This is the second part of Chapter 10 from the second edition of :

Networks, Routers and Transputers:

Function, Performance and applications

Edited by: M.D. May, P.W. Thompson, and P.H. Welch

© INMOS Limited 1993

This edition has been made available electronically so that it may be freely copied and distributed. Permission to modify the text or to use excerpts must be obtained from INMOS Limited. Copies of this edition may not be sold. A hardbound book edition may be obtained from IOS Press:

IOS Press
Van Diemenstraat 94
1013 CN Amsterdam
Netherlands

IOS Press, Inc.
P.O. Box 10558
Burke, VA 22009–0558
U.S.A.

IOS Press/Lavis Marketing
73 Lime Walk
Headington
Oxford OX3 7AD
England

Kaigai Publications, Ltd.
21 Kanda Tsukasa–Cho 2–Chome
Chiyoda–Ka
Tokyo 101
Japan

This chapter was written by C. Barnaby and N. Richards.

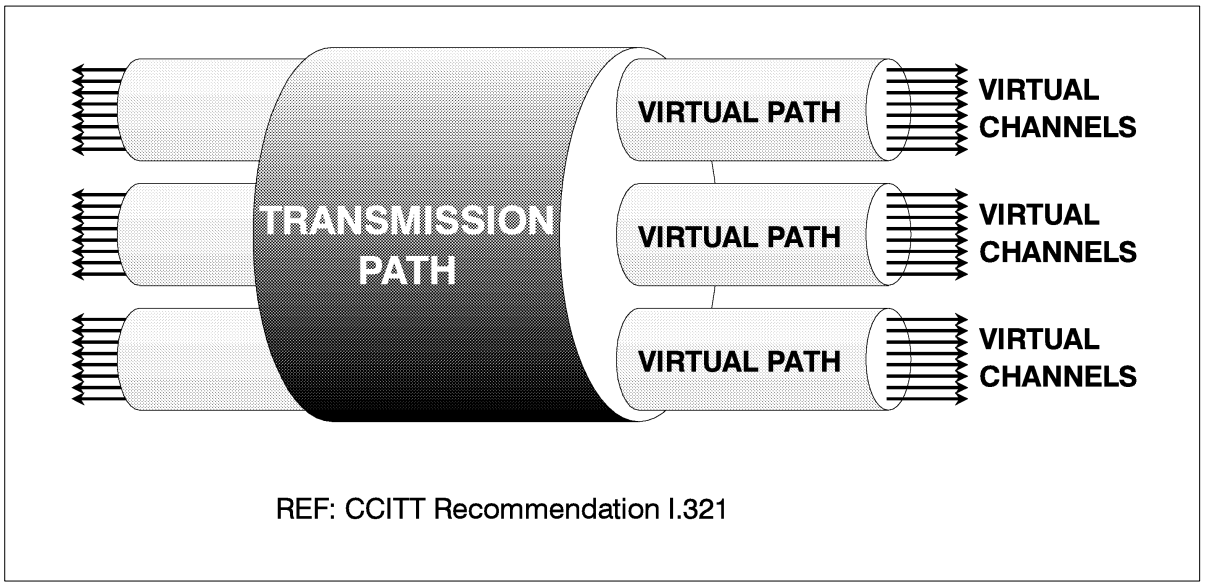

Figure 10.10 ATM VCI–VPI Relationships

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.

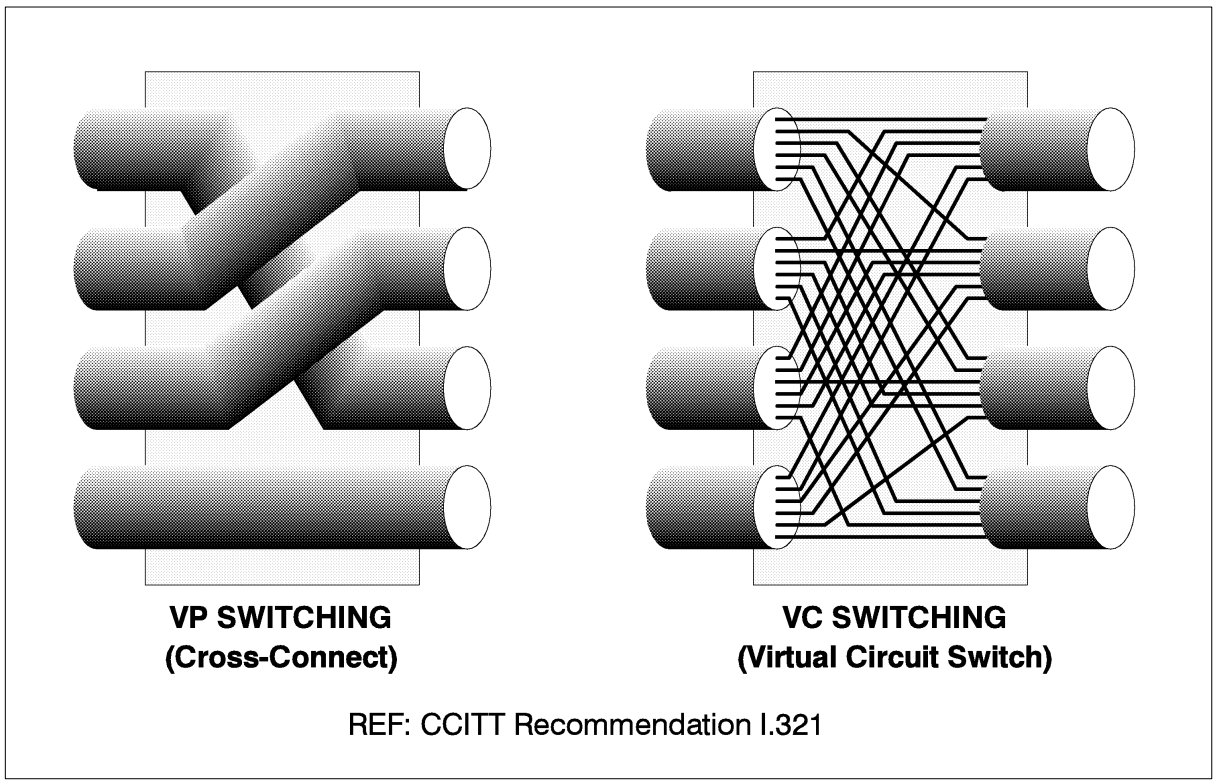

Figure 10.11 ATM Cell Routing

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)***)

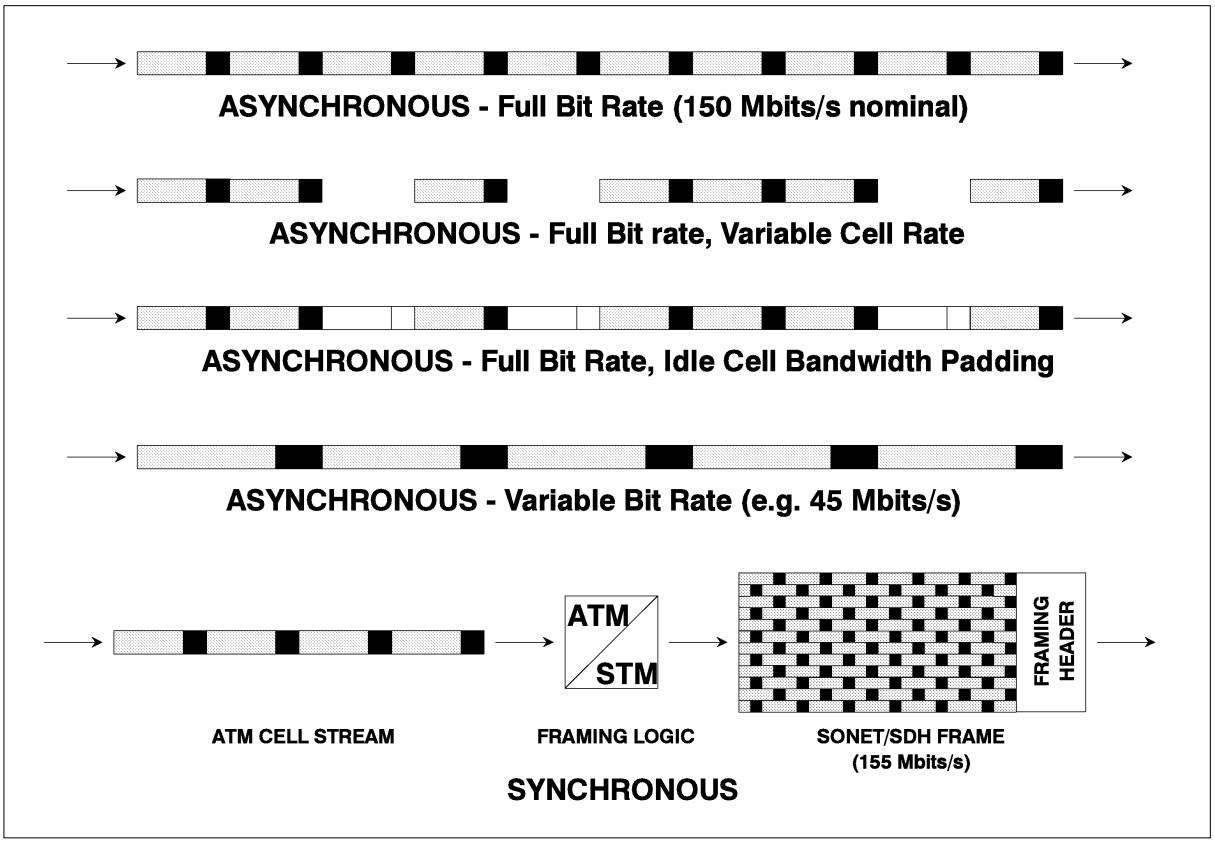

Figure 10.12 Transmission variations for ATM Cells

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.

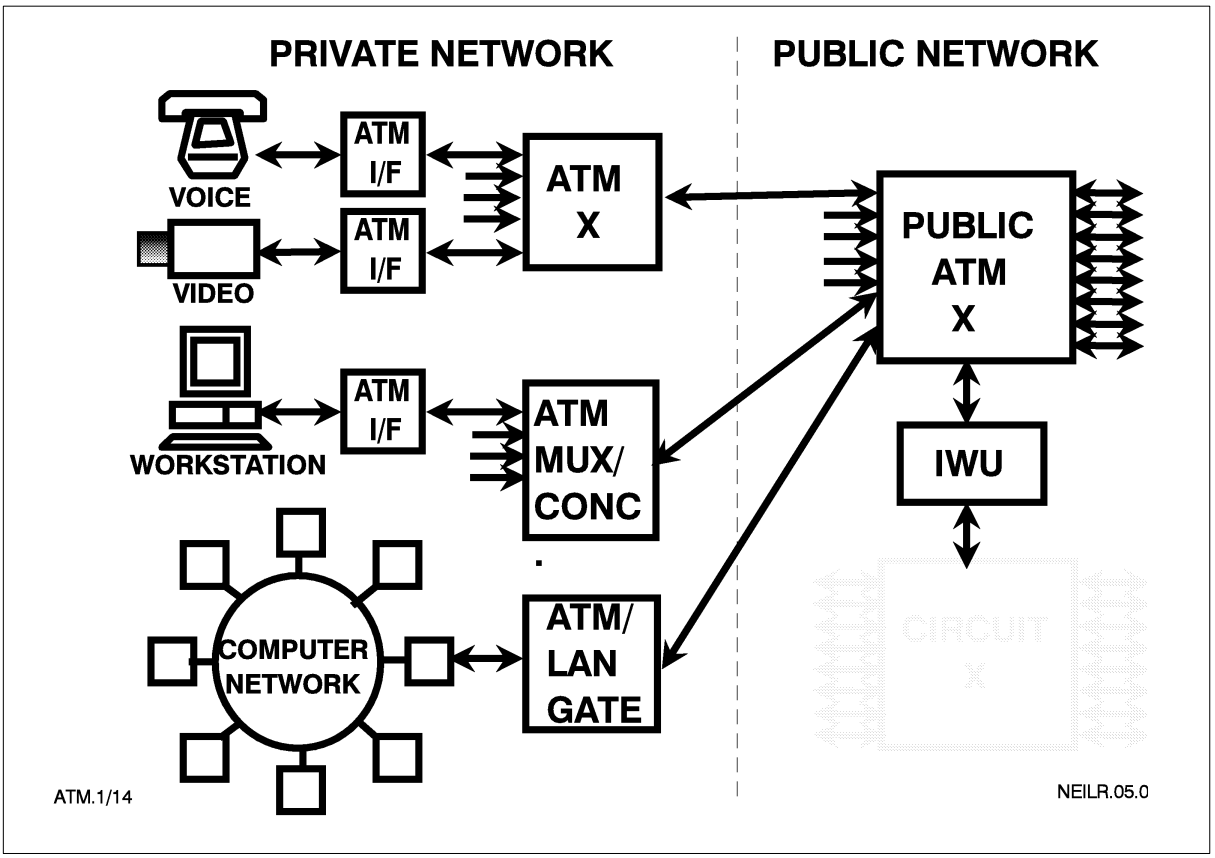

Figure 10.13 Possible ATM Network Equipment Environment

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.

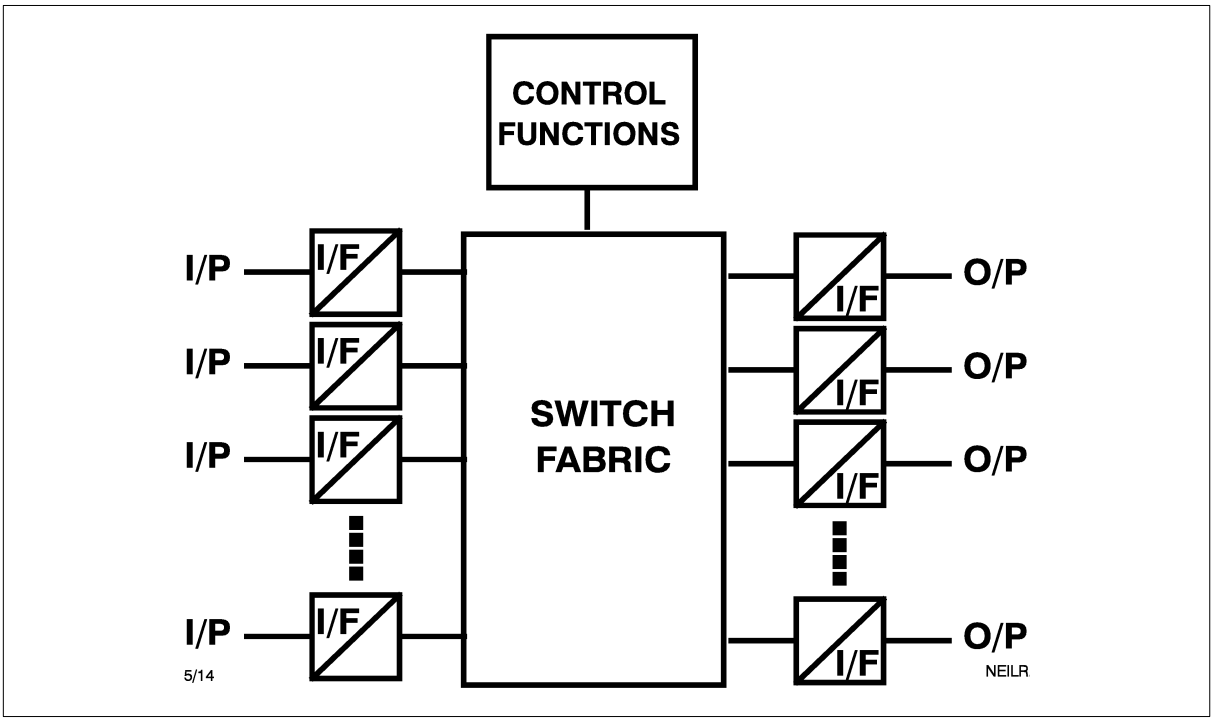

Figure 10.14 Generic Fast Packet Switching Architecture.

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.

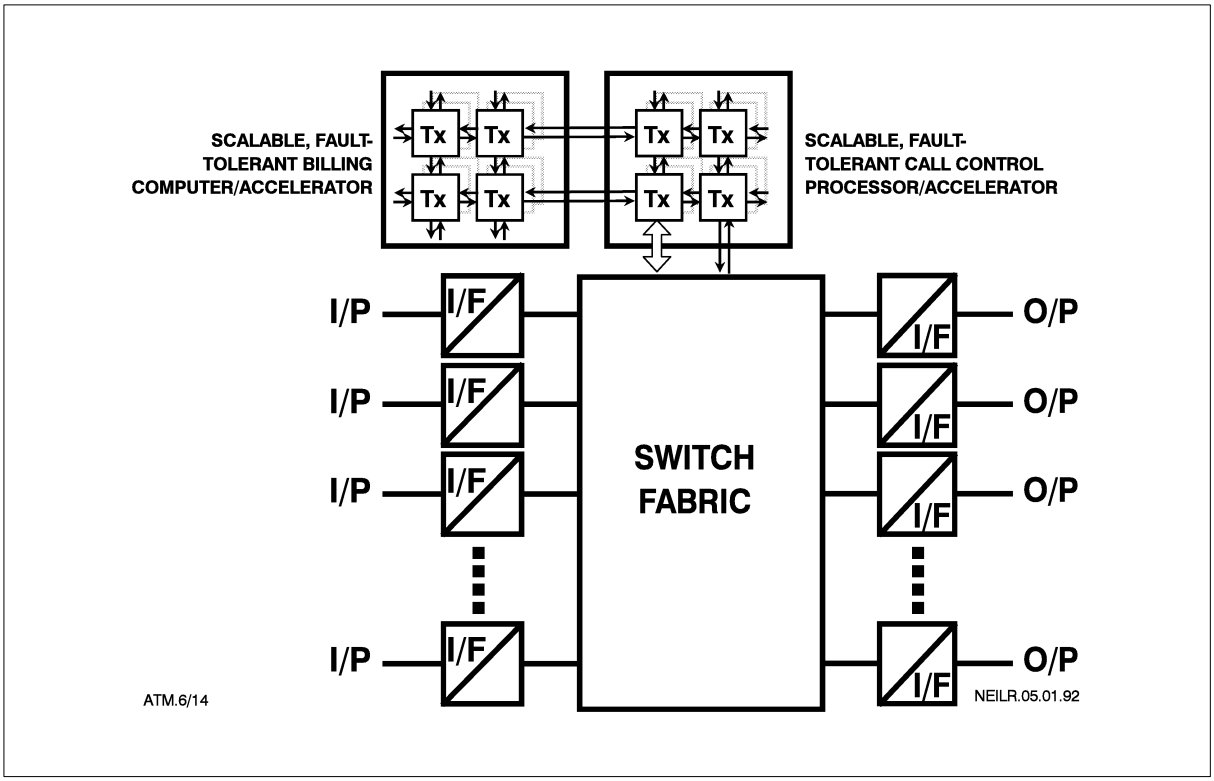

Figure 10.15 Billing/Call Control Application

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.

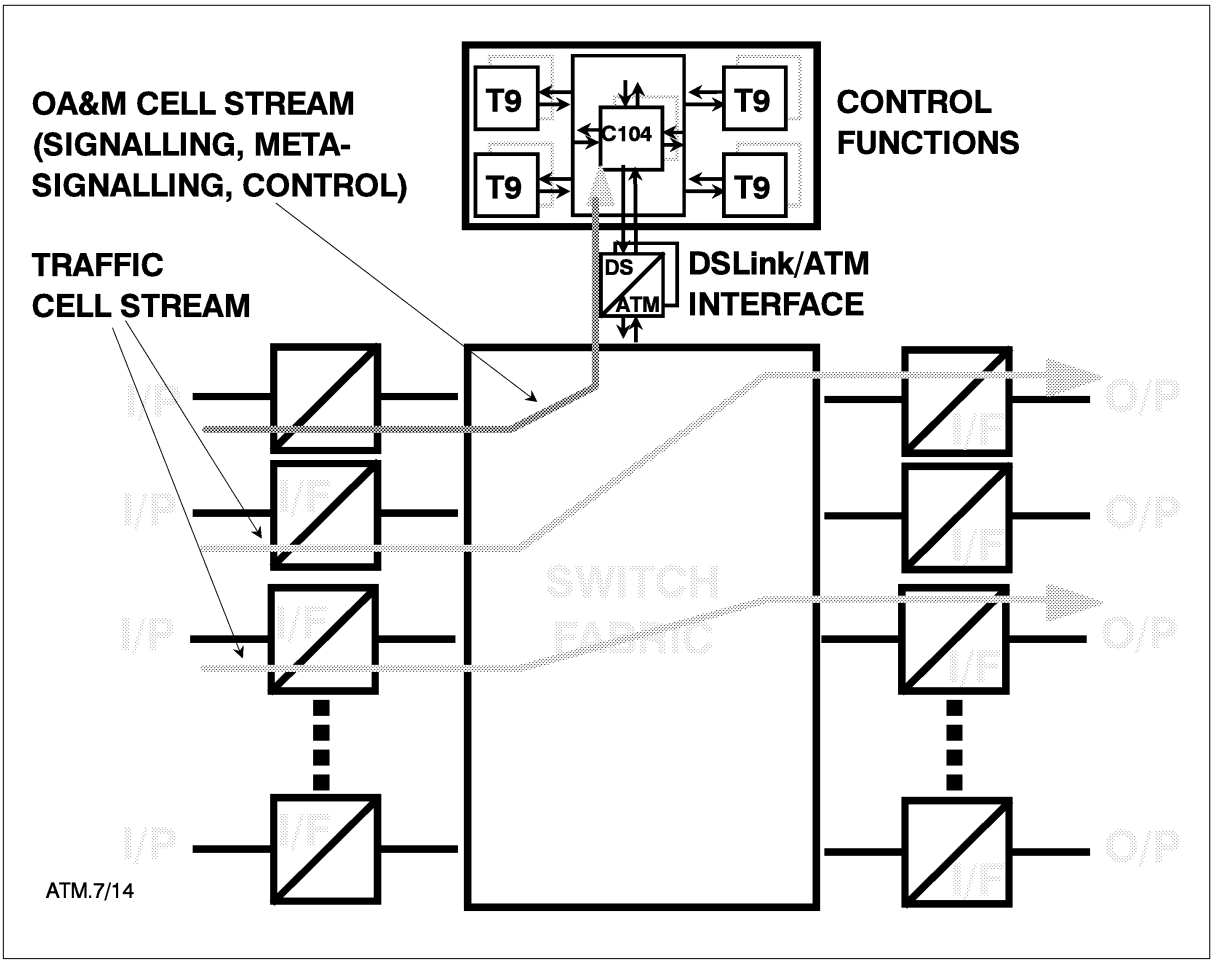

Figure 10.16 Interfacing to ATM Maintenance Traffic

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.

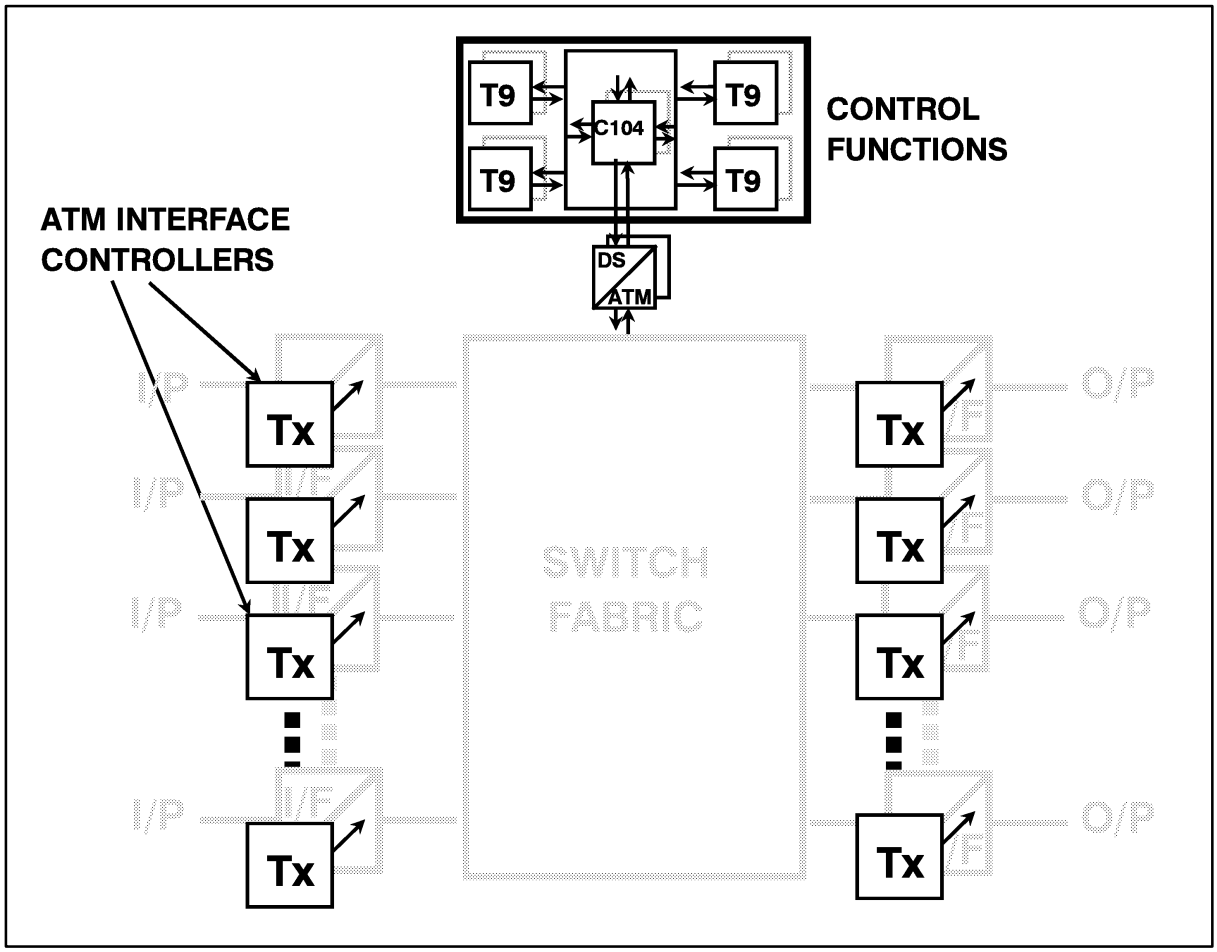

Figure 10.17 Transputers as Embedded ATM Interface Controllers

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.