10 A Generic Architecture for ATM Systems

10.1 Introduction

The rapid growth in the use of personal computers and high–performance workstations over the last ten years has fueled an enormous expansion in the data communications market. The desire to connect computers together to share information, common databases and applications led to the development of Local Area Networks and the emergence of ***distributed computing*****.** At the same time, the geographical limitations of LANs and the desire to provide corporate–wide networks stimulated the development towards faster, more reliable telecommunications networks for LAN interconnection, with the need to support data as well as traditional voice traffic. The resulting increase in the use of digital technology and complex protocols has resulted in the need for enormous computing capability within the telecommunications network itself, with the consequent emergence of the concept of the ***Intelligent Network***. With new, higher bandwidth applications such as video and multimedia on the horizon and user pressure for better, more seamless connection between computer networks, this convergence of computing and communications systems looks set to accelerate during the nineties.

A key step in this convergence is the development by the CCITT of standards for the ***Broadband Integrated Services Digital Network (B–ISDN)***. B–ISDN seeks to provide a common infrastructure on which a wide variety of voice, data and video services can be provided, thereby eliminating (hopefully) the final barriers between the world of computer networks and the world of telecommunications. The technological basis for B–ISDN chosen by the CCITT is the ***Asynchronous Transfer Mode (ATM)***, a fast–packet switching technique using small, self–routing packets called ***cells***.

The single most important element which has driven the development of both distributed computing and the intelligent network is the microprocessor. Indeed, as systems such as telecommunications networks have come to look more like distributed computers, so microprocessor architectures which support distributed multi–processing have come to look like communications networks. A message–passing computer architecture, such as that of the transputer, shares much in common with a packet switching system and thus provides a natural architecture from which to build communication systems. The communications architecture of the latest generation transputer, the T9000, shares much in common with ATM and is thus a natural choice for the implementation of ATM systems.

In this Chapter we describe the application of the transputer, in particular the serial links and packet routing capabilities of the communications architecture, to the design of ATM switching systems. We discuss their use in public switching systems and present a generic architecture for the implementation of private ATM switches and internetworking applications. We look at terminal adaption requirements and develop some ideas for interfacing transputers, routers and serial links to ATM networks. Finally, we consider various aspects of the performance of this architecture.

10.2 An Introduction to Asynchronous Transfer Mode

10.2.1 Background

Current communications systems split roughly into two basic categories:–

a) The existing telephone network, a ***Wide Area Network (WAN)***, predominantly designed around the requirements to transmit voice traffic around the globe

b) Existing ***Local Area Networks (LANs)***, designed to transmit digital data between computers over relatively short distances

As the idea of distributed computing and corporate–wide networks has gained acceptance, so has the desire to connect computers (predominantly PC's and workstations) across larger and larger distances. Unfortunately, seamless transmission of data from computer to computer across the globe using either of the existing types of networks is severely limited by the constraints inherent in each system:–

a) The telephone network is optimized for low–bandwidth, low latency point–to–point voice traffic (this traffic is relatively insensitive to noise and data errors)

b) Local area networks are optimized for high bandwidth computer data (which is not generally sensitive to latency, but is intolerant of data errors and usually uses some form of shared medium)

In summary, the telephone network is unreliable and too slow and LANs can't carry voice easily and don't go far enough. This split has led to communications networks developing from two directions over the past decade or so; one trying to make the telephone network faster and the other to make LANs go further.

Attempts to make the telephone network faster and more useful to data communications has resulted in a plethora of communications techniques and standards to transmit data between otherwise isolated computers. First came analogue modems (maximum 19kbits/s), then digital networks like X.25 (generally 64kbits/s), and latterly higher bandwidth access via basic/primary rate ISDN, frame relay, etc. However, the fastest access rates in common use are still no more than 1.5 – 2 Mbits/s, compared with 10–16 Mbits/s on LANs such as ethernet and token ring. Of more concern has been the need to use 'heavyweight' protocols to protect computer data as it travels over the existing, relatively unreliable, telephone network. The processing overhead of these protocols has a significant impact on the useable bandwidth available.

Progress on extending LANs has resulted in the development of ***Metropolitan Area Networks (MANs)*** designed to offer high bandwidth connections between computers over an area the size of, say, a reasonable city. An example is the ***Fibre Distributed Data Interface (FDDI)***, which can offer 100 Mbits/s connection over several kilometres. FDDI, however, is still a shared medium, is relatively expensive, requires new fibre cabling (although copper standards for short distances have been developed) and still requires expensive internetworking equipment to connect to WANs. In addition it cannot support voice traffic very easily. Another standard, ***IEEE 802.6***, shows greater promise in the longer term since it is designed to be 'media independent' and also to integrate more easily with WANs.

However, the situation has become exacerbated in recent years with the arrival of higher and higher bandwidth users (large CAD design databases, for example) and the expected growth of ***multimedia***, with its requirement to support voice, video and computer data applications (multimedia applications are described in more detail in Chapter 11 of this book). So, into the picture comes the CCITT with its efforts to provide the basis for the ***Broadband–ISDN***, a telecommunications infrastructure capable of supporting any type of traffic anywhere across the globe. The

CCITT has based this infrastructure on ***Asynchronous Transfer Mode (ATM)*** technology, which is described in the next section.

10.2.2 Basic ATM Concepts

ATM Cells

ATM is based on the concept of a universal ***cell*** (a very small packet) 53 bytes in length, of which the first 5 bytes are used for a routing header and the remaining 48 bytes are for carrying data. Each ATM cell is a self–contained entity which can be routed individually through each switching node in the network from source to destination. This cell has no awareness of the type of data it is carrying and can be considered to be a universal carrier of data, a sort of communications 'truck' (or 'lorry', for those of us in the UK) into which you can put voice, video, data, etc. The term 'asynchronous' is used since no clocking or timing relationship is maintained between the ATM cells.

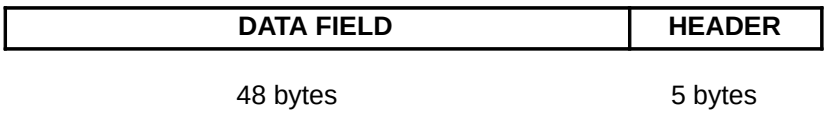

Figure 10.1 ATM Cell

The CCITT Recommendations for the public networks have so far defined ATM to run at a nominal 155 Mbits/s to fit in with the *Synchronous* (framed) bit rates used in the transmission systems between exchanges. In these systems, the ATM cells are packed in like bricks into a two–dimensional frame for transport to the next switch (described later). In reality the bit rate available for the ATM cells is about 149 Mbits/s once the framing overhead has been allowed for. It is expected that a 622 Mbits/s standard will follow (4 x 155 Mbit/s plus some extra overhead) with eventual data rates up to 2.4 Gbits/s being anticipated.

The situation for private networks is not yet clear, since the standards have not yet been set. 155 Mbits/s seems likely, but since ATM cells can be transmitted either framed (synchronously) or unframed (asynchronously) lower data rates (< 155 Mbits/s) for unframed cells may also be adopted. It is important to remember that this is the point–to–point bandwidth available to each connection, not the bandwidth of the network as a whole, which is the case of conventional shared–medium LANs/MANs like ethernet and FDDI.

ATM Connections

Any user who wishes to gain access to an ATM network must first establish a connection with the local switch. In the diagram below, our subscriber picks up a (very sophisticated) ATM telephone in order to send data across the network. During call set–up, the user negotiates with the network for the call and service characteristics desired. For example, the number dialled, bandwidth and service quality (error rates, etc.) required may be sent to the local switch. This is important, since different types of traffic require different performance from the network and the user will be charged accordingly. The local switch then negotiates with all the other switches necessary to connect to the desired destination. Assuming the connection is possible and that the user requested bandwidth and quality of service can be supported, the local switch confirms the connection to the user and allocates an ATM cell routing header from those available. If the requirements are not met and a lower standard of service is offered, it is up to the user to either accept this or terminate the call. Otherwise, the user equipment can now start sending data into the network using ATM cells and the routing header specified by the switch.

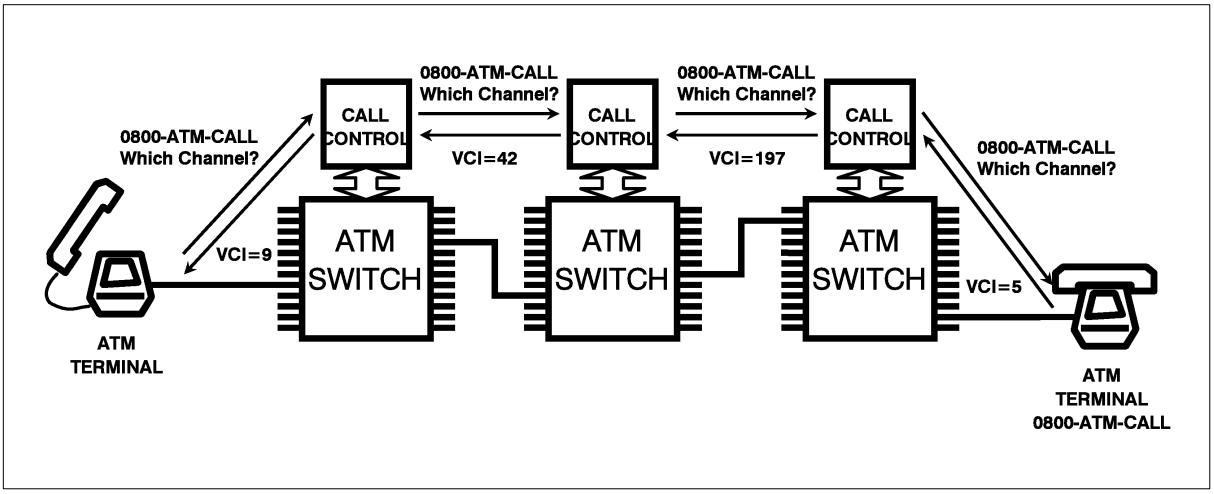

Figure 10.2 ATM call connections

Cell Header Policing

During the call set–up, the user negotiates with the network for certain service characteristics such as bandwidth. This may be specified in terms of the peak and average bandwidth required from the network (other parameters are under discussion). Since the user will be charged (on the public network) for his/her use of the system, and this charge will be dependent on the bandwidth negotiated, it is clearly necessary to monitor the actual use made to ensure nobody is cheating. It is proposed that this be done by monitoring the instantaneous and average bandwidth (or any other parameters) used by each cell on the network. This is referred to as ***Cell Header Policing*** and is done on a cell–by–cell basis on input by the network interface (ATM line card) at each ATM switch. Various algorithms have been proposed to perform this bandwidth policing, the most common of which is the ***Leaky Bucket*** algorithm. Depending on the type of service negotiated, transgressors of the negotiated policing limits may either be charged more (according to their use) or find their cells being discarded if they threaten the quality–of–service of other users.

Another important aspect of header policing arises due to the nature of ATM itself. On entering each ATM switch, each ATM cell is routed asynchronously (hence the name) from input to the appropriate output across the ATM switching fabric. Since cells may suffer delay in crossing this fabric due to internal traffic congestion, they may arrive at the output in 'clusters', resulting in a larger instantaneous bandwidth through no fault of the user (this is analogous to the behavior of buses in cities...). In extremis, if no flow control is provided across the switch fabric, cells may arrive at the output out of order. It would clearly be unreasonable to charge the user more or, worse, start discarding cells because of this behavior, so it is therefore necessary for the ATM switch itself to re–time the cells on output to the next switching node in order to meet the original user requirements. There is, therefore, a requirement to use header policing on output, as well as on input, and the system must ensure that cell order is maintained across the switch.

Cell Header Translation

The route that an ATM cell takes through the B–ISDN *network* is determined by the routing values in the cell header. Only a very limited routing 'space' is provided for each ATM cell since the header is only 5 bytes long and the bit–fields available are necessarily limited. To overcome this, the routing value is re–used (re–mapped) at each ATM switching point in the B–ISDN network. That is, the routing value only applies locally to one switching node and changes as the cell progresses through the network from one switching node to another. This constant re–mapping of the cell header is called ***Cell Header Translation*** and is performed when the cell is received by the ATM switch. Cell header translation is performed on a cell–by–cell basis by the network interface, or ATM 'line card', and with ATM operating at 155 or 620 Mbits/s, this re-

quires either very fast processing, custom hardware, or preferably an intelligent combination of the two.

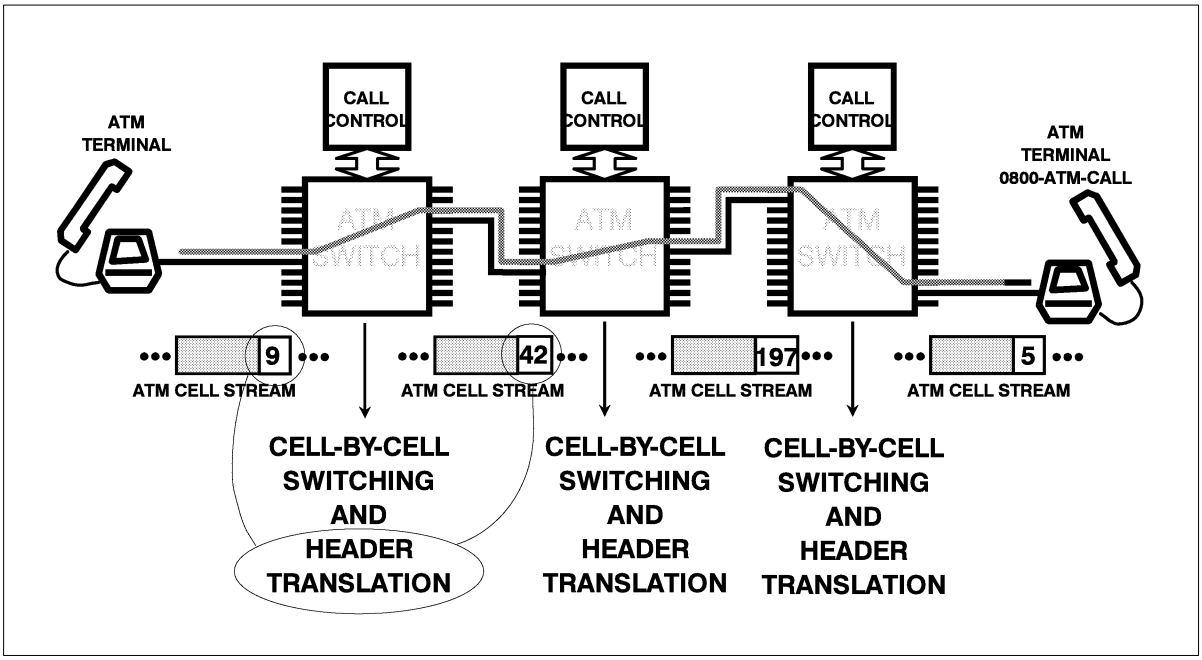

Figure 10.3 ATM Cell Header Translation

Within the ATM switch itself, routing decisions from network input to network output across the internal switching fabric also need to be made on a cell–by–cell basis. It may be necessary to perform another translation of the ATM cell header, to an internal format for routing purposes within the ATM fabric itself.

10.2.3 ATM Protocols and Standards

Having explained the basic principles it is now worth considering a few of the details. A good place to start is the CCITT Recommendations which apply to ATM. These are part of the I.xxx series of Recommendations which form the standards for ISDN networks.

ATM Protocol Reference Model

Like all good protocols, the ATM standard is defined as a series of layers. There are 3 basic layers which, from the top down, are:–

AAL: The ***'ATM Adaption Layer'*** defines various 'mapping' mechanisms from existing protocols (ISDN, voice, video, LAN data, etc.) onto ATM and vice versa.

ATM: This defines the ***ATM cell***, routing techniques and error mechanisms

PHY: This is the ***Physical*** layer and defines media (for example fibre/copper, connectors, etc.), bit timings, framing standards, etc.

In addition, the ATM standards describe Management and Control functions for each of the layers, such as call set–up and maintenance functions within the network. These layers constitute the ATM ***Protocol Reference Model (PRM)*** and are shown pictorially in Figure 10.4. The details of each layer are shown in Figure 10.5.

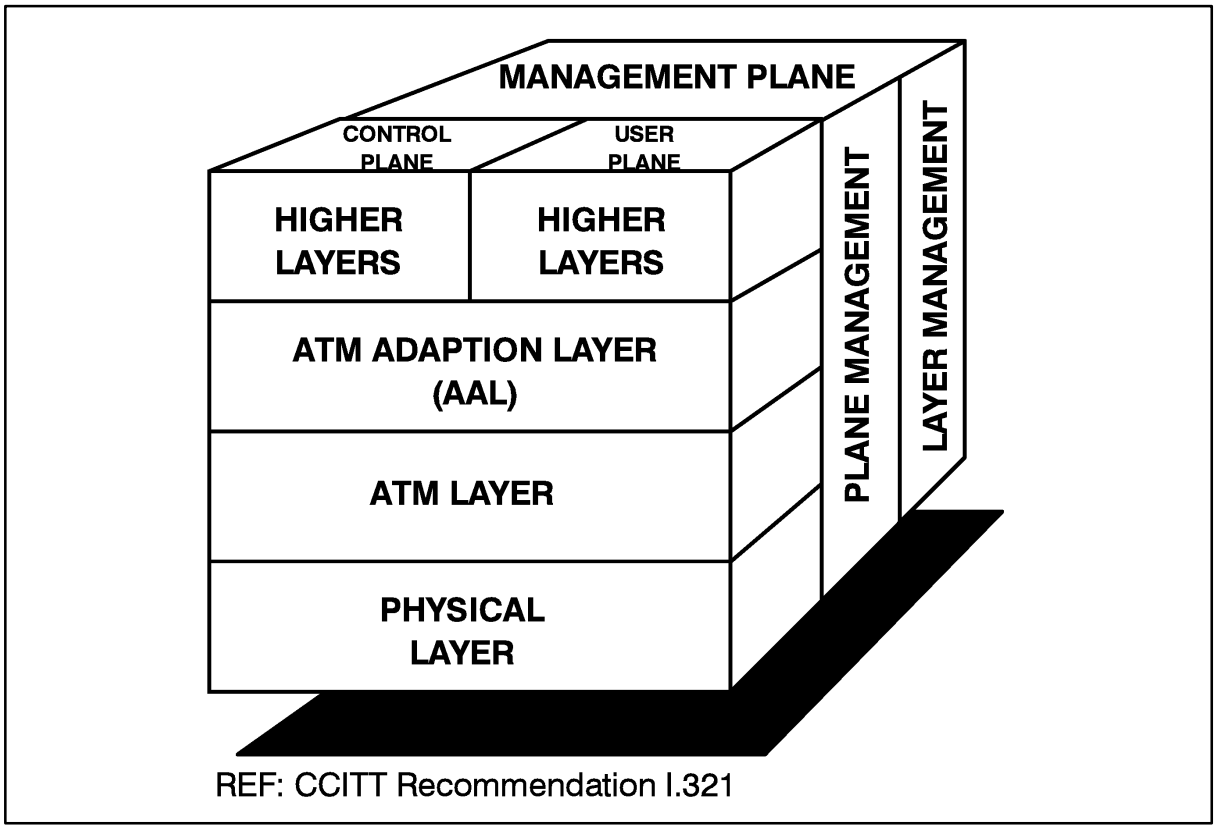

Figure 10.4 ATM Protocol Reference Model (PRM) [1]

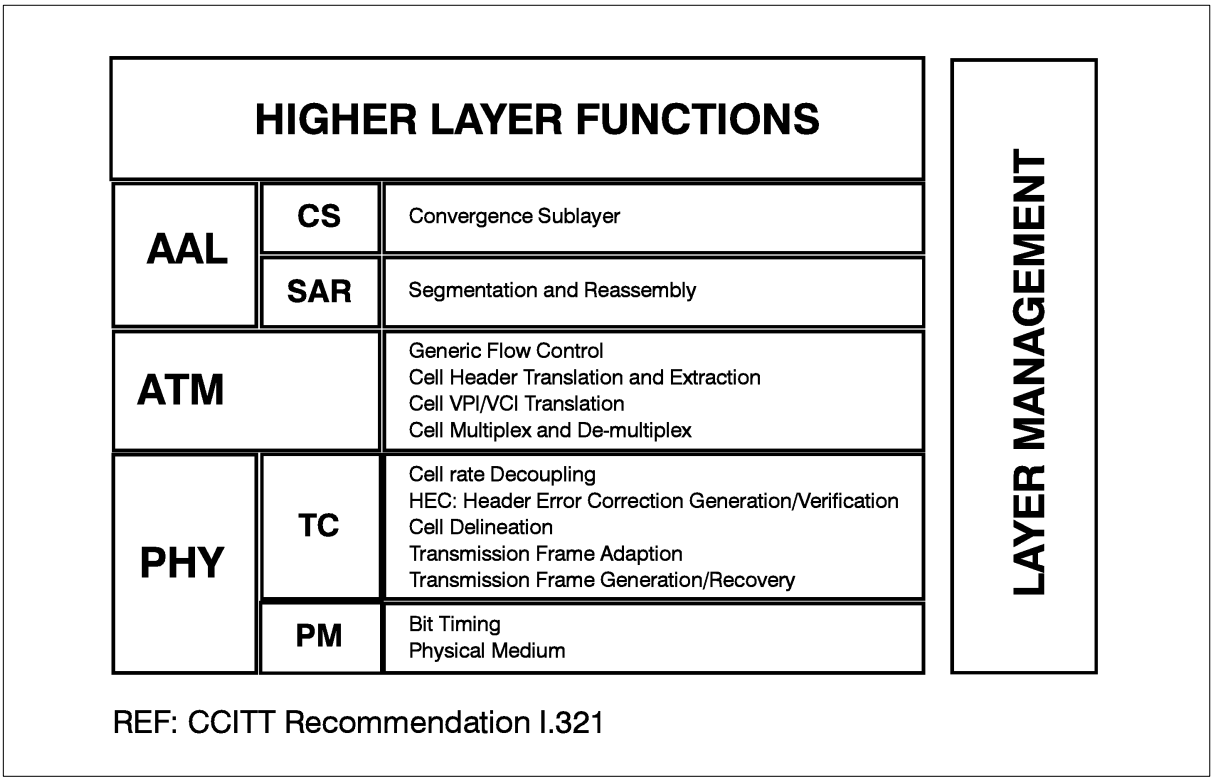

Figure 10.5 ATM PRM Layer Functions [1]

It is important to point out that many of the details in the ATM standards are still not yet finalized, particularly many of the management functions. However, a simplified diagram showing what all 3 layers do is given below and this may be referred to in the discussion of each layer in the following sections.

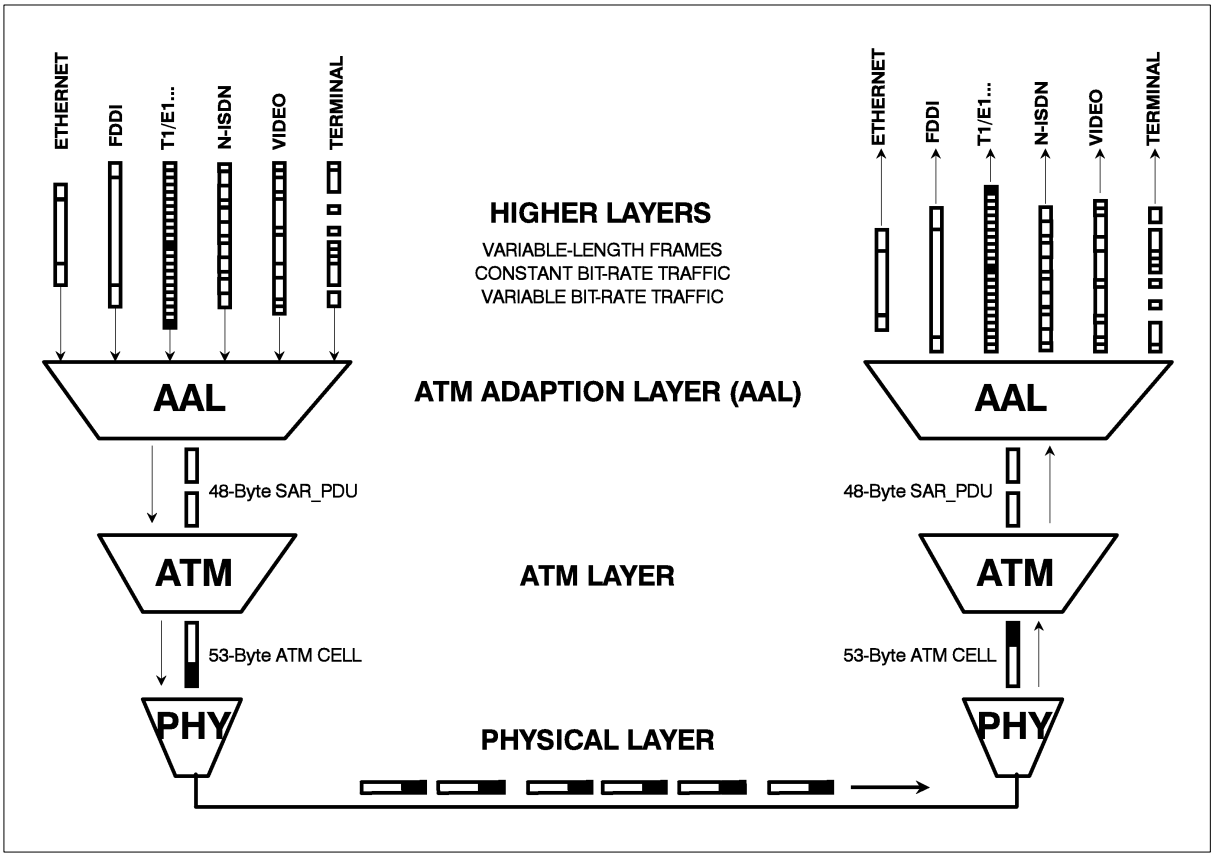

Figure 10.6 ATM Summary

The AAL Layer

The ‘ATM Adaption Layer’ is responsible for mapping other protocols onto the ATM cell format for transmission and switching. Examples of this would be to carry data traffic (in the form of ethernet, token ring or FDDI frames), voice traffic (64 kbit/s ISDN, for instance) or video traffic. Of necessity, the AAL layer comes in several varieties to suit the nature of the protocols being mapped. Data traffic is typically ‘bursty’ in nature and needs to be handled on a frame–by–frame basis. Voice traffic is referred to as ‘constant bit–rate’ traffic, that is, it is a constant flow of bits with no pause. Video traffic is referred to as ‘variable bit–rate’, since video coding algorithms typically generate an output which varies in bit–rate according to the contents of the picture being transmitted. The AAL layer provides functions to map all of these different types of traffic onto a flow of ATM cells. shown in the previous diagram.

There are four types of AAL specified in the CCITT standards, denoted as AAL1 to AAL4. Recently, a proposal for a fifth, AAL5, has been made with a view to providing a ‘lightweight’ AAL for frame (packet) based computer data (currently provided by AAL3). In each case, the AAL layer is responsible for ***Segmentation*** of the outgoing data, whatever it is, into small chunks of 48 bytes which then form the data field of the ATM cell. This 48–byte field will also contain overheads, such as CRC and payload type information which depend on which type of AAL is in use. For example, the actual user data field in AAL3 is only 44 bytes, with 2 bytes of header and 2 bytes of trailer added by the AAL to form the 48–byte field. Incoming data received from the ATM layer undergoes ***Reassembly*** by the AAL to provide an appropriate output stream, i.e. it undergoes the reverse of the segmentation process. An example is given in Figure 10.7, showing the AAL3 operation.

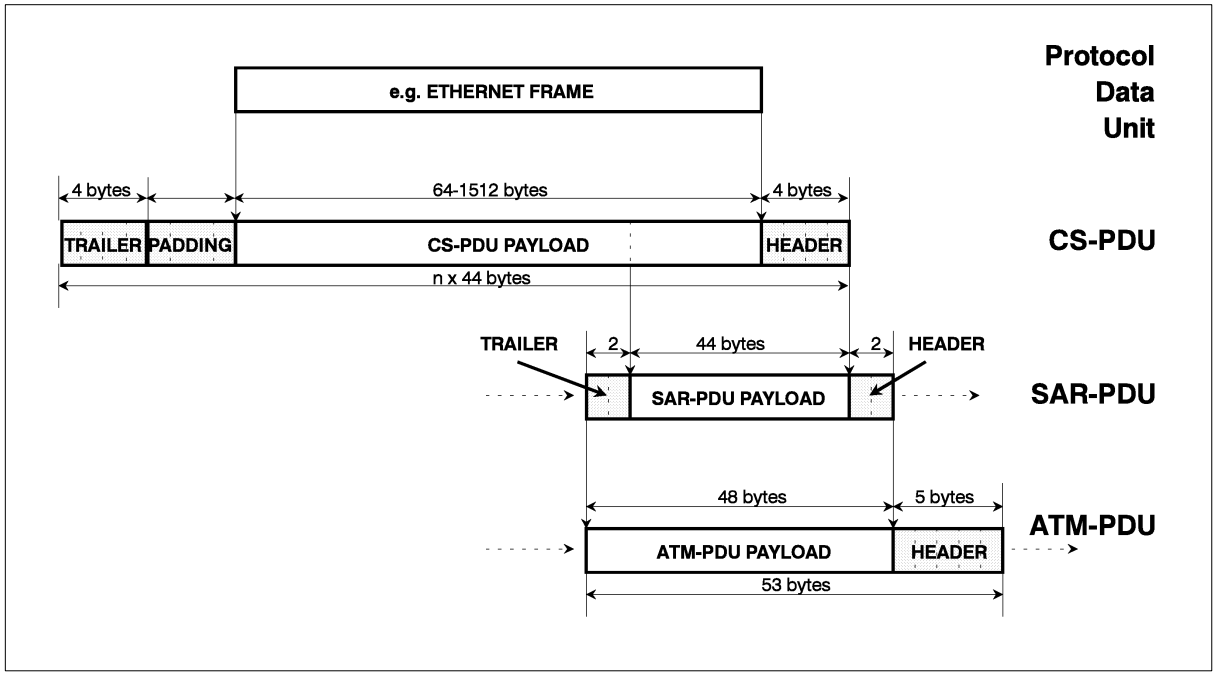

Figure 10.7 AAL3 Example

The use of each layer of the ATM protocol standard is illustrated in a simple form in Figure 10.8. ATM and PHY layer protocols are implemented everywhere in our simple network, but an AAL is only invoked at the termination points of the ATM network; that is, an AAL function is needed at:–

- the endpoints of the network (the user terminals)
- points where the ATM network meets another type of network (connecting to an ethernet network, for example)
- certain control nodes within the ATM network itself (passing signalling, management and control information between the control processors in the ATM exchanges, for instance).

There is insufficient space here to cover the AAL layer in detail so the reader is referred to the many papers on the subject for more detailed information, for example in [1] and [2]

The AAL layer is not needed as part of the switching function of an ATM network; this is handled entirely by the ATM layer.

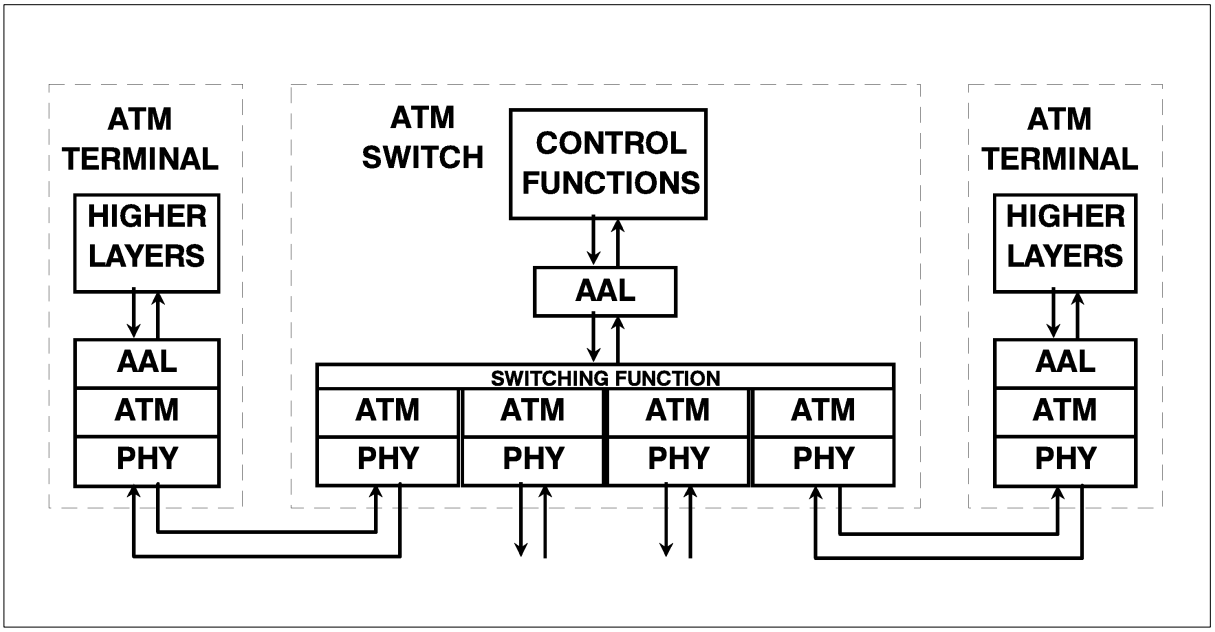

Figure 10.8 PRM Illustration in a simple Network

ATM Layer

There are two versions of the ATM cell format, one for the ***User–Network Interface (UNI)*** and another for the ***Network Node Interface (NNI)***. The basic structure of the ATM cell is shown in Figure 10.9.

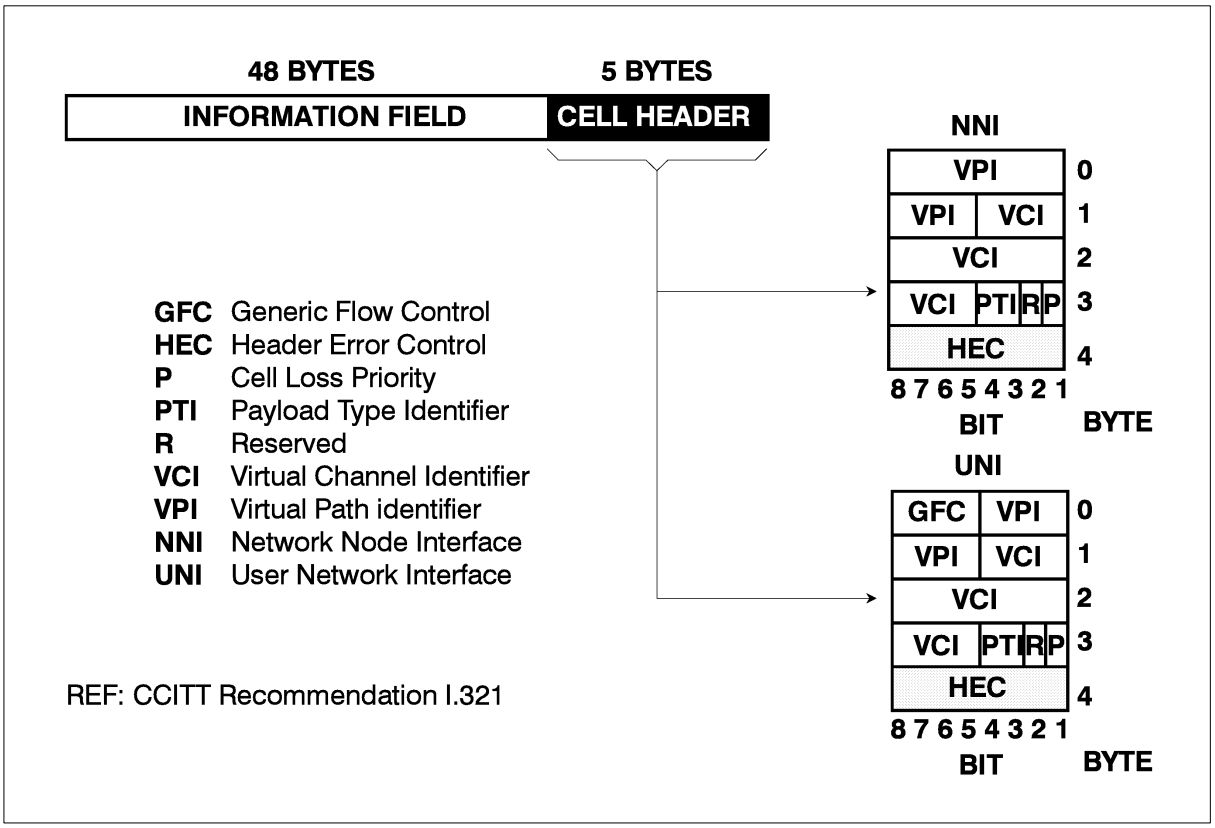

Figure 10.9 ATM Cell Structure

The cell header contains routing information, control bits and error detection features. Two methods of routing are provided; one is via the ***'Virtual Channel Identifier'*** and the other the ***'Virtual Path Identifier'*** (VCI and VPI respectively).

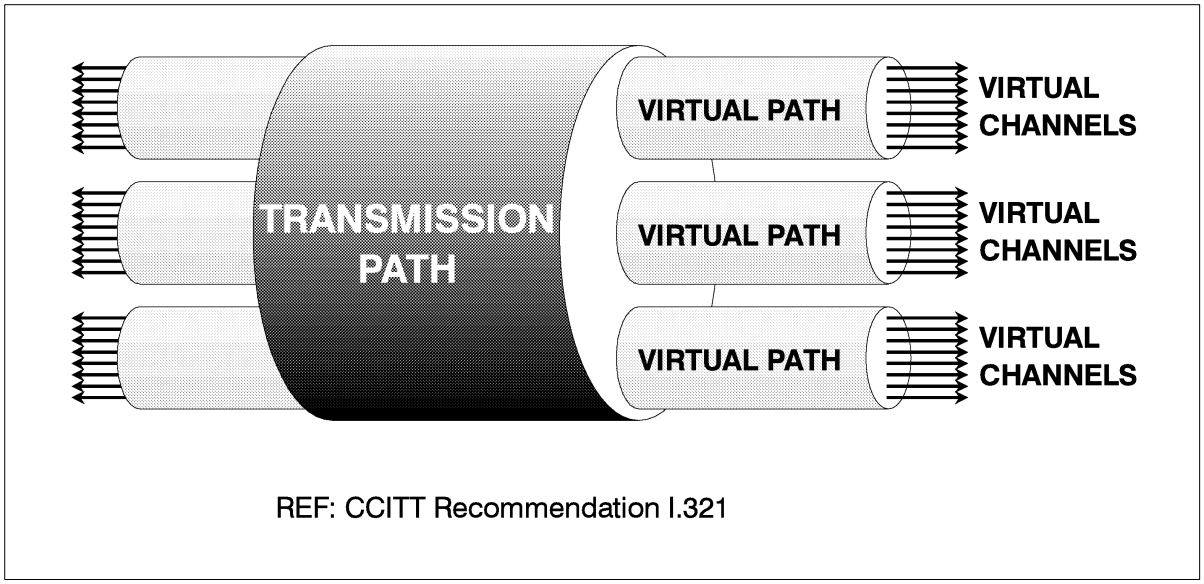

Figure 10.10 ATM VCI–VPI Relationships

Virtual Paths may be considered to be 'bundles' of Virtual Channels and may therefore be used to route a common group of cells together. An analogy would be that the VPI represents a virtual 'leased line' between two sites, with the VCI's being used to carry individual calls, as shown in Figure 10.11 below.

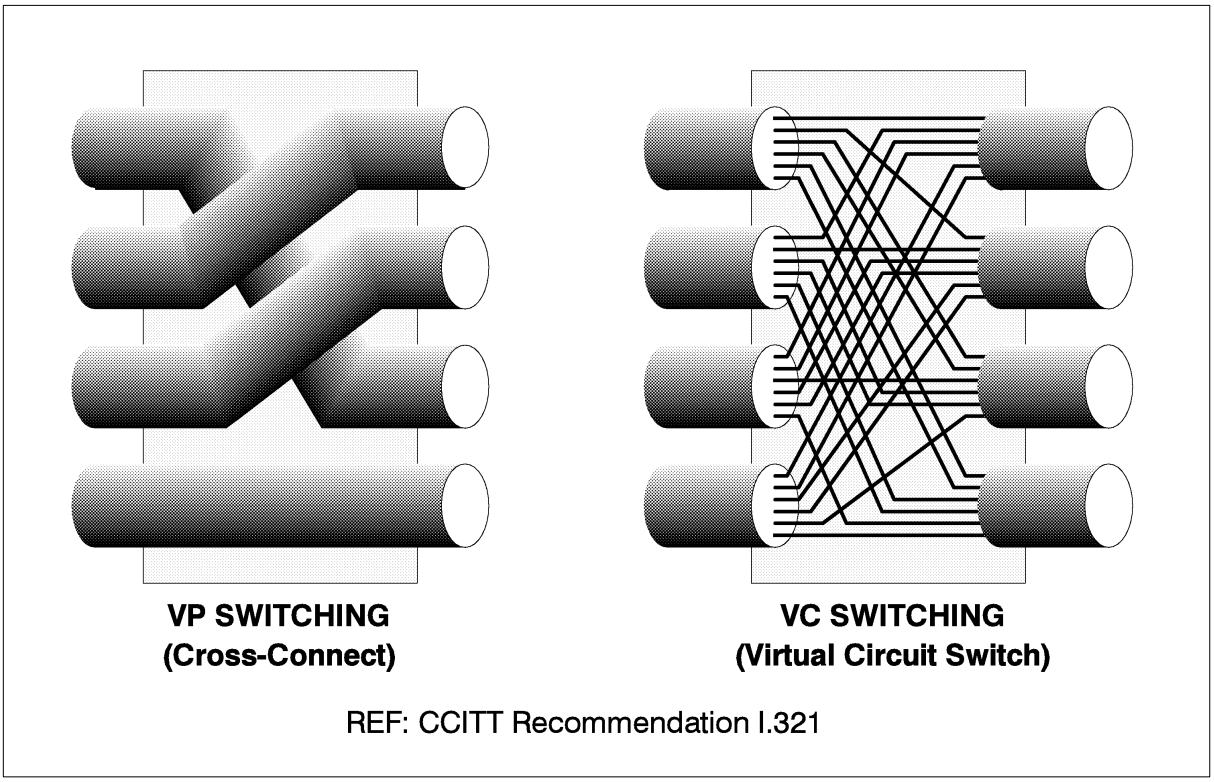

Figure 10.11 ATM Cell Routing

The ***Header Error Correction (HEC)*** byte is an error detection/correction mechanism for the cell header contents only to avoid mis–routing of cells. The definition of the HEC code and its intended use is actually part of the PHY layer standards, but is included here briefly for convenience. Protection of the data field is left to higher layer protocols. The HEC byte can detect and correct single–bit errors in the header and detect (only) multi–bit errors. It is up to the network to decide what to do with multi–bit errors, although the most likely course of action is to discard the cell and report the error. Another use of the HEC byte is for ***Cell Delineation***. The HEC is

continually evaluated on a bit–by–bit basis in order to provide a synchronization mechanism at the receiver – an ATM cell HEC has been identified when the HEC output is 0, so the location of the rest of the cell can be easily determined.

The ***Generic Flow Control (GFC)*** bits are not, currently, fully defined but are provided in order to support future flow control mechanisms within the network.

The ***Priority*** bit is used to indicate whether the cell can be discarded by the network in times of extreme congestion. For example, discarding a cell containing video data may result in a brief but acceptable sparkle on a monitor, whereas discarding maintenance and call set–up information may result in (an unacceptable) loss of service

The PHY Layer

This layer defines how cells are transported from terminal to network, between switching nodes within the network and then from the network to the destination terminal. The medium used in public networks is most likely to be optical fibre at 155 Mbits/s and above. As mentioned previously, ATM cells can be transmitted in a framed, synchronous format or in an unframed asynchronous format. For the public networks, a synchronous mechanism has been defined based on the bit rates defined in the CCITT ***Synchronous Data Hierarchy (SDH)*** and the ***SONET*** (Synchronous Optical NETwork) frame structure developed in the US. This mechanism allows the packing of ATM cells into the SONET/SDH 2–D frame format, rather like bricks or tiles (the use of a synchronous transmission medium is sometimes referred to as ***Synchronous Transfer Mode (STM)***)

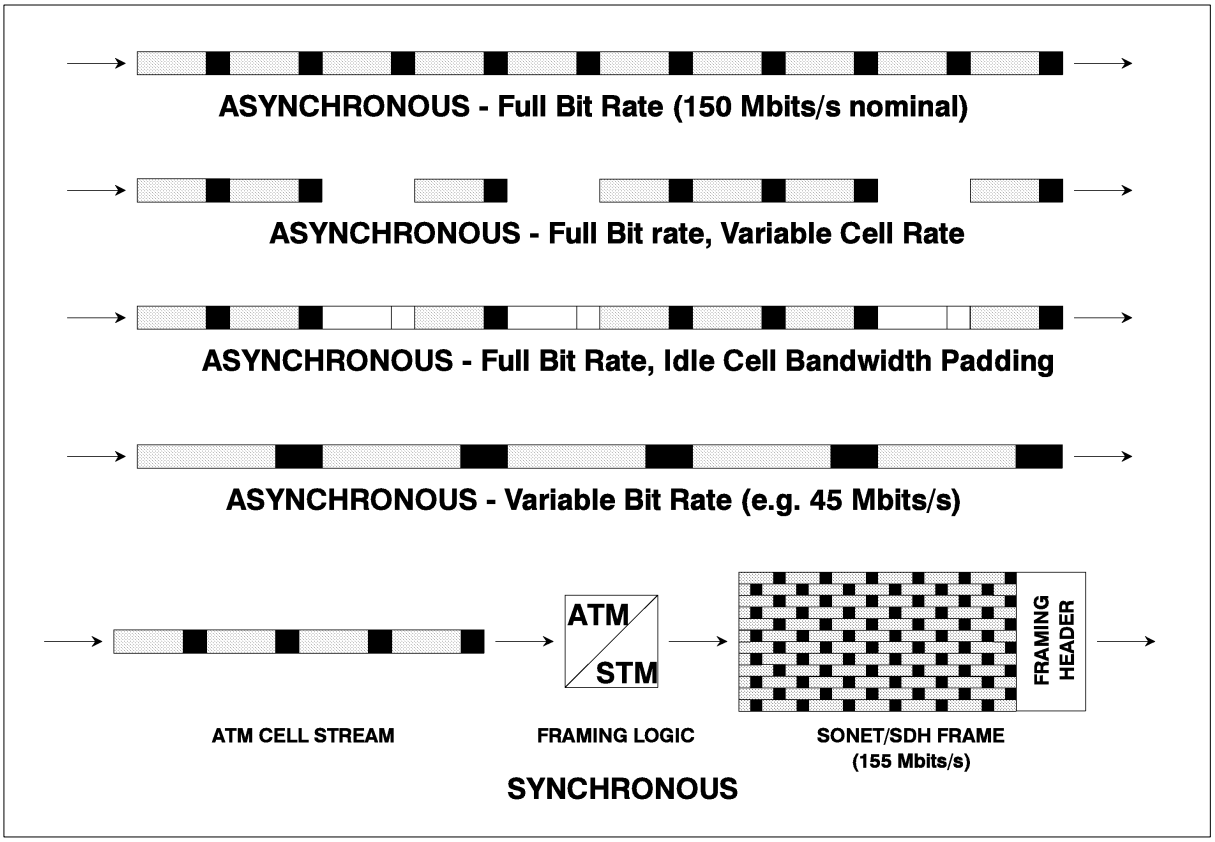

Figure 10.12 Transmission variations for ATM Cells

Various proposals have been made for PHY layer standards for private networks, including the use of FibreChannel. In private networks, however, there is an incentive to use existing, installed twisted pair cable where possible and this is likely to constrain the data rate available. Cost issues at the user terminal end are also likely to work against a full SONET/SDH implementation, at least initially. AT INMOS in Bristol we have been using transputer links as a physical medium for carrying ATM cells in our demonstrators, since they come free with every transputer. Work is in progress to develop drivers for ***DS-Links*** to copper and fibre, since they offer a cheap and attractive physical interconnect and could form the basis for low–cost ATM connections over distances of 10–100 metres, or even further, to a local ATM switch (further information on physical drivers for DS-Links can be found in Chapter 4 of this book and some of the issues surrounding their use to carry ATM cells are discussed later in this Chapter).

10.3 ATM Systems

In describing the use of DS-Links, routers and transputers in the construction of ATM systems we need to consider the types of equipment needed to build an ATM network. We make here a relatively naive split between ***Public*** Switching Equipment, ***Private*** Switching Equipment and ***Terminal*** Equipment, as shown in the diagram below, and then describe ways of applying the communications and processor architecture of the transputer to this equipment.

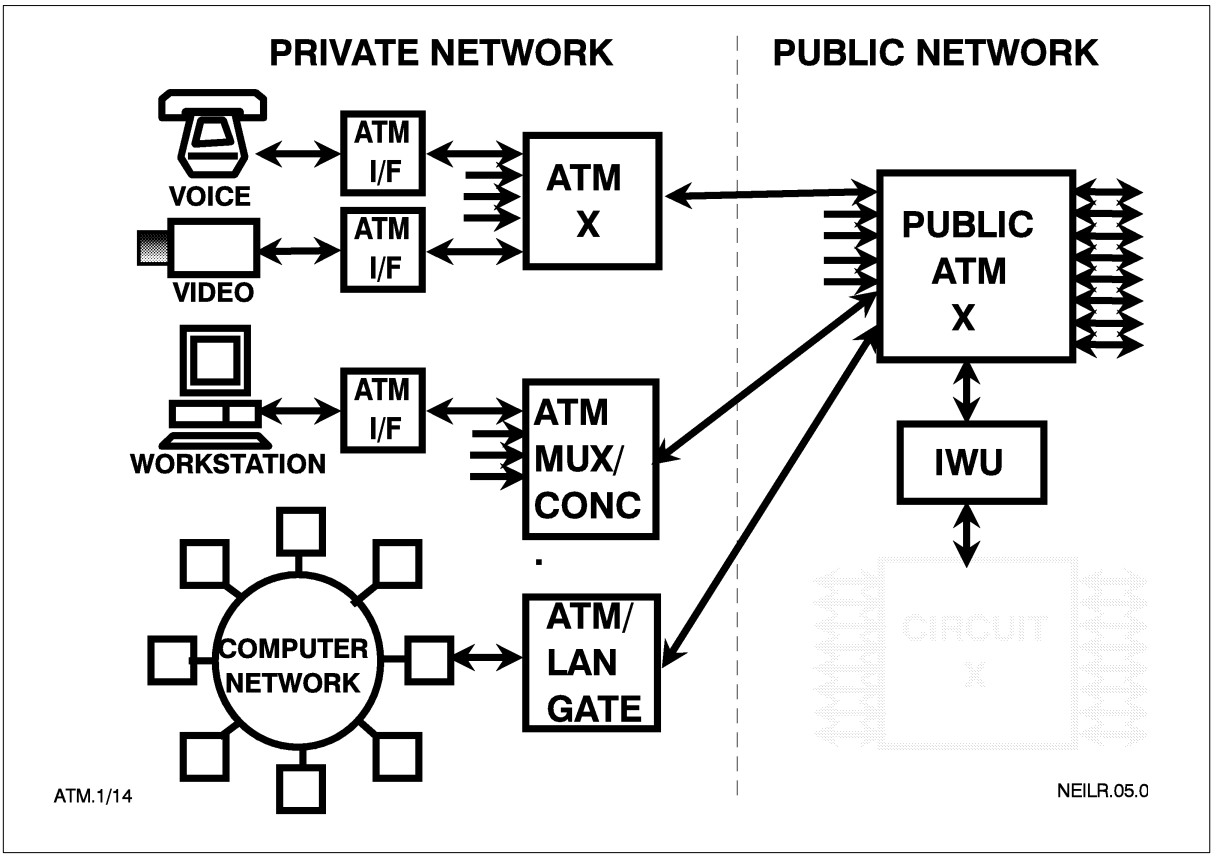

Figure 10.13 Possible ATM Network Equipment Environment

The first efforts in ATM date back to the early 1980's and until about 1991 the bulk of ATM development was focused on ***Public Switching*** systems, particularly in Europe. Field trials of public switching equipment have already started in some areas, but most public activity is expected to begin in late 1993/1994 with more widespread field trials of CCITT compliant equipment in the US, Europe and Japan. How long it will take for the general availability of 155 Mbits/s services on the public network is anyone's guess. As with any major infrastructure investment like this it must be expected to be a 10–20 year program. During this period, the developing ATM network

must coexist with existing networks [5], hence the requirement for ***Interworking Units (IWU)*** between the two.

Since late 1991/1992, there has been an enormous surge of interest in ATM for private use, mainly driven by computer manufacturers and users predominantly in the US. The creation of the 'ATM Forum' and the release of draft standards by Bellcore/Apple/Sun/Xerox for the use of ATM as a sort of local area network has spurred interest considerably. The initial use of ATM in this area is clearly as an interconnect fabric for existing ethernet/token ring/FDDI networks (there is considerable debate as to whether this interconnection will be done by 'pure' ATM or a MAN, such as IEEE 802.6). This would require ***Internetworking*** equipment capable of converting from LANs/MANs to ATM and then connecting into the public network. Ultimately, the possibility of building small, cheap, high–bandwidth ***Private ATM Switches*** for use in an office or building extends the idea closer to the user and offers the possibility of a seamless communication system, with the distinction between Local and Wide Area Networks finally disappearing.

If the cost of providing a physical ATM connection can be driven low enough, it becomes attractive to take ATM right to the desktop. An ATM ***Terminal Adapter*** in each workstation or PC would provide a fast communications medium capable of supporting voice, video and data traffic and would form the basis for widespread multimedia applications. Coupled with cheap ATM switches, mixed data could be sent or received from anywhere on the planet extremely quickly. First generation adapters would be board–level solutions, but there is plenty of scope to integrate this into a single–chip ATM terminal adapter later, when standards are firmer and the silicon technology more mature.

It seems reasonable to suppose that not all of these private terminals would necessarily require a full 155 Mbits/s ATM connection. Lower speeds between the terminals and the local switch would be sufficient, at least in the early years of use, and an ATM ***Concentrator*** could be provided to make efficient use of the connections to the local public switch. Operating at lower speeds, say sub–50 Mbits/s, also opens up the possibility of using existing cabling plant within buildings and offices.

So, having considered a possible environment for ATM equipment, let us now consider where the communications and processing architecture of the transputer can make a contribution towards realizing this network.

10.3.1 Public Switching Systems

Various fast packet switching architectures suitable for the implementation of ATM switches have been described. Indeed, this has been and still is the basis for an enormous amount of research and development activity around the world. Martyn De Pryckers book [2] gives a thorough description of most (if not all) of these architectures, as well as providing an excellent introduction to ATM principles and concepts.

Fast Packet Switch Model

In [4] a generic model of a fast packet switch is presented and we make use of such a simple model in order to illustrate where the DS–Link communications architecture and the transputer processor family contribute.

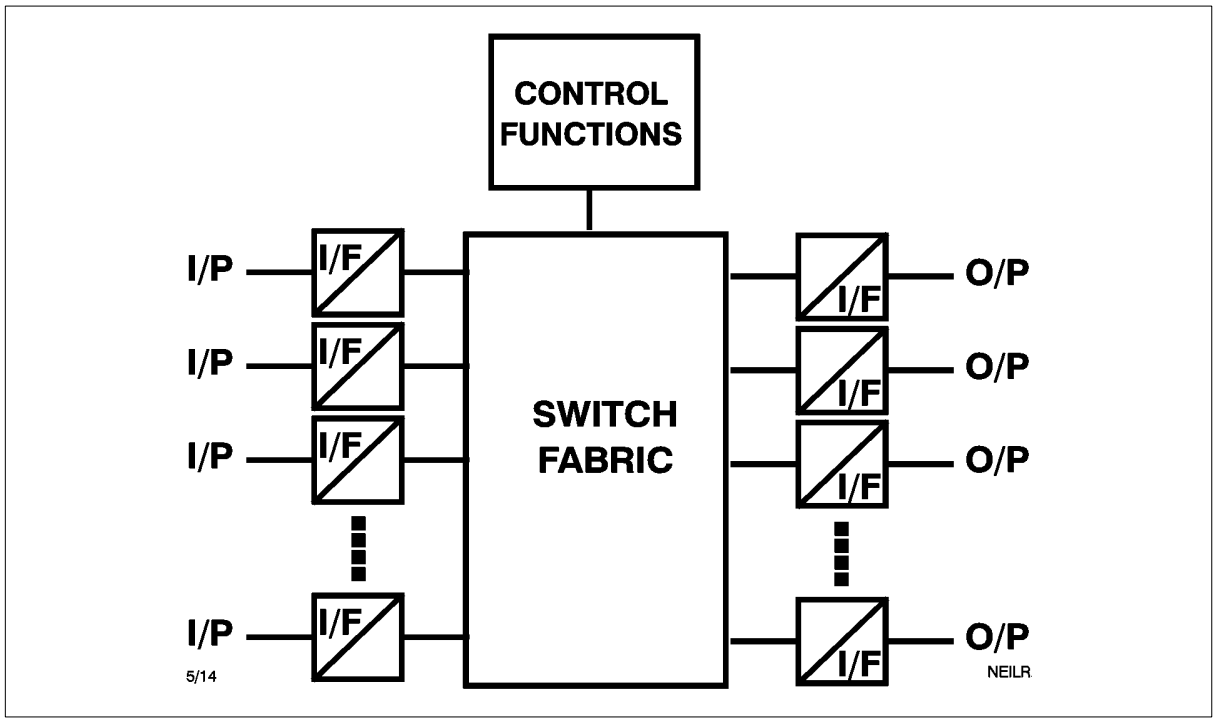

Figure 10.14 Generic Fast Packet Switching Architecture.

This basic architectural model has three main components:–

- Central Control functions (for signalling, control of the switch fabric and operations and maintenance)
- Input/output ports to and from the network
- A switching fabric

In a real switch each of these components will be a complex subsystem in its own right and each will require varying degrees of embedded computing and control. The usefulness of the transputer architecture is in providing the basis for the *control* of these complex subsystems and in particular as a ***distributed control system*** for the exchange as a whole.

Central Control Functions

Probably the most computationally intensive areas of the switch are the ***call–control computer*** and the ***billing*** (or ***call accounting) computer***, which form the central control and maintenance functions within the switch. The call control computer handles all of the signalling, call set–up/clearance and resource allocation for the exchange. It is a real–time function which, on a large exchange, has to handle hundreds of thousands or even millions of transactions per hour. It goes without saying that it needs to be reliable, since the allowable downtime for a main exchange is 2 hours every 40 years or so. Different manufacturers have different preferences as to whether a centralized or distributed architecture is used, but increasing processing requirements and the development of modular switches means that even centralized architectures are usually multi–processor in nature.

The billing computer tracks the use of the system by individual users in order, naturally, to provide billing information to the network operator. This is also a demanding task if millions of transactions per hour are involved and requires considerable processing power to handle the large transaction rate and database requirements. There is probably more emphasis on the fault–tolerant aspects of this part of the exchange than anywhere else; to the network operator, losing the billing computer means losing money!

Both the billing and call–control computer represent the major software investment in a public switch. The software maintenance effort is huge; hundreds, even thousands, of software engineers are needed to maintain the software on these systems in each of the major manufacturers. At a colloquium at the Royal Society in London called 'Telecommunications Beyond 2000', one of the senior executives at AT&T in the US pointed out that they have 6 million lines of code on their main switch, which grows at about 1/2 million lines a year. Supporting this sort of investment **and** adding new features and functionality for new services becomes increasingly difficult, especially when in time a mature, single–processor or shared–memory multiprocessor computer approaches the limits of its processing performance.

The advantage of the transputer architecture here is purely as a ***scalable, multiprocessing computer***, which is capable of being used in machines with up to many thousands of processors. The communications architecture of the T9000, for instance, is designed to provide a means of building such large computers free of the performance constraints experienced by shared–memory machines. This same architecture also supports various redundancy models economically (via the serial links), so ***fault–tolerant*** computer systems can be built in a straightforward fashion.

On existing (non–ATM) switches it should be possible to migrate towards such a parallel architecture for these computers, rather than outright replacement of existing machines, in order to preserve as much as possible of this existing software investment. A network of transputers could be provided as an ***accelerator*** to an existing billing computer, for example, to take some of the more intensive load off the existing machine. On new ATM switches, however, there is an opportunity to build a new architecture for these functions right from the beginning, one which is capable of growing with the demands of the application.

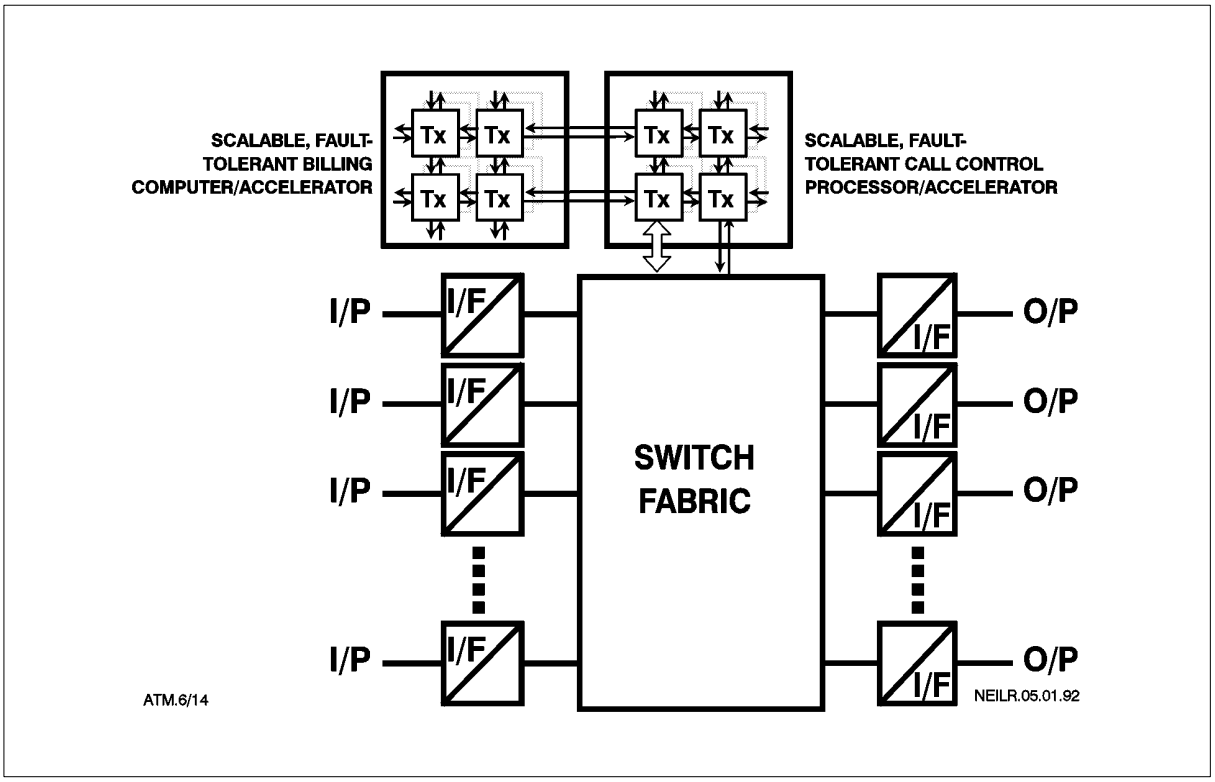

Figure 10.15 Billing/Call Control Application

If a transputer–based multiprocessor is used for the call–control functions, it will be necessary for it to communicate with the ATM traffic carrying the signalling and maintenance information around the network. This traffic is transmitted using ATM cells (naturally) with reserved values for the cell header, so that they can be detected, decoded and acted upon by the control functions in the exchange. This maintenance traffic rate is actually quite low (less than 5% of the total ATM

bandwidth) so carrying it around directly on the DS-Links within the control computer is no problem, even if the actual ATM traffic rates rise to 622 Mbits/s and beyond. A simple ASIC to interface between DS-Links and the ATM cell stream is all that would be required, with an AAL function provided in software on the transputer to extract the signalling data.

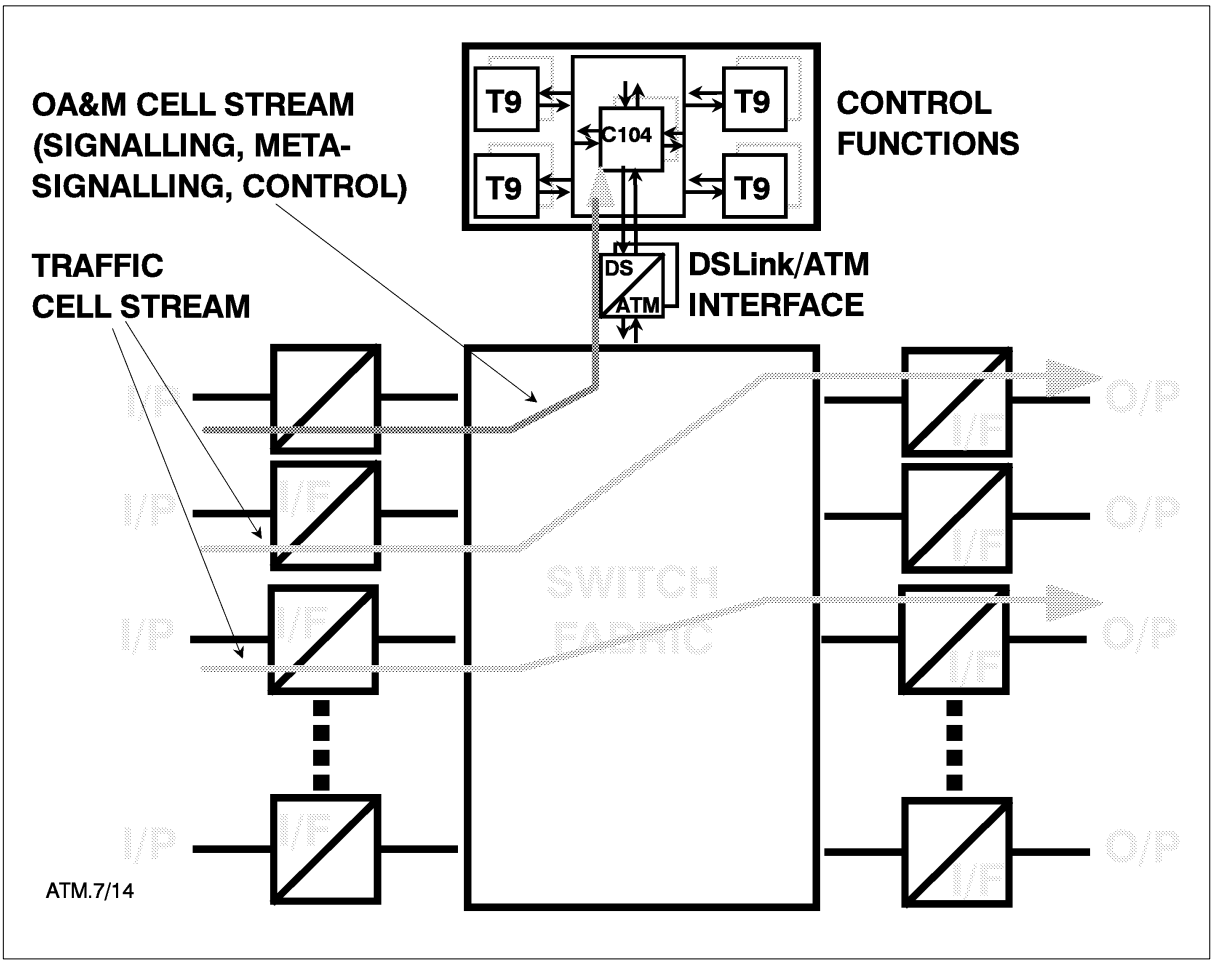

Figure 10.16 Interfacing to ATM Maintenance Traffic

ATM ‘line cards’ on a public switch need to be fast and reasonably intelligent. ATM cells arrive at the line card about every 3 μs and header translation, policing functions and error checks all need to be made on each cell on the fly. It isn’t possible to do all of this in software (certainly not economically) and a full hardware solution is expensive and inflexible. The combination of a fast, inexpensive micro like the transputer and some dedicated hardware functions is a good compromise that provides a balance between performance and flexibility. The context switch time of the T4 transputer of 600–950 ns means that some useful processing time is still available even if it is interrupted on every cell, although in most instances the hardware could be designed to interrupt the processor on exceptions only. It would be possible, for example, to perform the header translation operation using a direct table look–up, but use hardware for the HEC verification. However, the real value in having a fast but inexpensive micro on the card is the ability to track statistical information for use by the operations and maintenance functions, report faults and take recovery action where necessary.

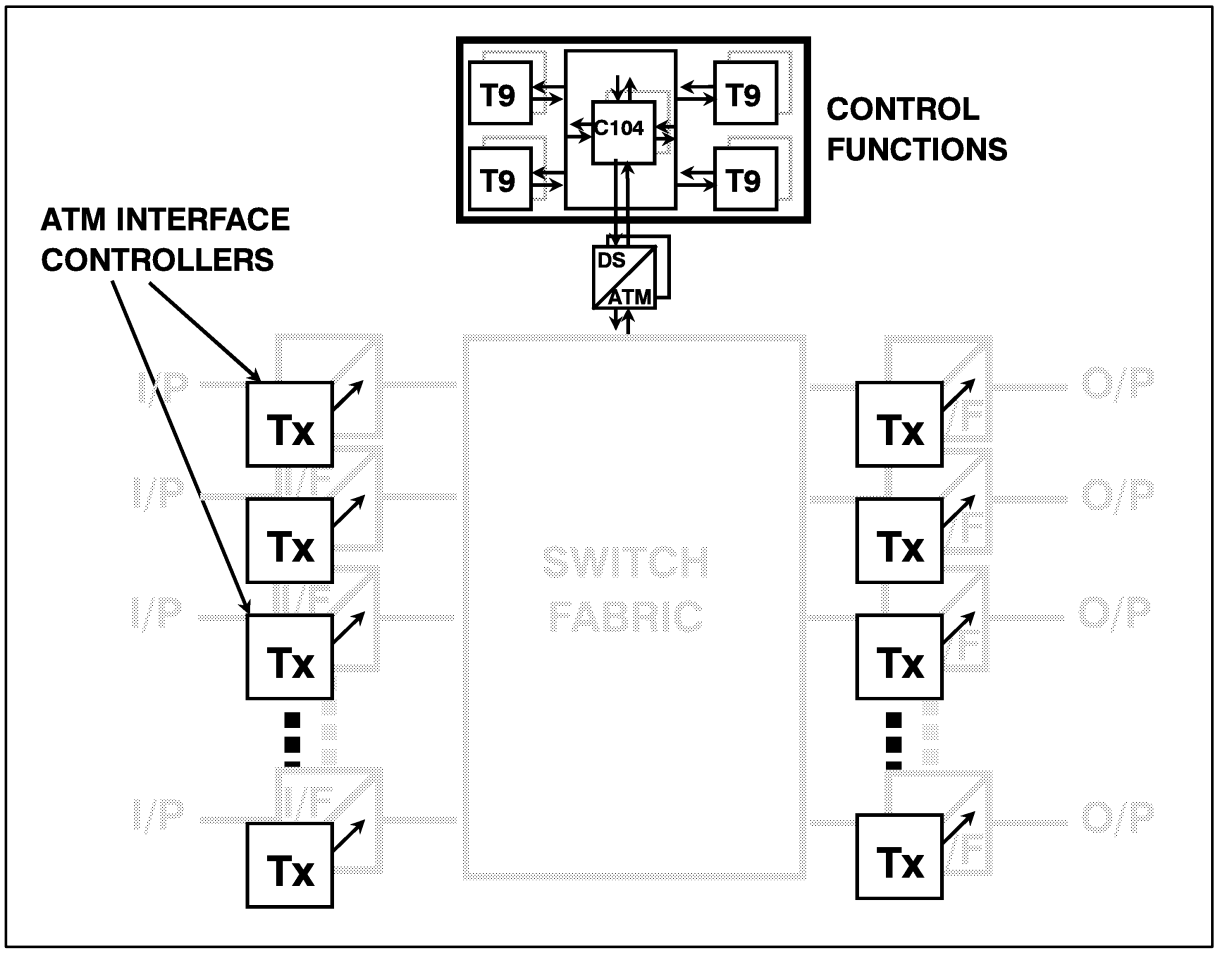

Figure 10.17 Transputers as Embedded ATM Interface Controllers

Network Interfaces

These line cards will typically consist of a hardware interface to the ATM/STM line, some logic to handle HEC checking, etc., an internal interface to the switching fabric and access to the transputer, via interrupts and memory. RAM will be required for program and data (translation look–up tables, etc.). The basic idea is shown below in Figure 10.18. The dotted line indicates where future integration is possible using semi–custom technology.

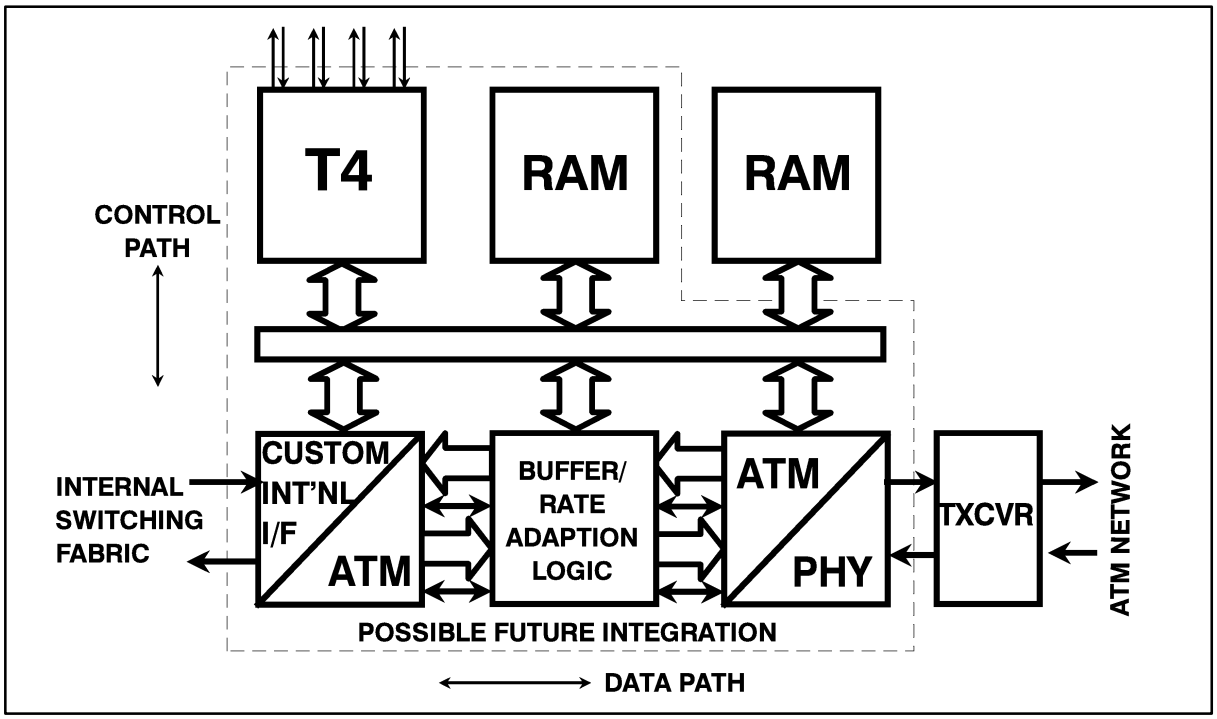

Figure 10.18 Possible ATM 'Line Card'

Such a line card is essentially a uniprocessor application, so the use of the transputer serial links for multiprocessing is not required. However, the serial links are very useful in other ways; for program download and debugging, test and diagnostics.

Putting software in ROM on the line card is undesirable from an upgrade and maintenance point of view. It would be better to be able to download code from some central point within the exchange. This could be achieved either by sending code via the switch fabric (possibly using a small boot ROM for cold–starts only) or by sending it down the transputer serial links (performing cold starts via the ***boot–from–link*** capability).

If the serial links are brought to the edge of the line card they can be used for testing in one of two ways. First, they can be used as part of the production test of the card by integrating them with an ATE system. Test code can be downloaded into the transputer (via the links) which runs entirely in the internal RAM. This code can exercise, at full speed, the external interfaces of the transputer as part of the test functions of the ATE system. Secondly, if the serial links are accessible while the card is in service in the exchange, it is a useful 'entry point' for a test engineer to interrogate the system. Better yet, if the serial links are internally interconnected, the switch control computer itself can use them to interrogate the system.

Switching Fabric

In a large public switch the data rates and requirements of the switching fabric are such that it is most likely to be built out of dedicated hardware and will in itself be a very complex subsystem. It is **not** appropriate to consider the use of the C104 for this fabric directly, nor to consider that the (non–maintenance) ATM traffic could be carried via transputers. However, like the network interfaces, there is considerable benefit in embedding processors within the hardware to provide intelligent control of the fabric. Maintenance and statistical measures can be provided, routing tables updated (if applicable) and the fabric monitored and reconfigured under fault or congestion conditions.

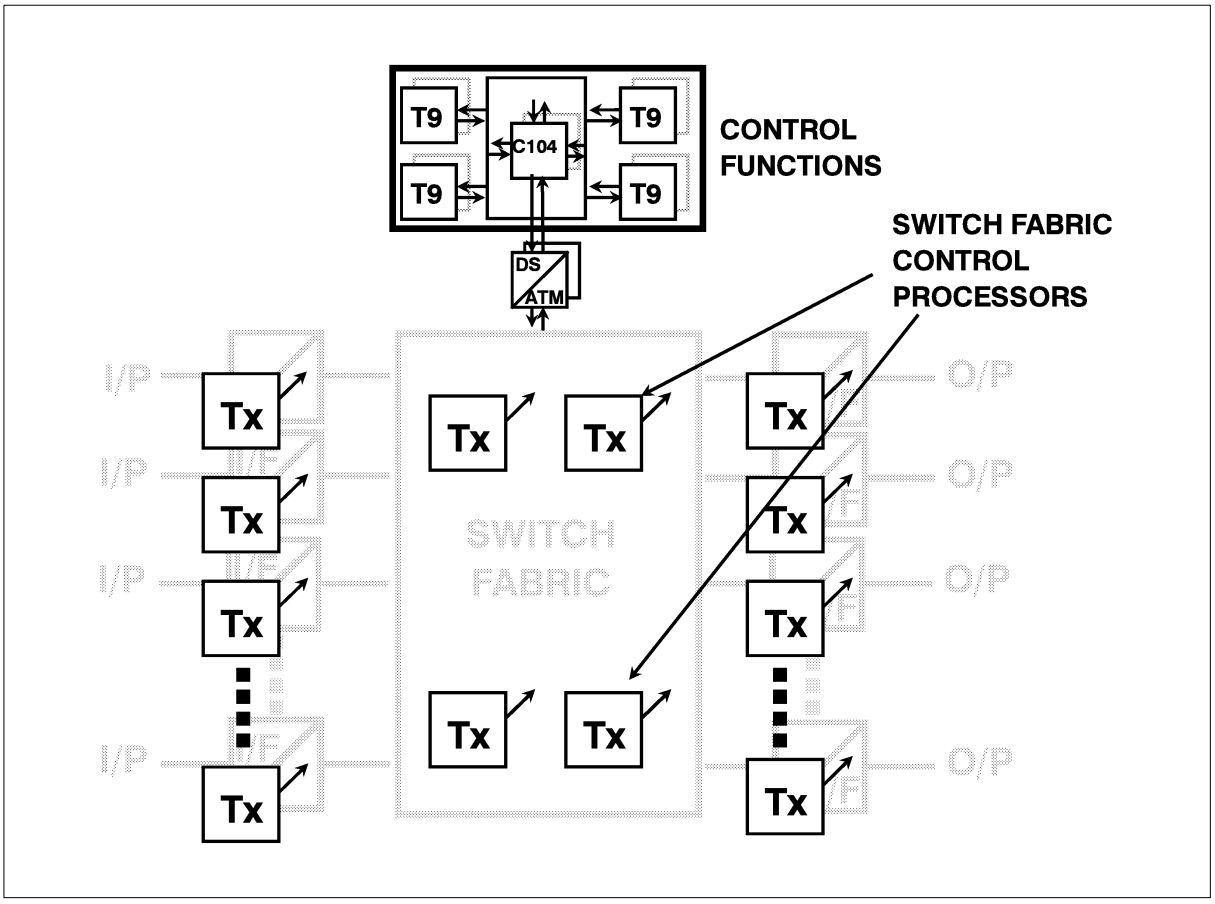

Figure 10.19 Embedded Switch Fabric Control

If desired, the links available from the control transputers can themselves be interconnected via a C104 network to provide a distributed control plane which is quite independent of the main ATM switch fabric, as illustrated in figure 10.20.

There are many other possibilities for mixed processor/hardware intelligent switching fabrics that remain to be investigated, and it is hoped that further ideas will be presented in future papers.

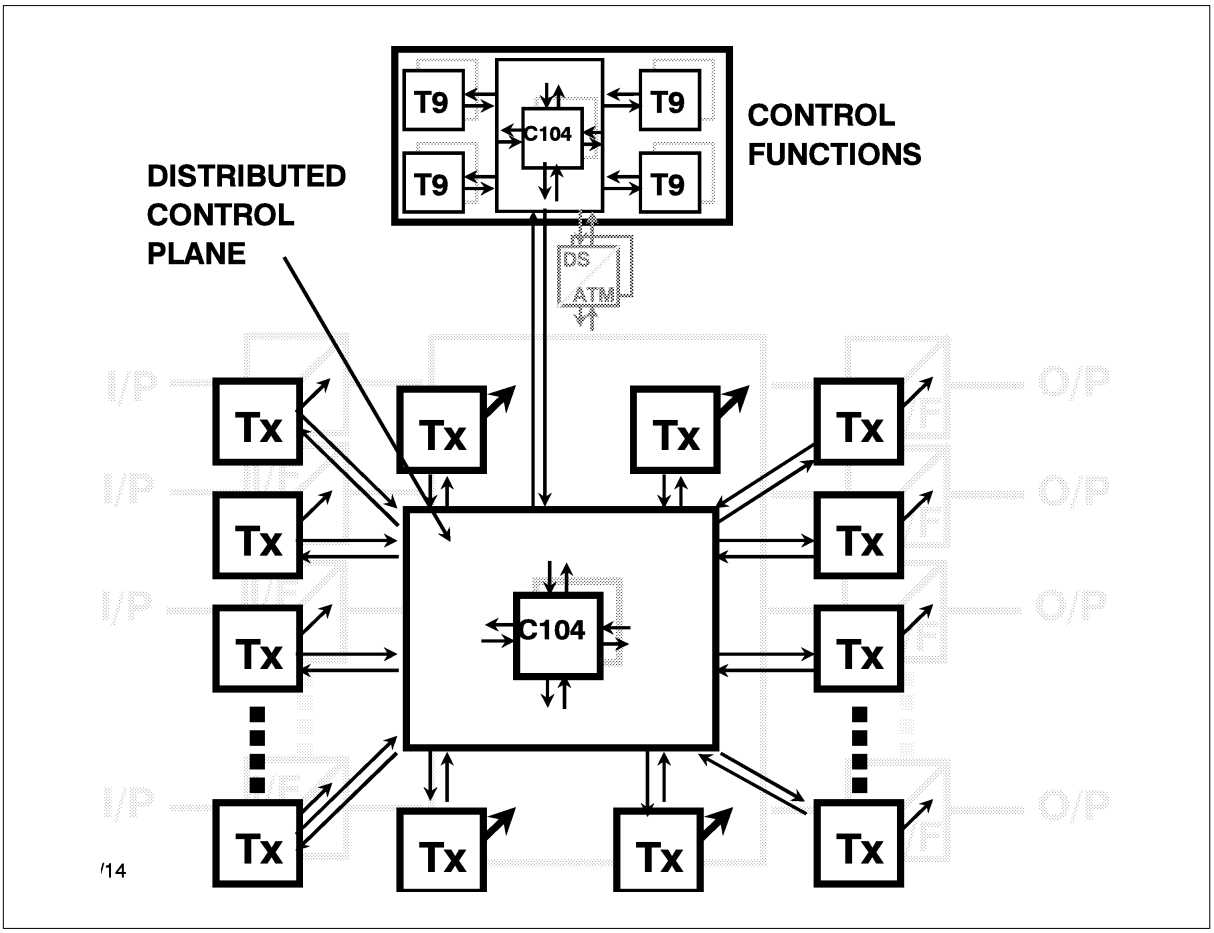

Figure 10.20 Distributed Control Plane

10.3.2 Private Switching Systems

All of the preceding discussion on public ATM switches also applies to private systems. However, there are some important differences:–

- the machines are not as large
- the bandwidth requirements are likely to be lower
- they are far more cost sensitive.

The nature of the ***Customer Premises Equipment (CPE)*** market is also likely to require much faster design cycles for the equipment, probably 1–2 years as the technology becomes established. The dynamics of the market are likely to place manufacturers under pressure to provide modular, flexible designs which can be upgraded, either in terms of performance, services or number of connections. Greater emphasis than in the past will be placed on network reliability, so the fault–tolerance aspects of the equipment will come under closer and closer scrutiny.

A Generic Private ATM Switch

The main difference from the point of view of applying the transputer architecture is that in private systems it is now possible to consider to use of the C104/DS-Link as the basis for an inexpensive switching fabric. Many current ‘campus’ ATM switches have been derived from existing bridge/router technology and are based on shared bus interconnect schemes. These do not provide scalable performance, as the common bus quickly becomes a bandwidth bottleneck. However, using the communications architecture of the transputer we can construct a scalable ***Generic ATM Switch*** for private applications [6].

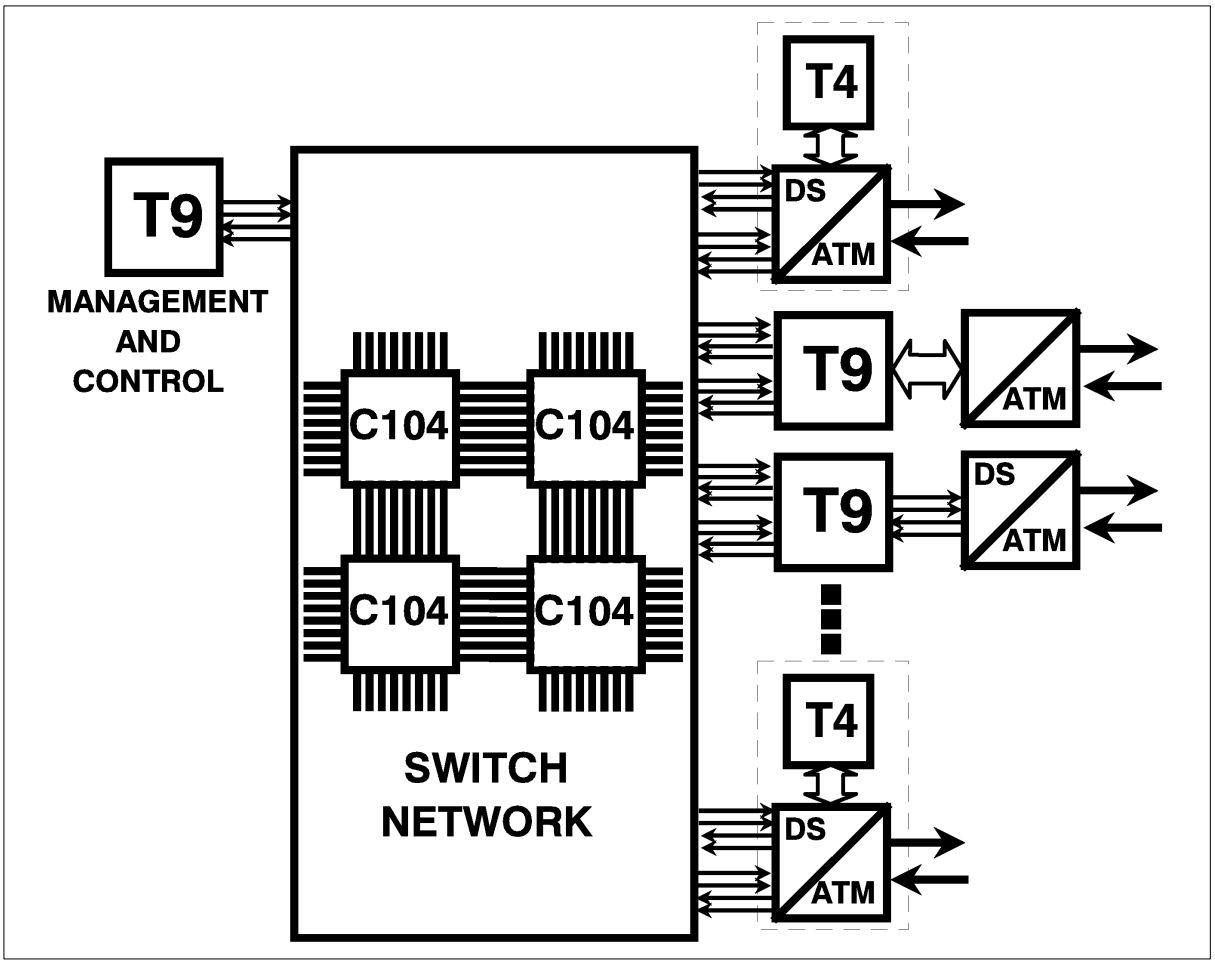

Figure 10.21 Generic Private ATM Switch

At its simplest, this switch may be considered to be a no more than a ‘black box’ multiprocessor computer running an ATM program. It has interfaces around the periphery to allow it to talk to the transmission network outside, but in essence it exploits the architectural similarity of message–passing/fast–packet–switching machines discussed earlier. Figure 10.21 shows illustrates several ways in which ATM interfaces can be built for such a switch, depending on cost/performance trade–offs required.

There are some important features of the DS-Link/C104 communications architecture which apply in its use as a fast packet switch:–

- DS-Links are ***cheap***
- The C104 can be used to build ***Scalable*** networks
- The in–built ***Flow Control*** mechanisms at the Token layer of the DS-Link protocol mean that the fabric is ***Lossless***, that is, no data packets/cells are ever lost internally due to buffer overflow within the fabric itself. Buffer dimensioning/overflow issues are moved outside the switch fabric to the network interfaces at the edge.
- DS-Links may be ***Grouped*** to provide high bandwidth connections within the fabric. This can be used to:–
    - minimize congestion for a given desired bandwidth
    - carry high–bandwidth traffic (for example, to 622 Mbit/s ATM).
    - provide redundant paths in the fabric for fault–tolerance reasons

- Link grouping on input can be used to avoid ***Head–Of–Line Blocking*** (congestion at the input to the switch fabric) by statistically increasing the chances of accessing the fabric (this is illustrated in the previous diagram)

- Universal (randomized) Routing can be used to avoid the 'hot–spot' congestion which can sometimes occur with certain systematic traffic patterns (for a full examination of this see Chapter 7).

- Traffic of ***any packet length*** may be carried by the C104 fabric. Only traffic intended directly for the (current) T9000 needs to be segmented into 32–byte packets, although longer packets may affect the congestion characteristics of the fabric.

Since the C104 fabric is simply an interconnect mechanism for a multi–processor computer, it is trivial to add further processors to this architecture to perform the Management and Control functions. As many as necessary can be attached to the switching fabric and they can communicate with the 'line cards' directly using the same fabric.

Multiple switching planes could also be used to provide either:–

- Separate control/data traffic planes

- Different planes to handle different traffic priorities

- Redundant Fault–tolerance within the overall switching fabric

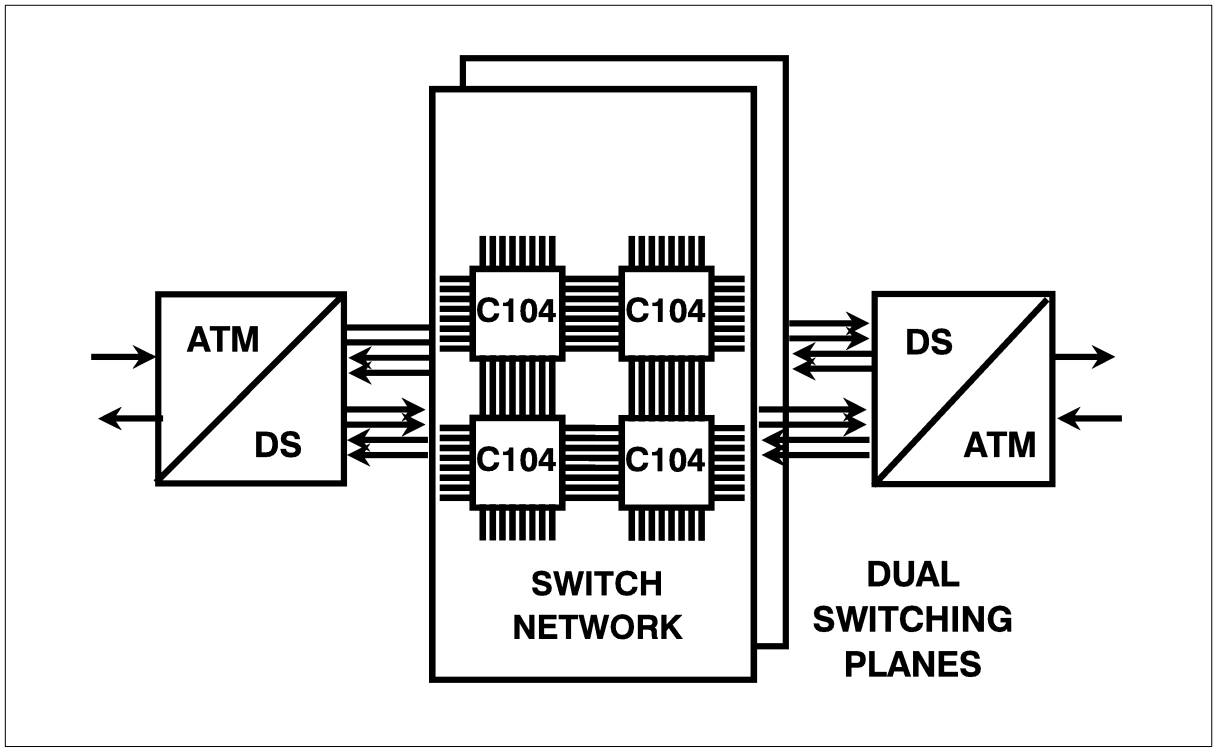

Figure 10.22 Multiple Switching Planes

Generic Internetworking Unit

One of the attractive aspects of this architecture is that interfaces to other networks, for example ethernet, token ring, FDDI, frame relay, etc., can be added very easily and so provide a ***Generic Internetworking*** architecture:–

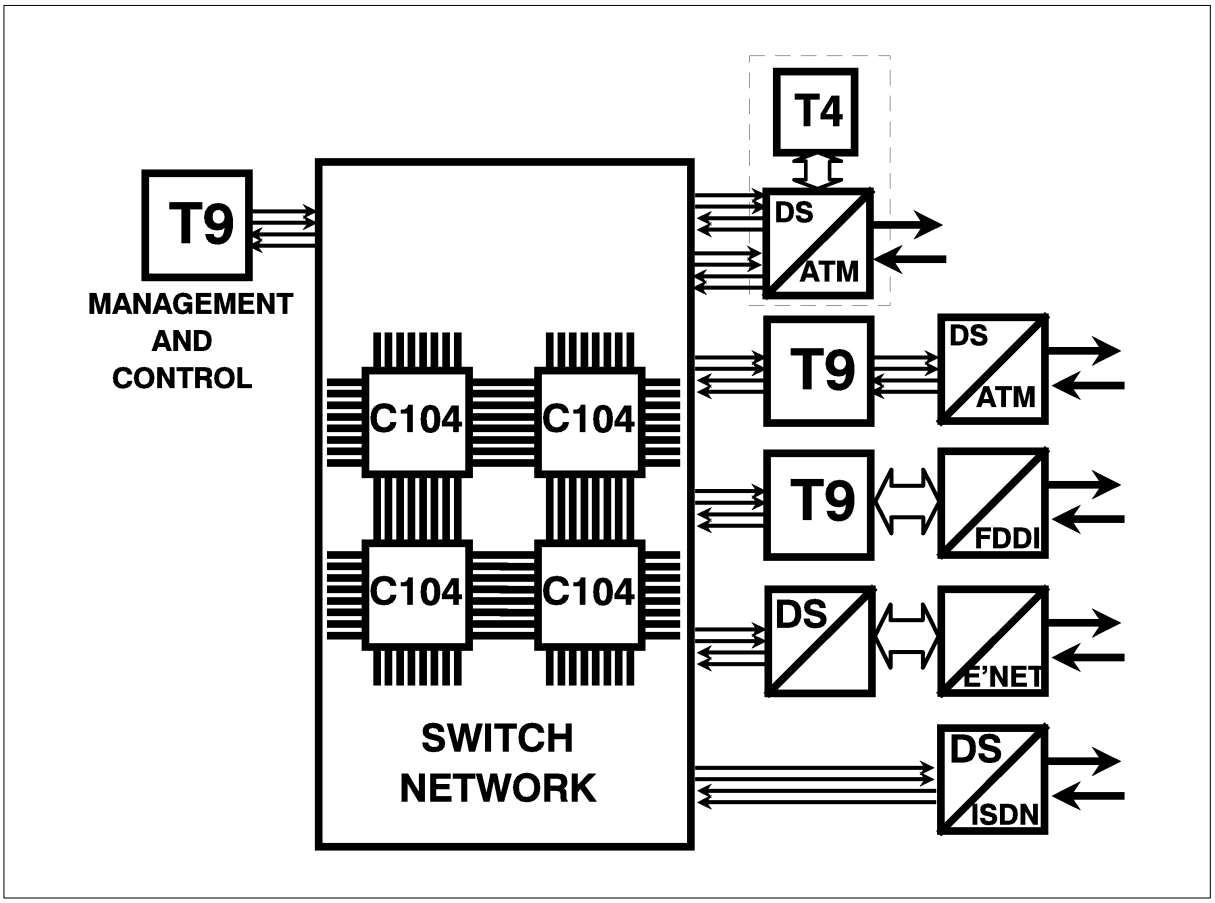

Figure 10.23 Generic Internetworking Architecture

Since we have simply built a computer (and one which is scalable in performance at that) we can add additional computing performance where required. A ‘‘pool’’ of processors can be added to this system to provide high–performance protocol processing between the various networks. Indeed, ‘‘***Parallel Protocol Processing***’’ techniques may be applied. For example, a ‘farm’ of T9000 processors may be made available to perform frame–by–frame AAL conversion from ethernet to ATM.

ATM Concentrator

We can extend the internal serial interconnect beyond the confines of our ‘black box’ ATM computer to provide a low–cost, lower speed entry point from an ATM terminal into the network, a sort of broadband serial concentrator. By using appropriate physical drivers, we can use the DS-Links directly to carry ATM cells asynchronously over local distances into the switch. Apart from cost advantages (since the DS-Links are inexpensive and the complication of full STM framing is not required) the DS-Links also provide an in–built ***flow–control mechanism*** which would provide an automatic means of ‘throttling’ the traffic flow back to the source. This is something which is currently missing from the ATM standards (GFC bits notwithstanding) and which could be added without requiring any alterations to the ATM standards by using the DS-Links. The availability of flow control to the source would considerably ease the buffering/performance design issues within the local switch as well as reducing the hardware/software costs associated with header policing on input.

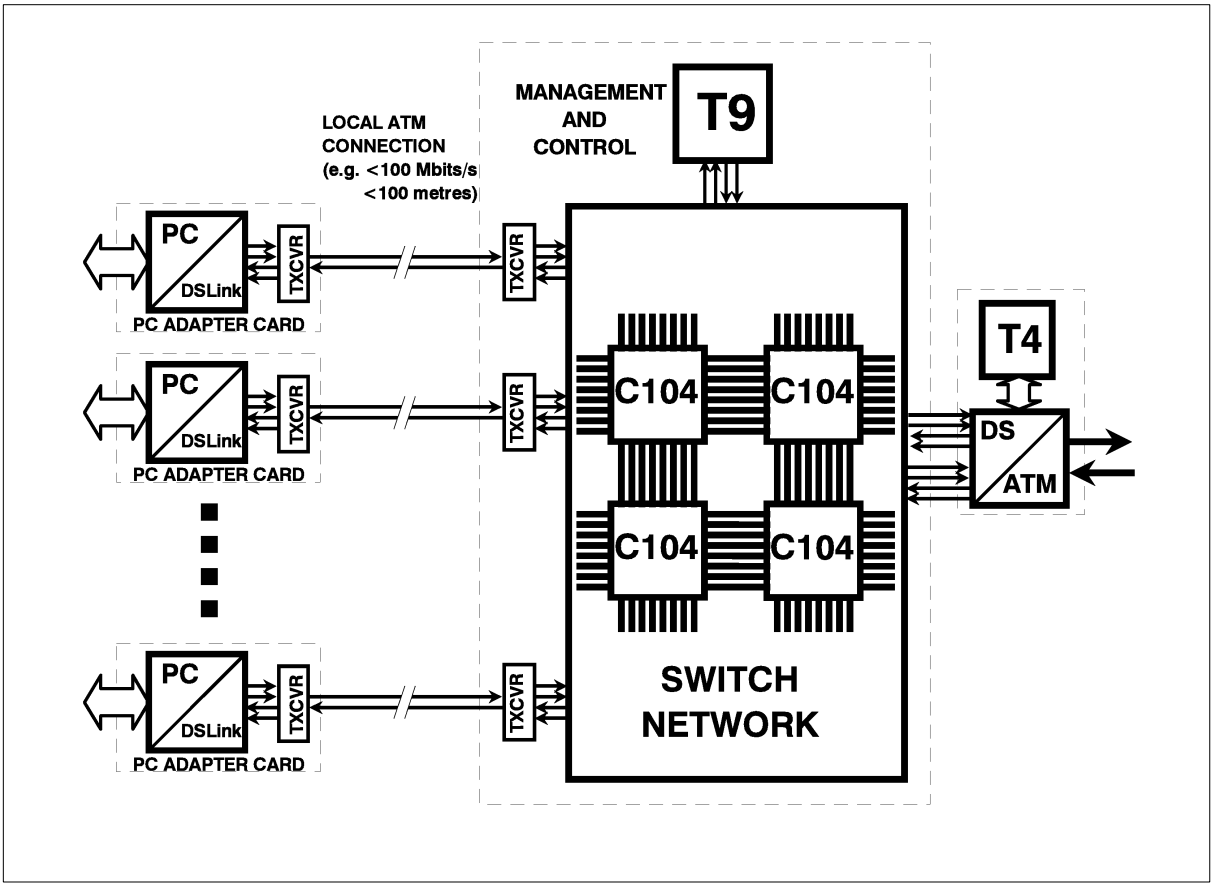

Figure 10.24 Low Cost ATM Concentrator

Issues and techniques for using DS-Links at a distance have been covered in Chapter 4 and such an interconnect could probably provide a very low cost entry–point into an ATM network for end user terminal equipment.

Private ATM Network Interface

The basic issue concerning the network interfaces for our private C104–based ATM switch is how to get ATM cells from the transmission system onto the DS-Links. Later in this Chapter a discussion is presented of the various ATM–DS-Link mappings that are possible and the performance issues that arise. Here, we consider the functional aspects of such interfacing for the moment.

The ATM line card must perform:–

1 **Rate adaption:**

- The need for rate adaption will vary depending on the speed and number of DS-Links provided at the line interface. In any case, some FIFO buffering will be needed to cope with slight rate mismatches caused by cell header processing, etc. More exotic methods may be added if the DS-Links are to run at a substantially different rate to the ATM line. Rate adaption between the DS-Link network and ATM can be provided by supporting one or more of the following:–
    - FIFO's to cope with traffic bursts
    - Inserting and deleting ATM 'Idle cells' (null cells for bandwidth padding) into a full–rate 155 Mbit/s ATM cell stream
    - Allowing the ATM clock rate to be varied (for example 1.5/2/34/45/155 Mbits/s. This may be allowable for private networks, but not on the public side).

2 ATM Cell Header Processing:

- HEC checking and generation for the ATM header
- Policing functions
- Header translation

3 Packetisation:

- Encapsulation of ATM cells into DS-Link packets for transmission via the DS-Links to the switching/processor network

4 STM/ATM Interfacing:

- Interfacing the ATM cell output stream to the synchronous, framed transmission system, where required on the public network. This will typically be done in hardware.

5 Management and Control:

- HEC error counts
- Policing parameters/algorithms
- Translation table updates, etc.

There is a hardware/software 'threshold' to be determined here which is the subject of further investigation. Some functions are obviously suited for hardware implementation, others for software. There is a grey area in between for functions such as policing and header translation, where the exact split between hardware and software could vary. A simple block diagram of a proposed network line card is given in the diagram below. The dotted line indicates where scope exists for a semi–custom integration of the card onto a single device in future.

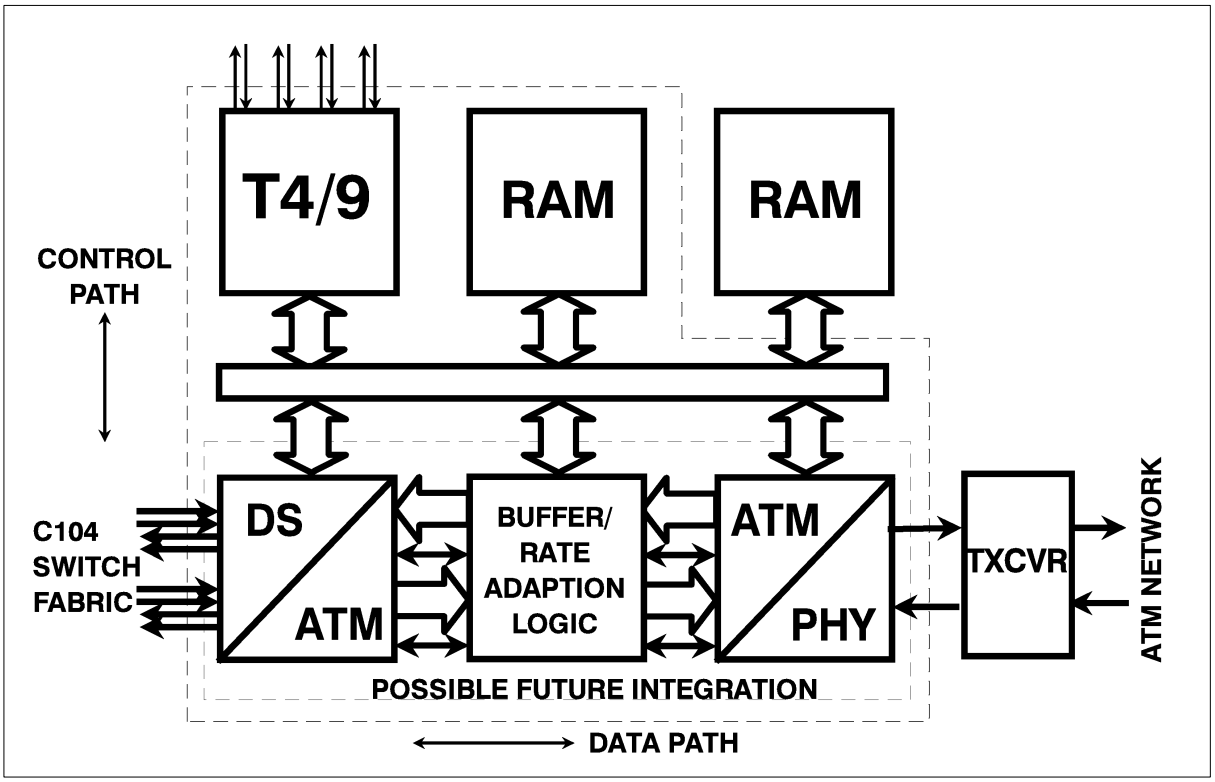

Figure 10.25 Simple ATM–DS-Link Network Interface Card.

10.3.3 ATM Terminal Adapters

Current PC's and workstations typically provide a fairly 'dumb' interface to a network in the form of a simple card to memory map an ethernet or token ring chip set into the hosts address space.

All interface control and higher layer protocol processing then falls on the host machine. It is becoming increasingly attractive to add a fairly powerful processor directly onto the network adapter cards in order to offload more of the protocol processing overhead from the host machine. As the bit–rate of the physical layer has increased in recent years, so the performance bottleneck in network access has moved to the higher layers of the protocol stack, which are more software/processor performance bound than the lower layers.

As 32–bit micro costs fall, we can apply many of the arguments for intelligent ATM line cards to an ATM Terminal Adapter and it becomes sensible to consider ‘smart’ rather than ‘dumb’ adapters. However, instead of providing an interface to a switching fabric (proprietary or DS-Link) we need a shared memory interface to one of the standard PC/workstation buses. A terminal adapter will also have to run one or more of the AAL standards and this is another reason for having a fast micro on the card – the AAL layer can be quite complex, the standards are changing and it may be necessary to run multiple AAL’s to support, say, multimedia applications. This tends to mitigate against a hardware–only implementation and, like the line card, a hardware/software ‘threshold’ needs to be determined. Also, an ATM terminal adapter may not need to run at a sustained 155 Mbits/s rate, so it may be possible to sacrifice some performance in order to save cost by using software functions. In the end, the application requirements will decide.

A simple block diagram for a shared–memory PC Adapter card is shown below. A suitable ATM/PHY interface chip is assumed (these are now becoming available) and some appropriate system interfacing logic to load and store ATM cells in memory. Again, the dotted line shows the integration possibilities.

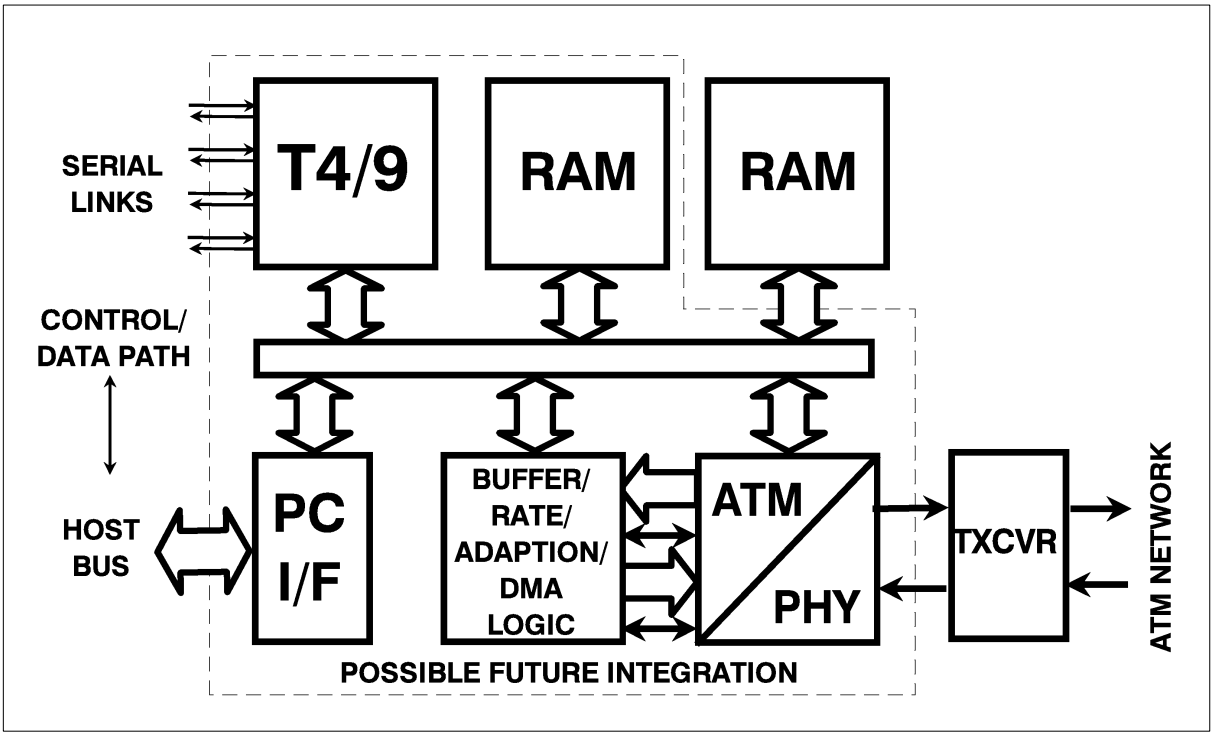

Figure 10.26 ATM–PC Terminal Adapter Card

In this example it is assumed that the AAL layer is handled in software by the transputer. A version of the AAL3 is currently being written for the transputer at INMOS in order to evaluate performance trade–offs and whether a software–only implementation is fast enough for modest applications. Details of this will form the basis of future papers.

An alternative form of Terminal Adapter can be envisaged for the control functions in a public or private switch. If a T9000 or multiple T9000’s are being used for the control then it may be necessary to interface the DS-Links of the T9000 straight to ATM. A relatively simple ASIC

would be required in order to do this and which would perform the rate adaption, ATM cell timing, packetisation and HEC functions described above. All other functions could potentially be performed in software, since the Maintenance and Control cell rate is very low.

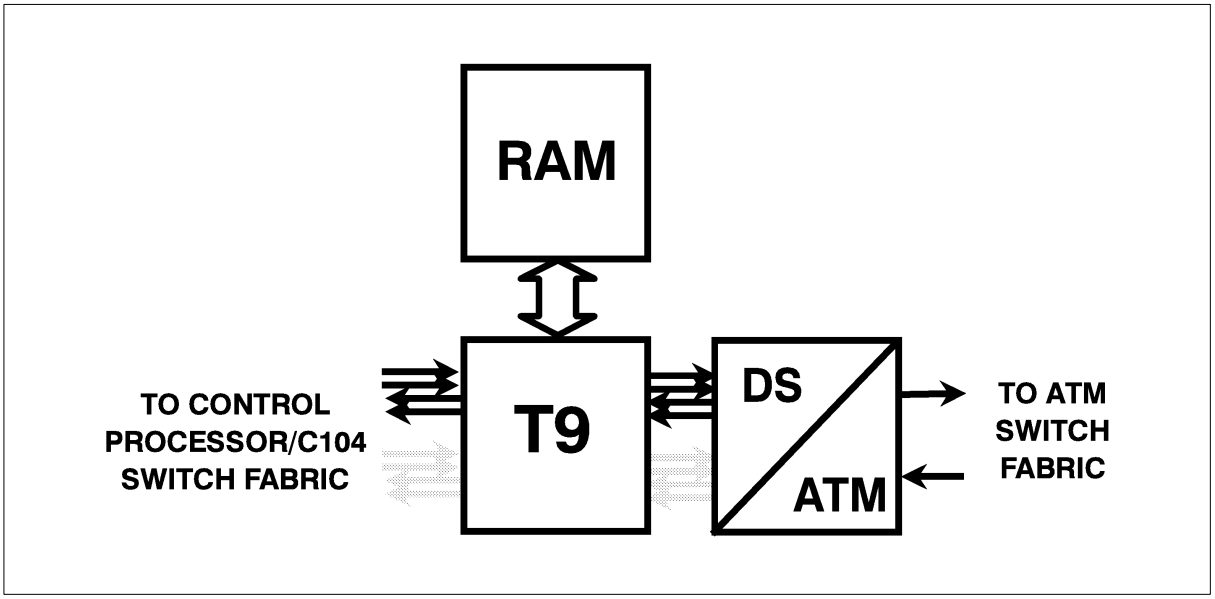

Figure 10.27 ATM–DS-Link Adapter Application

10.4 Mapping ATM onto DS-Links

In this section the issues associated with carrying ATM traffic over a DS-Link are considered. The DS-Link and the C104 do not require packets to be of a specified size, although the performance of the C104 chip has been optimized for use with small packets. This optimization is for parameters such as the amount of buffering on the chip and so variations in packet length will affect the blocking characteristics, although no packet data will ever be lost because the buffers cannot actually overflow. The current T9000 implementation, however, does place a constraint on packet length, presently of 32–bytes, and this means there are at least two ways of carrying ATM cells using DS–Links, depending on whether a T9000 is in the data path or not (this constraint could disappear in later T9000 versions if commercial issues justify a variant).

10.4.1 ATM on a DS–Link

In this section we consider the raw bandwidth the DS–Link can provide in order to carry ATM cells. We can consider 2 possible ways of using the DS–Links:–

- In a ‘T9000’ system with a full T9000 packet layer protocol implementation i.e. acknowledged packets of 32–byte maximum length
- In a ‘hardware’ system (built with no T9000’s in the data path) where the packet layer protocol implementation may be different, i.e. different packet length (and possibly without support for packet acknowledges).

A general performance model of the DS-Link is given in Chapter 6. This describes the data throughput of the DS-Link, given a specified message and packet size. It takes account of packet overheads, flow control and unidirectional and bidirectional use of the links. This basic model is extended here to show the throughput of the DS-Link carrying ATM cells, both with and without the full T9000 packet layer protocol. That is:–

- One ATM cell in single packet:–
    - One 53–byte packet on the DS–Link

- One ATM cell in 2 T9000 packets:–
    - One 32–byte data packet
    - One 21–byte data packet

Both unidirectional and bidirectional use of the DS-Links is considered in the following analysis. Data rates and throughput are calculated for DS–Links operating at 100 and 200 Mbits/s to give a representative performance spread.

10.4.2 Unidirectional Link Use

Single 53–byte packet

Suppose that the 53–byte ATM cell is sent as a single 53–byte packet. The packet has a one byte packet header and a four bit packet terminator. The flow control overhead is rounded up to one flow–control token, of four bits, per ATM cell. The total number of bits transmitted is the sum of the data bits, the header bits, the terminator bits, and the flow control bits, with the DS–Link transferring a byte of information as 10 bits.

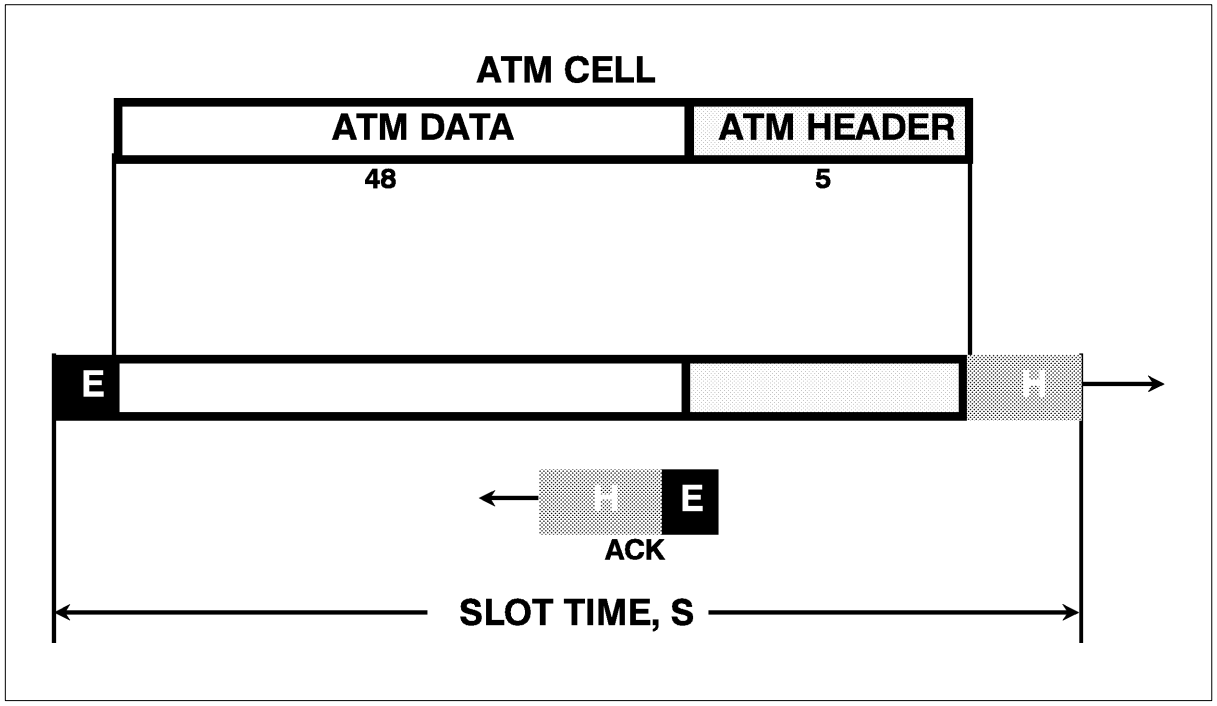

Figure 10.28 Unidirectional Single–Packet ATM–DS-Link Mapping

The net bandwidth available for the ATM traffic in this configuration has been calculated and is presented in Table 10.1 at the end of this section.

Double Packets

Now suppose that the largest packet contains 32 bytes of data, as is the case for a T9000. The 53–byte ATM cell will be transmitted as 2 packets, one 32 bytes long, the other 21 bytes. Each packet has a one–byte header and a 4 bit terminator. Again the overhead of flow control tokens is less than one token per ATM cell, and is rounded up to one token per ATM cell. This is an extra 4 bits.

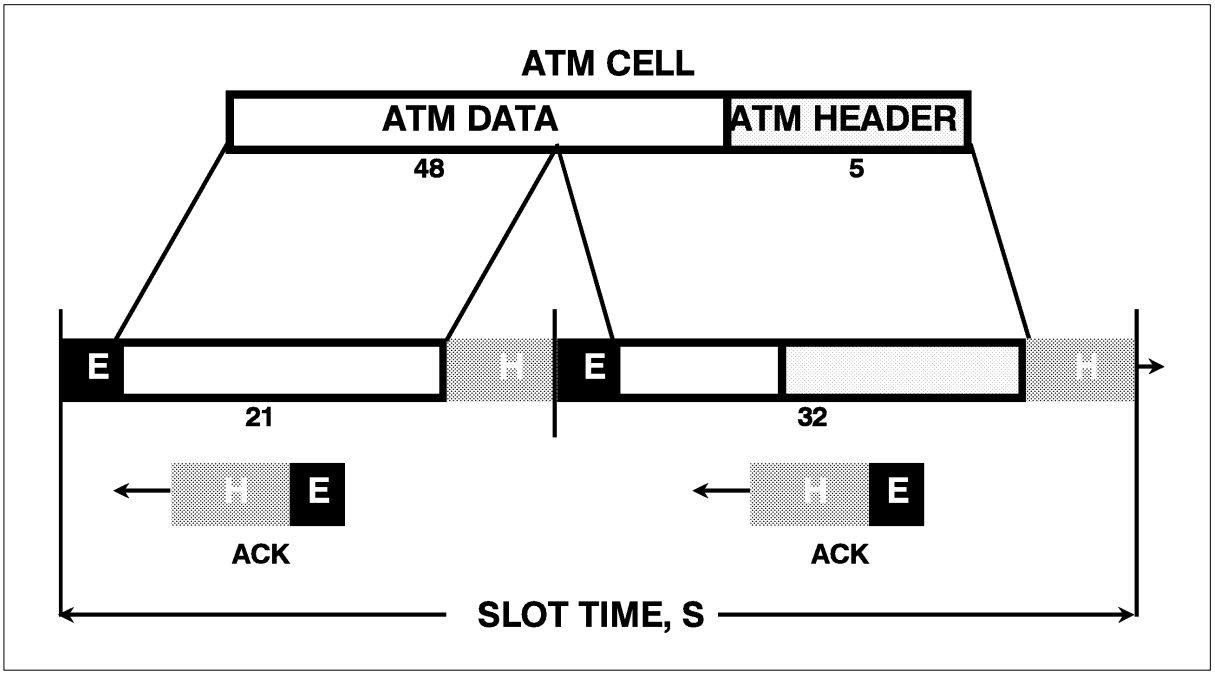

Figure 10.29 Unidirectional Double–packet ATM–DS-Link Mapping

Again, the bandwidth results are presented in Table 10.1 at the end of this section.

10.4.3 Bidirectional Link Use

When considering the effect of bidirectional operation, it is assumed that the inbound link carries a similar traffic load to the outbound link.

For bidirectional link use, the link overheads are greater. The link carrying the outbound data must now carry the acknowledge packets for the data on the inbound link, and vice versa. An acknowledge packet consists of a one–byte packet header, and a four–bit packet terminator. The outbound link must also carry flow control information for the inbound link.

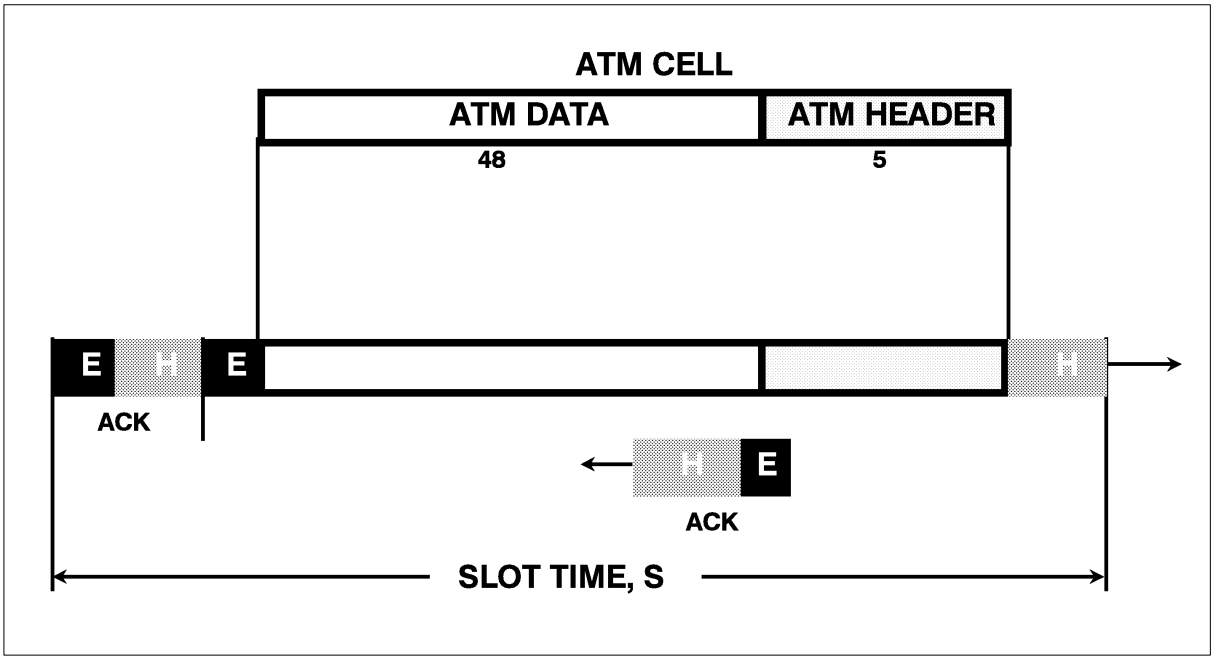

Figure 10.30 Bidirectional Single–Packet ATM–DS-Link Mapping

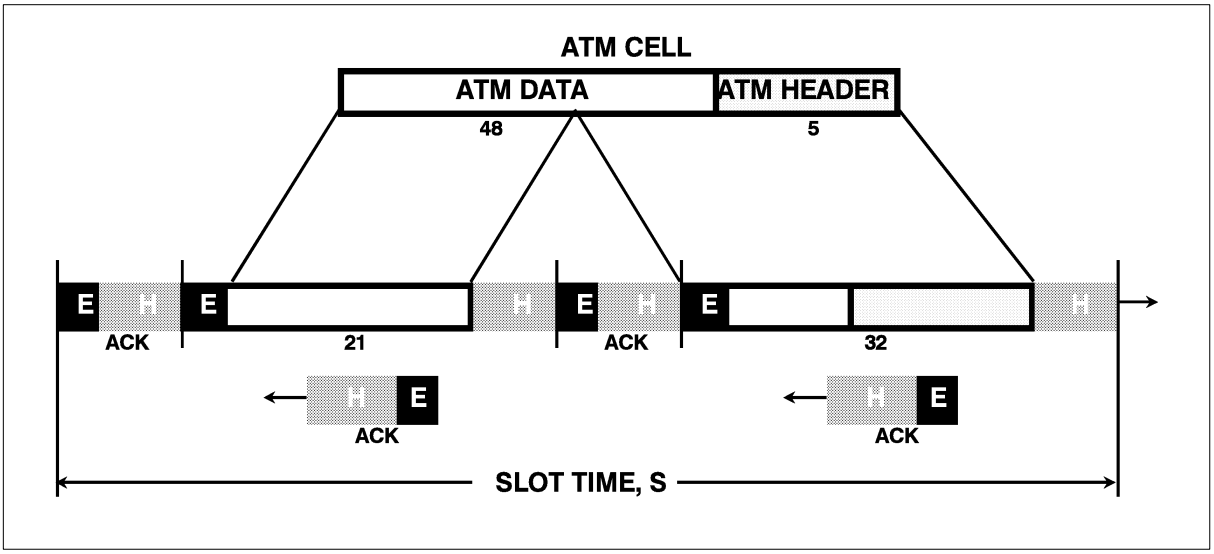

Figure 10.31 Bidirectional Double–Packet ATM–DS-Link Mappings

The results are presented in Table 10.1.

10.4.4 Summary of DS–Link Results

The performance results of the above configurations are summarized in the following table. The bidirectional throughput is available simultaneously in both directions on the link.

Link Speed (Mbits/s)	Max Packet Size (bytes)	ATM Cell Throughput (Mbits/s)		ATM Cell Rate (Cells/s)	
		Unidirectional	Bidirectional	Unidirectional	Bidirectional
100	32	75	69	177k	163k
	53	77	72	182k	175k
200	32	150	138	354k	325k
	53	154	144	363k	340k

Table 10.1 ATM Performance over DS-Links

The results indicate that 100 Mbits/s links would be more than sufficient to carry ATM at T3 rates, say sub–50 Mbits/s, so could be used to provide an economical point–to–point local connection from a terminal into an ATM concentrator. The DS-Link speed can be varied anywhere from 10 Mbits/s upwards in 5 Mbit/s increments, so the bit rate could be set appropriate to the physical medium used (only the transmit speed needs to be set, the receiver is asynchronous).

At 200 Mbits/s the DS-Links could provide a full–rate ATM connection unidirectionally and an only marginally slower (144 Mbit/s) bidirectional one, although if the traffic flow was asymmetrical this rate could be improved.

For interconnect use within a C104 switching fabric, single 200 Mbits/s DS-Links could provide full performance. However, traffic congestion issues would be far more significant than the marginal DS-Link bandwidth, so the use of grouped link pairs would be beneficial for blocking/congestion reasons. Assuming the fabric could support at best only 80% per–link throughput (based on the simulation models for a hypercube with Universal routing), this would mean that any DS-Link pair running at a bit–rate from about 120 Mbits/s up would support full–rate ATM traffic through the switch fabric (for a more complete treatise, see Chapter 7).

10.5 Conclusions

In this paper the use of the transputer architecture, its multiprocessing capability, its communication links and its packet switching interconnect capability, has been described in terms of applications within the emerging ATM systems market. Applications within public switching, private switching/internetworking and terminal adaption equipment have been considered. The main motivation in these discussions has been the convergence of architectures necessary to support message–passing multiprocessing computers (such as the transputer) and fast–packet switching systems (such as ATM). As each technology evolves and matures it is reasonable to expect an even closer relationship between the two.

A distinction has been drawn between the use of the transputer architecture in public versus private switching systems. In high–speed public switches the T9/C104 architecture is offered as a multiprocessing architecture for the *control plane* of the switch, with ATM traffic carried by a separate, dedicated (usually proprietary) switching fabric. Lower–speed private customer premises equipment has the potential to use a C104–based switching fabric directly, which could be used to carry both control **and** data traffic.

The use of transputers as uniprocessors, as opposed to multiprocessors, for building network termination and terminal adapter cards has also been considered. This area has a different set of constraints, mainly driven by cost, since ATM adapters and line cards will represent the volume end of the market. Silicon integration is the key, and the move to semi–custom techniques for transputer technology is an important factor here.

Given economical drivers for fibre and twisted pair, the DS-Links themselves offer their potential as a low–cost physical interconnect between terminals (PC's and workstations) and a local C104–based ATM concentrator. Transporting ATM cells and protocols across a DS-Link physical medium is very straightforward and provides relatively cheap office–scale connections with the added advantage of a built–in flow–control mechanism back to the source.

ATM is an exciting field and the transputer architecture offers a multitude of possibilities for building ATM systems. There are numerous combinations of ideas possible and no doubt in time many unique and interesting variations will emerge.

REFERENCES

[1] CCITT Draught Recommendations. Technical Reports I.150, I.321, I.327, I.361, I.362, I.363, I.413 and I.432. CCITT

[2] Martin De Prycker: 'Asynchronous Transfer Mode: Solution for Broadband–ISDN', Ellis Horwood, UK, 1991, ISBN 0–13–053513–3

[3] A.L.Fox and A.K.Joy: 'ATM–based Switching for the Integrated Broadband Network', Electronics and Communications Engineering Journal, 2(4), August 1990

[4] C. Hughes and A. Waters: 'Packet Power: B–ISDN and the Asynchronous Transfer Mode', IEE Review, October 1991

[5] Karl Anton Lutz: 'ATM Integrates Different Bit–Rates', Technical Report, Siemens AG, 1989

[6] C. Barnaby and N. Richards: 'A Generic Architecture for Private ATM Systems', Proceedings of the International Switching Symposium 1992, Session A8.4