Design of an Integrated Services *Packet* Network

Jonathan S. Turner

Washington University
St. Louis, Missouri 63130
jst@wucs.UUCP

ABSTRACT

The Integrated Services Digital Network (ISDN) has been proposed as a way of providing integrated voice and data communications services on a universal or near-universal basis. In this paper, I argue that the evolutionary approach inherent in current ISDN proposals is unlikely to provide an effective long term solution and advocate a more revolutionary approach, based on the use of advanced packet switching technology. The bulk of this paper is devoted to a detailed description of an Integrated Services *Packet* Network (ISPN), which I offer as an alternative to current ISDN proposals.

Introduction

An integrated voice and data packet communications system has several advantages over existing methods.

- It uses a common set of switching and transmission facilities for both voice and data communication. This is less costly than current systems that use separate mechanisms.
- It allows voice communication to be done using less than 25 percent of the bandwidth currently needed, without sacrificing quality. This allows major savings in long distance transmission costs. It also allows customers to carry on two or three simultaneous voice conversations along with a substantial amount of data traffic over a standard copper loop.
- It provides much higher performance data communication and at lower cost than current systems. This is largely due to the integration of data communication with voice, which allows one to take advantage of the economies of scale possible in the large systems needed for a national telephone network.

The system is based on high performance packet switches which are large and fast enough to effectively support both voice and data communication on a large scale. Each packet switch can have a raw throughput of up to 1.5 Gbs, allowing it to support as many as 50,000 simultaneous voice conversations using a 32 Kbs voice encoding scheme. The one-way cross-network delay in a worst-case connection in a national network in the U.S. can be limited to about 150 ms. The key elements of the design are

- The use of high speed digital transmission facilities (1.5 Mbs) with excellent error performance.
- Simple link level protocols. In particular, there is no flow control or error correction done at the link level, which eliminates the need for state information in the link level protocol processors.
- A predominantly connection-oriented service. This allows the routing of most packets to be handled by a very simple method and facilitates bandwidth allocation and overload control. A connectionless (datagram) service can also be supported.
- Hardware implementation of basic switching and protocol functions. Switching is done using a large self-routing network containing roughly 1300 custom VLSI chips. All per packet protocol functions are handled by protocol processors (one for each link) consisting of one custom controller chip plus one large memory chip.
- Implementation of higher level protocol functions (including error correction and flow control) on an end-to-end and application-dependent basis.

The governing philosophy behind the design is that the communications network should provide transport of information at the highest possible level of performance, but nothing else. The network achieves generality, not by providing every service a user might conceivably require, but by providing only those services that every user requires. The system can be implemented with currently available technology at a cost that is comparable to that of conventional telephone switching systems.

Appears in "Proceedings of the Ninth Data Communications Symposium"
9/85
Also, to appear in IEEE Journal on Selected Areas in Communication

The Trouble With ISDN

The Integrated Services Digital Network has been heralded as the mechanism that will usher in the Information Age. ISDN, it is said, will facilitate the development of new communications services, including a wide range of data services and maybe even video. It will allow such services to be implemented on a large scale at a cost most customers can afford, and spur the transformation to a 'post-industrial society.'

While I share the long range goals of ISDN proponents, I don't believe that current ISDN plans can take us very far. The fundamental problem is that current plans (see [1], for example) implicitly assume a network model based on circuit switched voice and a combination of circuit and packet switched data. This assumption is evident in current standardization efforts which focus on a transmission format for digital subscriber lines that provides two 64 Kbs circuit switched channels plus a 16 Kbs packet switched channel. While this plan can provide a limited data communications capability (it's certainly a vast improvement over the current situation), it is too inflexible to satisfy long term needs. What happens, for example to an application that requires a 75 Kbs channel? Do we implement it using the two 64 Kbs channels or one of them and the packet switched channel. Neither option is particularly attractive.

The source of the trouble is the reliance on circuit switching, which requires that the available bandwidth be divided up into fixed size channels. The channel size is a permanent feature which can't be changed easily, if at all. There's no reason to think that 64 Kbs is a particularly useful channel size. The only reason for picking it is that current telephone switching systems are based on that size. In fact, the driving force behind current ISDN plans is to preserve the investment in existing equipment while 'evolving' to a more flexible network.

Unfortunately, it won't work, because it's inherently a hybrid approach. The only thing integrated about ISDN is its name. Two (or more) switching networks are required to support ISDN as it's currently envisioned. Manufacturers may talk about their integrated architectures for ISDN, but what they mean is that they plan to put a packet switch and a circuit switch in the same box and call it an integrated system. This is not their fault—what else can they do? Packet switching and circuit switching are very different communications methods. They require different switching and transmission facilities and all the proposed schemes for combining them are little more than packaging.

What to do then? Is a hybrid network really necessary or is there an integrated solution that can satisfy the needs of both voice and data and remain flexible enough to meet satisfy new requirments as they arise? I claim there is such a solution, but it requires abandoning circuit switching and moving to new network designs based on packet switching.

Why Packet Switching?

Here are three reasons that make packet switching an attractive method for providing integrated voice and data services.

- *Adaptability to changing traffic.* Packet switching naturally provides the user with exactly the bandwidth required. As new services are developed with different bandwidth requirements, packet switching systems can adapt to the changing conditions easily. Circuit switching systems cannot.
- *Integrated internal architecture.* As outlined above, current ISDN plans require separate switching networks for different types of information. Packet switching can provide both an integrated customer interface and a single network solution for a wide range of communications needs, leading to substantial cost savings in switching systems and system administration.
- *Transmission efficiency.* Many data services are characterized by bursty communications patterns which make poor use of conventional circuit switched facilities. For example, interactive data users typically use only a few percent of the bandwidth available to them. Although it is less widely recognized, voice is also bursty. In the average telephone conversation, less than 40% of the available bandwidth is actually used. Packet switching can exploit this burstiness to double the number of conversations that can share a single transmission facility.

Given all these advantages, why has packet switching not been used extensively for voice? To answer this, we must take a closer look at the technical specifications and costs of conventional circuit switches and packet switches and see how they compare.

Circuit switching has been the technology of choice in the telephone network, since its origin about one hundred years ago. During that period, circuit switches have evolved from manually operated switchboards to small but automatic step-by-step switches, to larger panel switches to the computer-controlled electronic switching systems that currently dominate the scene. The current systems are very large—local switches can provide service to over 100,000 customers, large toll switches support as many as 50,000 simultaneous voice calls corresponding to a raw bandwidth of 6 gigabits per second (Gbs). The network as a whole is also very large. There are approximately 10^8 telephones in the United States at present and over 10,000 local switching offices. The performance of the switching systems is quite impressive—information passing through a modern digital switch is typically delayed no more than a few milliseconds, and new connections can be established in a fraction of a second. Equally impressive is the cost—while there is considerable variation, per line equipment costs for modern digital telephone systems is typically in the \$100–200 range.

Just as circuit switching has long dominated the telephone network, so has packet switching dominated the data communications scene, principally because of the advantages cited earlier. Packet switching is exemplified by the Arpanet, which was the first major example of a large scale data communications network. In the Arpanet, the endpoints of the communication are typically large timeshared computers, called hosts. Communication is provided by packet switches, each of which typically connects to a few hosts and several other packet switches. There are currently about 300 hosts in the Arpanet and 100 packet switches. The packet switches are implemented using general purpose computers (usually minicomputers), although more recent versions have been supplemented with front-end communications processors to reduce the load on the main processor. The transmission facilities used by the Arpanet include low speed modem connections (1–10 Kbs) and higher speed digital facilities (56 Kbs). The throughput of the packet switches is generally under 1 Mbs and delays can be substantial (50–100 ms per switch). The cost of packet switching as provided by the Arpanet is quite high, since each packet switch supports a small number of hosts. Commercial data networks do better, but there remains a large gap between per-host costs in data networks and per-line costs in the telephone network.

This comparison explains the conventional wisdom that packet switching is poorly suited to the needs of telephony and the resulting conclusion that an ISDN implementation must include a circuit switching component to support voice. The fallacy in the conventional wisdom is that the disadvantages attributed to packet switching by this comparison are not due to any inherent properties, but are side-effects of the conventional implementations. The requirements and design constraints that shaped the development of the Arpanet and commercial data networks were completely different from those that shaped the telephone network. The scale, the performance requirements and the cost sensitivities are all very different. It's because the *needs* differ that the resulting systems differ so in their technical specifications. In the remainder of this paper, I will attempt to correct the widely held, but erroneous view that packet switching is poorly suited to the needs of voice by describing a system capable of supporting voice and data communication on a large scale and at a cost that is competitive with conventional telephone systems.

ISPN Architecture

Let's begin by deciding what general properties a packet switching system must have if it is to be a suitable vehicle for providing voice and data communication on a large scale. First and most obviously, it must be big—comparable in raw bandwidth to conventional telephone systems. Second, it must be fast—long end-to-end delays are annoying in voice connections and hence unacceptable. While opinions differ on the exact amount of delay that can be tolerated, most experts would agree that 100 ms is acceptable, while more than 500 ms is not. Third, it must be inexpensive. Any system that seeks to replace circuit switching must be able to provide voice services at a competitive cost. It is not enough to offer a better product at a higher cost—for a system to succeed on a large scale, it cannot significantly increase costs for basic services.

How can we achieve the requisite scale, performance and economy in a packet switching network? Well, the telephone network is one place to look for the answer. One of the first things one notices is that the telephone network makes extensive use of high bandwidth digital transmission facilities. Modern digital switching systems are designed to interface directly to 1.5 Mbs transmission facilities carrying 24 voice channels in a 64 Kbs format. Interfacing costs are cheap, since they are shared by the 24 channels and the bit error rates are excellent. Conventional packet switching systems, on the other hand interface to much smaller channels, and must be prepared to cope with the much higher error rates present in modem connections. A second thing one notices about the telephone network is the amount of special purpose hardware used to provide the switching function. While telephone systems contain general purpose computers, their function is primarily connection establishment. The computers don't move the bits. That function is handled by large specialized networks. In contrast, most conventional packet switches include a general purpose computer that must do some processing on every packet and can quickly become a bottleneck.

These observations suggest that a high performance packet switch should (1) interface directly to high speed digital transmission facilities and (2) use special purpose hardware to perform all per packet processing. Now, those familiar with link level protocols used in conventional packet networks may recognize this as a challenging task. Typical protocols are quite complex, requiring extensive state information in the protocol processors and complicated error recovery procedures. This complexity almost demands a programmable processor (a hardware implementation would never be completely debugged), but a microprocessor-based implementation is unlikely to be fast enough to keep up with a 1.5 Mbs link, and also may be too expensive. Fortunately, there is a way out—simplify the protocol. Current packet switching protocols were designed for hostile environments—the low speeds and high error rates typical of modem connections. In a network composed entirely of high speed digital facilities, much of the complexity of typical protocols can be eliminated from the lower protocol layers. In particular, error correction and flow control can be taken out of the link layer and provided on an end-to-end basis rather than a link-by-link basis. This eliminates the need for extensive state information at the ends of each link and the synchronization and recovery procedures required to maintain it and in turn makes possible the construction of inexpensive, high performance protocol processors. In addition, if these functions are provided on an end-to-end basis, they can be pro-

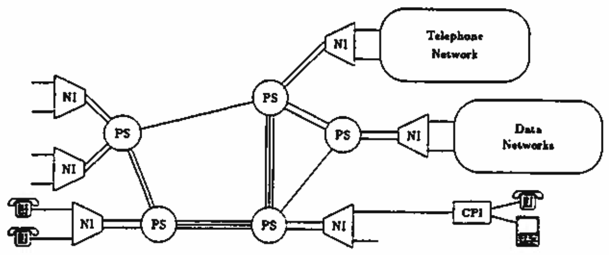


Figure 1: Network Architecture

vided selectively. Hence, delay-sensitive applications like voice can avoid the performance penalites they can cause.

Returning to the question of scale, how large must our packet switches be? In order to compete with conventional telephone switches, they should be capable of supporting at least 50,000 simultaneous voice conversations. How many conversations can a 1.5 Mbs link carry? When operated in a circuit switching mode using the 64 Kbs digital encoding method currently employed, the answer is 24. When operated in a packet switching mode that number jumps to about 50. One can obtain another factor of two improvement by using newer voice encoding methods. 32 Kbs adaptive differential pulse code modulation (ADPCM) is a good choice, since it provides comparable quality to the method currently used, but requires only half the bandwidth. This leads to a figure of 100 voice channels per 1.5 Mbs link, implying that our packet switch must terminate 1000 full-duplex links if it is to support 50,000 simultaneous voice conversations. This in turn, means that our switch must include some mechanism capable of receiving over two million packets per second, and sending each one out on the appropriate outgoing link.

Network Overview

Based on the discussion in the previous section, we can begin to develop an architecture for the ISPN. The major components are shown in Figure 1. The Packet Switches (PS) each terminate up to 1000 High Speed Links (HSL). Using 32 Kbs ADPCM coding for voice, they can support over 50,000 simultaneous voice conversations. Residential customers connect to the network over Medium Speed Links (MSL), which operate at 100 Kbs. High speed access can be provided to business customers. The Customer Premises Interface (CPI) can take a variety of forms depending on the kind of service required. At the low end of the spectrum would be a simple controller providing service for a single telephone and implemented using a single chip eight bit microcomputer. Customers wanting several phones and data communication would require a more complex controller. Businesses would typically have an interface to a Private Branch Exchange (PBX) or Local Area Network (LAN). The Network Interfaces (NI) provide concentration, accounting data collection and network protection. A configuration designed for residential customers would support about 500 customers and would be connected to a local packet switch by four HSLs. Thus, one PS could have as many as 125 NIs and support over 60,000 customers.

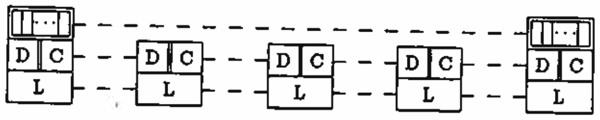


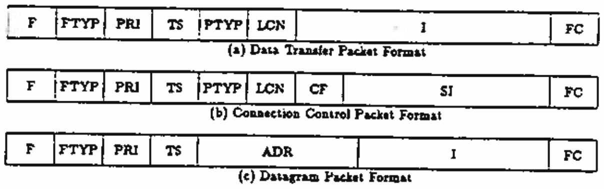
Figure 2: Protocol Structure

NIs can also be designed to provide a conventional line interface as supported by current telephone systems. Even higher concentration ratios can be supported in this case—up to 2000 customers could be supported on a single NI connected to its host PS by four HSLs. While this configuration doesn't exploit the advanced capabilities of the system, it does provide a mechanism for easing the transition from a circuit-oriented network to a packet-oriented one. In a similar fashion, NIs can be designed to interface to the current telephone network. In this case, the NI would interface directly to up to 960 digital trunks.

The network provides two communications services.

- *Point-to-point channels.* These are two-way channels joining pairs of customers. The customer establishes a channel by sending a connection request message to the network specifying the destination and the average bandwidth needed. If the network accepts the connection, it in effect guarantees that the customer can expect to have the requested bandwidth available. The network may refuse to accept the connection if there are not adequate resources available. The connections do not provide perfectly reliable information transport. In particular, the network does not provide mechanisms for error correction and flow control. The network can provide connections at any speed up to 1.5 Mbs (although larger connections are more likely to be blocked). The bandwidth requirement may be asymmetric—that is, it may depend on the direction of transmission. No distinction is made between voice and data connections. A voice connection is simply one with an average bandwidth of about 12 Kbs.
- *Datagrams.* These are individually addressed packets, not associated with a pre-established connection. The network makes an effort to deliver them, but does not guarantee delivery.

The communications protocols are divided into three levels—the *link level*, the *network level* and the *customer level*. The interrelationships among the different levels are indicated in Figure 2. There is a single link level protocol, two network level protocols (one for connection-based communication and one for datagrams) and a variety of customer level protocols. The customer level protocols are not discussed in detail here, but the intention is that these protocols would be application-dependent and built on top


F
FTYP
PRI
TS
PTYP
LCN
I
FC
(a) Data Transfer Packet Format
F
FTYP
PRI
TS
PTYP
LCN
CF
SI
FC
(b) Connection Control Packet Format
F
FTYP
PRI
TS
ADR
I
FC
(c) Datagram Packet Format

F:	Flag (1 byte)	CF:	Control function (1 byte)
FTYP:	Frame type (4 bits)	SI:	Supplementary information
PRI:	Priority (4 bits)	ADR:	Address (8 bytes)
TS:	Time stamp (1 byte)	I:	Information field
PTYP:	Packet type (4 bits)	FC	Frame check sequence (2 bytes)
LCN:	Logical channel number (12 bits)		

Figure 3: Packet Formats

of one of the two network level protocols. One of these would be a telephony protocol. Another might be an internet protocol such as the DARPA IP, to facilitate communication among different data networks. Still another might be a connection-oriented internet protocol.

There are essentially three packet formats of interest to the network—*connection control packets*, *data transfer packets* and *datagram packets*. These packet formats are shown in Figure 3.

Link Level Protocol. The link level protocol provides frame delimiting, link transparency, error detection, packet timing, and congestion control, but not error correction. There are four fields used by the link level protocol, the Frame Type field (FTYP), the Priority field (PRI), the Time Stamp field (TS) and the Frame Check field (FC). The FTYP field identifies the frame as a test frame, a datagram or a frame belonging to a connection. The Priority field (PRI) contains a customer-specified priority. The network preferentially discards low priority frames to alleviate short-term overload conditions in the network. The network uses the Time Stamp field (TS) to record the delay encountered by the packet as it crosses the network. More precisely, it records the number of milliseconds the packet spent in each PS and NI it passed through (see [11]). This is important for applications such as voice that are sensitive to delay variations and need a mechanism to remove the timing 'jitter' that packet networks can introduce. It can also be used in distributed programming applications for clock synchronization. The Frame Check field (FC) uses a sixteen bit cyclic redundancy code to detect errors in the frame. Frames with errors are simply discarded.

Network Level Protocols. There are two protocols at the network level, the Simple Datagram protocol (SDP) and the Simple Connection protocol (SCP). SCP packets contain a four bit Packet Type field (PTYP) and a twelve bit Logical Channel Number field (LCN). The Packet Type field (PTYP) identifies each packet as either a data packet or a control packet and contains a Congestion Control subfield, used to inform the NIs and customers of internal network congestion. SCP data packets also contain an Information field, which can have any length up to 144 bytes. SCP control packets are used to establish and control connections and contain a Control Function field (CF) and a Supplementary Information field (SI). SDP packets contain an eight byte address field. Two bytes are reserved as a socket number, to be interpreted by the customer protocols. The remaining six bytes define a hierarchical address space organized on a geographical basis, to permit routing based on local knowledge of network topology.

Network Performance

Given the network architecture described above, we can make some approximate calculations of network performance. To do this we hypothesize a worst-case reference connection in which two customers access the network using MSLs and communicate over a 4000 mile connection passing through two NIs and six PSs. We assume that the access links each carry two voice calls and that the seven internal HSLs are all operating at an occupancy of 85%.

The delay experienced by a packet passing across the reference connection contains several components.

- *Speed-of-light delay.* This is the constant delay caused by the finite speed of light. It is approximately 40 ms for a 4000 mile connection.
- *Packetizing delay.* This is the delay involved in producing enough data to put in a packet. Using 32 Kbs ADPCM voice encoding, information is generated at the source at the rate of 32 bits per millisecond. 512 bits are placed in each packet, giving a packetizing delay of 16 ms.
- *Queueing delay.* This is the delay encountered by the packet as it is waiting to be transmitted over each of the seven HSLs and two MSLs in the reference connection. We will analyze this component in detail below.
- *Switching delay.* This is the delay encountered by a packet as it passes through a PS or NI, excluding queueing delays. We will see in the next section, this can be kept below 0.5 ms per switch, hence we allow 4 ms of switching delay in the reference connection.

We use an M/M/1 queueing model to approximate the queueing delay on the seven HSLs in the reference connection. The formula for the average delay through n tandem queues is

$$q = \frac{bn}{s(1-\rho)}$$

where b is the packet length in bits, s is the bit rate of the transmission lines (1.544 Mbs) and ρ is the link occupancy (0.85). We use an average packet length of 650 bits to model a mixture of voice packets with 1000 bit data packets. For seven queues we calculate $q \approx 20$ ms. Of course, this is just the average delay. For voice, it is important to have an upper bound on the delay experienced by *most* packets. We can determine this by using an Erlang distribution with a rank of seven and a mean of 20 ms. With this technique we estimate that 99% of all packets in the

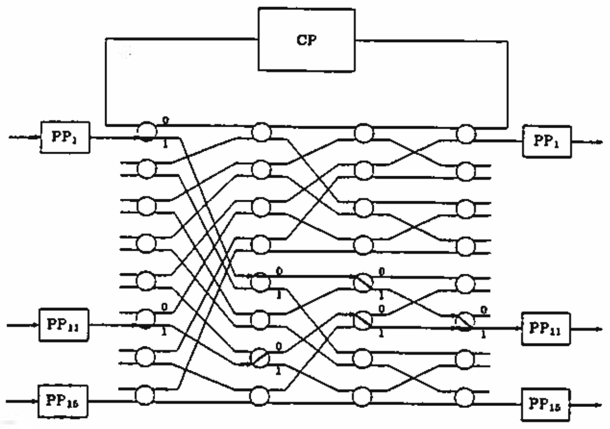


Figure 4: Packet Switch Structure

reference connection experience a delay of no more than 42 ms on the internal network links.

We still need to determine the queueing delays experienced on the access links. Here the M/M/1 model is not really applicable. On the access link going into the network, the delay is bounded by 12 ms, since we are assuming two simultaneous voice calls generating packets at 16 ms intervals. The worst thing that can happen is that packets are generated simultaneously from both voice encoders meaning that one packet must wait for the other to be transmitted. The delay of 12 ms results from the observation that voice packets are less than 600 bits long, and with an access line speed of 100 Kbs can be transmitted in 6 ms—hence it takes 12 ms to transmit a packet for each call. Packets experience a similar delay as they leave the network. Of course, the actual delays experienced at the access links is highly dependent on the nature of the services the customer is using at the time. One way to ensure good performance for voice at the access links is to use the packet priorities to give preference to voice packets. With this mechanism, we can add data to the traffic carried by the access link and be guaranteed a maximum delay of 22 ms on each end (assuming a non-preemptive priority mechanism). Based on these considerations, we assume 50 ms of access delay.

Summing the various delay components yields an estimate of about 150 ms for 99% of all packets.

Packet Switch Design

The network is built using large high performance packet switching systems, each terminating up to 1023 HSLs. The structure of such a packet switch is illustrated in Figure 4, which shows a small version with 15 HSLs.

The system is controlled by a Control Processor (CP) which performs all connection control functions, plus administrative and maintenance functions. The CP is a large general-purpose computer. Its role is analagous to that of the Control Processor in large telephone switching systems such as the No. 4 ESS [2].

Each HSL is terminated by a Packet Processor (PP), which performs the link level protocol for all packets and the network level protocol for data transfer packets. It also forwards connection control packets to the CP and datagram packets to the Datagram Routers (DR).

The heart of the switch is the Switch Fabric (SF) which consists of a large binary routing network. The important property of such networks is that the path each packet takes through the network is determined by successive bits of its destination address. The figure shows paths from two different SF input ports to output port 1011. Note that at the first stage the packets are routed out the lower port of the nodes (corresponding to the first '1' bit of the destination address), at the second stage they are routed out the upper port (corresponding to the '0' bit) and in the third and fourth stages they are routed out the lower ports. The self-routing property is shared by a variety of interconnection patterns, including the so-called delta, shuffle-exchange and banyan networks (see [5]). The PS uses a ten stage binary routing network with 1024 ports.

The Datagram Routers (not shown) are special-purpose devices used to route datagrams. The number of DRs can be engineered to suit the traffic. Each occupies a port on the SF, replacing one PP. Since most of the traffic is expected to be connection-oriented, the number of DRs required should be modest.

Duplication is required to provide the highly reliable operation expected in a large scale public communications network. The scheme used here (but not shown in the figure) is to duplicate the SF, but not the PPs. In normal operation, the duplicate SFs are operated in a load-sharing fashion, with each carrying half the traffic. If one side fails, all traffic is shifted to the other side.

Packet Processing

When a packet is received by a PP, it is placed in a buffer with several additional header fields added. The Destination field (D) identifies the destination port on the SF. The Source field (S) identifies the port where the packet arrived. The Length field (LNG) gives the packet length in bytes. The Switch Packet Type field (SPTYP) is used to identify various packet types within the PS. The Arrival Time field (AT) gives the time at which the packet arrived at the PS (this is used for processing the TS field).

For data transfer packets, the destination port is determined by the PP using the packet's LCN field and the PP's Logical Channel Translation Table (LCXT). Each entry in the LCXT contains an outgoing port number and a new LCN. The outgoing port is placed in the D field of the packet and the new LCN goes in the LCN field. The

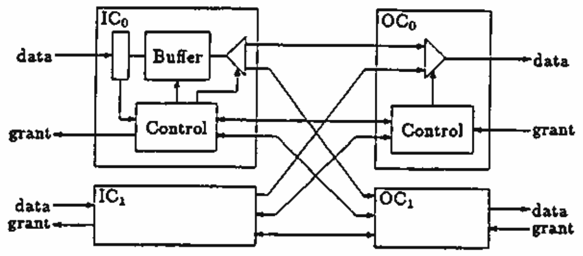


Figure 5: Switch Node

packet is then sent on to the switch fabric, which uses the D field to route the packet to the proper outgoing port as described earlier. When a packet arrives at the outgoing PP, it is buffered and then transmitted on the HSL with the extra header information stripped off.

The contents of the LCXTs is controlled by the CP, which can read and write the LCXT using special control packets sent to the PPs through the SF. Thus, the connection establishment process includes the sending of messages from the CP to the two PPs selected for the connection, updating their LCXTs appropriately.

Switch Fabric

The basic operation of the switch fabric has already been described. The SF's highly regular and parallel structure allows the construction of very large switches, without the bottlenecks that can arise in bus or ring based interconnection networks.

The nodes of the SF operate as miniature packet switches. Each node has a buffer at each input port capable of holding one maximum length packet. The data paths joining the nodes are bit serial and operate at 12 Mbs. This gives the SF an 8:1 speed advantage over the external HSLs. Thus, if all the external links are operated at an occupancy of 85%, the internal links will have an average occupancy of less than 11%. (This is assuming just one switch plane active. When both are active, the average occupancy is less than 6%.) There is also an upstream control lead joining each pair of adjacent nodes. This is used to implement a simple hardware flow control mechanism, which prevents buffer overflows within the SF.

A more detailed look at the switch node appears in Figure 5. It consists of two Input Controllers (IC) and two Output Controllers (OC). The ICs contain a buffer large enough to hold one packet and a controller implemented as a state machine. The OCs are simple state machines that arbitrate requests for their ports.

When a packet is received, the IC determines the proper outgoing port by examining the appropriate bit of the packet's destination field, then requests permission to use that port. If the desired port is immediately available, the packet is sent to it directly, bypassing the buffer.

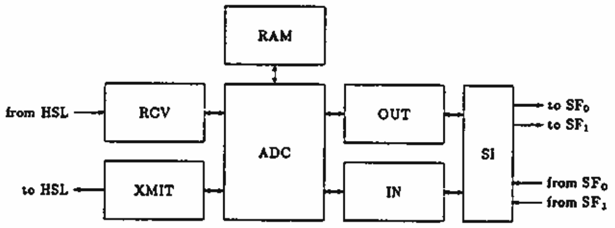


Figure 6: Packet Processor

Hence, a packet can pass through a switch node after experiencing a delay of just a few bit times. In fact, this is the normal case due to the relatively low occupancy of the internal data paths.

If the desired port is not available, the packet is shifted into the buffer. As soon as the desired port becomes available, the packet is sent out—even if not all of the packet has been received. If the port is still not available when the end of the packet is received, the IC holds its *grant* lead low to prevent the arrival of new packets. The grant lead is re-asserted as soon as the desired link becomes available, allowing a new packet to enter the buffer, while another is leaving.

Packet Processors

The structure of the Packet Processors (PP) is shown in Figure 6. It is organized around a 16 Kbyte RAM, which contains four packet buffers and the Logical Channel Translation Table (LCXT). The principal buffers are the Receive Buffer (RCB) used for packets received from the HSL on their way to the switch, and the Transmit Buffer (XMB) for packets going from the switch to the HSL. The Link Test Buffer (LTB) and the Switch Test Buffer (STB) are small buffers that provide loop-back paths for testing the HSL and switch respectively.

Access to the memory is provided through the Address Controller (ADC), which contains read and write pointers for the buffers and arbitrates memory access.

The Receive circuit (RCV) receives incoming packets from the HSL, removes the flag field, discards packets with errors, adds the extra header fields, initializes the length (LNG) and arrival time (AT) fields, converts from bit serial to eight bit parallel format and writes the packet to the RCB through the ADC.

The Output circuit (OUT) takes packets from the RCB, performs the logical channel translation described above and sends the packets onto the switch in bit serial format. The Input circuit (IN) takes packets from the switch and writes them to the XMB.

The Transmit circuit (XMIT) takes packets from the XMB, performs the time stamp calculation, strips the extra header information, adds the flag field and transmits the packets on the HSL.

The Switch Interface (SI) connects to the duplicated switch planes, normally routing packets to and receiving packets from the active plane. The SI can also send and receive packets from the standby plane—this is used for testing.

The PP can be implemented using two chips—one custom controller chip and one memory chip. The simplicity of the protocols makes this possible. There is no need to buffer unacknowledged frames that may need to be retransmitted as in conventional link level protocols that perform error correction and flow control. Similarly, there is no need for the recovery and synchronization procedures that are required by such protocols.

Performance of the Switch Fabric

Switching networks similar to the one considered here have been analyzed by several authors to determine their performance characteristics (see [4] and [8], for example). Unfortunately the earlier analyses don't quite suit the network considered here, since they assume packets are buffered at each stage. This is not the case for the network considered here, which permits packets to pass through a stage without being buffered at all. This buffering technique is called *virtual cut-through* by Kermani and Kleinrock [10], who analyze its performance in a somewhat different context. Their analysis does not apply to our situation either.

The analysis given by Jenq in [8] can be modified to suit our needs. Jenq provides an algorithm for calculating the delay through a binary routing network preceded by packet processors containing infinite buffers. The main difference between his model and the network described here is that it assumes that packets are buffered at each stage. This means that in his model, packets encounter a delay of at least nb/s, where n is the number of stages in the network, b is the packet length and s is the speed of the internal data paths. This is true even at very low occupancies. In our network, on the other hand, the delay experienced at low occupancies is $\approx 10n/s$, since each packet is delayed about 10 bit times per stage. Hence, at low occupancies the delay through the network can be obtained by subtracting $(b-10)n/s$ from the estimate obtained using Jenq's method. Since the network is normally operated at occupancies below 20%, this approach can be expected to yield satisfactory estimates.

Two other assumptions of Jenq's method need to be mentioned. First, it assumes that all packets have the same length—we will use it to estimate the delay under the assumption that all packets are of maximum length ($b = 1264$), which is a worst-case assumption for this network. Second, Jenq's analysis assumes that traffic is uniformly distributed throughout the network—more precisely, the destination field of each arriving packet is assumed to be selected at random from the integers 0–1023. This is called the *uniform traffic assumption* and while it appears reasonable, it ignores the effect of the network's

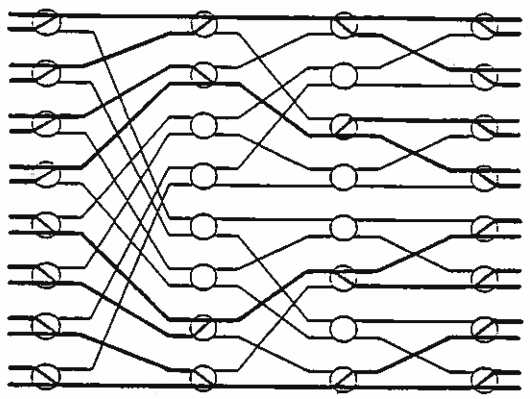


Figure 7: Congestion in Binary Routing Networks

connection-orientation on traffic patterns. This assumption is examined more closely in a later section and justified.

Let $d_J(\rho, n)$ be the average delay through an n stage binary routing network for a given occupancy ρ, using the method described in section V of Jenq's paper [8]. We find, $d_J(.05, 10) = 1070\,\mu s$, $d_J(.1, 10) = 1090\,\mu s$, $d_J(.15, 10) = 1113\,\mu s$ and $d_J(.2, 10) = 1141\,\mu s$, Subtracting $1045 = (b-10)n/s$, as described above gives corresponding estimates of 35, 45, 68 and 96μs for our network. When all the HSLs coming into the PS are operated at occupancies of 80%, the occupancy of the internal links is about 11%, leading to an average switch fabric delay of about 50 μs.

Switch Fabric Revisited

One problem with binary routing networks is that they can become congested in the presence of certain traffic patterns. This is illustrated in Figure 7, which shows a traffic pattern corresponding to several *communities of interest*. In this pattern, all traffic entering the first four inputs is destined for the first four outputs, all traffic entering the second group of four inputs is destined for the second group of four outputs, and so forth. Note that with this pattern, only one fourth of the links joining the second and third stages are carrying traffic. Thus, if the inputs are heavily loaded the internal links will be hopelessly overloaded and traffic will back up. In a 1024 × 1024 network, there are ten stages and the links between the fifth and sixth stages can in the worst-case be carrying all the traffic on just 32 of the 1024 links.

This problem can be solved by using two networks instead of one. One of these is called the Routing Network (RN) and is a standard binary routing network. The other network sits in front of the RN and is called the Distribution Network (DN). It has the same structure as the RN, but instead of routing packets based on their destination address, it attempts to distribute packets evenly across all its output ports. This is done by having each switch node route packets alternately out its two ports. The alternate-port strategy is modified if one or both ports is unavailable. In this case, the first port to become available is used. This approach breaks up any communities of interest and makes the combination of the DN and RN robust in the face of pathological traffic patterns.

Note, that if the DN used a purely random distribution algorithm, the uniform traffic assumption made in the performance analysis section would be validated. The distribution algorithm used by the DN is actually superior to a random one, since it distributes traffic more evenly and can route around congested areas. These observations justify the simple analysis based on the uniform traffic assumption.

One drawback of the DN is that it doubles the number of stages in the SF, thus roughly doubling the packet delay and the circuit complexity. This loss can be recovered by using larger switch nodes for the RN and DN. Instead of 2×2, we can use 4×4 nodes. With the larger nodes we can construct a network with half the number of stages required with 2×2 nodes. This also halves the delay and the circuit complexity.

Other Issues

The circuit complexity of the switch nodes is dominated by the buffer, which is basically a 1280 bit shift register. Thus a 4×4 node contains about 5000 bits of memory and will fit comfortably on a single chip with 20 pins. These chips are combined onto circuit boards, each board containing either 32 or 48 chips. A single RN will contain 32 of these boards.

The PPs can be implemented using one custom controller chip and one $16K \times 9$ memory chip. Sixteen PPs are packaged together on a single board, meaning that 64 of these boards are required for the entire PS. The Datagram Routers, which haven't been described in detail can be implemented in a similar fashion to the PPs—they're actually a bit simpler.

The packaging of the PS can be remarkably compact. A single equipment frame, 3 feet wide by 6 feet high by 1.5 feet deep is all that's required for the duplicated switch fabric and all the packet processors.

Summary

This paper has described the design of a high performance packet switching network capable of supporting both voice and data communication on a large scale and at a cost to the user, comparable to current telephone service. The key features of the design are the use of high speed digital transmission facilities and simple link level protocols, a predominantly connection-oriented service, hardware implementation of all per packet functions and implementation of higher level protocol functions on an end-to-end and application-dependent basis. I claim that the advanced packet switching technology described here is inherently better suited for the provision of advanced communications services than the circuit switching technology which underlies current ISDN proposals.

Many aspects of the packet switching system described here have been patented by AT&T Bell Laboratories. See references [14]–[20] for further details.

Acknowledgements. The work described here was carried out between 1981 and 1983 when I was a Member of Technical Staff at Bell Laboratories in Naperville, Illinois. Many individuals contributed ideas. I particularly want to thank Harold Andrews, Danny Creed, Bill Hoberecht, Warren Montgomery and Len Wyatt who helped turn the original idea into a workable design. I also want to thank Bob Cardwell and Eric Nussbaum for recognizing the potential and providing the backing to develop it. Special thanks go to Maurizio Dècina, whose refusal to accept half-baked ideas forced me to make sure that mine were always well-cooked.

REFERENCES

[1] Aoki, D. J., D. H. Florin, S. D. McKenna, G. R Welsh and P. E. White. "Digital Feature Evolution in the 5ESS^{TM} Digital Switch," *Proceedings of Globecom 83*, 11/83, 601–605.

[2] *Bell System Technical Journal*, vol. 56, no. 7, 9/77 (Special issue devoted to the No. 4 ESS).

[3] Dècina, Maurizio. "Performance Requirements for Integrated Voice/Data Networks," *IEEE Transactions on Communications*, vol. COM-30, no. 9, 9/82, 2117–2130.

[4] Dias, Daniel M. and J. Robert Jump. "Packet Switching Interconnection Networks for Modular Systems," *Computer*, vol. 14, no. 12, 12/83, 43–54.

[5] Feng, Tse-yun. "A Survey of Interconnection Networks," *Computer*, vol. 14, no. 12, 12/83, 12–30.

[6] Gruber, John G. "Performance Requirements for Integrated Voice/Data Networks," *IEEE Journal on Selected Areas in Communications*, vol. SAC-1, no. 6, 12/83, 981–1005.

[7] Hoberecht William L. "A Layered Network Protocol for Packet Voice and Data Integration," *IEEE Journal on Selected Areas in Communications*, vol. SAC-1, no. 6, 12/83, 1006–1013.

[8] Jenq, Yih-Chyun. "Performance Analysis of a Packet Switch Based on a Single-Buffered Banyan Network," *IEEE Journal on Selected Areas in Communications*, vol. SAC-1, no. 6, 12/83, 1014–1021.

[9] Kasahara, M., H. Shimizu, M. Imaizumi. "Basic Concept of the Digital Network for the Information Network System," *Proceedings of Globecom 83*, 11/83, 969–973.

[10] Kermani, Parviz and Leonard Kleinrock. "Virtual Cut-Through: A New Computer Communication Switching Technique," *Computer Networks*, vol. 3, 1979, 267–286.

[11] Montgomery, Warren A. "Techniques for Packet Voice Synchronization," *IEEE Journal on Selected Areas in Communications*, vol. SAC-1, no. 6, 12/83, 1022–1028.

[12] Rettberg, R., C. Wyman, D. Hunt, M. Hoffman, P. Carvey, B. Hyde, W. Clark and M. Kraley. "Development of a Voice Funnel System: Design Report," Bolt Beranek and Newman, Report No. 4098, 8/79.

[13] Turner, Jonathan S. and Leonard F. Wyatt. "A Packet Network Architecture for Integrated Services," *Proceedings of Globecom 83*, 11/83, 45–50.

[14] Turner, Jonathan S. "Packet Load Monitoring By Trunk Controllers," United States Patent #4,484,326, 11/20/84.

[15] Turner, Jonathan S. "Packet Switching Loop-Around Network and Facilities Testing," United States Patent #4,486,877, 12/4/84.

[16] Turner, Jonathan S. "End-to-End Information Memory Arrangement in a Line Controller," United States Patent #4,488,288, 12/11/84.

[17] Turner, Jonathan S. "Interface Facility for a Packet Switching System," United States Patent #4,488,289, 12/11/84.

[18] Turner, Jonathan S. "Packet Error Rate Measurements by Distributed Controllers," United States Patent #4,490,817, 12/25/84.

[19] Turner, Jonathan S. "Fast Packet Switch," United States Patent #4,491,945, 1/1/85.

[20] Turner, Jonathan S. "Fast Packet Switching System," United States Patent #4,494,230, 1/15/85.

[21] Turner, Jonathan S. "Duplicated Network Arrays," U.S. patent application filed.

[22] Turner, Jonathan S. "Integrated Self-Checking Packet Switch Node," U.S. patent application filed.

[23] Turner, Jonathan S. "Distributed Monitoring of Packet Transmission Delay," U.S. patent application filed.

[24] Turner, Jonathan S. and Leonard F. Wyatt. "Alternate Paths in a Self-Routing Packet Switching Network," U.S. patent application filed.

[25] Weinstein, Clifford J. and James W. Forgie. "Experience with Speech Communication in Packet Networks," *IEEE Journal on Selected Areas in Communications*, vol. SAC-1, no. 6, 12/83, 1022–1028.