	3		ns
t_2	IOR High to SA<3:0> invalid; IOR high to CS high.	3		ns
t_3	IOR low pulse width.	30		ns
t_4	IOR low RDY low	0	26	ns
t_5	IOR low to RDY Tristate		175	ns
	For registers and Buffer Port when is ready before the read cycle begins	0	7	ns
	For port access only. if system makes contiguous system read cycles at less than 100 ns intervals, and the network is busy.		175	ns
	For bus read error.		2.15	µs
t_6	IOR low to RDY low	0	175	ns
	For all registers.		28	ns
	For port access only, if system makes contiguous system read cycles at less than 100 ns intervals, and the network is busy.		175	ns
	For bus read error.		2.15	µs
t_7	IOR high to RDY Tristate		28	ns
t_8	IOR low to SD<15:0> valid (registers).		44	ns
t_9	RDY Tristate to SD<15:0> valid (buffer port).		8	ns
t_{10}	RDY low to SD<15:0> valid .		10	ns
t_{11}	IOR high to SD<15:0> valid (data hold).	15		ns

Table 43. Write Cycle (Mode 2-0)



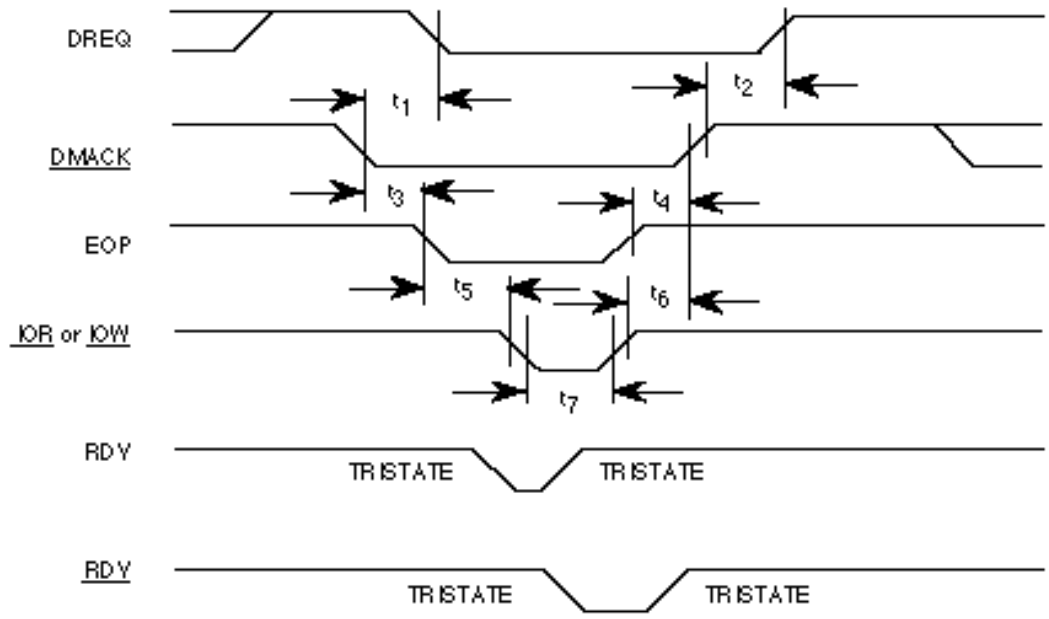
SYMBOL	PARAMETER DESCRIPTION	MINIMUM	TYPICAL	MAXIMUM	UNITS
t_1	SA, SBHE and AEN setup time to IOW low	15			ns
t_2	Hold time of SA, SBHE and AEN after IOW high (up to leading edge of next cycle ALE)	21			ns
t_3	Setup time of SD to IOW high	5			ns
t_4	Hold time of SD to IOW low	33			ns
t_5	IOW to IOCHRDY not ready		14		ns
t_6	SA, AEN, SBHE to IOCS16 low to Tristate			38	ns
t_7	SA, AEN, to IOCS16 Tristate to low			71	ns
t_8	IOW to ENHB active			47	ns
t_9	IOW to ENLB active			24	ns
t_{10}	IOW to ENLB inactive			45	ns
t_{11}	IOW to ENLB inactive			25	ns
AC characteristics: V_{CC}=5.0±5%; T_A=0°C to +70°C					

Table 44. Write Cycle (Mode 3)



SYMBOL	PARAMETER DESCRIPTION	MINIMUM	MAXIMUM	UNITS
t_1	SA<3:0> valid to IOW low; CS low to IOW Low.	3		ns
t_2	IOW High to SA<3:0> invalid; IOW high to CS high.	3		ns
t_3	IOW low pulse width.	36		ns
t_4	IOW low RDY low	0	26	ns
t_5	IOW low to RDY Tristate		175	ns
	For registers and Buffer Port when is ready before the write cycle begins	0	7	ns
	For port access only. if system makes contiguous system write cycles at less than 100 ns intervals, and both the transmitter and receiver are active in loopback reception.		175	ns
t_6	IOW low to RDY low	0	175	ns
	For all registers.		28	ns
	For port access only, if system makes contiguous system write cycles at less than 100 ns intervals, and both the transmitter and receiver are active in loopback reception.		175	ns
t_7	IOW high to RDY Tristate		28	ns
t_8	SD<15:0> valid (data setup) .	5		ns
t_9	IOW high to SD<15:0> valid (data hold) .	6		ns

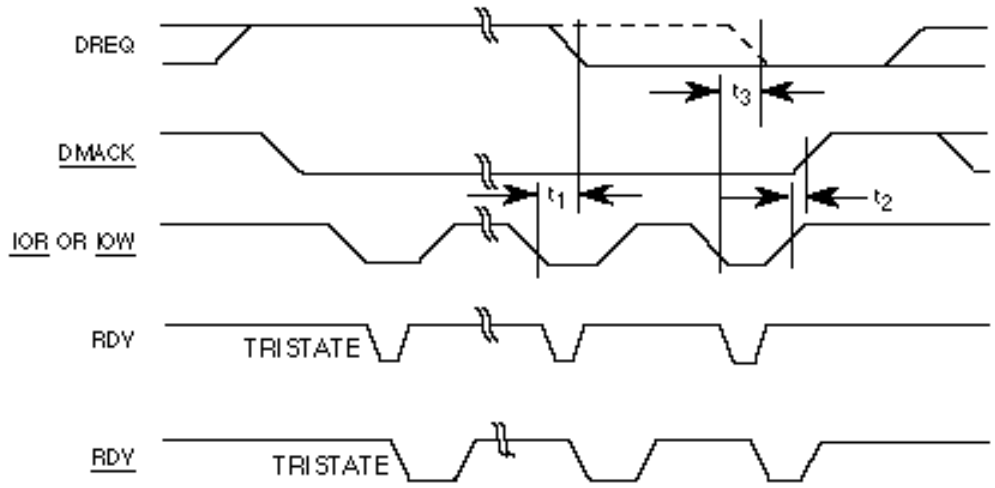
Table 45. Single-cycle DMA Timing



SYMBOL	PARAMETER DESCRIPTION	MINIMUM	MAXIMUM	UNITS
t_1	DMACK low to DREQ low	0	21	ns
t_2	DMACK high to DREQ high	0	19	ns
t_3	DMACK low to IOR or IOW low	0		ns
t_4	IOR or IOW high to DMACK high	3		ns
t_5	IOR or IOW low to EOP low	0		ns
t_6	EOP high to DMACK high	3		ns
t_7	EOP low pulse width	19		ns

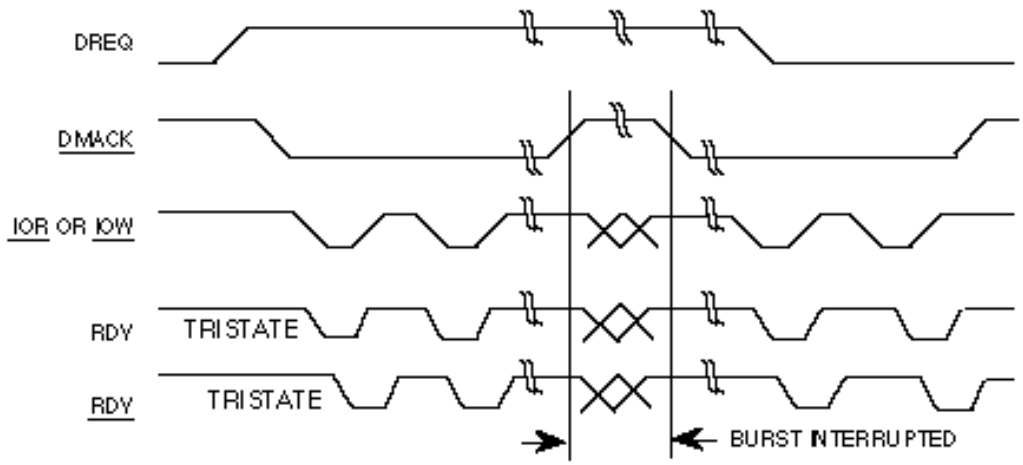
All of the IOR, IOW and EOP low-going pulses must fall inside of the DMACK low-going pulse. An asserted EOP terminates any further DREQ after DMACK returns high. The DMA cycle uses DMACK as the chip select. DMACK overrides SA<3:0>, forcing selection of the Buffer Memory Port as in a DMA cycle.

Table 46. Burst DMA Timing



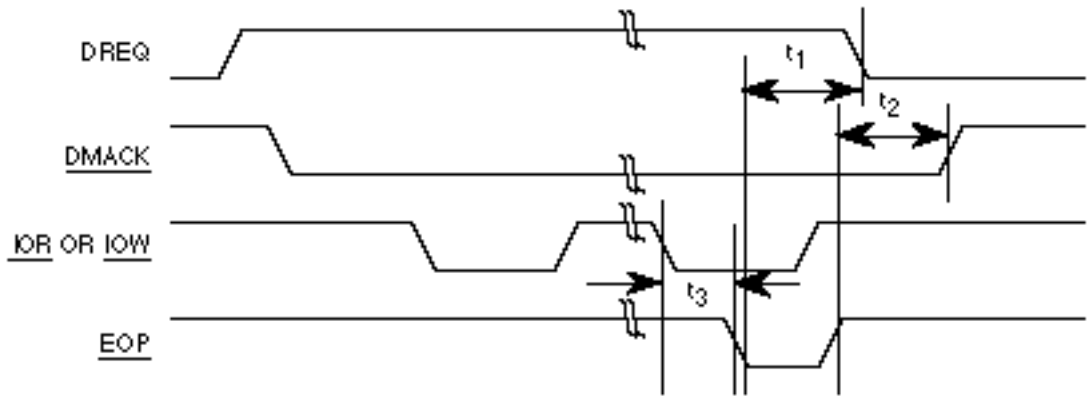
SYMBOL	PARAMETER DESCRIPTION	MINIMUM	MAXIMUM	UNITS
t_1	IOR or IOW low to DREQ low		32	ns
t_2	IOR or IOW high to DACK high	3		ns
t_3	IOR or IOW low to DREQ low, if DREQ EXTND (DLCR4<2>) is set to 1 (default)		32	ns

All of the IOR, IOW and EOP low-going pulses must fall inside of the DMACK low-going pulse. The DMA cycle uses DMACK as the chip select. DMACK overrides SA<3:0>, forcing selection of the Buffer Memory Port as in a DMA cycle. DREQ goes low during the next-to-last transfer of the burst, and can be extended from the next-to-last transfer cycle when DREQ EXTND, DLCR4<2>, is low to the last transfer cycle when DREQ EXTND is high.



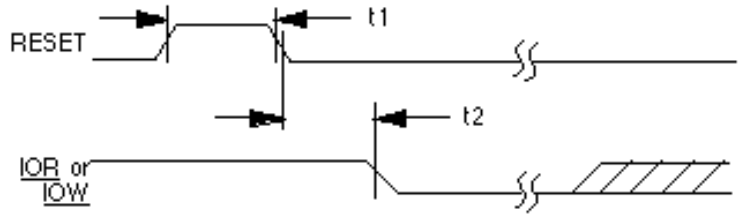
Burst can be interrupted by DMACK high-going pulse during the burst. Burst will resume when DMACK returns low. DREQ goes low during the nest-to-last transfer of the burst.

Table 47. Burst DMA Terminated by EOP



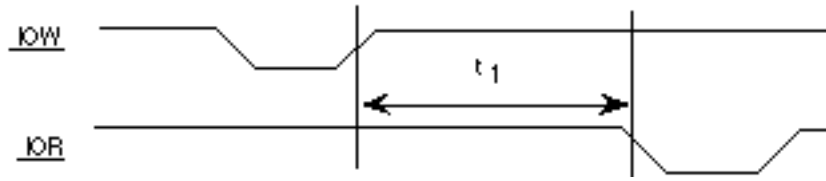
SYMBOL	PARAMETER DESCRIPTION	MINIMUM	MAXIMUM	UNITS
t_1	EOP low to DREQ low	4	28	ns
t_2	EOP high to DMACK high	3		ns
t_3	IOR or IOW low to EOP low	0		ns

Table 48. RESET Timing



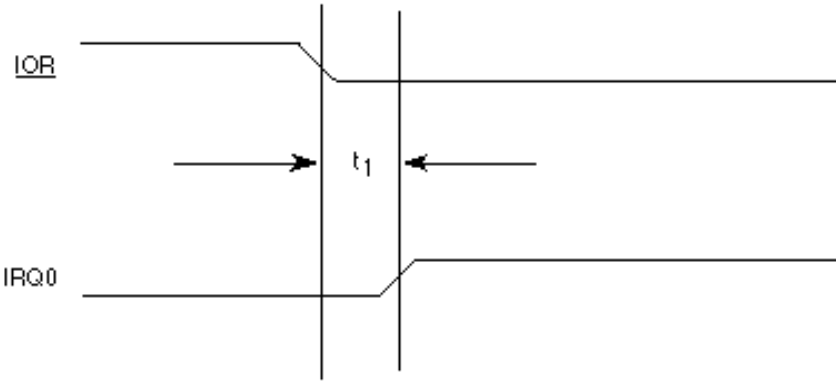
SYMBOL	PARAMETER DESCRIPTION	MINIMUM	MAXIMUM	UNITS
t_1	RESET pulse width	200		ns
t_2	RESET low to first IOR or IOW low	300		ns
Before enabling Transmit and Receive functions, wait 200µ sec after reset pulse for stabilization of receiver PLL.				

Table 49. Skip Packet Timing



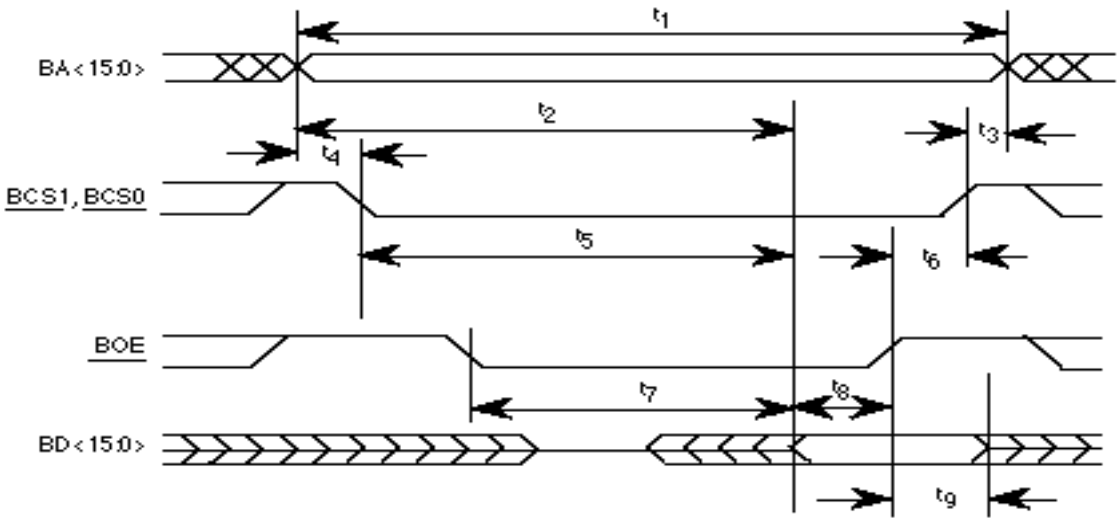
SYMBOL	PARAMETER DESCRIPTION	MINIMUM	MAXIMUM	UNITS
t_1	Writing skip packet high to next Buffer Memory Port read	300		ns

Table 50. IRQ Timing



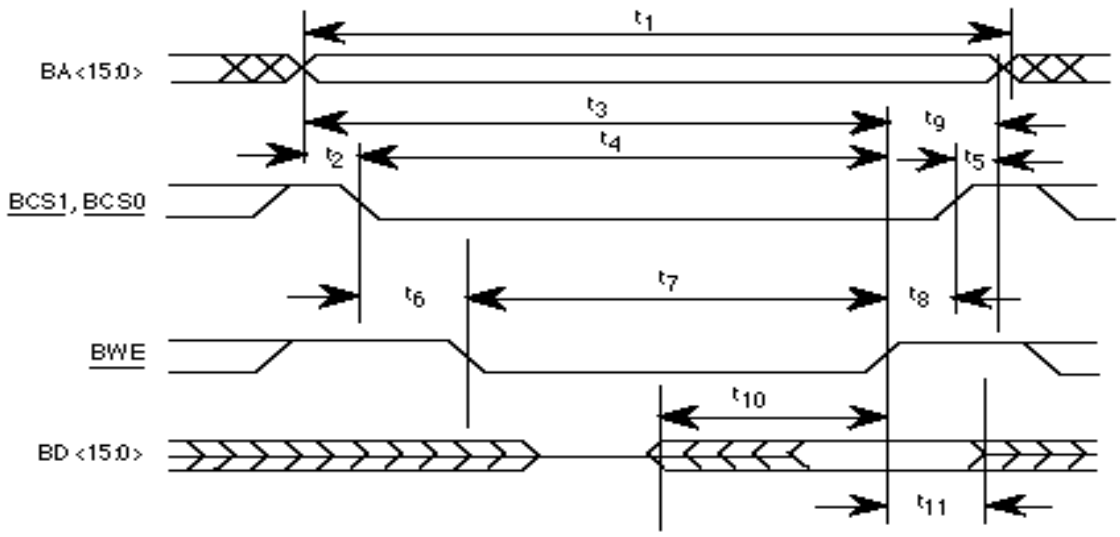
SYMBOL	PARAMETER DESCRIPTION	MINIMUM	TYPICAL	MAXIMUM	UNITS
t_3	IRQ0 signal-clearing delay	30		80	ns

Table 51. SRAM Read Timing



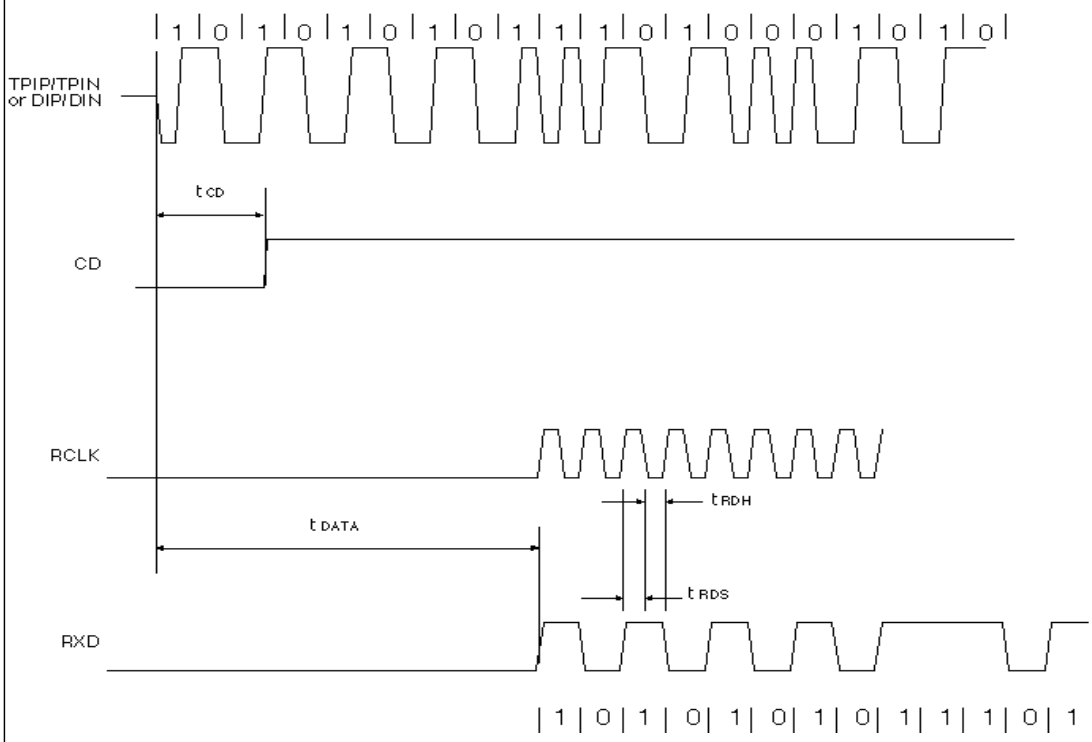
SYMBOL	PARAMETER DESCRIPTION	MINIMUM	TYPICAL	MAXIMUM	UNITS
t_1	Read cycle, 100 ns	95	100	105	ns
	Read cycle, 150 ns	145	150	155	ns
t_2	T_{AA}, address access time			81	ns
t_3	BCS!, BCS0 high to address invalid	0			ns
t_4	Address valid to BCS!, BCS0 low	0		8	ns
t_5	T_{AC} chip select access time			81	ns
t_6	BOE high to BCS!, BCS0 high	0		2	ns
t_7	T_{OE} output enable access time			49	ns
t_8	Data setup time	15			ns
t_9	Data hold time	0			ns
Use SRAM with T_{AA} of 80 ns or less					

Table 52. SRAM Write Timing



SYMBOL	PARAMETER DESCRIPTION	MINIMUM	TYPICAL	MAXIMUM	UNITS
t_1	Write cycle, 100 ns	95	100	105	ns
	Write cycle, 150 ns	145	150	155	ns
t_2	Address valid to BCS!, BCS0 low	2		8	ns
t_3	Address valid to BWE low	71			ns
t_4	BCS!, BCS0 low to BWE high	62			ns
t_5	BCS!, BCS0 high to address invalid	0			ns
t_6	BCS!, BCS0 low to BWE low	0			ns
t_7	BWE pulse width	60			ns
t_8	BWE high to BCS!, BCS0 high	0			ns
t_9	BWE high to address invalid	12			ns
t_{10}	Data setup time	41			ns
t_{11}	Data hold time	14			ns
Use SRAM with T_{AA} of 80 ns or less					

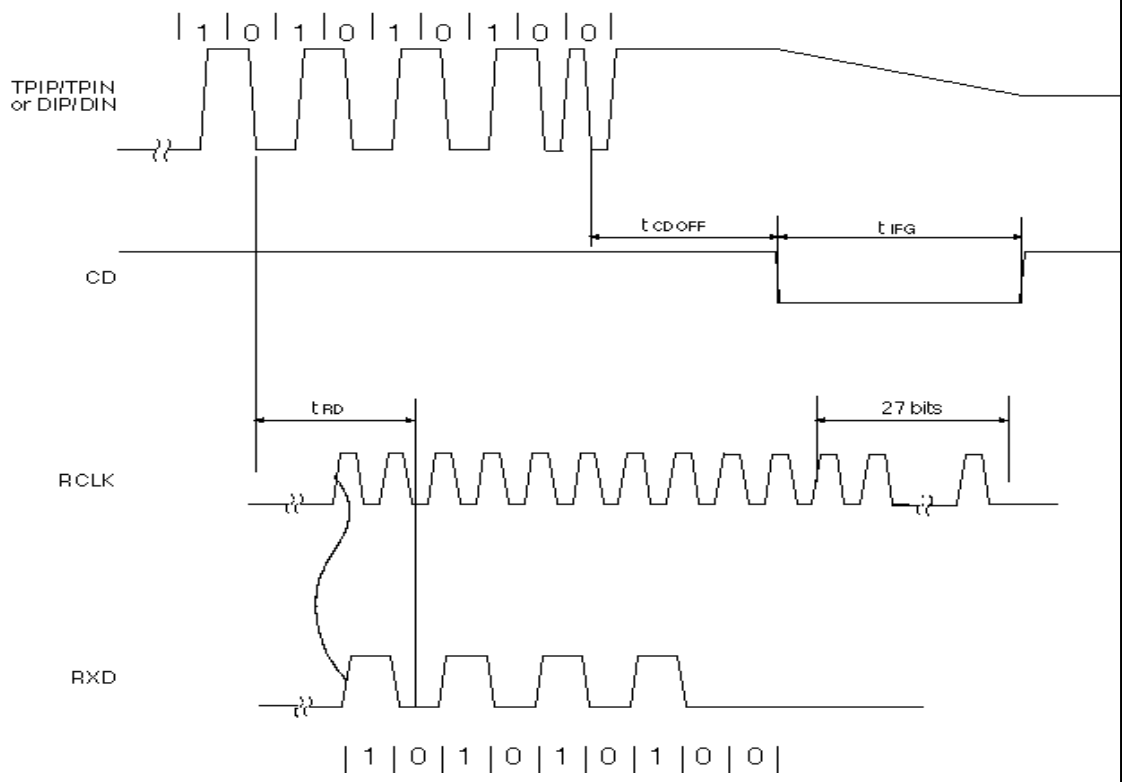
Table 53. RCLK/Start of Frame Timing (Mode 3) — For Reference Only



SYMBOL	PARAMETER DESCRIPTION	MINIMUM	TYPICAL	MAXIMUM	UNITS
t_{CD}	CD turn-on delay - AUI	—	900	—	ns
t_{CD}	CD turn-on delay - TP	—	1300	—	ns
t_{DATA}	Decoder acquisition time - AUI	—	50	—	ns
t_{DATA}	Decoder acquisition time - TP	—	400	—	ns
t_{RDH}	Receive data hold from RCLK	30	—	—	ns
t_{RDS}	Receive data setup from RCLK	30	—	—	ns

The CD, RCLK and RXD signals are internal to the EtherCoupler chip, and are provided for reference only. The RXD signal changes at the rising edge of RCLK. The controller is sampled at the falling edge.

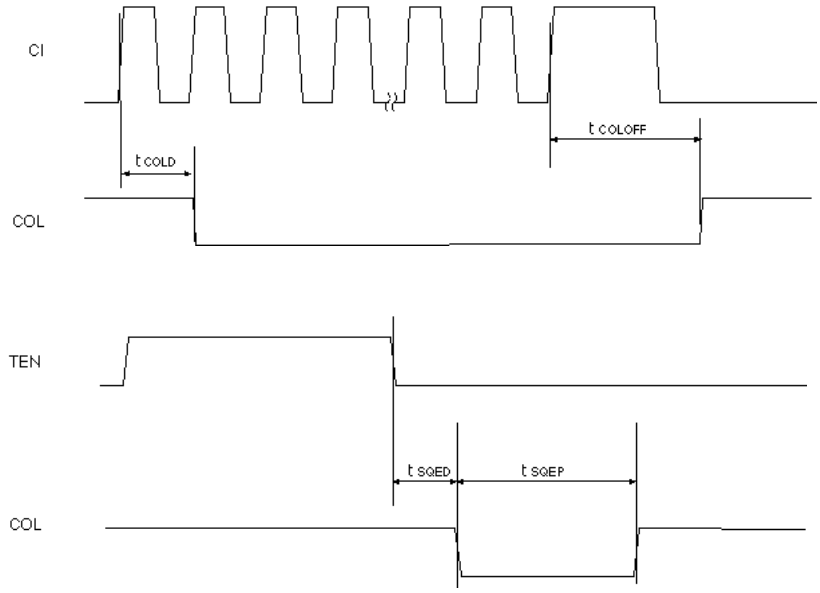
Table 54. RCLK/End of Frame Timing (Mode 3) — For Reference Only



SYMBOL	PARAMETER DESCRIPTION	MINIMUM	TYPICAL	MAXIMUM	UNITS
t_{RCH}	RCLK hold after CD low	27	—	—	bit time
t_{RD}	RCV data throughput delay	—	275	—	ns
t_{CDOFF}	CD turn-off delay, measured from middle of last bit, so that timing specification is unaffected by the value of the last bit.	—	375	—	ns
t_{IFG}	Receive block out	—	400	—	bit time

The CD, RCLK and RXD signals are internal to the EtherCoupler chip, and are provided for reference only. The RXD signal changes at the rising edge of RCLK. The controller is sampled at the falling edge. Typical figures are at 25°C and are for design aid only, not guaranteed and not subject to production testing.

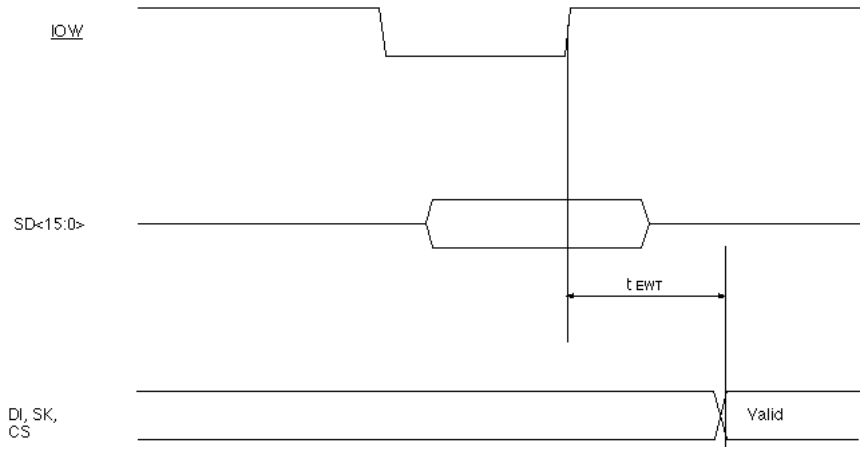
Table 55. Collision Detect and COL/CI Output Timing (Mode 3) — For Reference Only



SYMBOL	PARAMETER DESCRIPTION	MINIMUM	TYPICAL	MAXIMUM	UNITS
t_{COLD}	COL turn-on delay	—	50	—	ns
t_{COLOFF}	COL turn-off delay	—	160	—	ns
t^{SQED}	SQE delay	0.65	—	1.6	ns
t_{SQEP}	SQE pulse duration	500	—	1500	ns

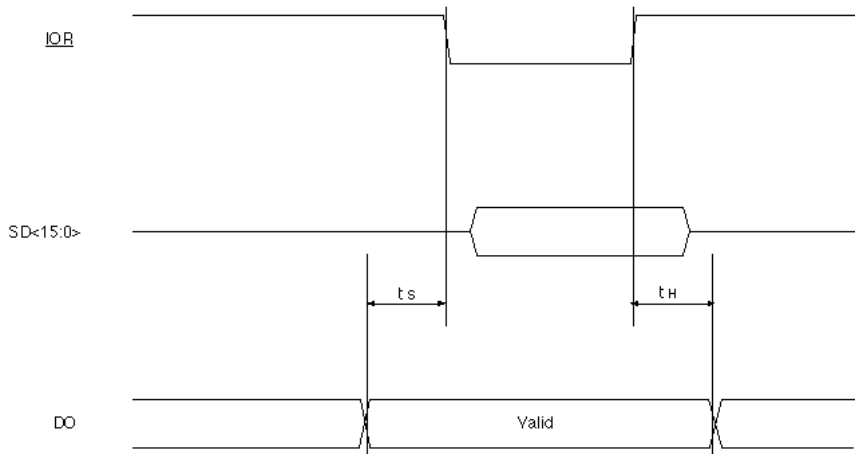
The CI, COL and TEN signals are internal to the EtherCoupler chip, and are provided for reference only. Typical figures are at 25°C and are for design aid only, not guaranteed and not subject to production testing.

Table 56. EEPROM Write Timing (Mode 0-2)



SYMBOL	PARAMETER DESCRIPTION	MINIMUM	TYPICAL	MAXIMUM	UNITS
t_{EWT}	IOW to DI, SK CS valid	9	—	24	ns
The SD signal is for reference only.					

Table 57. EEPROM Read Timing



SYMBOL	PARAMETER DESCRIPTION	MINIMUM	TYPICAL	MAXIMUM	UNITS
t_S	DO valid to IOR	2	—	—	ns
t_H	IOR to DO invalid	2	—	—	ns
The SD signal is for reference only.					

ETHERCOUPLER BOARD

INTRODUCTION

The EtherCoupler Ethernet adapter card is an I/O board that provides a reliable, high-performance networking interface between an IBM XT or AT-compatible personal computer and Ethernet. It supports three types of cables: standard Ethernet (10BASE5), thin Ethernet (10BASE2), and twisted-pair Ethernet (10BASE-T). 10BASE5 is supported by use of an external remote transceiver unit; 10BASE2 is supported on board directly by the MBL8392A transceiver chip; 10BASE-T is supported on board directly by the MB86965 Ether-Coupler chip.

The EtherCoupler board occupies 32 bytes of the 1 kilobyte I/O space of the PC, and 16 kilobytes of the memory space for the Boot PROM. The EtherCoupler controller is completely I/O mapped, and occupies 16 of the 32 I/O addresses. Eight of the other 16 addresses are used to read the Ethernet ID PROM: one is used for jumperless operation control, while the remaining seven are reserved for future use. When using the board in 8-bit mode, all I/O is presented to the bus as 8-bit transfers on the lower byte of the data bus. When using the 16-bit mode, the EtherCoupler controller is presented as a 16-bit device, while the ID PROM and boot PROM are still 8-bit devices. The boot PROM is the only device on the board that resides in the memory space of the bus, and is only used in special applications.

Figure 28 illustrates the location of EtherCoupler board components, and Table 58 lists EtherCoupler board parts.

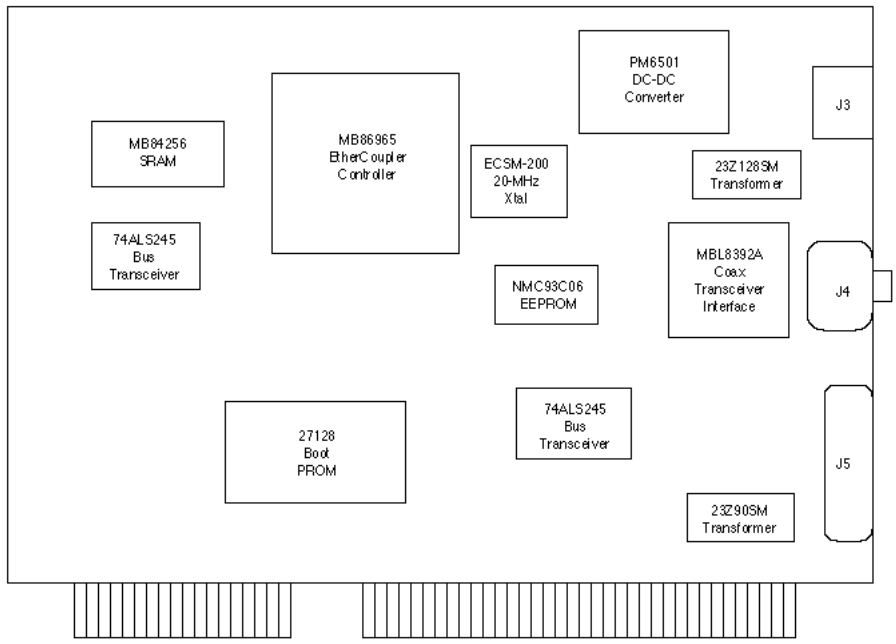


Figure 28. Ethercoupler Board Component Locations

Table 58. EtherCoupler Board Parts List

Item	Quantity	Reference	Value/PART	Vendor Part Number
1	3	C1, C4, C8	22 mF	
2	8	C2, C3, C5, C10, C11, C12, C13, C14	0.1 mF	
3	2	C6, C7	20 pF	
4	1	C9	10mF	
5	2	C15, C16	.001 mF	
6	1	C17	0.01 mF	
7	1	C18	0.01 mF	
8	2	DS1, DS2	LED	
9	1	F1	2.5 A	
10	1	J1	CON AT62	
11	1	J2	CON AT36	
12	1	J3	RJ45	
13	1	J4	BNC	
14	1	J5	CON DB15	
15	2	R1, R2	120 kilohms	
16	1	R3	12.4 kilohms ± 1%	
17	2	R4, R6	75 ohms ± 1%	
18	2	R5, R7	37.5 ohms ± 1%	
19	2	R8, R9	50 ohms ± 1%	
20	4	R10, R11, R12, R13	470 ohms	
21	1	R14	82 ohms	
22	4	R15, R16, R17, R18	1.5 kilohms	
23	1	R19	1 kilohm ± 1%	
24	1	R20	1 megaohm	
25	4	R21, R22, R23	78 ohms	
26	1	R24	60 ohms	
27	1	SA1	Spark Arrestor	
28	1	T1	Transformer	Fil-Mag 23Z90SM
29	1	T2	Transformer	Fil-Mag 23Z128SM
30	1	U1	SRAM	Fujitsu MB84256
31	2	U2, U7	Bus Transceiver	74ALS245
32	1	U3	EtherCoupler Controller	Fujitsu MB86965
33	1	U4	Boot PROM	27128
34	1	U6	EEPROM	NMC93C06
35	1	U8	10BASE2 Transceiver	Fujitsu MBL8392A
36	1	U9	DC-DC CONverter	Valor PM6501
37	1	Y1	20 MHz Crystal	Ecliptek ECSM-200

ETHERCOUPLER PC BOARD LAYOUT INSTRUCTIONS

Guidelines

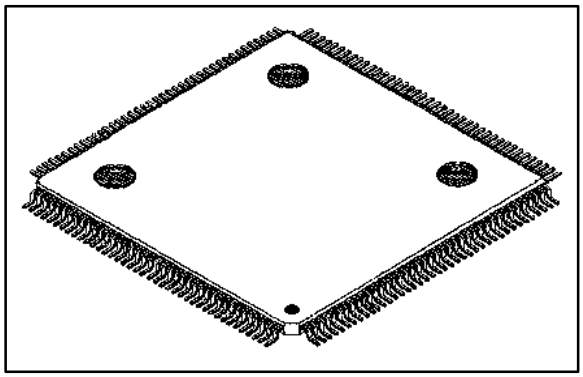
When laying out a PC board which uses the Fujitsu EtherCoupler chip set, the following guidelines should be observed:

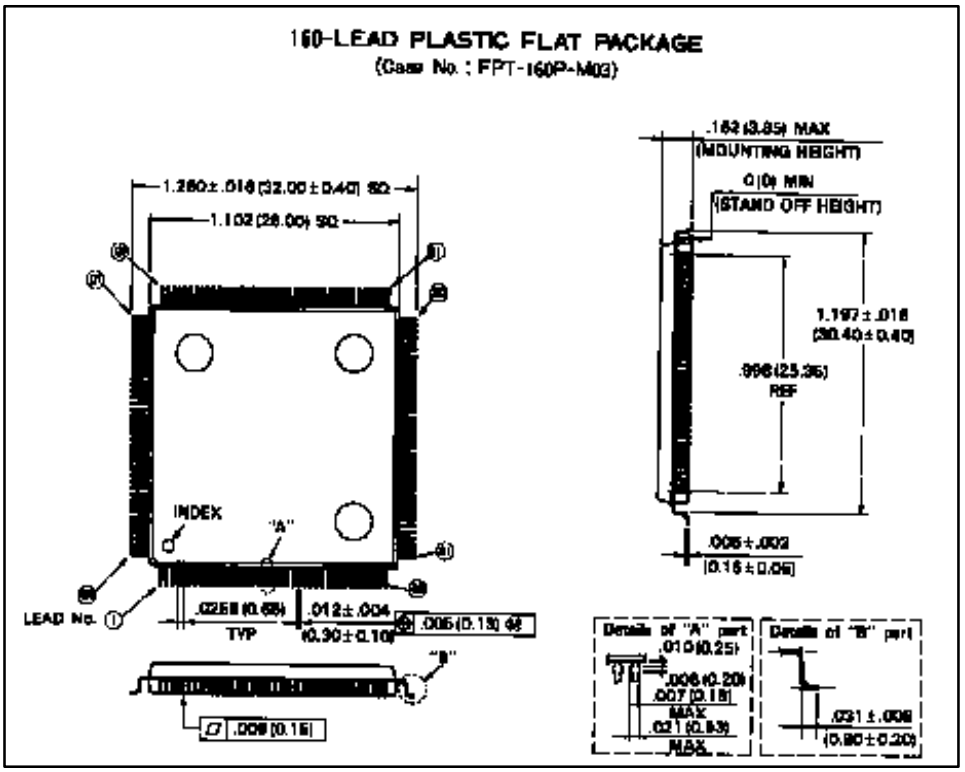
1. Use a 0.1-micro farad bypass capacitor at the top or bottom of every IC. Bypass 12V and 5V. Include several bypass capacitors for the MB86965; this devices has many output drivers. The trace length from the device VCC and ground pins to the associated bypass capacitor should be as short as possible.
2. Use several 22-mF tantalum filter capacitors on +5V, and space these evenly throughout the board.
3. For two layer boards, use a 50-mil traces for both VCC and ground. Be sure that they are well grounded throughout the board. (However, we don't recommend building a two-layer board)
4. Ensure that there are no traces or feedthroughs under crystals or crystal oscillators. Provide adequate clearance for components around crystals and crystal oscillators. Also, put a ground pad under the crystals and associated resistors and capacitors. Do not put a solder mask over the ground pad that is under the crystal.
5. Crystals and associated discrete components should be placed as close as possible to the corresponding oscillator IC.
6. Due to a lack of ground signals on the PC bus connector, the CHRESET signal is susceptible to crosstalk from the data bus. Two-layer boards are especially vulnerable to this problem. To eliminate glitches on this signal, use a 50-mil trace, put a ground trace next to it, and put a 0.1-mF capacitor near the CHRESET receiver. An RC filter next to the bus connector is recommended.
7. The RBIAS resistor is used as a reference by the MB86965's analog section. It should be 12.4 KΩ ± 1% tolerance, and care must be taken to tie it to a low-noise point on the ground plate.
8. Keep digital signals away from the Thinnet BNC signals.
9. The 8392 COAX Transceiver should be placed as close to the BNC connector as possible to minimize stray capacitance and inductance on the tap.

Checklist

1. Check IC pack placement.
2. Do a preliminary mechanical check for board size, connector locations, and mounting-hole locations.
3. Check the wire list.
4. Check the IC pin numbers on the schematics.
5. Verify that the above layout instructions have been followed.
6. Check pinouts of special devices on the layout (PLCCs).
7. Check orientation, pinouts, and pin spacing of all connectors.
8. Check the silkscreen.
9. Check the clearance of special components, e.g., connectors, capacitors, CPU.
10. Check the clearance around mounting holes.
11. Check mechanical dimensions shown in the layout specification drawing.
12. Check the orientation of polarized capacitors. The position of pin 1 should be consistent.
13. Check the orientation of three-pin jumpers. The position of pin 1 should be consistent.
14. Check the VCC and ground grid.
15. Visually inspect plots for each layer. Check for air gaps, pad sizes, and cross-net shorts.

ORDERING INFORMATION





Dimensions in inches (millimeters)