# Designing with Cypress QuadPort™ (CY7C0430BV) Backplane Switch

## Introduction

The QuadPort™ is much more than a simple bus matching tool. It is a data transport enabler which gives a designer maximum flexibility in creating a data path architecture which will maximize the throughput of the system while reducing the complexity and cost of design. This application note will explore the QuadPort and its uses in the following sections:

I) Introduction to QuadPorted Memories

II) Enhancing Communications designs in a

A) SONET Wide Area Network Design

B) Frame Relay WAN Design

C) ATM WAN Design

D) Storage Area Network Design

III) Introduction to Quad Port arbitration

The QuadPort is a revolutionary step in the development of specialty memory products. It is the first step in migrating the specialty memory product line from simple data storage to data path management.

## Why design a QuadPorted Memory?

Dual Ported or FIFO memories have been traditionally used in the communications industry as a data speed matching tool. These memories were used as simple data buffers which stored data coming from a high speed bus until a slower bus was ready for the data. This is shown in *Figure 1*. Because the busses were much slower than they are today, the data could be stored and processed without impacting the overall throughput of the system.

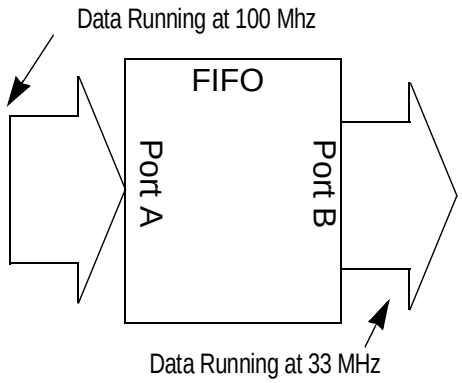


**Figure 1. Data Speed Matching.**

In today's high data rate communications systems, optimal throughput can not be maintained if the processing of the data can not be done at system speed. This means that the data must be manipulated as it moves through the system, and requires that either the data is processed at least twice as fast as the overall system throughput or that extensive buffering is designed into the system.

In order to accomplish this, most communications designers today utilize an ASIC or an FPGA. As the data throughput continues to increase, the size of these FPGAs/ASICs have also grown to require 1 Million gates or more. To optimize the system throughput, most of these gates are being used either to buffer the data or manage the data path. This is an expensive solution and is a poor use of an FPGA, whose main task is to evaluate and act on information inside the communications packet. If the processing element can be removed from the data path, the designer can greatly reduce the size and complexity of the ASIC/FPGA. The QuadPort was designed with this in mind.

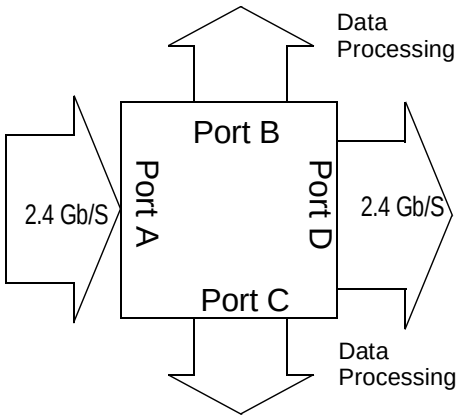


**Figure 2. The Quad Port.**

In designing the QuadPort, Cypress has taken a radically different approach to the function of specialty memory. The QuadPort is not so much a memory device as it is a data transport vehicle. Its multiple ports allow a network processor to act upon the data independent of the data path. The multiple ports also gives a designer the flexibility to create divergent data paths with the QuadPort at its nexus like a 3:1 Multiplexer. Memory now becomes an integral part of the communications pipeline allowing a designer to reduce the number of the devices required in the system and reduce the size of the ASIC/FPGAs used in a system.

The QuadPort provides a unique challenge to a communications engineer because it gives the engineer so much flexibility that it merits re-evaluating the entire data path design. The effort will be more than worthwhile because designing with a QuadPort can reduce the complexity and cost of a design. The following sections will describe how to use the QuadPort in some common applications.

## Using the Quad Port in a Communications Design

A router/switch consists of a series of network cards. Each network card consists of two interfaces: A port side and a backplane side. *Figure 3* illustrates a generic network card.

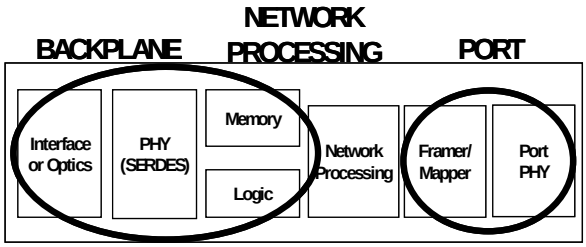


**Figure 3. Block diagram of a typical network card.**

The port side of the network card is where data is sent or received across a dedicated media link (PHY device) to the outside world. Next to the PHY device is a framer; the framer byte aligns the data as it comes across serial link and presents the data in packet format across its parallel side to the network processor.

The backplane side of the network card is generally used to transport data from one network card to another internal to the router. This is generally done with a proprietary, LVDS, or fibre channel bus. There is a movement to evolve the interface to Infiniband or 10 Gigabit Ethernet in the future. The SERDES device draws the data in parallel from the processing block and encodes it based upon the selected protocol for delivery across the backplane as a serial stream.

Between these two sides of the card sits the network processor which performs the primary tasks of the router. Usually there is a specialty memory device that buffers the data between the processor and the two ends. This is required because the network processor runs at speed different from the communications paths attached to either end of the network card. The speed of the data though has forced the migration of the processor into an ASIC/FPGA with an integrated FIFO memory.

The reason for the integration of a FIFO into the ASIC is that the network processor sits directly in the data path. Because of this all of the packet data must be read into, processed, and retransmitted by the network processor at speed to ensure that there is not a break in the transmission path. Since most processors do not run at speed the integration of the buffer into the ASIC allows the processor time to act on the data. If the processor could be removed from the data path, this extra memory is no longer needed and the system can be optimized.

The network processor's primary functions are logical inter connectivity, route calculation/maintenance, and security. Functions that do not require the network processor to handling the packet payload directly. Most of these calculations are done on the packet header and so there is no need for the processor to store or handle the entire packet.

Since the processor is not required to handle the entire package, removing it from the data path could lead to substantial reduction in the size and complexity of the network processor ASIC/FPGA. Which translates to savings in part cost and design cycle time. For example, a network processor which must handle a entire OC-48 frame will have to handle 48 STS-1 frames which consist of 6480 bits per frame. The total memory needed than is 304K bits. If the data is not completely processed before the next frame enters the device, you need to buffer more than one frame. This could change the amount of memory needed to 608K or more. This is a substantial portion of a 1 million gate FPGA just to store the data before you can operate on it. Additional logic is also required to manage and maintain the data path.

The QuadPort can significantly reduce the amount of memory needed in the network processor. The QuadPort can interface directly with a SERDES device or with a network framer. Creating a data channel through the device. The two additional ports on the QuadPort can be used to interface with one or more network processors. This removes the processor from the data path but still allows it access to process the data. *Figure 4* is an illustration of the revised system. The following sections will discuss specific applications using *Figure 4* as the basis for design.

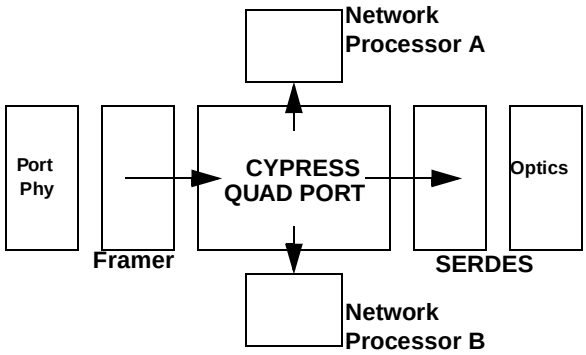


**Figure 4. Quad Port Network Card.**

### SONET Transport

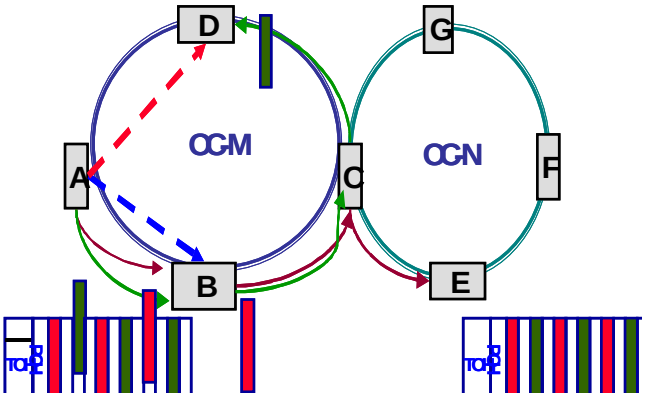


**Figure 5. SONET Packet Processing.**

A typical SONET system consists of two optical rings. An active and redundant data path. A Sonet packet is sent in one direction along the ring and only if the ring is broken does the network switch to the redundant path.

Each packet consists of multiple virtual tributaries (VTs). A virtual tributary is a dedicated channel in which an end customer can place data whether it is in IP, ATM, or Frame Relay format.

Each VT is time division multiplexed and added to the packet as it is sent serially along the SONET ring. At each node, the VTs inside the packet can be added or deleted. Only those customers who have paid for that virtual channel can change the data in the VT. Figure #5 shows a SONET packet as VTs are added and deleted along the data path at each node.

A packet is deleted if the node recognizes that the data in the packet is intended for an address inside the node. The VT is then processed to remove the data in the packet and send it to an end machine in the LAN. The Quad Port can be used to minimize the amount of circuitry required to perform the Add Drop Multiplexor (ADM) function and aid in the processing of the VT itself.

*The QuadPort as an Add Drop Multiplexer*

Since SONET is a TDM based transmission method, it is easy to integrate the QuadPort into the design. The QuadPort has a burst counter feature that allows the user to set a start address and as long as the CNTINC# pin is held low data will be written into the QuadPort on each clock. There is no additional address cycle. The address is automatically incremented for as long as CNTINC# is held low. Each VT time slice can have its own address range within the QuadPort with an external PLD handling the switching between VTs based upon a pre-determined timing scheme. When the entire packet has been written into the QuadPort, a processor can examine the contents of the packet and read out the data intended for an internal address or can write data into that address range. When the processor is done with the packet, the entire packet can be burst out of the QuadPort and sent along the ring. Since the VT channels are predetermined, the address control can be very simple and the processor will know which address to read its data from based upon a pre-programmed scheme.

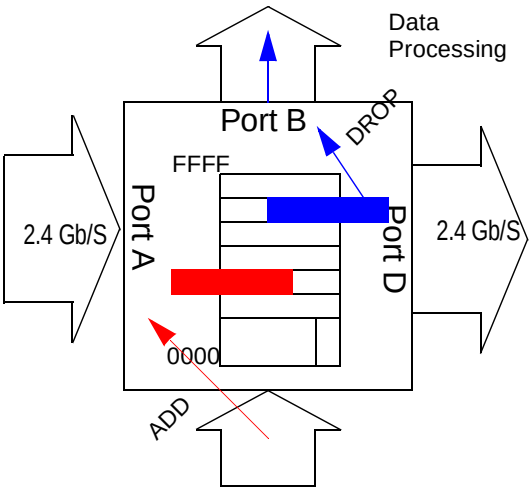


**Figure 6. The Quad Port as an ADM.**

As you can see from *Figure 6*, very little is done to the actual SONET payload other than adding and removing the VT frames. There is simply no need to read the entire packet into the network processor. The processor only needs to know what address range a specific VT is assigned. It can then read or write VTs to that location thus performing the Add/Drop function with minimal circuitry.

*The QuadPort as part of a packet processing solution*

The QuadPort can do much more than just act as an Add Drop Multiplexor. The QuadPort's density is sufficient to allow the storage of multiple SONET packets. So as data comes into the Quad Port, the processor can not only add/drop VTs but can actually index into the VT itself. This can be done using a Ternary Content Addressable Memory or network processor. Both choices can handle the table lookups required to parse the VT and route its payload to its final destination.

In the case of an IP packet, the processor has to be able to read the packet header out of the QuadPort, compare the network IP address inside the address to a routing table, and then address a second port to send the packet across its backplane. This is illustrated in *Figure 7*.

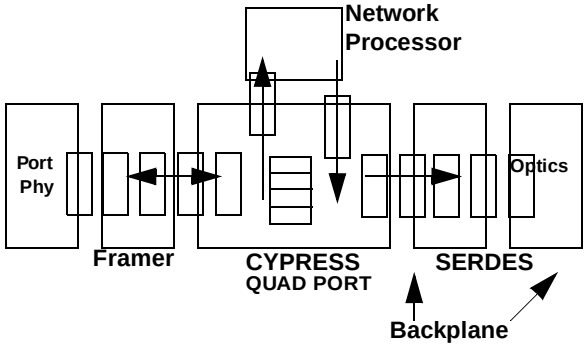


**Figure 7. SONET Packet Processing in a Quad Port.**

To do this the Network processor simply has to index into the packet (which are standard size and format) to remove the data that it needs. This makes the problem not one of data throughput but rather one of memory management greatly simplifying the amount of processing required and the amount of logic needed in the network processor.

So we reduce the required size of the network processor by removing the registers that would have been used to store the packet and the logic required to control the data path. The I/O pin count can also be reduced since the processor only has to maintain a single data interface. If the processor was in the data path, the processor would have to have enough I/O to support a data in interface and a data out interface. This means having enough I/O to support data, address, and control for both interfaces.

**b) Frame Relay Packet Processing**

A frame relay edge router can receive data packets from the backplane. The packet then has a two byte frame header and a Frame Check Sum (FCS) appended to it. Figure 8 illustrates the steps taken in creating a frame relay packet.

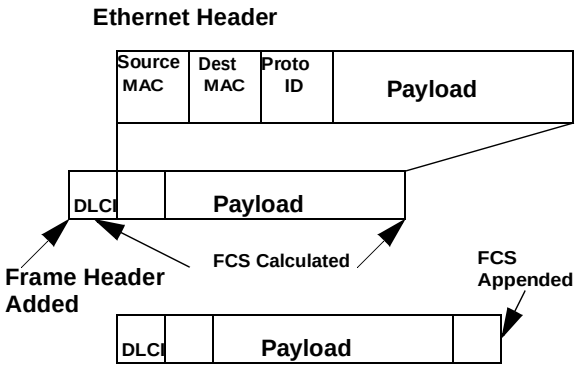


**Figure 8. Transmit Frame Relay Packet Processing.**

Using the system detailed in *Figure 3*, network processor A upon initial receipt of the packet into the QuadPort can index down 14 bytes past the beginning of the packet. At this point network processor A can add the two bytes of the frame header into the packet. This includes the DLCI or address of the receiving frame node. network processor B can read the data in as it comes in to generate the FCS. Since it does not need to change the payload this is a read and not a write operation. The FCS can then be written to the QuadPort at the end of the packet. While the FCS is calculated, processor A can start the transmission of the packet on the port side. Because data can be transmitted while it is processed, the overall latency of the system can be reduced. In a standard Frame Relay system, the data is read into the network processor completely before any processing is done to it. This is not necessary since the FCS involves non-destructive reads and so the data could be read out without impacting the data integrity of the packet while the FCS is calculated. This is only possible if the network processor is independent of the data path.

*Figure 9* shows a break out of the frame relay packet and the DLCI bytes and the DE byte. As you can see from the figure, to determine DLCI or DE requires very data from the packet itself.

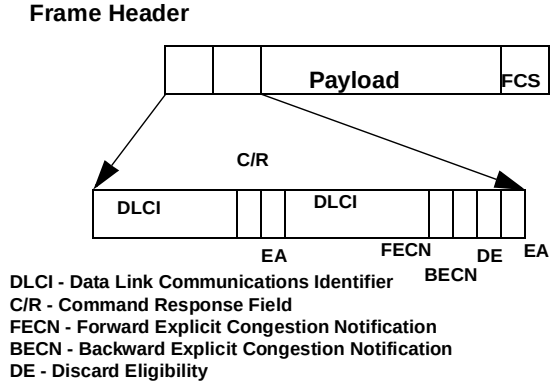


**Figure 9. Receive Frame Relay Packet Processing.** If the frame relay router is receiving the packet on its port side, then network processor A can quickly check the first byte of the packet to decide if the packet's DLCI is recognized by that node. If it is not then there is no reason to check the FCS of that packet and it can be discarded. Thus the packet can be ignored and the next packet received can be checked. In the case of a congested pipe, several packets can be stored in the QuadPort without impacting the network processor. The network processor A can review a packet's Quality Of Service (QOS) and its Discard Eligibility (DE). Network processor A can decide to either send high priority packets or discard low priority packets based solely on the packet header. A high priority packet's address can be sent to network processor B so it can begin checking the FCS of the packet. Processor A now is free to review the next packet's QOS/DE. After it does a simple 16 bit read on all the packets in the QuadPort (1 read cycle per packet), network processor A can start reading out the packets if the FCS is correct. Processor A only has to wait for processor B to finish checking the highest priority packets and not all received packets. Packets with incorrect FCS are discarded so processor A must wait for processor B to check a packet before it can transmit the packet.

The benefits of a QuadPort include latency reduction, cells count reduction by not storing the data, and reduction in the I/O count by eliminating the data pins. This makes the QuadPort an attractive solution for a Frame Relay system. Again this is a two processor implementation that can be combined in a single larger FPGA.

### ATM Packet Processing

Having already delved into both SONET and Frame Relay, lets take a look at what advantages the QuadPort would supply to an ATM line/switch card. First, the protocol must be discussed and understood.

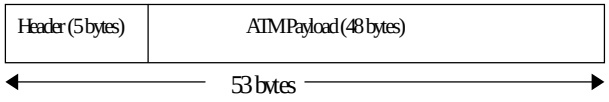


**Figure 10. ATM Cell.**

ATM is a connection based packet-switched technology that transmits packets that have a fixed length of 53 bytes as shown in *Figure 10*. These packets are in fact called cells. Before the transmittal of any data has occurred, ATM sets up a virtual connection between the origin and destination. In doing so, every ATM switch that is needed along the way allocates sufficient resources to meet the required class of service. Thus, every cell that belongs to the same virtual connection travels through the same sequence of ATM switches.

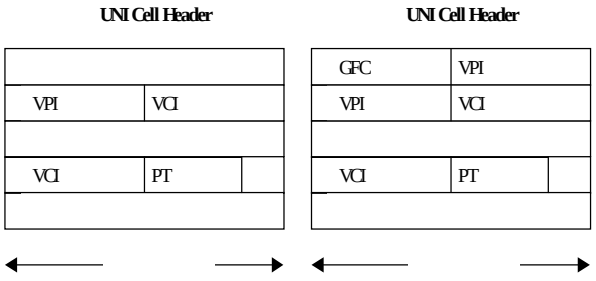


**Figure 11. ATM Header.**

At any one ATM switch there are a few processing tasks that are necessary to perform on the ATM headers shown in *Figure 11*. An 8-bit CRC must be performed on the first 4 bytes of a 5 byte ATM cell header. This value is then validated against the Header Error Check (HEC) field of the ATM cell header. Thus, it can be verified that the header has not been altered in transit. The must important task to perform after this validation, is that of reading and updating the Virtual Path Identifier (VPI) and Virtual Channel Identifier (VCI) fields. These values are updated based upon a lookup table that was established during the process in which the virtual connection was established. It is these values that are used to identify which virtual connection the cells belong to. The only processing done by the network processor in an ATM system is on the first five bytes of the cell.

Once again, the QuadPort becomes an excellent place to store the data as it goes through the system. It removes the processor from the data path and allows the processing to occur at wire speed. This is because the processor only needs to operate on less than 10% of the packet and therefore the header can be processed while the rest of the packet is received.

**Using the QuadPort in a SAN Device**

The storage area network consist of several servers with Redundant Array of Independent Drives (RAID) hard drive arrays, Just a Bunch Of Disks (JBOD) hard drive arrays, and tape drives. The server and storage elements are connected using either fibre channel or ESCON. The actual logic needed to support either ESCON or fibre channel is actually quite small. What drives the size of these controllers is the I/O pin count necessary to allow them to switch data down multiple paths. This switching allows storage of the data in various medias devices and allows for a robust disaster recovery system because the same data exists in multiple locations. *Figure 12* shows a high level block diagram of a SAN system.

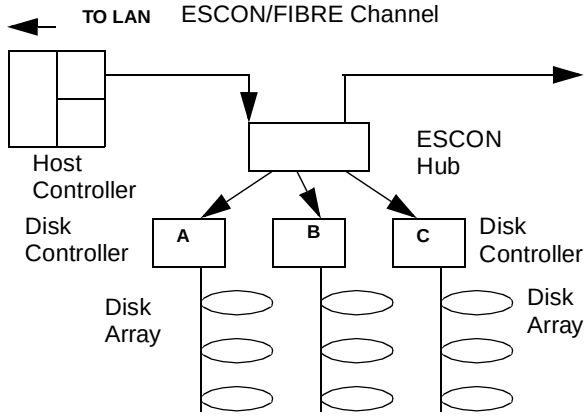


**Figure 12. SAN Architecture.**

The host controller is usually a server with access to an Ethernet LAN on one side and the SAN on the other. The host controllers connection to the SAN is usually through a fibre channel hub which contains several connections to disk arrays whether they are RAID, JBOD, or tape arrays. These arrays contain redundant copies of data that is being driven from the host machine. The hub is part of a fibre ring that can contain several disk arrays depending upon the network architecture of the SAN.The hub itself can contain several cards, the purpose of each card is to handle the ESCON/Fibre Channel protocol, check the integrity of the data packet, and finally route that packet to its end destination.

The ESCON data path usually consists of a three parts. The first part is a ESCON fibre optic physical layer (PHY) device. The second part is a protocol engine and the last part is an output physical layer device. The protocol engine consists of logic blocks that perform parity and CRC checks on the data as well as logic blocks that demultiplex command and data then interprets the command field. In a single SAN card, Several of these data paths exists allowing the SAN card to mirror the data it receives along multiple data paths. The protocol engine and the data multiplexor are usually combined into a single large FPGA. The primary reason to use a large FPGA is to obtain a large enough I/O pin count to support the multiple data paths.

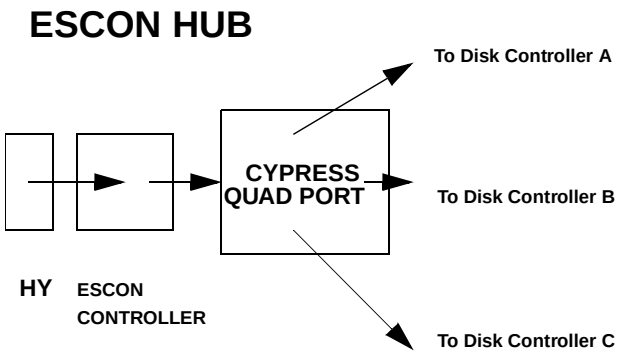


**Figure 13. SAN Card using a Quad Port.**

*Figure 13* shows a SAN card which utilizes the quad ports inherent data switching capabilities. The way the system could work is that data is received across the PHY and converted from optical to electrical. The PHY also recovers the clock, unencodes the data (usually its 8B/10B encoded), and frames it to recover packet boundaries. Once the data is recovered, the ESCON Controller reads the data in and checks the integrity of the data in the packet. This is done by a one bit parity check as well as a more intensive 16 bit CRC on the data packet. Once the packet is validated, the controller will separates the command bytes from the data bytes. It acts on any valid commands then repackages the data for transmit. This steps involves creating the appropriate headers onto the packet. The controller can then write the data into the Quad Port.

Once the data is in the QuadPort, the controller will drive the starting address of the packet inside the QuadPort on the remaining ports address lines loading the starting address of QuadPort's burst counter. The controller can then load the size of the packet to be sent and then can simply toggle the read line of the other ports and the transmit the data down all three paths. The same data will be mirrored down each data path and continually written until the counter expires.

The logic required to perform the tasks of the ESCON controller is normally less than 200k gates. A larger FPGA of a million or more gates is required to supply enough I/Os to support a data interface for multiple paths and to store the

data. The QuadPort handles this function natively with only a small logic block required to manipulate the addressing of its ports. This results in a lower cost and more efficient system.

### QuadPort Arbitration

The memory inside of the QuadPort is a shared resource between all four ports. Since reads are non-destructive, all four ports can read the same memory cell at the same time. Writes through are destructive in nature. So when data is written to a memory cell, the system has to ensure that no other port is writing to the same cell or reading from it. If two ports write to the same cell or if one port reads the cell while another is writing to it, the data that is read or written is indeterminate.

To ensure that a conflict does not occur, the designer that is integrating a QuadPort in a system must enforce an arbitration scheme. There are several ways to do this. These include address comparison to determine port access, enforcing a timing circuit that does not allow for simultaneous operations, or by using the QuadPort's built in mailboxes. Since the first two schemes are highly system dependent, the following will describe the best way to employ the mailbox.

The upper four addesses of the QuadPort can be used to past messages between ports. Address FFFFh is for port 1, address FFFEh is for port 2, address FFFDh is for port 3, and address FFFCh is for port 4. If port 4 has a message for port 1, it would write to address FFFFh. Once the message is written the INT pin for port 1 will assert. This alerts port 1 that there is a message in its mailbox. Once port 1 reads the message the interrupt is cleared. The other ports can read address FFFFh but only when port 1 reads it does the interrupt get cleared.

This feature can be used to pass tokens. The token can consist of up to 18 bits. If the upper ten bits were the ten most significant bits of the address, the other eight bits can specify the number of 1K blocks that the port wanted to reserve. So if port 4 wanted to reserve an address 4k block starting at 0000h, it would write 00004h to addresses FFFFh to FFFDh. This will cause an interrupt to occur on Ports 1, 2, and 3. External logic attached to the ports will read addresses FFFFh to FFFDh and will not write to addresses 0000h to 1000h. Port 4 can proceed to write to that block. When port 4 was done with the address block it could then write the same value to address FFFFh to FFFDh to let the other ports know that it no longer needs that block of memory.

The format of the 18 bits can change based upon system requirements. The mailbox feature is a quick and convenient way to design an arbitration scheme for the Quad Port.

## Conclusion

The Cypress Semiconductor CY7C0430 QuadPort is a revolutionary step in the development of specialty memory. Specialty memory products are traditionally an afterthought in the design process. A "band-aid" device that is added to buffer data between a high speed bus and a lower speed bus. With the advent of the QuadPort, the network designer has more flexibility in designing the next generation of network line cards. As we have shown, the QuadPort Backplane Switch is a versatile tool used to remove the network processor from the data path. BY doing so, it reduces the size and costs of these processors and lead to a more efficient system. It also is a great inherent switch which allows for greater flexibility in designing data paths.

For further information, please visit the Cypress web site at www.cypress.com. The web site also provides the latest data sheets, models, and any related documentation.