Notes on links
Sept 2025
Copied From Transputer Databook, 3rd edition

Transputer Architecture

1.3.1 Point to point communication links
The transputer architecture simplifies system design by using point to point communication links. Every member of the transputer family has one or more standard links, each of which can be connected to a link of some other component. This allows transputer networks of arbitrary size and topology to be constructed.
Point to point communication links have many advantages over multi-processor buses:
There is no contention for the communication mechanism, regardless of the number of transputers in the system.
There is no capacitive load penalty as transputers are added to a system.
The communications bandwidth does not saturate as the size of the system increases. Rather, the larger the number of transputers in the system, the higher the total communications bandwidth of the system. However large the system, all the connections between transputers can be short and local.

1.4 Communication
To provide synchronized communication, each message must be acknowledged. Consequently, a link requires at least one signal wire in each direction.

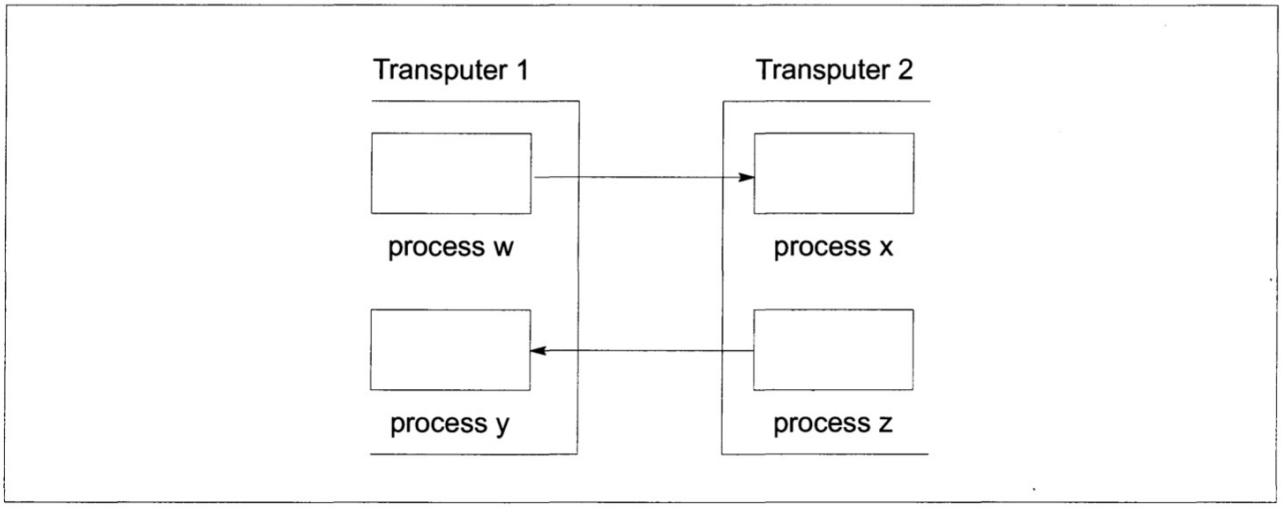


Figure 1.4 Links communicating between processes

A link between two transputers is implemented by connecting a link interface on one transputer to a link interface on the other transputer by two one-directional signal lines, along which data is transmitted serially.
The two signal wires of the link can be used to provide two occam channels, one in each direction. This requires a simple protocol. Each signal line carries data and control information.

The link protocol provides the synchronized communication of occam. The use of a protocol providing for the transmission of an arbitrary sequence of bytes allows transputers of different word length to be connected.

Each message is transmitted as a sequence of single byte communications, requiring only the presence of a single byte buffer in the receiving transputer to ensure that no information is lost. Each byte is transmitted as a start bit followed by a one bit followed by the eight data bits followed by a stop bit. After transmitting a data byte, the sender waits until an acknowledge is received; this consists of a start bit followed by a zero bit. The acknowledge signifies both that a process was able to receive the acknowledged byte, and that the receiving link is able to receive another byte. The sending link reschedules the sending process only after the acknowledge for the final byte of the message has been received.
Data bytes and acknowledges are multiplexed down each signal line. An acknowledge can be transmitted as soon as reception of a data byte starts (if there is room to buffer another one). Consequently transmission may be continuous, with no delays between data bytes.

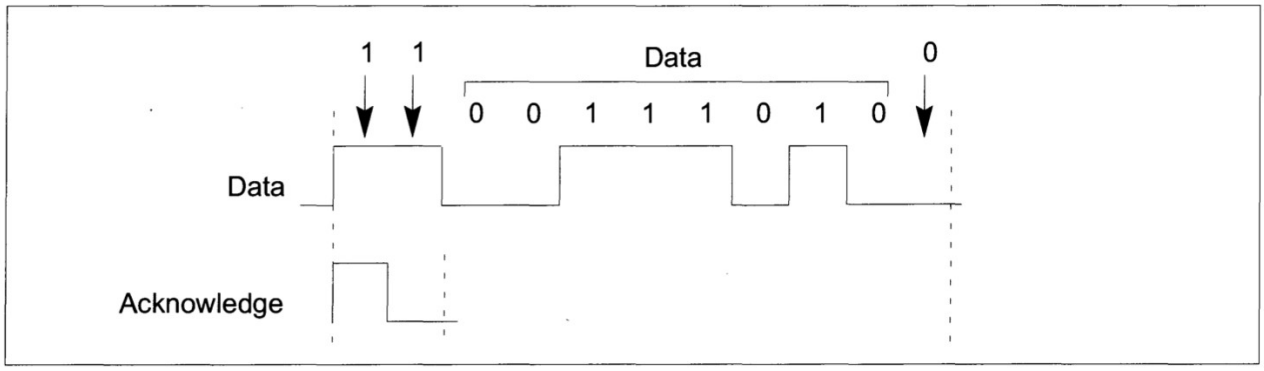


Figure 1.5 OS link protocol

The links are designed to make the engineering of transputer systems straightforward. Board layout of two wire connections is easy to design and area efficient. All transputers will support a standard communications frequency of 10 Mbits/sec, regardless of processor performance. Thus transputers of different performance can be directly connected and future transputer systems will directly communicate with those of today.

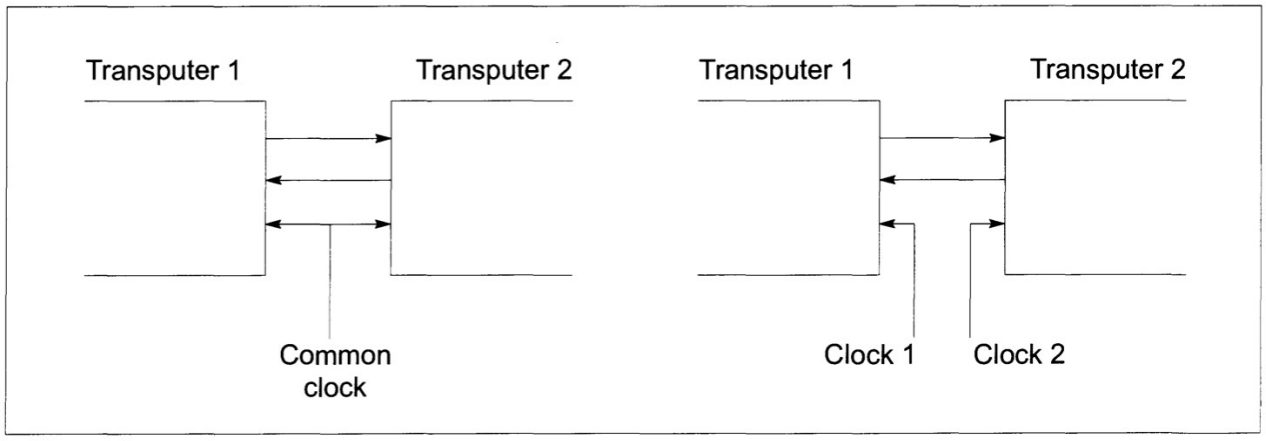


Figure 1.6 Clocking transputers

Link communication is not sensitive to clock phase. Thus, communication can be achieved between independently clocked systems as long as the communications frequency is the same.
The transputer family includes a number of link adaptor devices which provide a means of interfacing transputer links to non-transputer devices.

5 Physical architecture
5.1 INMOS serial links
5.1.1 Overview

All transputers have several links. The link protocol and electrical characteristics form a standard for all INMOS transputer and peripheral products.

All transputers support a standard link communications frequency of 10 Mbits/sec. Some devices also support other data rates. Maintaining a standard communications frequency means that devices of mixed performance and type can intercommunicate easily.

Each link consists of two unidirectional signal wires carrying both data and control bits. The link signals are TTL compatible so that their range can be easily extended by inserting buffers.

The INMOS communication links provide for communication between devices on the same printed circuit board or between printed circuit boards via a back plane. They are intended to be used in electrically quiet environments in the same way as logic signals between TTL gates.

The number of links, and any communication speeds in addition to the standard speed of 10 Mbits/sec, are given in the product data for each product.

5.1.2 Link electrical specification
The quiescent state of the link signals is low, for a zero. The link input signals and output signals are standard TTL compatible signals.

For correct functioning of the links the specifications for maximum variation in clock frequency between two transputers joined by a link and maximum capacitive load must be met. Each transputer product also has specified the maximum permissible variation in delay in buffering, and minimum permissible edge gradients. Details of these specifications are provided in the product data.

Provided that these specifications are met then any buffering employed may introduce an arbitrary delay into a link signal without affecting its correct operation.

5.3 Bootstrapping from ROM or from a link
…
If bootstrapping from a link, the transputer bootstraps from the first link to receive a message. The first byte of the message is the count of the number of bytes of program which follow. The program is loaded into memory starting at a product dependent location MemStart, and then control is transferred to this address.
Messages subsequently arriving on other links are not acknowledged until the transputer processor obeys a process which inputs from them. The loading of a network of transputers is controlled by the transputer development system, which ensures that the first message each transputer receives is the bootstrap program.

1.4 Communications

Communication between processes is achieved by means of channels. occam communication is point- to-point, synchronized and unbuffered. As a result, a channel needs no process queue, no message queue and no message buffer.
A channel between two processes executing on the same transputer is implemented by a single word in memory; a channel between processes executing on different transputers is implemented by point-to-point links. The processor provides a number of operations to support message passing, the most important being

input message
output message

The input message and output message instructions use the address of the channel to determine whether the channel is internal or external. This means that the same instruction sequence can be used for both hard and soft channels, allowing a process to be written and compiled without knowledge of where its channels are connected.
As in the occam model, communication takes place when both the inputting and outputting processes are ready. Consequently, the process which first becomes ready must wait until the second one is also ready.
A process performs an input or output by loading the evaluation stack with a pointer to a message, the address of a channel, and a count of the number of bytes to be transferred, and then executing an input message or an output message instruction.

1.4.1 Internal channel communication

At any time, an internal channel (a single word in memory) either holds the identity of a process, or holds the special value empty. The channel is initialized to empty before it is used.
When a message is passed using the channel, the identity of the first process to become ready is stored in the channel, and the processor starts to execute the next process from the scheduling list. When the second process to use the channel becomes ready, the message is copied, the waiting process is added to the scheduling list, and the channel reset to its initial state. It does not matter whether the inputting or the outputting process becomes ready first.
In figure 1.6, a process P is about to execute an output instruction on an 'empty' channel C. The evaluation stack holds a pointer to a message, the address of channel C, and a count of the number of bytes in the message.

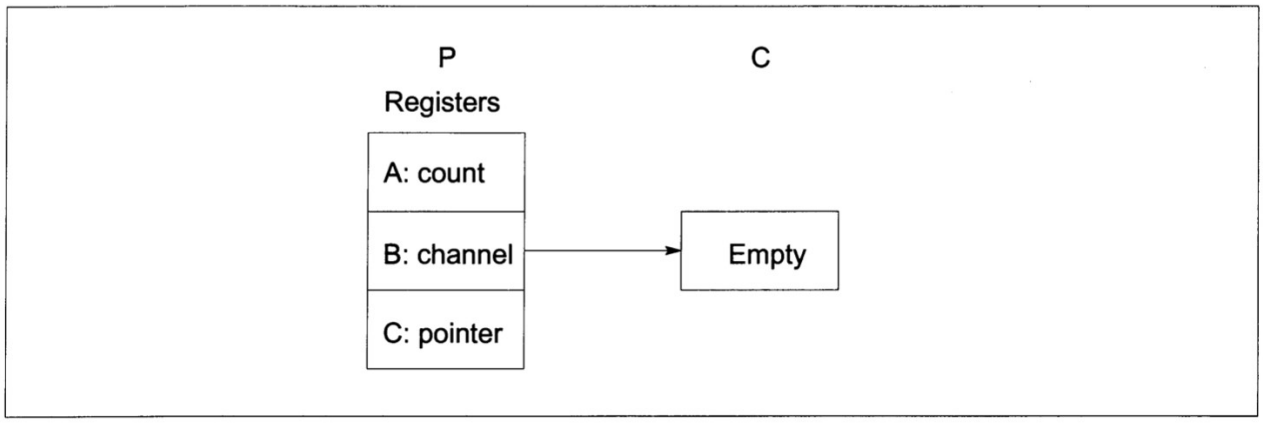


Figure 1.6 Output to empty channel

After executing the output instruction, the channel C holds the address of the workspace of P, and the address of the message to be transferred is stored in the workspace of P. P is descheduled, and the process starts to execute the next process from the scheduling list.

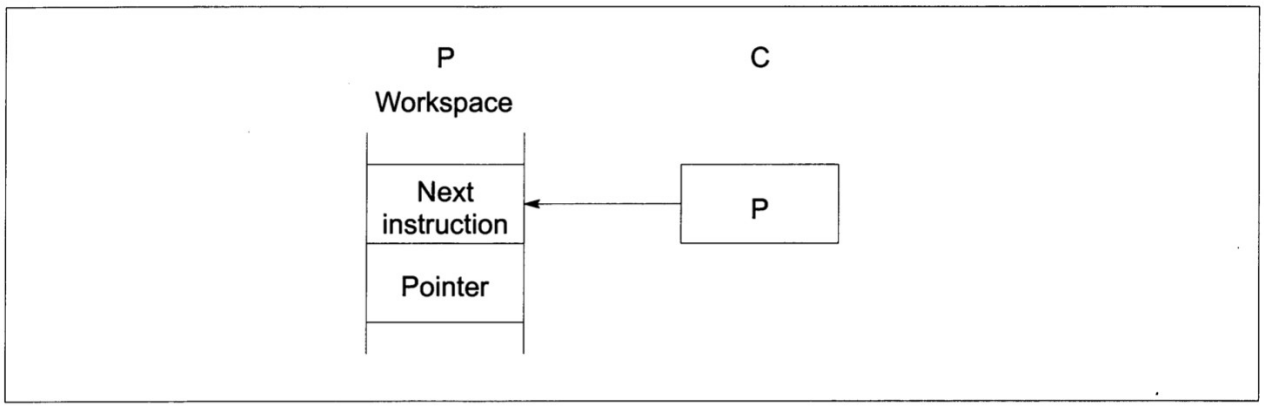


Figure 1.7

The channel C and the process P remain in this state until a second process, Q executes an output instruction on the channel.

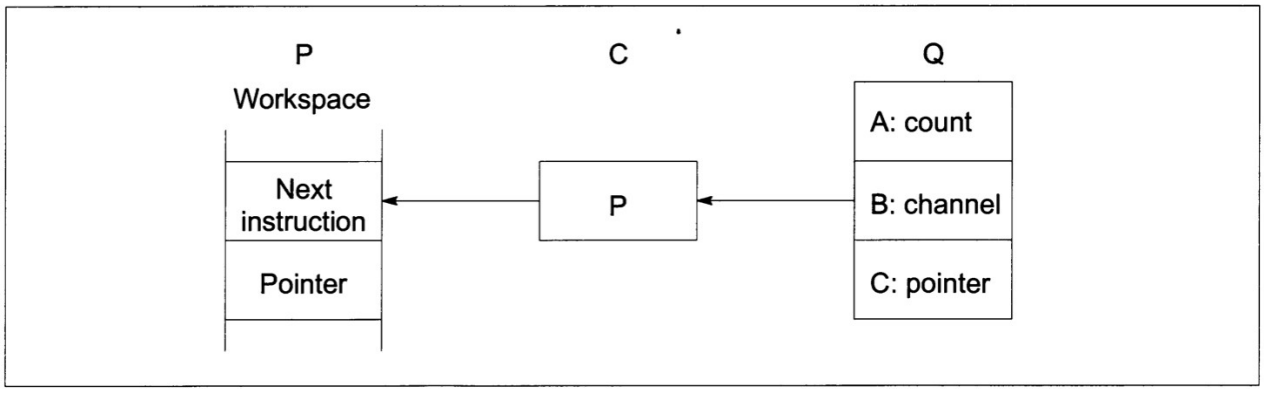


Figure 1.8

The message is copied, the waiting process P is added to the scheduling list, and the channel C is reset to its initial 'empty' state.

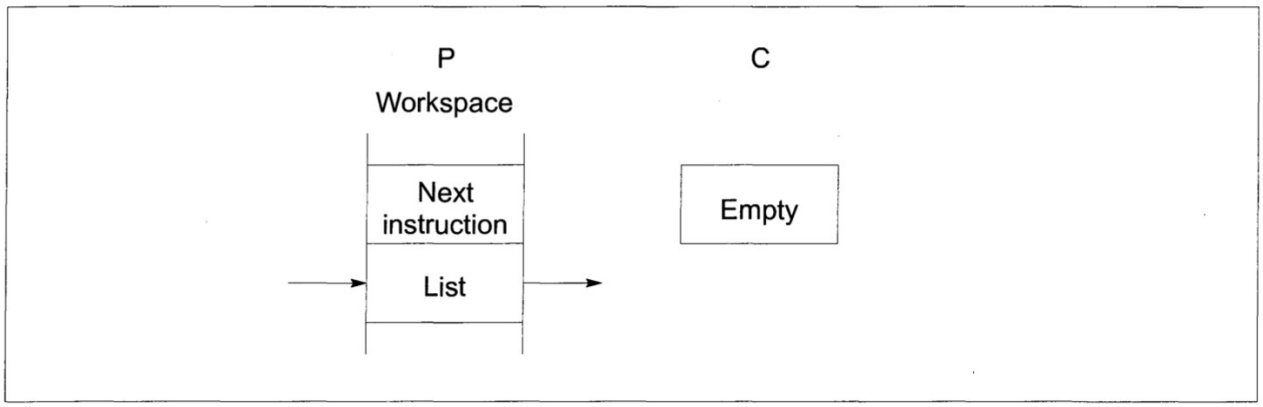


Figure 1.9

1.4.2 External channel communication

When a message is passed via an external channel the processor delegates to an autonomous link interface the job of transferring the message and deschedules the process. When the message has been transferred the link interface causes the processor to reschedule the waiting process. This allows the processor to continue the execution of other processes whilst the external message transfer is taking place.

Each link interface uses three registers:

- a pointer to a process workspace
- a pointer to a message
- a count of bytes in the message

In figure 1.10 processes P and Q executed by different transputers communicate using a channel C implemented by a link connecting two transputers. P outputs, and Q inputs

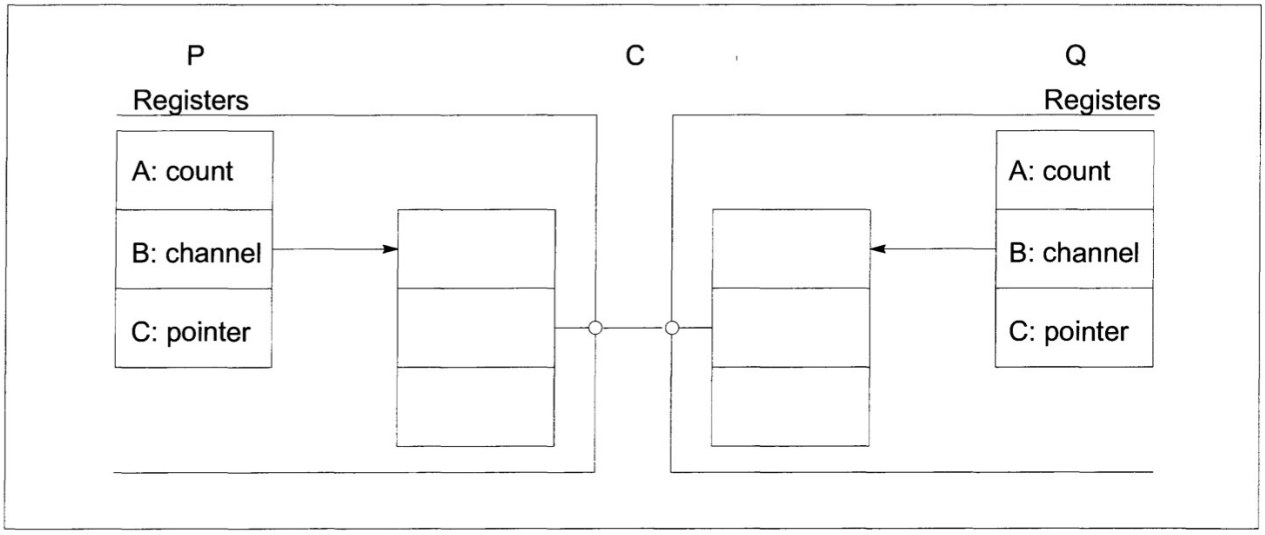


Figure 1.10 Communication between transputers

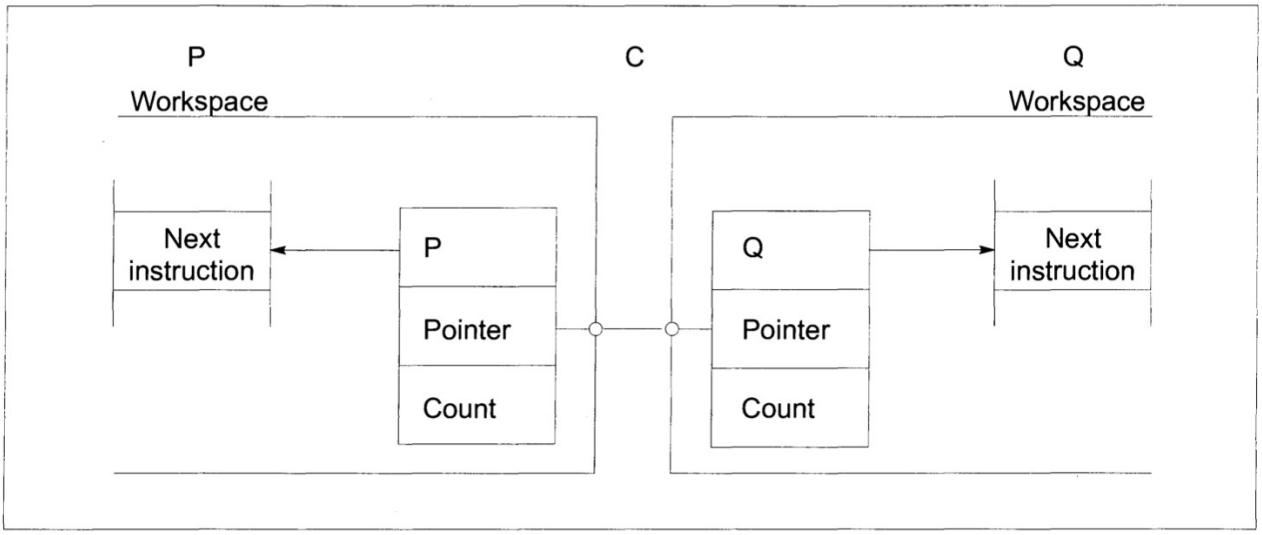


Figure 1.11

When P executes its output instruction, the registers in the link interface of the transputer executing P are initialized, and P is descheduled. Similarly, when Q executes its input instruction, the registers in the link interface of the process executing Q are initialized, and Q is descheduled (figure 1.11). The message is now copied through the link, after which the workspaces of P and Q are returned to the corresponding scheduling lists (figure 1.12). The protocol used on P and Q ensures that it does not matter which of P and Q first becomes ready.

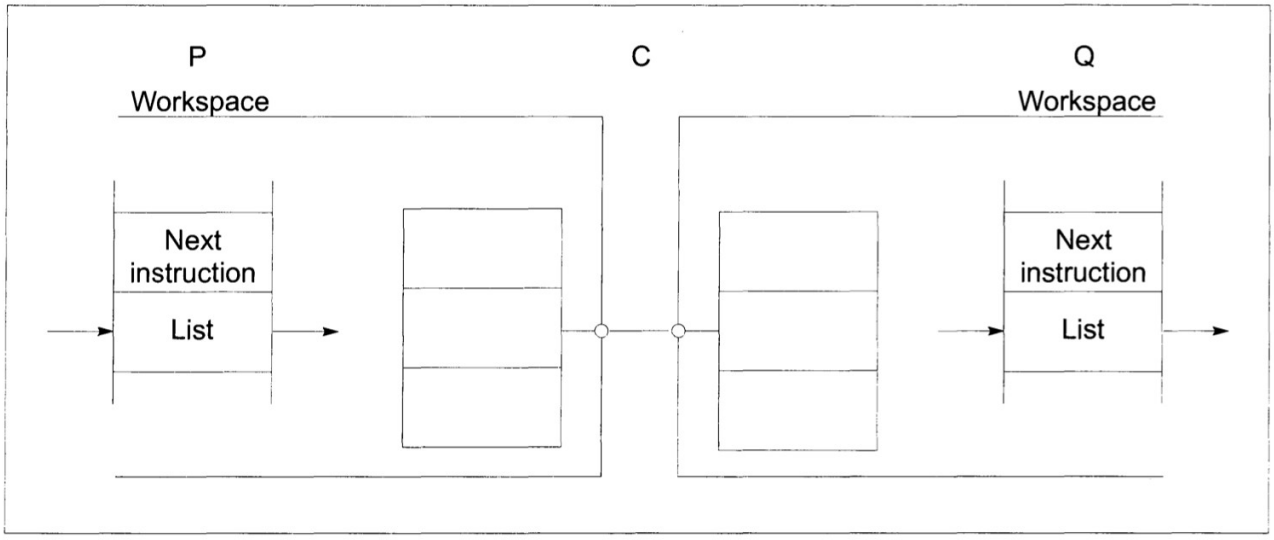


Figure 1.12

1.4.3 Communication Links

A link between two transputers is implemented by connecting a link interface on one transputer to a link interface on the other transputer by two one-directional signal wires, along which data is transmitted serially. The two wires provide two occam channels, one in each direction. This requires a simple protocol to multiplex data and control information. Messages are transmitted as a sequence of bytes, each of which must be acknowledged before the next is transmitted. A byte of data is transmitted as a start bit followed by a one bit followed by eight bits of data followed by a stop bit. An acknowledgement is transmitted as a start bit followed by a stop bit. An acknowledgement indicates both that a process was able to receive the data byte and that it is able to buffer another byte.

The protocol permits an acknowledgement to be generated as soon as the receiver has identified a data packet. In this way the acknowledgement can be received by the transmitter before all of the data packet has been transmitted and the transmitter can transmit the next data packet immediately. Some transputers do not implement this overlapping and achieve a data rate of 0.8 Mbytes/sec using a link to transfer data in one direction. However, by implementing the overlapping and including sufficient buffering in the link hardware, the rate can be more than doubled to achieve 1.8 Mbytes/sec in one direction, and 2.4 Mbytes/sec when the link carries data in both directions. The diagram below shows the signals that would be observed on the two link wires when a data packet is overlapped with an acknowledgement.

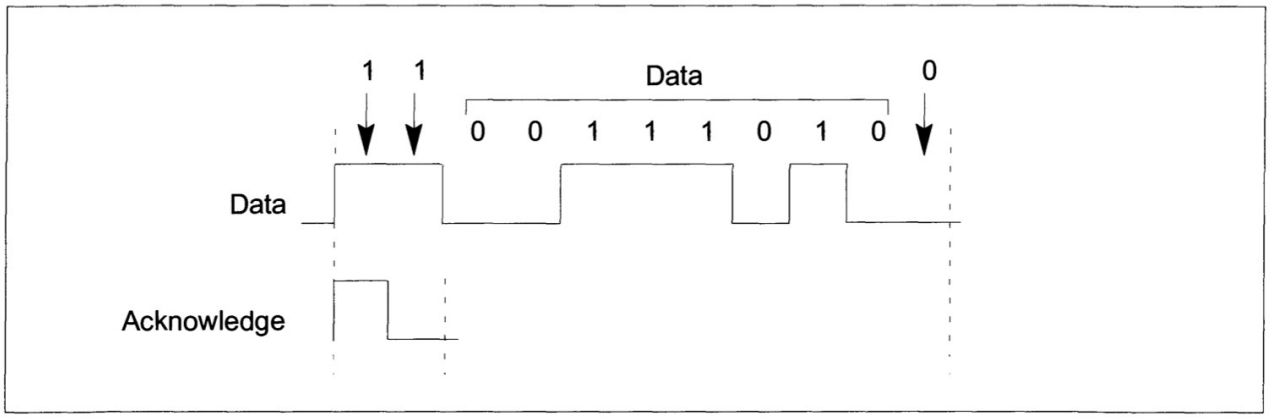


Figure 1.13 OS link data and acknowledge formats

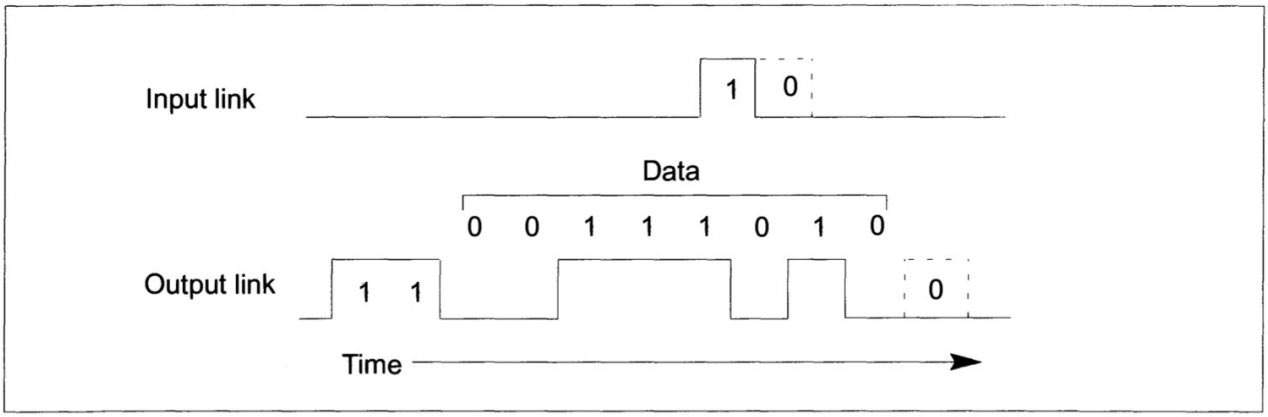
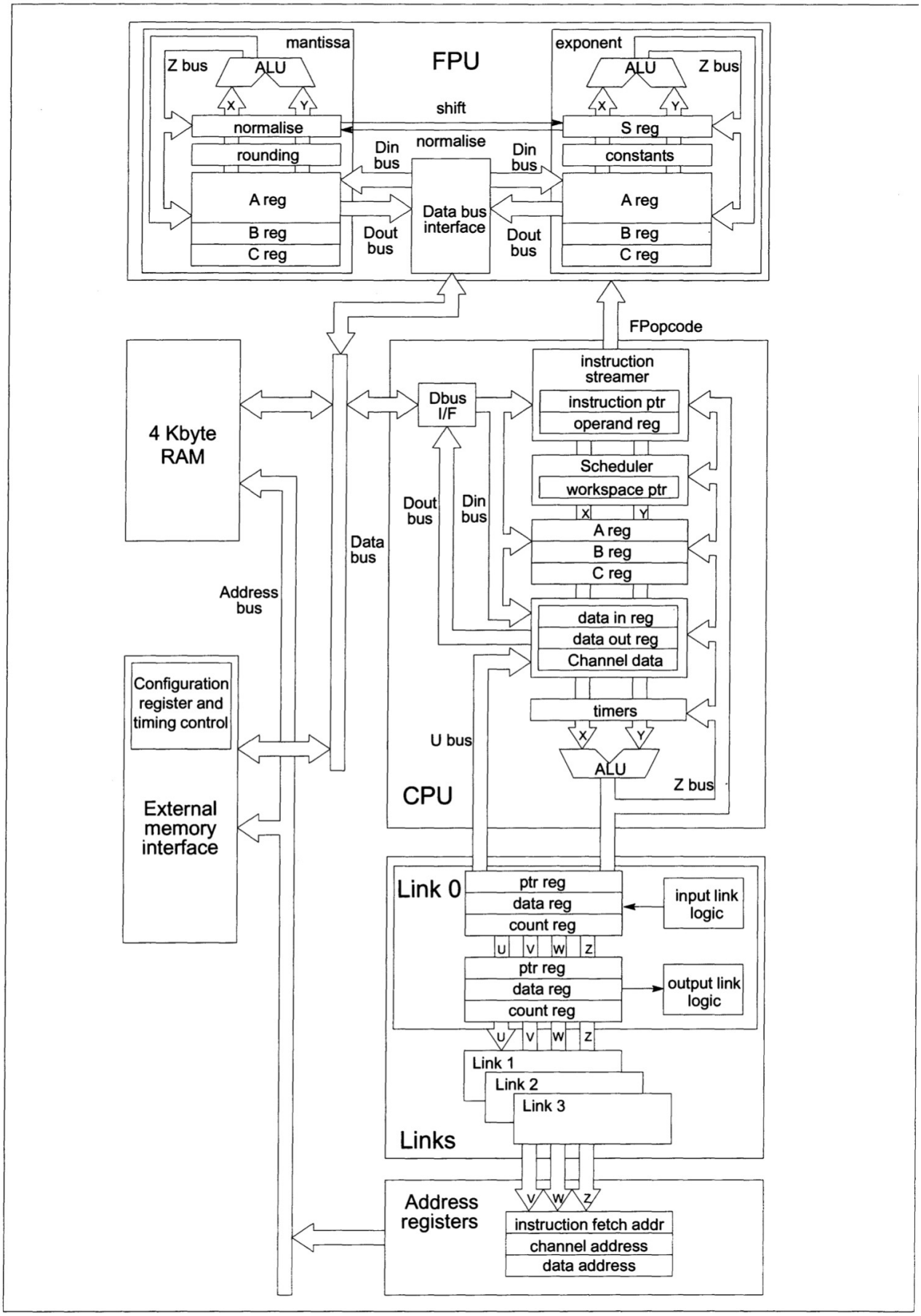


Figure 1.14 OS link data and acknowledge formats

6. IMS T805 Transputer Engineering Data



8 Links

Four identical lNMOS bi-directional serial links provide synchronised communication between processors and with the outside world. Each link comprises an input channel and output channel. A link between two transputers is implemented by connecting a link interface on one transputer to a link interface on the other transputer. Every byte of data sent on a link is acknowledged on the input of the same link, thus each signal line carries both data and control information.

The quiescent state of a link output is low. Each data byte is transmitted as a high start bit followed by a one bit followed by eight data bits followed by a low stop bit. The least significant bit of data is transmitted first. After transmitting a data byte the sender waits for the acknowledge, which consists of a high start bit followed by a zero bit. The acknowledge signifies both that a process was able to receive the acknowledged data byte and that the receiving link is able to receive another byte. The sending link reschedules the sending process only after the acknowledge for the final byte of the message has been received.

The IMS T805 links support the standard INMOS communication speed of 10 Mbits/sec. In addition they can be used at 5 or 20 Mbits/sec for 20 MHz and 25 MHz devices, and 20 Mbits/sec for faster devices. Links are not synchronised with Clockln or ProcClockOut and are insensitive to their phases. Thus links from independently clocked systems may communicate, providing only that the clocks are nominally identical and within specification.

Links are TTL compatible and intended to be used in electrically quiet environments, between devices on a single printed circuit board or between two boards via a backplane. Direct connection may be made between devices separated by a distance of less than 300 millimetres. For longer distances a matched 100 ohm transmission line should be used with series matching resistors RM. When this is done the line delay should be less than 0.4 bit time to ensure that the reflection returns before the next data bit is sent.

Buffers may be used for very long transmissions. If so, their overall propagation delay should be stable within the skew tolerance of the link, although the absolute value of the delay is immaterial.

Link speeds can be set by LinkSpecial, Link0Special and Link123Special. The link 0 speed can be set independently. Table 8.1 shows uni-directional and bi-directional data rates in Kbytes/sec for each link speed; LinknSpecial is to be read as Link0Special when selecting link 0 speed and as Link123Special for the others. Data rates are quoted for a transputer using internal memory, and will be affected by a factor depending on the number of external memory accesses and the length of the external memory cycle.

Symbol	Parameter		Min	Nom	Max	Units	Notes
TJQr	**LinkOut** rise time				20	ns	1
TJQf	**LinkOut** fall time				10	ns	1
TJDr	**LinkIn** rise time				20	ns	1
TJDf	**LinkIn** fall time				20	ns	1
TJQJD	Buffered edge delay		0			ns	
TJBskew	Variation in TJQJD	20 Mbits/s			3	ns	2
		10 Mbits/s			10	ns	2
		5 Mbits/s			30	ns	2
CLIZ	**LinkIn** capacitance	@ f=1MHz			7	pF	1
CLL	**LinkOut** load capacitance				50	pF	
RM	Series resistor for 100Ω transmission line			56		ohms	

Notes

1 Guaranteed, but not tested.

2 This is the variation in the total delay through buffers, transmission lines, differential receivers etc., caused by such things as short term variation in supply voltages and differences in delays for rising and falling edges.

Table 8.2 Link

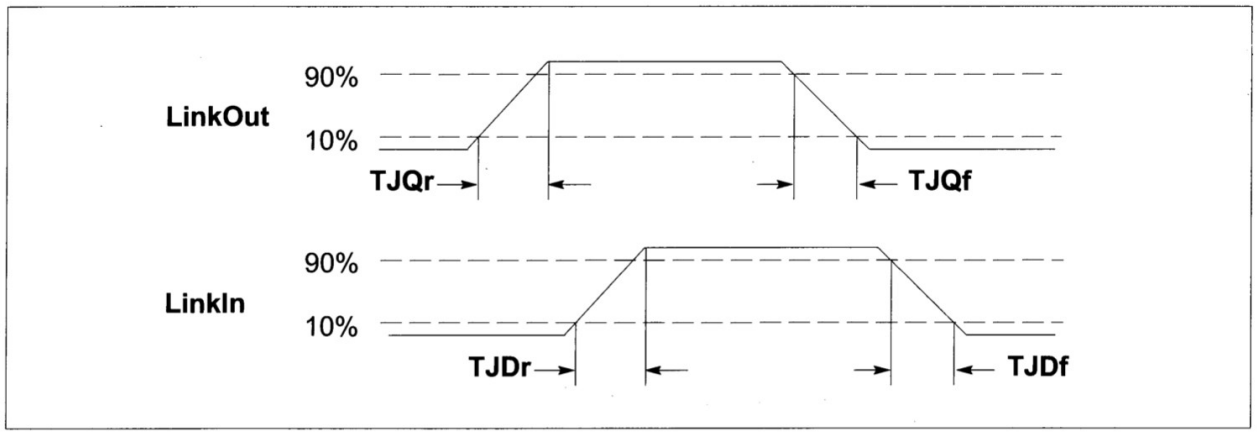


Figure 8.2 IMS T805 link timing

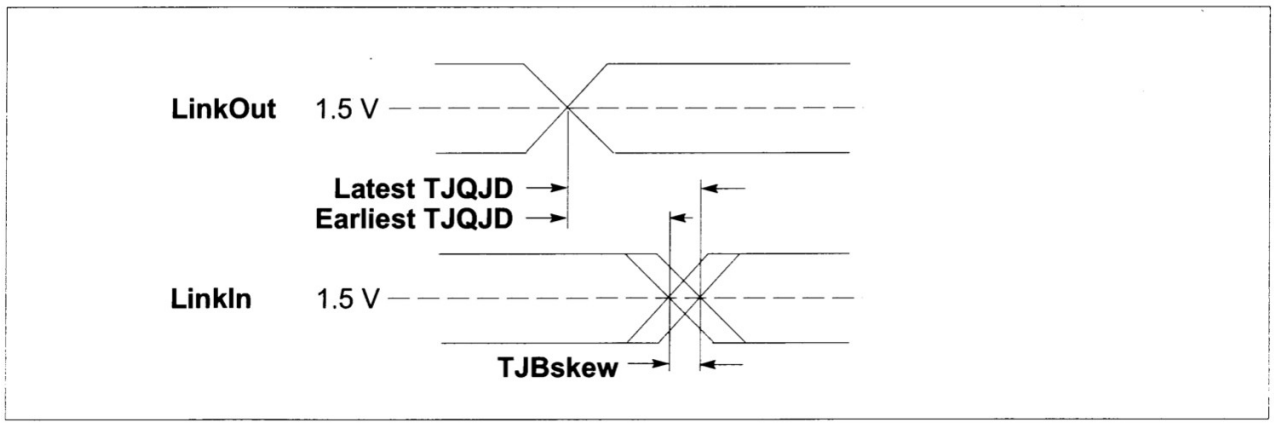


Figure 8.3 IMS T805 buffered link timing

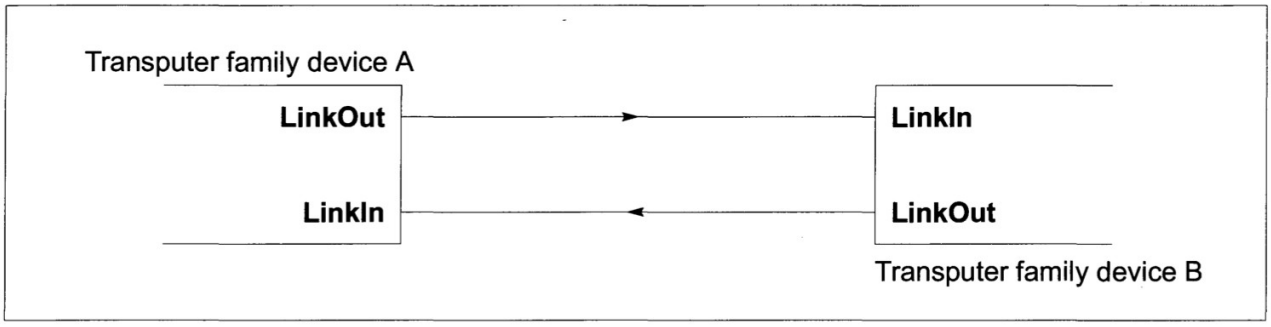


Figure 8.4 Links directly connected

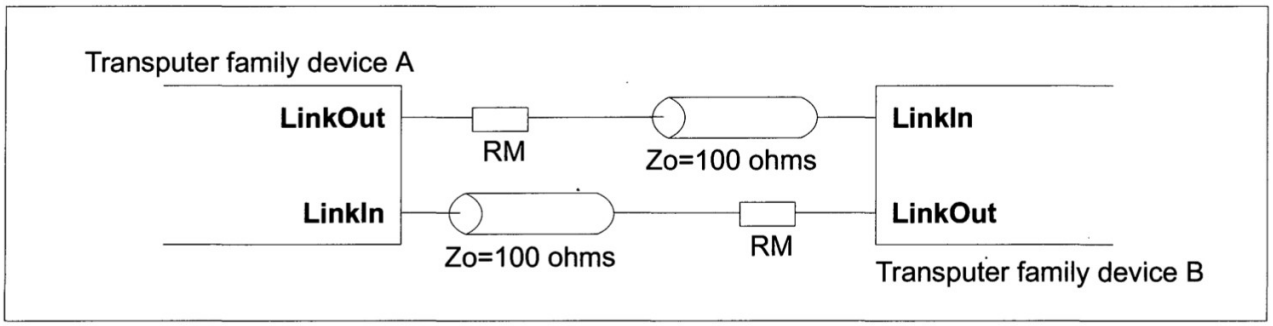


Figure 8.5 Links connected by transmission line

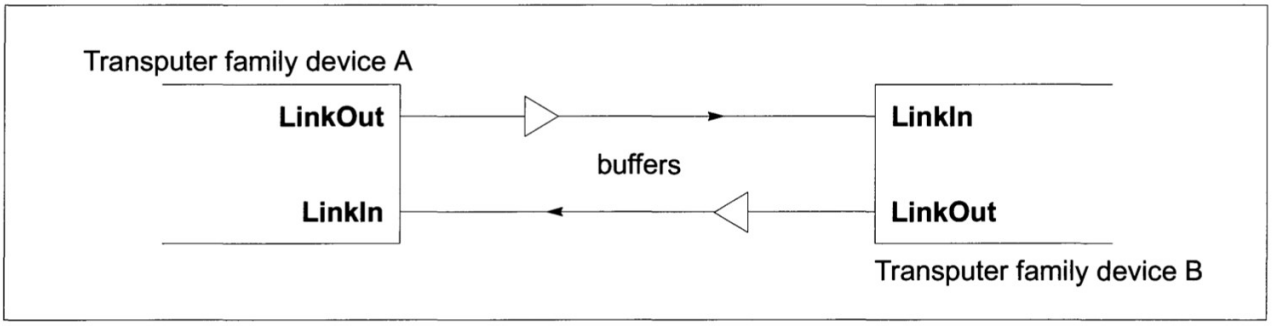


Figure 8.6 Links connected by buffers