EXHIBIT D

to Declaration of Dr. Gregory L. Chesson in Support of Microsoft's Opposition to Alacritech's Motion for Preliminary Injunction

Protocol Engines Incorporated

Protocol Engine® Handbook

PE CHIPSET

PE Chipset

1. Features

- provides up to 200 Mbps end-to-end throughput
- hardware support for XTP, TCP, ISO, and XNS checksums
- programmable support for multiple protocols
- supports multicast, priority sorting, and other advanced features.

2. General Description

The Protocol Engine® chipset offers real-time protocol processing for high-speed networks. A wide range of cost-performance subsystem solutions are available through various configurations based on the PE Chipset. The chipset (shown in Figure 1) consists of four chips: MPORT, HPORT, BCTL, and CP. A basic configuration consists of MPORT, HPORT, and BCTL.

MPORT (MAC Port) provides an intelligent medium access control (MAC) layer interface to FDDI and other generic MACs. MPORT performs various tasks in its datapath, including checksumming, protocol identification, and address recognition. One to three MAC interfaces are supported by the architecture. A high-speed MAC, such as FDDI, requires a dedicated MPORT. Lower-speed MACs, such as Ethernet, can share an MPORT with the aid of external logic and buffering.

HPORT (Host Port) provides an intelligent direct memory access (DMA) engine with a synchronous bus interface (HBus) to the host. The HBus is compatible with industry standard buses (e.g., SBus, VME) and is easily interfaced to others. HPORT can interface to a data source (e.g., sensor) or a data sink (e.g., image buffer). One or two HPORT chips can be used in the PE subsystem. The architecture supports combinations of MPORT and HPORT chips up to a combined maximum of four.

BCTL (Buffer Controller) manages the external direct random access memory (DRAM) for buffering frames and storing state information. The chipset assumes the use of generic, page-mode DRAM chips.

CP (Control Processor) provides low latency, high performance processing for management functions, error recovery, and protocol processing functions not handled by MPORT, HPORT, or BCTL components. The multi-thread instruction architecture of this chip minimizes context switching and branching overhead, and maximizes processor efficiency.

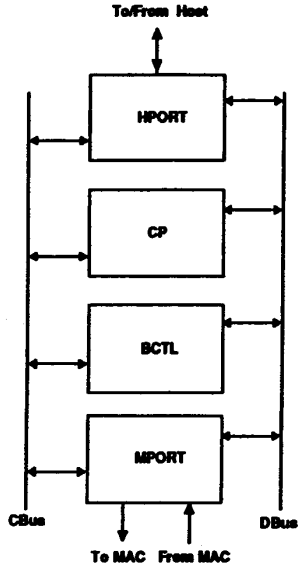

Figure 1. PE Chipset.

3. Basic Configuration

A basic configuration of the PE Chipset uses the HPORT, MPORT, and BCTL chips. (Figure 2 illustrates a basic configuration.) In addition to these three chips, two memory blocks are required: Control Memory (CM) and Network Buffer Memory (NBM). Control Memory is based on static, random access memory (SRAM); Network Buffer Memory is based on dynamic, random access memory (DRAM).

3.1 HPORT

Data arriving from the host is processed by Host Port (HPORT). HPORT packetizes the data into proper packet size and inserts proper values into the header/trailer fields (e.g., sequence number, checksum). The packets are temporarily stored in Network Buffer Memory before they are sent to the MAC layer by MPORT, when the next service opportunity arrives.

3.2 MPORT

MAC Port (MPORT) provides a duplex interface to one medium access control (MAC) device. Each received frame is processed appropriately depending on its address and protocol type. Various address types are supported, including local, broadcast, and multicast addresses, as well as a number of wild card conditions. MPORT executes checksumming, byte swapping, and other data manipulations, while data flows through the datapath. The header/trailer information is filtered to an on-chip Packet Processor. The packet processor is a programmable micro-engine. It processes the header/trailer data and decides the disposition of the packet.

The received packets are stored in Network Buffer Memory. A number of programmable modes are provided for deciding if and how packets will be further processed. "Raw data" mode causes packets to be delivered to the host via HPORT without further processing. This allows the use of existing protocol codes with minimum or no modification. Other modes activate further processing of the packets in order to achieve faster end-to-end throughput. The chipset acts as a fastpath protocol processor. Software must be provided (in CP, in the host, or in another nearby CPU) to do non-fastpath processing.

3.3 BCTL

The Buffer Controller (BCTL) both manages Network Buffer Memory and arbitrates access to it. BCTL generates DRAM control signals and address signals for reading from and writing to Network Buffer Memory. BCTL also performs periodic DRAM refresh cycles.

3.4 Control Memory

Control Memory (CM) is used for storing control information. It is accessed by the MPORT, HPORT, and BCTL chips for storing/retrieving various data structures, including tables, maps, control blocks, and command blocks. The locations where the data structures reside are programmable through on-chip registers. The structures are accessed through page registers and pointers.

Control Memory is accessible by words only. (Each word is 32 bits.) Optional byte parity can be included. The maximum Control Memory size supported by the chipset is 256k words. For an end-host with two thousand active connections, a minimum of 32k words of Control Memory is recommended.

3.5 Network Buffer Memory

Network Buffer Memory (NBM) is used to buffer network traffic. Both received frames and transmitting frames are buffered in Network Buffer Memory.

Like Control Memory, Network Buffer Memory is accessible by 32-bit words only. Network Buffer Memory can be constructed by 256kx4 or 1Mx4 DRAMs. It is organized by modules. Each module is 32-bits wide with four bits of optional byte parity. Up to four modules are supported in each bank of memory. Up to two banks are supported by this architecture. In a non-interleaved NBM (Network Buffer Memory) structure, one bank of memory is used. In an interleaved memory, two are used. The minimum Network Buffer Memory size is 256k words (1 MByte) and the maximum size is 8M words (32 MBytes).

In addition to being a temporary repository for frame data, Network Buffer Memory is also used for storing the bulk of context-related information for each connection. Various data structures reside in Network Buffer Memory to facilitate storing, organizing, and moving data. The data structures, their locations, and their sizes are programmable via the on-chip registers in MPORT, HPORT, and BCTL.

3.6 OPTIONS

BCTL can have optional Instruction Memory (BIM) connected to it directly. This memory provides an expansion area for firmware that overflows the on-chip instruction memory.

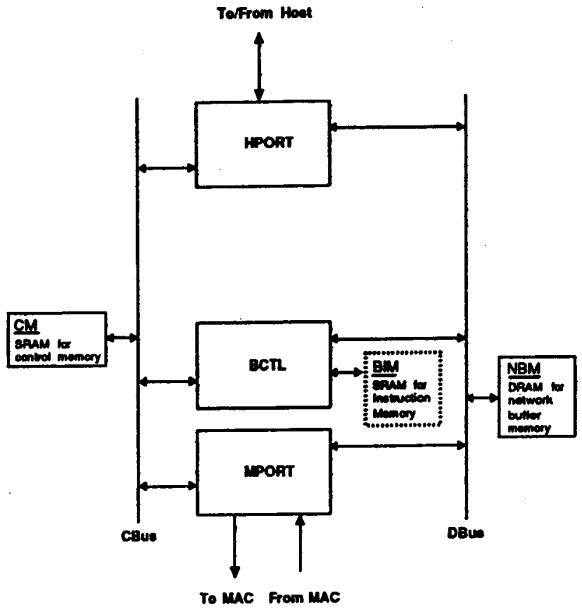

Figure 2. Basic Configuration.

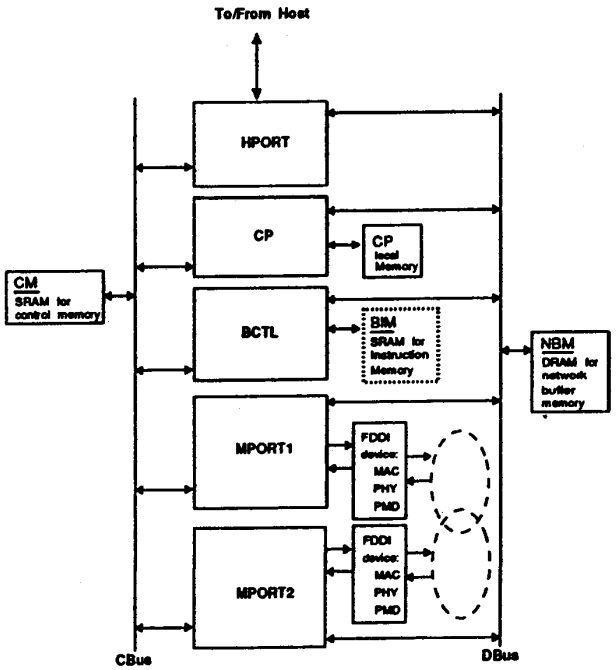

Figure 3. An Extended Configuration for Dual-attach FDDI.

4. Extended Configurations

A number of extended configurations are available. These involve adding a CP chip and/or increasing the number of MPORT and/or HPORT chips.

4.1 CP

Addition of a CP chip to the basic configuration provides significant performance enhancement. As shown in Figures 3, 4, and 5, CP is connected to CBus and DBus. These two buses provide CP direct access to the memories that store pertinent information for protocol processing. With multi-thread architecture, CP is able to attend to multiple processes without incurring context switching overhead. CP also has dedicated hardware to perform routine network processing functions.

Like the chips of the basic configuration, CP can own some of the data structures in Network Buffer Memory (NBM).

4.2 Two MPORTs

A second MPORT can be added to the basic configuration. It could be connected to a second MAC layer device that may be the same type as the first one, or may be different. This configuration could be used to construct a dual MAC, dual attach FDDI station in order to offer a higher level of redundancy. (See Figure 3.) Or, the second MPORT could be used as a gateway to connect two networks as illustrated in Figure 4. In this configuration, the packets received from one MAC go to Network Buffer Memory and then are delivered to the other MPORT, which transfers the data to the other MAC. The packets are not processed by HPORT and are not passed over the HBus, and hence do not require any host intervention. A third use for a second MPORT could be to construct a bridge. This could be done by connecting the second MPORT to a different type of MAC than the first.

4.3 Three MPORTs

A third MPORT can be added thus making it possible to implement any combination of the above. For example, a three MPORT system can be configured for both dual MAC FDDI and a bridge/gateway application.

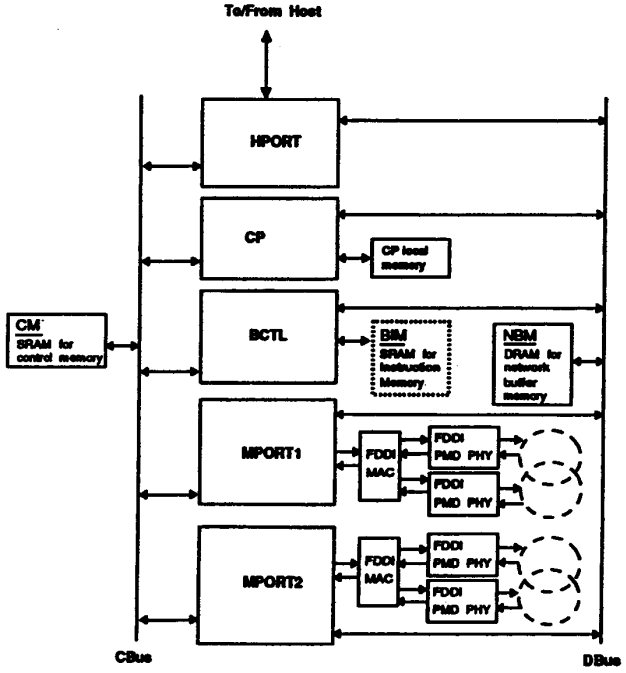

Figure 4. An Extended Configuration for an FDDI Gateway.

4.4 Two HPORTs

A second HPORT can be added to the basic configuration. This HPORT can be connected to a data sink or a data source, or both. Typical data sources include sensors and analog-to-digital converters with adequate buffers. A typical data sink is an image buffer. A typical data sink and source is a frame buffer or mass storage device. Data can be fed directly into the data sink/source device via the second HPORT without traversing the host bus. This significantly improves the data throughput to these devices.

The second HPORT could also be connected to the router backplane. This configuration provides high performance routing for high bandwidth network media. (See Figure 5.)

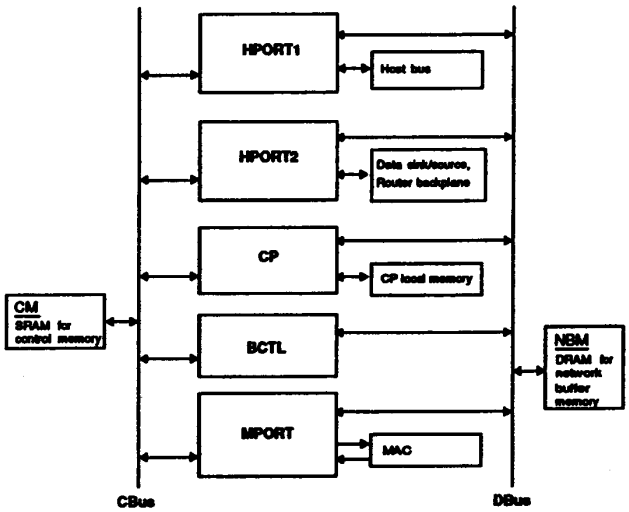

Figure 5. Extended Configuration with two HPORTs.

MPORT

1. Features

- one micron CMOS technology
- 33MHz system clock
- synchronization to external MAC device clock, up to 25MHz
- asynchronous MAC device clock and system clock
- two simplex, 9-bit channels: one to and one from MAC device
- handles peak data rate to 200M bits/s for each simplex channel
- separate, on-chip receiver and transmitter FIFOs
- 32-bit packet processing engine
- hardware support for XTP, TCP, ISO, and XNS checksums
- programmable support for multiple protocols

2. General Description

MAC Port (MPORT) of the Protocol Engine® chipset is a programmable controller that supports high-speed protocol processing. It provides hardware interfaces to the medium access control (MAC) device, the data bus (DBus), and the control bus (CBus).

The MAC interface provides a byte-wide, duplex data channel to the MAC device. This channel is operated by the MAC device clock, up to 25 MHz. The MAC device clock may be asynchronous to the PE Chipset's system clock.

Through the DBus interface, MPORT accesses Network Buffer Memory to store received frames and to retrieve frames for transmission.

Through the CBus interface, MPORT accesses Control Memory and sends/receives control messages to peer devices.

With these three interfaces, an on-chip programmable processor, and datapath functional units, MPORT performs checksumming, header processing, packet demultiplexing, and other functions. The user has the freedom to program a variety of protocols and policies, limited to the on-chip hardware resources.

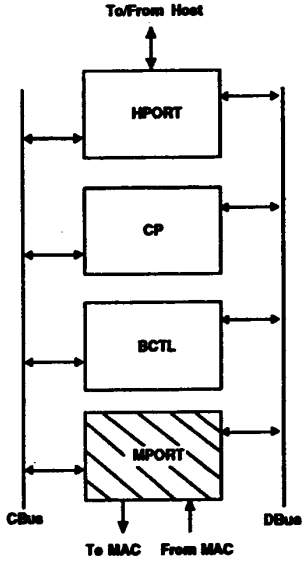

Figure 1. PE Chipset.

PIN	*Function*	*Type*	*# of pins*
CAD	CBus address/data	I&O	32
CDP	CBus address/data parity	I&O	4
$\overline{\text{CADS}}$	CBus address strobe	I&O	1
$\overline{\text{CRD}}$	CBus read	I&O	1
$\overline{\text{CWR}}$	CBus write	I&O	1
$\overline{\text{CRQ}}$	CBus request	O	1
$\overline{\text{CGT}}$	CBus grant	I	1
$\overline{\text{CERR}}$	CBus error	I&O	1
$\overline{\text{CSI}}$	CBus select in	I	1
$\overline{\text{CSCM}}$	CBus select out for CM (control memory)	O	1
$\overline{\text{CSBCTL}}$	CBus select out for BCTL (buffer controller)	O	1
$\overline{\text{CSCP}}$	CBus select out for CP (control processor)	O	1
DDATA	DBus data	I&O	32
DDP	DBus data parity	I&O	4
DTG	DBus tag	I&O	4
$\overline{\text{DRQ}}$	DBus request	O	1
$\overline{\text{DGT}}$	DBus grant	I	1
$\overline{\text{DWR}}$	DBus write	O	1
MCLK	MAC device clock	I	1
MDO	MAC data output	O	8
MPO	MAC parity output	O	1
MDI	MAC data input	I	8
MPI	MAC parity input	I	1
others	TBD		
CLK	Clock	I	1
$\overline{\text{RESET}}$	Reset	I	1
$\overline{\text{TIS}}$	Test internal scan	I	1
TCK	Test clock	I	1
TMS	Test mode select	I	1
TDI	Test data in	I	1
TDO	Test data out	O	1
TRST	Test reset	I	1

Table 1. MPORT Pin Summary.

CBus Pins

Symbol	I/O	Description
CAD[31:0]	I/O	Address/data bus. It is tri-stated when CBus is not granted. When CBus is granted, address is driven onto this bus during address cycle, and data is driven onto this bus during data cycle. The data is driven by MPORT in a write transfer, or it is driven by a slave device in a read transfer.
CDP[3:0]	I/O	Byte parity for the CAD[31:0] bus. CDP has the same timing as CAD[31:0]. CDP[3] covers the CAD[31:24], and so on.
$\overline{\text{CADS}}$	I/O	Address strobe. When CBus is granted, it is an output. Its assertion indicates that address is being driven onto CAD[31:0] by MPORT. When CBus is not granted, it is an input. Its assertion along with the assertion of $\overline{\text{CSI}}$ indicates that CAD[31:0] carries address driven by the current CBus master to address a slave register on MPORT.
$\overline{\text{CRD}}$	I/O	Read strobe. When CBus is granted, it is an output. Its assertion indicates a read transfer. When CBus is not granted, it is an input. Its assertion indicates that the current CBus master is making a read transfer.
$\overline{\text{CWR}}$	I/O	Write strobe. When CBus is granted, it is an output. Its assertion indicates a write transfer. When CBus is not granted, it is an input. Its assertion indicates that the current CBus master is making a write transfer.
$\overline{\text{CRQ}}$	O	CBus request. This signal is asserted to request CBus mastership. It may remain asserted to request multiple transactions for a bus tenure.
$\overline{\text{CGT}}$	I	CBus grant. This input indicates the granting of CBus mastership.
$\overline{\text{CERR}}$	I/O	CBus error. This pin is both input and output. The open drain, active low output reports CBus error to the peer devices which have $\overline{\text{CERR}}$ pins connected together. The input circuitry detects CBus errors reported by any device connected on CBus.
$\overline{\text{CSI}}$	I	Chip select input. The assertion of this pin indicates that the current CBus master is making a transaction with MPORT, as a slave device.
$\overline{\text{CSCM}}$	O	Control Memory select. It is tri-stated when CBus is not granted. It is asserted when CBus is granted and MPORT is addressing Control Memory.
$\overline{\text{CSBCTL}}$	O	BCTL select. It is tri-stated when CBus is not granted. It is asserted when CBus is granted and MPORT is addressing BCTL.
$\overline{\text{CSCP}}$	O	CP select. It is tri-stated when CBus is not granted. It is asserted when CBus is granted and MPORT is addressing CP.

DBus Pins

Symbol	I/O	Description
DDATA[31:0]	I/O	Data bus. It is tri-stated when DBus is not granted. When DBus is granted, data is driven onto this bus in a write transaction to the Network Buffer Memory. Data is driven by the Network Buffer Memory in a read transaction.
DDP[3:0]	I/O	Byte parity for the DDATA[31:0]. It has the same timing as DDATA[31:0]. DDP[3] covers the DDATA[31:24], and so on.
DTG[3:0]	I/O	Tags. These tag bits are tri-stated when DBus is not granted. When DBus is granted, they are outputs in a write transaction, driven with the same timing as DDATA[31:0]. These tag bits are sensed by BCTL to determine the data types and the status on DDATA[31:0]. They are inputs in a read transaction. MPORT senses these tag bits to determine the data types and status on DDATA[31:0], indicated by BCTL.
$\overline{\text{DRQ}}$	O	DBus request. This signal is asserted to request DBus mastership. It may remain asserted to request multiple transactions for a bus tenure.
$\overline{\text{DGT}}$	I	DBus grant. This input indicates the granting of DBus mastership.
$\overline{\text{DWR}}$	O	DBus write strobe. This signal, when negated, indicates the requested DBus transactions are reads. When asserted, it indicates the requested DBus transactions are writes.

MAC Interface Pins

Symbol	I/O	Description
MCLK	I	MAC clock. This clock is used to synchronize with the external MAC device. It may be asynchronous to the system clock, CLK. The on-chip dual-clock FIFO synchronizes MCLK and CLK.
MDO[7:0]	O	Data output to MAC device.
MPO	O	Parity for the data output to MAC device.
MDI[7:0]	I	Data input from MAC device.
MPI	I	Parity for the data input from MAC device.
TBD		The control signals for the MAC device interface are to-be-determined.

System Control Pins

Symbol	I/O	Description
CLK	I	System clock. This input signal provides synchronization for the on-chip circuitry.
$\overline{\text{RESET}}$	I	System reset. This input signal resets on-chip circuitry to a known state. It should remain asserted for at least 512 CLK cycles to assure proper reset of the chip.

Test Pins

Symbol	I/O	Description
$\overline{\text{TIS}}$	I	Test internal scan. Assertion of this signal selects the internal scan test.
TCK	I	Test clock. This input signal is used to synchronize on-chip testing circuitry.
TMS	I	Test mode select. This signal is used to select among various test modes.
TDI	I	Test data input. This pin is used for input test data.
TDO	O	Test data output. This pin is used for output test data.
TRST	I	Test reset. This pin is used to reset test modes.

Power Pins

Symbol	I/O	Description
VCC		Power pins.
GND		Ground pins.

4. Functional Description

MPORT (MAC Port) transfers frames between the external MAC device and Network Buffer Memory (NBM). When the input frame passes through MPORT, processing is done in the datapath. This processing includes checksumming, byte swapping, header/trailer parsing, protocol identification, demultiplexing, etc.

The processing is done in various functional blocks that are illustrated in Figure 3.

There are six major functional blocks in MPORT: MAC Interface Unit (MIU), Receive Pipeline Unit, Transmit Pipeline Unit, DBus Interface Unit (DIU), Packet Processing Unit (PPU) and CBus Interface Unit (CIU).

4.1 MAC Interface Unit (MIU)

The MAC Interface Unit contains the state machines to interface to the external MAC devices. It has an eight bit data input bus and an eight bit data output bus to connect to the MAC. Both buses have parity bits. The parity can be optionally turned off. The control signals interfacing to the MAC device have not been finalized.

4.2 Receive Pipeline Unit

The Receive Pipeline Unit has a single directional data flow. The data flow starts from MIU. It is fed into a Receive MAC FIFO. The data coming out of the Receive MAC FIFO flows into a Receive Datapath, then goes to a Receive DBus FIFO. The data coming out of Receive DBus FIFO goes to a DBus Interface Unit (DIU). DIU will store the data into Network Buffer Memory via DBus.

The Receive MAC FIFO is operated by both MCLK and CLK. MCLK is used to feed data into the FIFO and CLK is used to retrieve data from FIFO. This scheme allows the MAC device, running on MCLK, to be asynchronous to the system clock, CLK.

The data, embodied in frames, flows into the Receive Datapath to obtain a series of processing steps. The data are word aligned, padded with fill-pattern if necessary, byte swapped if necessary to become big-endian. These processing steps are controlled by a Proto Parser. Proto Parser is a programmable processor. Its micro program is down-loaded at boot time. It can be programmed to adapt to various media and various protocols. Proto Parser counts the bytes in frames and examines their header to determine further action. Proto Parser recognizes the protocol used by the packet in the frame, extracts certain information, and delivers it to the Packet Processing Unit.

A Checksummer validates the checksum on the packet flowing through the Receive Pipeline Unit. Packet Processor firmware determines the actions to be taken in the event of checksum failure or other failure.

4.3 Transmit Pipeline Unit

Like the Receive Pipeline Unit, the Transmit Pipeline Unit is a single directional datapath. The data flow starts from DIU. The data flows into a Transmit DBus FIFO, through a Transmit Datapath and a Transmit MAC FIFO and ends at the MAC Interface Unit (MIU).

The Transmit MAC FIFO is similar to the Receive MAC FIFO with opposite data flow direction. The data flow into the FIFO is clocked by CLK and the data flow out of the FIFO is clocked by MCLK to provide synchronization between two clock systems.

4.4 DBus Interface Unit (DIU)

The DBus Interface Unit contains state machine for DBus control. It performs burst read/write on the DBus to maximize DBus efficiency. Two operating modes are available: the non-interleaved mode and the interleaved mode.

The non-interleaved mode works with non-interleaved Network Buffer Memory to deliver 320 Mbps bandwidth on DBus. The interleaved mode works with interleaved Network Buffer Memory to deliver 700 Mbps bandwidth on DBus.

4.5 Packet Processing Unit (PPU)

The Packet Processing Unit contains a Packet Processor and Instruction Memory. The program for the Packet Processor is down-loaded into Instruction Memory at boot time. The Packet Processor processes the information received from the Receive Pipeline Unit. As a result of the processing in Packet Processor, the receiving packets are demultiplexed to each context. They are also properly disposed according to the information retrieved from the header/trailer in the packet.

4.6 CBus Interface Unit (CIU)

The CBus Interface Unit interfaces to CBus to access the Control Memory and other devices in the PE Chipset. The Packet Processor may access the data

base stored in Control Memory, or request slave access to the other devices attached on CBus. These transactions provide information for the Packet Processor to decide the disposition of the packet being processed, updates information in Control Memory to facilitate processing for the following packets, and receives and provides commands to peer devices.

HPORT

HPORT

1. Features

- one micron CMOS technology
- 33MHz system clock
- synchronization to external host bus clock, up to 25MHz
- asynchronous host bus clock and system clock
- separate receiver and transmitter internal FIFOs
- 32-bit host interface with parity
- performs checksum generation, packetization, byte swapping, word alignment, and other functions
- hardware support for XTP, TCP, ISO, and XNS checksums
- programmable support for multiple protocols
- supports both big- and little-endian hosts.

2. General Description

Host Port (HPORT) of the Protocol Engine® chipset is a programmable controller that supports high-speed protocol processing. It provides three external hardware interfaces: host bus (HBus), data bus (DBus), and control bus (CBus).

Through HBus, HPORT provides DMA and slave interfaces to the host bus or to a dedicated bus connecting to a data sink/source device, such as image buffer and sensor. HBus is operated by a separate clock, HCLK, to synchronize to the host bus.

Through DBus, HPORT accesses Network Buffer Memory to store output packets, to retrieve received packets, and to update the buffer management data structures.

Through CBus, HPORT accesses Control Memory and sends/receives control messages to peer devices. It also provides a datapath for the host to make slave accesses to HPORT's peer devices.

Through these buses, an on-chip programmable processor, and the datapath functional unit, the HPORT performs checksum generation, packetization, intelligent DMA, and other functions. HPORT can support a variety of protocols and custom host interfaces.

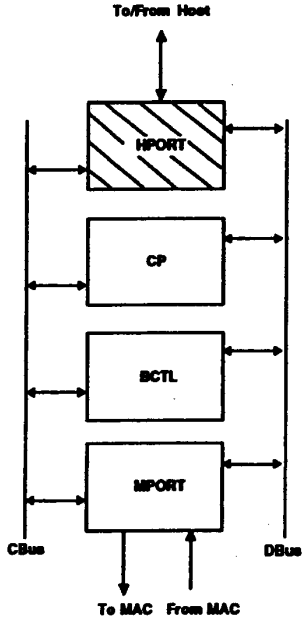

Figure 1. PE Chipset.

3. Pin Description

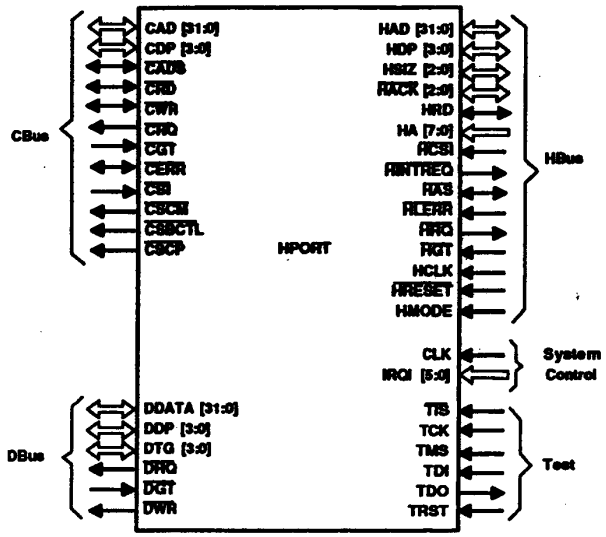

Figure 2. HPORT Symbol Diagram.

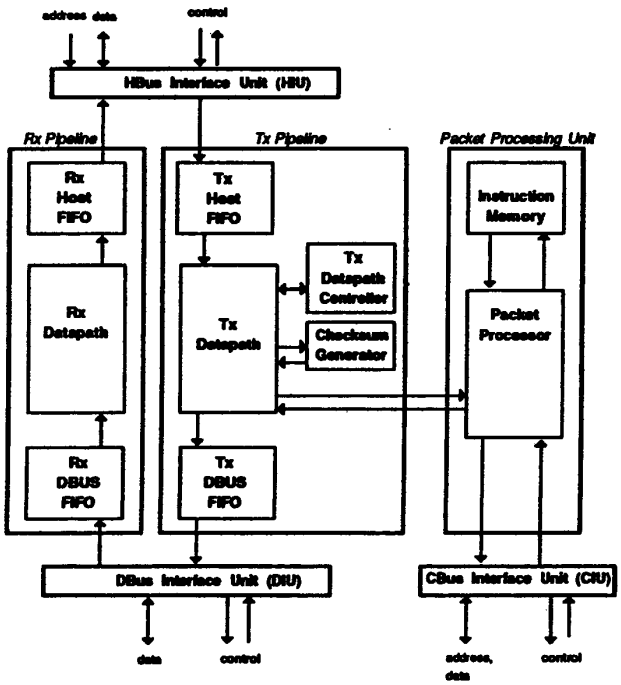

Figure 3. HPORT Block Diagram.

PIN	*Function*	*Type*	*# of Pins*
CAD	CBus address/data	I&O	32
CDP	CBus address/data parity	I&O	4
$\overline{\text{CADS}}$	CBus address strobe	I&O	1
$\overline{\text{CRD}}$	CBus read	I&O	1
$\overline{\text{CWR}}$	CBus write	I&O	1
$\overline{\text{CRQ}}$	CBus request	O	1
$\overline{\text{CGT}}$	CBus grant	I	1
$\overline{\text{CERR}}$	CBus error	I&O	1
$\overline{\text{CSI}}$	CBus select in	I	1
$\overline{\text{CSCM}}$	CBus select out for CM (control memory)	O	1
$\overline{\text{CSBCTL}}$	CBus select out for BCTL (buffer controller)	O	1
$\overline{\text{CSCP}}$	CBus select out for CP (control processor)	O	1
DDATA	DBus data	I&O	32
DDP	DBus data parity	I&O	4
DTG	DBus tag	I&O	4
$\overline{\text{DRQ}}$	DBus request	O	1
$\overline{\text{DGT}}$	DBus grant	I	1
$\overline{\text{DWR}}$	DBus write	O	1
HAD	HBus data.	I&O	32
HDP	HBus address/data parity	I&O	4
HSIZ	HBus size.	I&O	3
$\overline{\text{HACK}}$	HBus acknowledgement	I&O	3
HRD	HBus read	I&O	1
HA	HBus slave address	I	8
$\overline{\text{HCSI}}$	HBus chip select in	I	1
$\overline{\text{HINTREQ}}$	HBus interrupt request	O	1
$\overline{\text{HAS}}$	HBus address strobe	I&O	1
$\overline{\text{HLERR}}$	HBus late error	I	1
$\overline{\text{HRQ}}$	HBus request	O	1
$\overline{\text{HGT}}$	HBus grant	I	1
HCLK	HBus clock	I	1
$\overline{\text{HRESET}}$	HBus reset	I	1
HMODE	HBus mode	I	1
CLK	Clock	I	1
IRQI	Interrupt request in	I	6
$\overline{\text{TIS}}$	Test internal scan	I	1
TCK	Test clock	I	1
TMS	Test mode select	I	1
TDI	Test data in	I	1
TDO	Test data out	O	1
TRST	Test reset	I	1

Table 1. HPORT Pin Summary.

CBus Pins

Symbol	*I/O*	*Description*
CAD[31:0]	I&O	Address/data bus. It is tri-stated when CBus is not granted. When CBus is granted, address is driven onto this bus during address cycle, and data is driven onto this bus during data cycle. The data is driven by HPORT in a write transfer, or it is driven by a slave device in a read transfer.
CDP[3:0]	I/O	Byte parity for the CAD[31:0] bus. It has the same timing as the CAD[31:0]. CDP[3] covers the CAD[31:24], and so on.
$\overline{\text{CADS}}$	I/O	Address strobe. When CBus is granted, it is an output. Its assertion indicates that address is being driven onto CAD[31:0] by HPORT. When CBus is not granted, it is an input. Its assertion along with the assertion of $\overline{\text{CSI}}$ indicates that CAD[31:0] carries address driven by the current CBus master to address a slave register on HPORT.
$\overline{\text{CRD}}$	I/O	Read strobe. When CBus is granted, it is asserted to indicate a read transfer. When CBus is not granted, it is an input. Its assertion indicates that the current CBus master is making a read transfer.
$\overline{\text{CWR}}$	I/O	Write strobe. When CBus is granted, it is asserted to indicate a write transfer. When CBus is not granted, it is an input. Its assertion indicates that the current CBus master is making a write transfer.
$\overline{\text{CRQ}}$	O	CBus request. This signal is asserted to request CBus mastership. It may remain asserted to request multiple transactions for a bus tenure.
$\overline{\text{CGT}}$	I	CBus grant. This input indicates the granting of CBus mastership.
$\overline{\text{CERR}}$	I/O	CBus error. This pin is both input and output. The open drain, active low output reports CBus error to the peer devices which have $\overline{\text{CERR}}$ pins connected together. The input circuitry detects CBus errors reported by any device connected on CBus.
$\overline{\text{CSI}}$	I	Chip select in. The assertion of this pin indicates that the current CBus master is making a transaction with HPORT, as a slave device.
$\overline{\text{CSCM}}$	O	Control Memory select. It is tri-stated when CBus is not granted. It is asserted when CBus is granted and HPORT is addressing Control Memory.
$\overline{\text{CSBCTL}}$	O	BCTL select. It is tri-stated when CBus is not granted. It is asserted when CBus is granted and HPORT is addressing BCTL.
$\overline{\text{CSCP}}$	O	CP select. It is tri-stated when CBus is not granted. It is asserted when CBus is granted and HPORT is addressing CP.

DBus Pins

Symbol	*I/O*	*Description*
DDATA[31:0]	I/O	Data bus. It is tri-stated when DBus is not granted. When DBus is granted, data is driven onto this bus in a write transaction to the Network Buffer Memory. Data is driven by the Network Buffer Memory in a read transaction.
DDP [3:0]	I/O	Byte parity for the DDATA[31:0]. It has the same timing as DDATA[31:0]. DDP[3] covers the DDATA[31:24], and so on.
DTG[3:0]	I/O	Tags. These tag bits are tri-stated when DBus is not granted. When DBus is granted, they are outputs in a write transaction, driven with the same timing as DDATA[31:0]. These tag bits are sensed by BCTL to determine the data types and the status on DDATA[31:0]. They are inputs in a read transaction. HPORT senses these tag bits to determine the data types and status on DDATA[31:0], indicated by BCTL.
$\overline{\text{DRQ}}$	O	DBus request. This signal is asserted to request DBus mastership. It may remain asserted to request multiple transactions for a bus tenure.
$\overline{\text{DGT}}$	I	DBus grant. This input indicates the granting of DBus mastership.
$\overline{\text{DWR}}$	I	DBus write strobes. This signal, when negated, indicates the requested DBus transactions are reads. When asserted, it indicates the requested DBus transactions are writes.

HBus Pins

Symbol	*I/O*	*Description*
HAD[31:0]	I/O	HBus address/data. Data and addresses are transferred over these lines. When HPORT is HBus master, these lines are inputs in reads and outputs in writes for data transfers. When HPORT is HBus slave, they are the other way around.
HDP[3:0]	I/O	HBus data byte parity. These signals provide byte parity for HAD[31:0]. HDP[3] covers HAD[31:24], and so on. They can be optionally turned off.
HSIZ[2:0]	I/O	HBus size. These encoded lines indicate the transfer size of HAD[31:0]. They are outputs when HPORT is HBus master. They are inputs when it is not. The coding of these signals are as follows. *HSIZ[2:0]* — *Function* 000 — Word (four byte) transfer 001 — Byte transfer 010 — Half-word (two byte) transfer 011 — Reserved 100 — Four word burst (16 bytes) 101 — Eight word burst (32 bytes) 110 — Sixteen word burst (64 bytes) 111 — Two word burst (8 bytes)

Continued on next page

HBus Pins continued

Symbol	*I/O*	*Description*
$\overline{\text{HACK}}$ [2:0]	I/O	HBus acknowledgment. These encoded acknowledgment signals are used to terminate HBus cycles. They are inputs when HPORT is HBus master. They are outputs when it is not. The coding of these signals are as follows. $\overline{\text{HACK}}$ [2:0] Function 000 Idle/wait 001 Error acknowledgment 010 Byte (data) acknowledgment 011 Rerun acknowledgment 100 Word (data) acknowledgment 101 Reserved 110 Half-word (data) acknowledgment 111 Reserved
HRD	I/O	HBus read. This signal indicates HBus data transfer direction. When it is asserted, it indicates a read cycle. Else, it indicates a write cycle. This line is an output when HPORT is HBus master. It is input when HPORT is not.
HA[7:0]	I	HBus slave address. These address lines presents physical addresses during slave cycles.
$\overline{\text{HCSI}}$	I	HBus slave select. This signal, when asserted and qualified by $\overline{\text{HAS}}$, indicates that the current HBus master is accessing HPORT as a slave device.
$\overline{\text{HINTREQ}}$	O	HBus interrupt. This line is used to signal host CPU that interrupt conditions occurred.
$\overline{\text{HAS}}$	I	HBus address strobe. This input signal indicates that HA[7:0] and $\overline{\text{HCSI}}$ are in valid state. The assertion of $\overline{\text{HCSI}}$ and $\overline{\text{HAS}}$ causes HPORT to respond as a slave.
$\overline{\text{HLERR}}$	I	HBus late error. When HPORT is HBus master, the assertion of this line indicates the addressed slave device is acknowledging a bus error.
$\overline{\text{HRQ}}$	O	HBus request. This signal is asserted to request HBus mastership. Once asserted, it must be left asserted until grant is received.
$\overline{\text{HGT}}$	I	HBus grant. This signal indicates the granting of HBus mastership.
HCLK	I	HBus clock. This signal synchronizes to the external host bus. The maximum clock rate is 25 MHz. It can be asynchronous to the Protocol Engine system clock, CLK.
$\overline{\text{HRESET}}$	I	System reset. This input signal resets on-chip circuitry to a known state. It should remain asserted for at least 512 HCLK cycles to assure proper reset of the chip.
HMODE	I	HBus mode. This signal selects HBus modes. When it is strapped to VCC, the HBus behaves in conformance to SBus specification. When it is strapped to GND, it behaves as a generic host bus.

System Control Pins

Symbol	*I/O*	*Description*
CLK	I	System clock. This input signal provides synchronization for the on-chip circuitry.
IRQI[5:0}	I	Interrupt request in. These signals detect interrupt conditions in the system.

Test Pins

Symbol	*I/O*	*Description*
$\overline{TIS}$	I	Test internal scan. Assertion of this signal selects the internal scan test.
TCK	I	Test clock. This input signal is used to synchronize on-chip testing circuitry.
TMS	I	Test mode select. This signal is used to select among various test modes.
TDI	I	Test data input. This pin is used for shifting input test data.
TDO	O	Test data output. This pin is used for shifting output test data.
TRST	I	Test reset. This pin is used to reset test modes.

Power Pins

Symbol	*I/O*	*Description*
VCC		Power pins.
GND		Ground pins.

4. Functional Description

HPORT transfers data between Network Buffer Memory and a host memory or data sink/source device, such as a scanner or a frame buffer. Data passing through HPORT is processed in several ways. On output to a network, the processing includes packet header/trailer generation, checksum insertion, sequence number insertion, byte swapping, word alignment, and padding. For received packets, the processing includes byte swapping, word alignment and header/ trailer stripping. Many of these functions can be optionally turned off.

This processing is performed in various functional blocks as illustrated in the HPORT block diagram (Figure 3).

There are six major functional blocks in HPORT: HBus Interface Unit (HIU), Receive Pipeline Unit, Transmit Pipeline Unit, DBus Interface Unit (DIU), Packet Processing Unit (PPU) and CBus Interface Unit (CIU).

4.1 HBus Interface Unit (HIU)

The HBus Interface Unit contains the state machines to interface to an external host bus (HBus) or data sink/source device. HIU has a 32-bit bi-directional data bus. There are byte parity bits for the data bus, which can be disabled. HPORT has both master and slave interfaces to the HBus. An external HBus controller is required to arbitrate the HBus mastership.

The HIU master interface does burst mode DMA on HBus with various burst sizes to get high throughput with low HBus occupancy. An intelligent DMA controller further enhances DMA efficiency.

The HIU slave interface is used to initialize HPORT and other peer devices in Protocol Engine subsystem. It is also used to down-load instructions to the on-chip RAMs in the subsystem.

4.2 Transmit Pipeline Unit

The Transmit Pipeline Unit has a single directional data flow. The data flow starts from HIU and is fed into a Transmit HBus FIFO. The data coming out of the Transmit HBus FIFO flows into a Transmit Datapath, then goes to a Transmit DBus FIFO. The data coming out of Transmit Dbus FIFO goes to a DBus Interface Unit (DIU). DIU stores the data into Network Buffer Memory via DBus.

The Transmit HBus FIFO is operated by both HCLK and CLK. HCLK is used to feed data into the FIFO and CLK is used to retrieve data from FIFO. This scheme allows the host or the data sink/source device, running on HCLK, to be asynchronous to the system clock, CLK.

The transmitting data are obtained by DMA read cycles. A user definable data structure, Host Control Block (HCB), is used to determine the size and the location of the data blocks within host memory. The degree of packetization of data in the host varies from fully packetized (i.e., header/trailer are appended, with correct checksum and sequence number) to raw data. The Transmit Pipeline Unit performs the remaining packetization process based on the information obtained through the Host Control Block.

A Transmit Datapath Controller controls the data flow within the datapath. It manages the insertion of header/trailer information, the injection of sequence number, checksum and other information to the packets. A checksum generator generates checksum for the transmitting packets.

4.3 Receive Pipeline Unit

Like the Transmit Pipeline Unit, Receive Pipeline Unit is a single directional datapath. The data flow starts from the DBus Interface Unit. The data flows into a Receive DBus FIFO, through a Receive Datapath and a Receive HBus FIFO and ends at HIU.

The Receive HBus FIFO is similar to the Transmit HBus FIFO, with opposite data flow direction. The data flow into the FIFO is clocked by CLK and the data flow out of the FIFO is clocked by HCLK to provide synchronization between two clock systems.

The data going through Receive Datapath will be byte-swapped and word-aligned if necessary.

4.4 DBus Interface Unit (DIU)

The DBus Interface Unit contains the state machine for DBus access. It performs burst read/write on the DBus to maximize DBus efficiency. There are two operating modes available: non-interleaved mode and the interleaved mode.

The non-interleaved mode works with non-interleaved Network Buffer Memory to deliver 320 Mbps bandwidth on DBus. The interleaved mode works with interleaved Network Buffer Memory to deliver 700 Mbps.

4.5 Packet Processing Unit (PPU)

The Packet Processing Unit consists of a Packet Processor and an Instruction Memory. The programs for the Packet Processor are down-loaded into the Instruction Memory at boot time. The Packet

Processor manages the Host Control Block and other HPORT-owned data structures. It also executes host interface policies to achieve high efficiency on the host bus and to make minimum interrupt to the host.

4.6 CBus Interface Unit (CIU)

The CBus Interface Unit interfaces to CBus to access the Control Memory (CM) and other devices in the PE Chipset. Packet Processor uses this interface to access the data structures stored in CM and to access the slave registers in the peer devices. This interface is also used to down-load micro code to peer devices.

Protocol Engines
Incorporated

PRELIMINARY
(subject to change without notice)

BCTL

1. Features

- one micron CMOS technology
- 33MHz system clock
- programmable selection of DRAM: either 256kx4 or 1Mx4
- programmable selection of DRAM operation: either interleaved or non-interleaved
- optional, external, additional instruction memory, up to 32k x 24 bits.
- support for wide range of Network Buffer Memory: 1 to 32 MBytes
- provides DRAM control functions
- 32-bit on-chip Buffer Control Processor
- on-chip instruction and data memory
- manages data structures in Network Buffer Memory

2. General Description

Buffer Controller (BCTL) of the Protocol Engine® chipset is a programmable controller that provides memory management functions for Network Buffer Memory to achieve high-speed protocol processing. It provides external hardware interfaces to four buses: DBus, CBus, MBus, and BBus. It alsoarbites for DBus access.

Through the DBus interface, BCTL accesses Network Buffer Memory to update memory management data structures.

Through the CBus interface, BCTL accesses Control Memory and sends/receives control messages to peer devices.

Through the MBus interface, BCTL provides access (for other devices) to Network Buffer Memory.

Through the BBus, BCTL provides access to additional instruction memory.

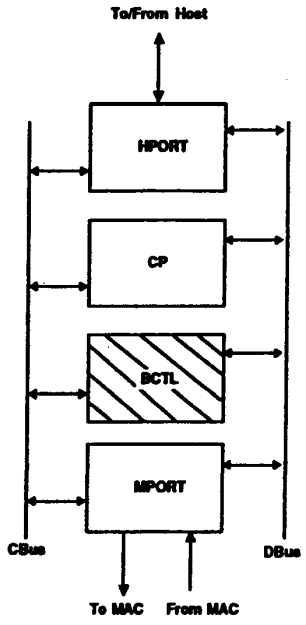

Figure 1. PE Chipset.

3. Pin Description

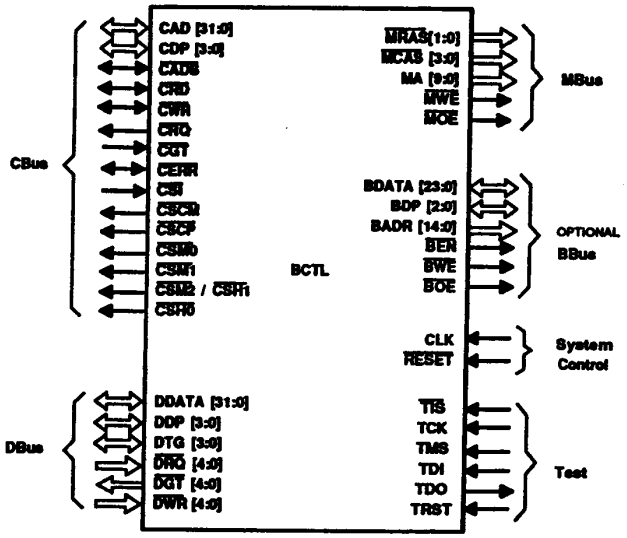

Figure 2. BCTL Symbol Diagram.

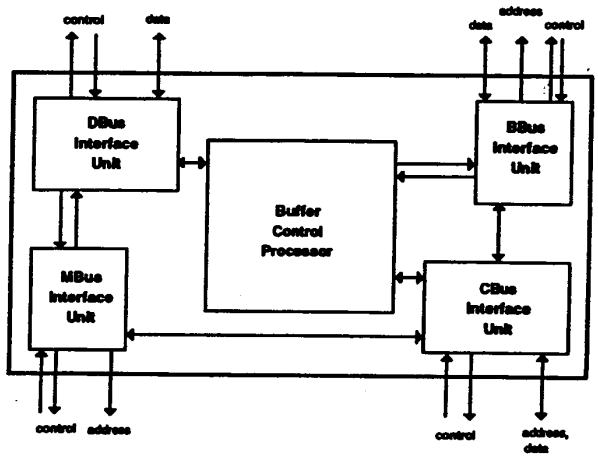

Figure 3. BCTL Block Diagram.

PIN	*Function*	*Type*	*# of pins*
CAD	CBus address/data	I&O	32
CDP	CBus address/data parity	I&O	4
$\overline{\text{CADS}}$	CBus address strobe	I&O	1
$\overline{\text{CRD}}$	CBus read	I&O	1
$\overline{\text{CWR}}$	CBus write	I&O	1
$\overline{\text{CRQ}}$	CBus request	O	1
$\overline{\text{CGT}}$	CBus grant	I	1
$\overline{\text{CERR}}$	CBus error	I&O	1
$\overline{\text{CSI}}$	CBus select in	I	1
$\overline{\text{CSCM}}$	CBus select out for CM (control memory)	O	1
$\overline{\text{CSCP}}$	CBus select out for CP (control processor)	O	1
$\overline{\text{CSM0}}$	CBus select out for first MPORT	O	1
$\overline{\text{CSM1}}$	CBus select out for second MPORT	O	1
$\overline{\text{CSM2/CSH1}}$	CBus select out for third MPORT or second HPORT	O	1
$\overline{\text{CSH0}}$	CBus select out for first HPORT	O	1
DDATA	DBus data	I&O	32
DDP	DBus data parity	I&O	4
DTG	DBus tag	I&O	4
$\overline{\text{DRQ}}$	DBus request	O	5
$\overline{\text{DGT}}$	DBus grant	I	5
$\overline{\text{DWR}}$	DBus write	O	5
CLK	Clock	I	1
$\overline{\text{RESET}}$	Reset	I	1
$\overline{\text{TIS}}$	Test internal scan	I	1
TCK	Test clock	I	1
TMS	Test mode select	I	1
TDI	Test data in	I	1
TDO	Test data out	O	1
TRST	Test reset	I	1
$\overline{\text{MRAS}}$	Row address strobe	O	2
$\overline{\text{MCAS}}$	Column address strobe	O	4
MA	DRAM addresses	O	10
$\overline{\text{MWE}}$	Write enable	O	1
$\overline{\text{MOE}}$	Output enable	O	1
BDATA	BBus data	I&O	24
BDP	BBus byte parity	I&O	3
BADR	BBus word address	O	15
$\overline{\text{BEN}}$	BIM enable	O	1
$\overline{\text{BWE}}$	BIM write enable	O	1
$\overline{\text{BOE}}$	BIM output enable	O	1

Table 1. BCTL Pin Summary.

CBus Pins

Symbol	*I/O*	*Description*
CAD[31:0]	I/O	Address/data bus. It is tri-stated when CBus is not granted. When CBus is granted, address is driven onto this bus during an address cycle, and data is driven onto this bus during an data cycle. The data is driven by BCTL in a write transfer, or it is driven by a slave device in a read transfer.
CDP[3:0]	I/O	Byte parity for the CAD[31:0] bus. It has the same timing as the CAD[31:0]. CDP[3] covers the CAD[31:24], and so on.
$\overline{\text{CADS}}$	I/O	Address strobe. When CBus is granted, it is an output. Its assertion indicates that address is being driven onto CAD[31:0] by BCTL. When CBus is not granted, it is an input. Its assertion along with the assertion of $\overline{\text{CSI}}$ indicates that CAD[31:0] carries address driven by the current CBus master to address a slave register on BCTL.
$\overline{\text{CRD}}$	I/O	Read strobe. When CBus is granted, it is asserted to indicate a read transfer. When CBus is not granted, it is an input. Its assertion indicates that the current CBus master is making a read transfer.
$\overline{\text{CWR}}$	I/O	Write strobe. When CBus is granted, it is asserted to indicate a write transfer. When CBus is not granted, it is an input. Its assertion indicates that the current CBus master is making a write transfer.
$\overline{\text{CRQ}}$	O	CBus request. This signal is asserted to request CBus mastership. It may remain asserted to request multiple transactions for a bus tenure.
$\overline{\text{CGT}}$	I	CBus grant. This input indicates the granting of CBus mastership.
$\overline{\text{CERR}}$	I/O	CBus error. This pin is both input and output. The open drain, active low output reports CBus error to the peer devices which have $\overline{\text{CERR}}$ pins connected together. The input circuitry detects CBus errors reported by any device connected on CBus.
$\overline{\text{CSI}}$	I	Chip select. The assertion of this pin indicates that the current CBus master is making a transaction with BCTL, as a slave device.
$\overline{\text{CSCM}}$	O	Control Memory select. It is tri-stated when CBus is not granted. It is asserted when CBus is granted and BCTL is addressing Control Memory.
$\overline{\text{CSCP}}$	O	CP select. It is tri-stated when CBus is not granted. It is asserted when CBus is granted and BCTL is addressing CP.

Continued on next page

CBus Pins (continued)

Symbol	*I/O*	*Description*
$\overline{\text{CSM0}}$	O	MPORT select. It is tri-stated when CBus is not granted. It is asserted when CBus is granted and BCTL is addressing the first instance of MPORT. One to three instances of MPORT chips can be used in a Protocol Engine System. This pin is always used to connect to one of them.
$\overline{\text{CSM1}}$	O	MPORT select. It is tri-stated when CBus is not granted. It is asserted when CBus is granted and BCTL is addressing the second instance of MPORT. One to three instances of MPORT chips can be used in a Protocol Engine System. This pin is used to connect to one of them if two or three MPORT chips exist in the system.
$\overline{\text{CSM2}}$ /$\overline{\text{CSH1}}$	O	MPORT select or HPORT select. It is tri-stated when CBus is not granted. It is asserted when CBus is granted and BCTL is addressing the third instance of MPORT or the second instance of HPORT. Either the third instance of the MPORT or the second instance of HPORT can exist is a Protocol Engine system, but not both.
$\overline{\text{CSH0}}$	O	HPORT select. It is tri-stated when CBus is not granted. It is asserted when CBus is granted and BCTL is addressing the first instance of HPORT. One or two instances of HPORT chips can be used in a Protocol Engine System. This pin is always used to connect to one of them.

DBus Pins

Symbol	*I/O*	*Description*
DDATA[31:0]	I/O	DBus data. It is tri-stated when DBus is not granted. When DBus is granted, data is driven onto this bus in a write transaction to the Network Buffer Memory. Data is driven by the Network Buffer Memory in a read transaction.
DDP[3:0]	I/O	Byte parity for the DDATA[31:0]. It has the same timing as DDATA[31:0]. DDP[3] covers the DDATA[31:24], and so on.
DTG[3:0]	I/O	DBus Tags. These tag bits are tri-stated when DBus is not granted to any DBus master. When DBus is granted to a DBus master other than BCTL itself, they are inputs in a write transaction, driven with the same timing as DDATA[31:0] by the current DBus master. BCTL senses these tag bits to determine the data types and the status on DDATA[31:0]. These tag bits are outputs in a read transaction while BCTL granted DBus mastership to a peer device. BCTL presents the data types and status of the DBus by driving these lines. The encoding of these tags bit is to-be-determined.

Continued on next page

DBus Pins (continued)

Symbol	*I/O*	*Description*
$\overline{\text{DRQ}}$ [4:0]	I	DBus requests. These signals are asserted by the peer DBus masters to request DBus mastership. It may remain asserted to request multiple transactions for a bus tenure. The assignments of these lines to the peer DBus masters are as follows. $\overline{\text{DRQ}}$ [0]: the first instance of MPORT $\overline{\text{DRQ}}$ [1]: the second instance of MPORT $\overline{\text{DRQ}}$ [2]: the third instance of MPORT or the second instance of HPORT $\overline{\text{DRQ}}$ [3]: the first instance of HPORT $\overline{\text{DRQ}}$ [4]: CP BCTL requests DBus mastership via an on-chip signal, since it is the DBus arbiter itself.
$\overline{\text{DGT}}$ [4:0]	O	DBus grants. These outputs indicate the granting of DBus mastership. Similar assignments as those of DRQ[4:0] are used. As the DBus arbiter, BCTL grants DBus mastership to itself via an on-chip signal.
$\overline{\text{DWR}}$ [4:0]	I	DBus write strobes. These signals, when negated, indicate the requested DBus transactions are reads. When asserted, they indicate the requested DBus transactions are writes. Similar assignments as those of DRQ[4:0] are used. As the DBus arbiter, BCTL indicate the write requests or the read requests via on-chip signals.

MBus Pins

Symbol	*I/O*	*Description*
$\overline{\text{MRAS}}$ [1:0]	O	Row address strobe. These signals are used to drive the RAS lines of the DRAMs in the NBM. $\overline{\text{MRAS}}$ [0]: Low Bank $\overline{\text{MRAS}}$ [1]: High Bank
$\overline{\text{MCAS}}$ [3:0]	O	Column address strobe. These lines are used to drive the CAS lines of the DRAMs in the NBM. The assignments of these lines are as follows. $\overline{\text{MCAS}}$ [0]: Module 0 $\overline{\text{MCAS}}$ [1]: Module 1 $\overline{\text{MCAS}}$ [2]: Module 2 $\overline{\text{MCAS}}$ [3]: Module 3
MA[9:0]	O	DRAM addresses. When 256Kx4 DRAM is used, MA[9] should be left open.
$\overline{\text{MWE}}$	O	Write enable. This signal is used to drive the $\overline{\text{WE}}$ lines of the DRAMs in the NBM.
$\overline{\text{MOE}}$	O	Output enable. This signal is used to drive the $\overline{\text{OE}}$ lines of the DRAMs in the NBM.

BBus Pins

Note: The existence of the BBus in the first generation of BCTL is to-be-determined.

Symbol	*I/O*	*Description*
BDATA[23:0]	I/O	BBus data. This pins are outputs during BBus writes. They are inputs during BBus reads.
BDP[2:0]	I/O	BBus byte parity bits. BDP[2] covers BDATA[23:16], and so on.
BADR[14:0]	O	BBus word address. These address lines address up to 32K words of BCTL Instruction Memory (BIM).
$\overline{\text{BEN}}$	O	BIM enable. When this line is asserted, BIM is enabled.
$\overline{\text{BWE}}$	O	BIM write enable. This signal is used to drive the $\overline{\text{WE}}$ lines of the SRAMs in the BIM.
$\overline{\text{BOE}}$	O	BIM output enable. This signal is used to drive the $\overline{\text{OE}}$ lines of the SRAMs in the BIM.

System Control Pins

Symbol	*I/O*	*Description*
CLK	I	System clock. This input signal provides synchronization for the on-chip circuitry.
$\overline{\text{RESET}}$	I	System reset. This input signal resets on-chip circuitry to a known state. It should remain asserted for a least 512 CLK cycles to assure proper reset of the chip.

Test Pins

Symbol	*I/O*	*Description*
$\overline{\text{TIS}}$	I	Test internal scan.
TCK	I	Test clock. This input signal is used to synchronize on-chip testing circuitry.
TMS	I	Test mode select. This signal is used to select among various test modes.
TDI	I	Test data input. This pin is used for shifting input test data.
TDO	O	Test data output. This pin is used for shifting output test data.
TRST	I	Test reset. This pin is used to reset test modes.

Power Pins

Symbol	*Description*
VCC	Power pins.
GND	Ground pins.

4. Functional Description

The Buffer Controller (BCTL) manages queue structures in Network Buffer Memory (NBM) and arbitrates DBus. The on-chip 32-bit Buffer Control Processor can be programmed to implement different Network Buffer Memory algorithms.

There are five major functional units in BCTL. They are shown in Figure 3. These units are: DBus Interface Unit, CBus Interface Unit, MBus Interface Unit, BBus Interface Unit, and Buffer Control Processor.

4.1 DBus Interface Unit (DIU)

The DBus Interface Unit contains two major blocks: a DBus Master Block and a DBus Arbiter Block.

The DBus Master Block contains the state machine for DBus control. The request/grant handshake is performed with the signals between the this block and the DBus Arbiter Block.

The DBus Master Block performs burst read/write on the DBus to maximize DBus efficiency. There are two operating modes are available: the non-interleaved mode and the interleaved mode.

The non-interleaved mode works with non-interleaved Network Buffer Memory to deliver 320 Mbps bandwidth on DBus. The Interleaved Mode works with interleaved Network Buffer Memory to deliver 700 Mbps.

The DBus Arbiter Block receives six bus request signals, of which five are from the off-chip peer devices and one is from the on-chip DBus Master Block. The DBus Arbiter executes a fair and efficient arbitration scheme, which is a combination of round-robin, priority and conditional refusal. The acknowledgement of the bus grant is through six grant signals. Like the bus request, five of the grant signals go to the peer devices and one goes to the DBus Master Block.

4.2 CBus Interface Unit (CIU)

The CBus Interface Unit interfaces to CBus to access the Control Memory (CM) and other peer devices in the PE Chipset. Also, through this interface, other peer devices can access the BCTL slave registers, on-chip RAM and the external BCTL Instruction Memory.

BCTL reads information within Control Memory to determine proper action. It writes Control Memory to update predefined data structure to facilitate the peer devices to recognize the state changes in the system.

4.3 MBus Interface Unit (MIU)

The MBus Interface Unit contains state machines to generate DRAM addresses and timing control signals. It executes various DRAM cycles to allow random access, page mode access, non-interleaved memory access, interleaved memory access, and refresh.

MIU contains a number of memory address registers. These registers can be loaded by Buffer Control Processor or by the peer devices through CIU.

4.4 BBus Interface Unit (BIU)

The BBus Interface Unit interfaces to the external BCTL Instruction Memory (BIM). BIM facilitates the applications requiring instruction memory larger than the on-chip one for the Buffer Control Processor. The BBus interface is a generic SRAM interface to provide ease of system applications.

4.5 Buffer Control Processor (BCP)

The Buffer Control Processor is a 32-bit, special purpose processor. It is programmable to provide flexibility in defining data structures in Network Buffer Memory, in setting service policies, and in tuning performance. There are separate instruction and data memories in the unit. If the on-chip instruction memory is insufficient for the micro program, the external BIM can be used. BIM is not required for a basic configuration.

BCP has an instruction set including integer arithmetic operations, logical operations, data movement operations and program control operations. Certain operations are tied closely with the on-chip hardware resources to achieve high performance. A number of special hardware units are used to execute simple routine operations without requiring constant supervision from the processor.

CP

CP

1. Features

- one micron CMOS technology
- 33MHz system clock
- multi-thread 32-bit processor core
- multi-bank register file
- integrated CBus and DBus interface
- dedicated hardware for routine functions
- SRAM and DRAM external interface
- on-chip programmable timers

2. General Description

Control Processor (CP) of the Protocol Engine® chipset is a 32-bit, multi-thread execution unit that provides high speed protocol processing.

CP can execute multiple threads concurrently. This increases parallelism and reduces context switching overhead. Processes can also be prioritized. A hardware scheduler can preempt a low-priority thread with a higher priority thread in response to programmable hardware events.

CP is designed to extend the basic packet-handling abilities of MPORT, HPORT, and BCTL by adding additional processing capability to the packet handling pipeline. Address and connection management, flow, error, and rate control are examples of CP functions.

The CP achieves a high integration factor by combining both CBus and DBus interfaces on chip. This permits efficient communication with the other chips. The on-chip instruction memory can be configured as a cache or as dedicated RAM.

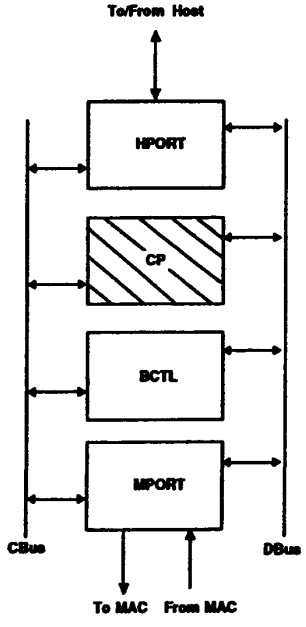

Figure 1. PE Chipset.

3. Pin Description

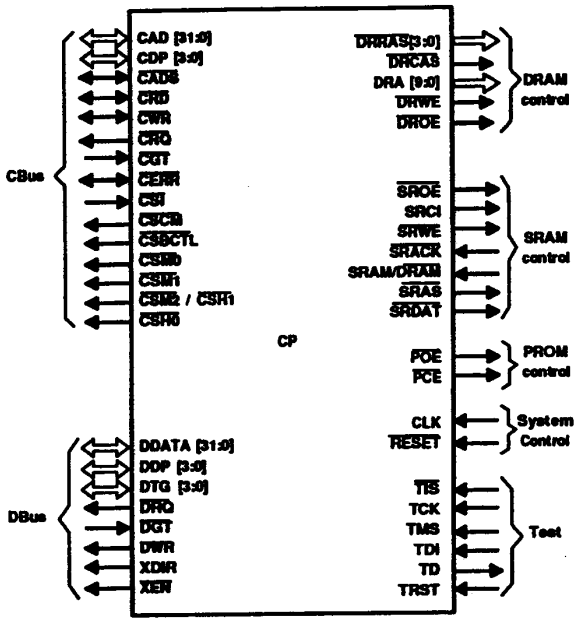

Figure 2. CP Symbol Diagram.

CBUS

1. Introduction

CBus facilitates interdevice communication within the Protocol Engine® chipset as well as communication with Control Memory (CM).

2. General Description

CBus is designed to connect up to eight CBus devices. The CBus masters access the slave interfaces and Control Memory (CM). Control Memory is static RAM based.

A CBus arbiter arbitrates the CBus mastership. The maximum CBus clock rate is 33 MHz.

3. CBus Cycles

The timing diagrams for CBus cycles are not available at the time this document is written.

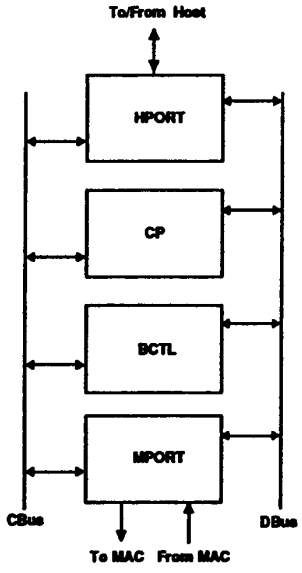

Figure 1. PE Chipset.

4. Signals

Signal	*Description*
CAD[31:0]	CBus address/data bus. This is a tri-state bus. During an address cycle, it is driven by the current bus master with address to select the Control Memory or the register locations within the selected device. During a data cycle, it is driven by the current bus master or the selected slave device with data. During a wait cycle, it is not driven by any device.
CDP[3:0]	CBus data parity. These signals have the same timing as CAD[31:0]. CDP[3] covers CAD[31:24], and so on.
$\overline{\text{CADS}}$	CBus address strobe. It is a tri-state, active low signal, driven by the current bus master. It is asserted when a valid address is being driven onto CAD[31:0] by the master.
$\overline{\text{CRD}}$	CBus read strobe. It is a tri-state, active low signal, driven by the current bus master. It is asserted to provide timing reference for a read transfer.
$\overline{\text{CWR}}$	CBus write strobe. It is a tri-state, active low signal, driven by the current bus master. It is asserted to provide timing reference for a write transfer.
$\overline{\text{CRQ}}$ [5:0]	CBus request. These active low signals are driven by the bus masters to request CBus mastership. Each signal is used by one bus master. Up to six bus masters are supported. They may remain asserted to request multiple transactions for a bus tenure. They are inputs for the CBus arbiter.
$\overline{\text{CGT}}$ [5:0]	CBus grant. These active low signals are driven by the CBus arbiter. They are inputs for the corresponding CBus masters. At any given time, at most one of these lines may be asserted by the arbiter. The assertion of these signals indicates the corresponding bus master is granted bus mastership.
$\overline{\text{CERR}}$	CBus error. It is an open drain signal from CBus devices. External pull-up device is used to bring the line up to high voltage level. Any CBus device asserting this signal indicates the occurrence of CBus error.
$\overline{\text{CSEL}}$ [7:0]	CBus device select. These are tri-state signals driven by the current bus master. They are inputs for the corresponding CBus slave devices.

DBUS

DBUS

1. Introduction

DBus provides a burst data transfer between Network Buffer Memory in the Protocol Engine® subsystem and the devices in the Protocol Engine chipset. Other devices that conform with the DBus specification may be connected to the bus.

2. General Description

DBus accommodates up to six DBus masters. The Protocol Engine chipset incorporates an integrated DBus arbiter within BCTL. BCTL supports up to five DBus masters in addition to itself.

DBus is used by its masters to access Network Buffer Memory (NBM) which is the only slave device on Dbus. It is assumed that Network Buffer Memory is direct, random-access memory (DRAM) based. The DBus arbiter generates bus grant signals according to DRAM cycle timing. It also provides page mode cycles to access this memory efficiently.

There are two modes of DBus: Non-interleaved Mode and Interleaved Mode. In Non-interleaved Mode, data transfer happens every two clocks, excluding overhead. In Interleaved Mode, it happens every clock.

3. DBus Cycles

The timing diagrams for DBus cycles are not available at the time this document is written.

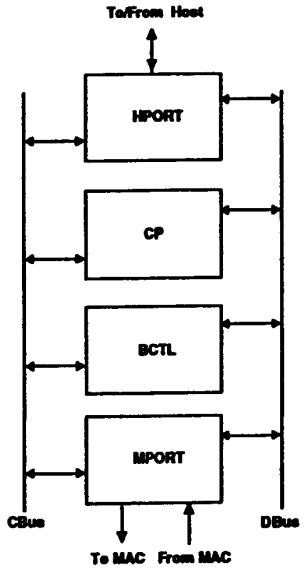

Figure 1. PE Chipset.

4. Signals

Signal	*Description*
DDATA[31:0]	Tri-state bi-directional data bus. Active high. The data is driven by a current DBus Master in a write cycle. It is driven by Network Buffer Memory in a read cycle.
DDP[3:0]	Tri-state bi-deirectional byte-parity for the 32-bit DDATA. Active high. DDP[3] covers DDATA[31:24], and so on. They have the same timing as DDATA[31:0]. They should be valid when DDATA[31:0] is valid.
DTG[3:0]	Tri-state bi-directional tag bits. Active high. It has the same timing as DDATA[31:0]. It should be valid when DDATA[31:0] is valid. The coding of these tag bits is to be determined.
$\overline{\text{DRQ}}$ [4:0]	DBus Request. Active low signal. They are outputs driven by the DBus Masters. They are inputs to DBus arbiter. A DBus Master asserts its request line when it needs DBus transaction. Once asserted, the request line should be left asserted until the transaction is granted. DBus master can keep asserting the request line after the bus is granted to get a burst of transactions on DBus.
$\overline{\text{DGT}}$ [4:0]	DBus grant. Active low signal. They are outputs driven by DBus arbiter. They are inputs for the corresponding DBus masters.
$\overline{\text{DWR}}$ [4:0]	DBus write. Active low signal. They are outputs driven by the DBus Masters. They are inputs to DBus arbiter. When asserted, they indicate that the DBus request is for a write. When negated, they indicate it is for a read. The timing is the same as $\overline{\text{DRQ}}$ [4:0]. They make state transition when the corresponding $\overline{\text{DRQ}}$ lines do.