The plaNET/ORBIT High Speed Network

I. Cidon, I. Gopal, P. M. Gopal, R. Guérin, J. Janniello and M. Kaplan

High Performance Computing and Communications
IBM T. J. Watson Research Center
Yorktown Heights, NY 10598

ABSTRACT

This paper presents an overview of the plaNET/ORBIT Gigabit networking system being developed at IBM. It identifies the various network components and their functionality, and describes how the many services provided by the network are supported. The paper focuses on providing a general view on how the different components operate and interact with each other, rather than on giving detailed technical descriptions of their implementations. Instead, pointers to relevant publications are used to allow interested readers to find additional details.

1.0 INTRODUCTION

The increasing availability of fiber optic transmission facilities has made Gigabit/second networks feasible. Within the next few years, it is likely that such networks will be widely deployed. They will carry a variety of traffic types such as interactive video, file transfers, multimedia mail, remote visualization, distributed computation, etc.

The protocols, switching structures and control algorithms necessary to realize such a network have been the subject of extensive research activity (see, for example the many recent special issues of the IEEE J. Select. Areas Commun. or Commun. Mag. [4-10].) The fact that these networks will have to support different traffic types, each with its own requirements in terms of bandwidth, delay and loss, adds considerably to the difficulty of the design. Within IBM, a major effort was undertaken to understand and develop a comprehensive set of hardware and software components that will enable the construction of a Gigabit/sec network that extends from desktop to desktop. In other words, unlike the traditional carrier view where the "network" ends at some well defined service interface, the view taken at IBM is to consider the end-user machine and the associated hardware and software interfaces as an integral part of the overall network architecture.

The current name of the project is plaNET. The basic components are a Gigabit/sec WAN composed of plaNET switches and a Gigabit/sec MAN/LAN called ORBIT which acts as a feeder or local access mechanism for the backbone WAN. To clear up a frequent confusion, plaNET and ORBIT are based on, respectively, the PARIS [14, 19, 20] and METARING [27, 42] projects. Thus, in many other technical papers, these names are used instead of plaNET and ORBIT.

In this paper we present an overview of the plaNET/ORBIT system. We focus on the functions provided by the network and on the essential hardware and software components. There is little detailed technical description of the hardware implementation or of the algorithms or protocols. Rather, we make suitable references to related publications or conference presentations. In many ways, the paper can be viewed as a road map which points to and pulls together the numerous publications from the plaNET/ORBIT (and the PARIS/METARING) projects.

The various components described in this paper will be deployed in a series of field trial activities planned for 1992 [28, 29, 38, 39]. One such field trial is the AURORA project [28, 29]. AURORA is a high speed network testbed planned under the sponsorship of the Corporation for National Research Initiatives (CNRI), and funded by NSF and DARPA. The research participants are Bellcore, IBM Research, MIT and University of Pennsylvania. We plan to deploy a 4 node network using the components described in this paper and to use it as an experimental testbed to study Gigabit/sec protocols, applications and services. Other field trials planned for the plaNET technology include a trial with Rogers Cable Services in Toronto, Canada [39] and a trial with Bell South Services in Tennessee [38].

2.0 THE NETWORK MODEL

The basic structure of the system is depicted in Figure 1. The backbone WAN is composed of plaNET switches interconnected by point-to-point links. The plaNET switches are shown as octagons in the figure. The topology of the WAN is arbitrary, with a geographical span ranging from within a single room to across a continent. Networks composed of plaNET switches can grow up to a few thousand switches in size with the point-to-point links interconnecting the nodes ranging in length from a few meters to several thousand kilometers. The plaNET switches are currently capable of sustained data transfer rates of 1Gbps per port. They are capable of working with either ATM or variable packet formats and have extensive hardware support for packet processing, buffer management and network control. Details on the switch are provided in section 4.0.

The primary access to the plaNET switch is through an ORBIT ring. ORBIT is a Gb/s buffer insertion ring [27] that is optimized to work in a "seamless" fashion with the plaNET switches. In other words, the network services, packet formats, control protocols, etc. used in ORBIT are identical to those used in the backbone WAN. This eliminates the need for a gateway or router box interconnecting LAN and WAN, thereby eliminating a potential bottleneck. Through the ORBIT ring, it is possible to multiplex onto a single plaNET switch port traffic from multiple access points to the network. As shown in Figure 2, each plaNET node has at least one and maybe several ORBIT ring(s) attached to it to permit access. Fault tolerance is achieved by having an ORBIT ring attached to more than one plaNET node. ORBIT can also act as a stand-alone Gbps LAN in a campus or metropolitan environment. Details on the ORBIT ring (topologies other than a ring are also possible) appear in a later section.

The access points to the network are divided into two categories: “native” and “agent.” A “native” access point is one in which the source of traffic is directly interfaced to the ORBIT adaptor. For example, a native access point could be an RS/6000 or PS/2 workstation (e.g., the video conference machine of Figure 2) with an ORBIT adaptor attached to the I/O bus of the workstation. Programs running in the system processor can directly send and receive data from the network interface. A native access point has a potential performance advantage as it is possible for the transmitting programs to format the data in a manner that is matched to the plaNET/ORBIT network, thereby eliminating the need for any intervening networking software to manipulate or reformat the data. This is further facilitated by the availability of multiple routing modes and packet formats (see section 3.1), which provide application with a wide range of choices and let them select the approach they deem most appropriate for their own requirements. (This flexibility permits the construction of a “transparent” network (see [32])). In the case of voice or video, a native access point may involve a direct hardware interface between the ORBIT adaptor and the video or voice packet stream.

An “agent” access point is one where the source of data is not directly attached to the network. For example as illustrated in Figure 1, to provide IP connectivity across the plaNET network, an RS/6000 could act as a relay for packets from a token ring or ethernet card that is the interface to the IP network. The RS/6000 would interpret the IP destination address and, if the destination was across the plaNET/ORBIT network, would encapsulate the packet in the appropriate plaNET header for transport across the backbone. In such a case, the RS/6000 is acting as an “agent.” In some cases, it may be appropriate to have an agent implemented directly in hardware without involving software running in a general purpose processor.

In the plaNET/ORBIT system, the primary software platform is the RS/6000. As shown in Figure 2, each plaNET node has at least one ORBIT ring and at least one RS/6000 attached as a native access point to that ring. One such RS/6000 is called the Control Point (CP) and executes most of the network control and management software. For reliability purposes plaNET provides a dynamic selection mechanism for the CP among all the contending RS/6000s attached to a given plaNET node. This process is further detailed in section 7.0, and allows for quick recovery of a backbone node in case of CP failure. This recovery can in fact be carried out without impacting the data flow of existing connections.

3.0 TRANSPORT ARCHITECTURE

The philosophy behind the plaNET design is "transparency" [32, 33]. In other words, we attempt to modify the packet as little as possible in order to transmit it across the network. In [32], we outline the advantages of this approach and it is argued that the advantages heavily outweigh the cost. As a result, the transport architecture for plaNET is a heterogeneous one, providing support for various transfer (or routing) modes such as ATM, source routing, multicast, and various priority and loss sensitivity classes.

3.1 Routing modes

We now briefly describe the routing modes available in plaNET and the rationales behind them. All routing modes are natively supported in the plaNET hardware which will be detailed in another section. This ensures that plaNET users can select the routing mode best suited to their own requirements, without sacrificing performance. There are four basic routing modes in plaNET, the source routing mode, the label swapping routing mode, the cell or ATM routing mode, and the multicast routing mode. A short field at the beginning of each packet identifies the routing mode used (see Figure 3 for the plaNET packet format.) All modes except the cell mode allow for variable packet sizes.

The source routing mode (also called Automatic Network Routing or ANR) is a routing mode where the origin prepends all the necessary routing information to the packet to be sent. This information consists of a sequence of labels, where each label corresponds to a local link identifier that uniquely identifies the desired outgoing link within a particular node. The labels are stripped as the packet progresses on the path, exposing each time the label relevant to the next node. For flexibility, labels are either one or two bytes in length, and this choice is local to each node rather than network wide, i.e., nodes with different label sizes coexist within the same network.

A source routing mode has a number of advantages. It eliminates the need for maintaining connection tables at intermediate nodes. As it requires no interaction with the network, it is, therefore, suitable for connectionless services. It can also support connection-oriented services as it provides a fixed path that can be fully selected and defined by the source.

With the help of optional bits in the packet header the source routing mode can also perform additional functions. For example, it is possible to perform a simple reverse path accumulation process. Upon receipt of a packet, the incoming link will insert its own label at the head of the current reverse path field. At any node, the accumulated information then always gives the complete path back to the origin. This process is carried out by dedicated hardware in the link adapters (see section 4.2) without slowing down the progress of the packet. Similarly, an extension called the COPY function permits "copies" of a packet to be dropped at intermediate nodes on its path. Again, the copy is performed “on the fly” without slowing down the packet flow.

As discussed in [32], the flexibility available from the source routing mode is key to the transparent support of a number of applications, and to the more efficient implementation of network control functions and services, e.g., fast setup, non-disruptive path switching, etc. In particular, the combined use of the ANR routing mode, COPY function, and Reverse Path Accumulation, simplifies the negotiations needed to establish connections across the network.

The second routing mode available in plaNET, is a label swapping routing mode similar to the ones used in many other networks, e.g., ATM [3], frame relay [1], etc. In this mode, routing information is carried in a fixed size field or label, that at each node (link) uniquely identifies the connection and the desired local outgoing link. The swapping of this label from node to node ensures that the assigned value remains local to that node (or link), and eliminates the need for a unique network wide identifier for each connection.

The third routing mode is, in fact, a special case of label swapping restricted to fixed 53 byte packets with 5-byte network headers. This is the cell mode that is designed to carry ATM information without the need for any additional overhead. Support for ATM is provided by performing a “syntactic transformation” (which is simply a bit shuffling) of the ATM header at the input link interface so that it appears within the plaNET switch as a plaNET label swapping routing mode packet. The switching hardware deals with the packet in its plaNET format and the output link interface performs the reverse transformation to reconvert the packet to the standard ATM format. The cell routing mode allows plaNET to transport ATM cells as efficiently as any “native” ATM network, i.e., without any additional overhead beyond the regular 5-byte header. The exact “syntactic transformation” that is performed when mapping an ATM cell to/from a plaNET packet is outlined in [33].

In general, label swapping (either the ATM or the variable size packet variety) operates with a fixed-size routing field, which may be better suited to certain applications, e.g., those with small packet sizes. The use of a label that uniquely identifies a connection may also be useful in providing various levels of quality-of-service (QOS) in the network, i.e., apply different buffering and scheduling policies as a function of a connection's identity. On the other hand, the need for a connection identifier also translates in additional cost and limitations for the label swapping routing mode. In particular, it requires tables in each node/link, that store for any assigned incoming label the value of the corresponding outgoing label. The tables have a size which grows with the number of connections supported and more important, require additional mechanisms to update and maintain them. In addition, the need to initialize table entries imposes certain constraints on the connection establishment process, e.g., the flow of packets cannot start before all labels have been assigned. The approach taken in plaNET is to support the label swapping mode for those applications that may require it, but take advantage of the benefits available from the ANR routing mode for most network control flows, e.g., connection setup.

The fourth routing mode supported by plaNET is the TREE routing mode. Its main purpose is to provide a native multicast capability. The TREE routing mode is based on a packet format similar to that of the label swapping mode with a fixed size routing field, that at each node uniquely identifies the links that are "branches" on the multicast tree. This label is either unique network wide, or optionally swapped independently at each output link. The use of a global tree ID simplifies tree establishment, and is useful in implementing certain network control functions, e.g., all network control points are linked together through a unique, well-known spanning tree [14, 21, 23-25, 31]. On the other hand as for the label swapping routing mode, the ability to swap IDs lowers the probability of contention and simplifies the verification process when assigning new tree IDs [40].

The COPY and reverse path accumulation options are available in the TREE routing mode as well. Another useful option available with the TREE routing mode is that of *Remote Access to TREE*. This feature allows a remote station which is not member of a tree, to multicast messages to all stations on that tree. This is achieved by combining the ANR and TREE routing modes. The ANR routing mode is first used to deliver the packet from the remote station to the tree. At this point the TREE routing mode takes over and ensures delivery of the packet to all the stations on the tree.

The availability of a low level (hardware) multicast capability is a key requirement for integrated broadband networks. It is necessary to efficiently support emerging applications such as multimedia teleconferencing as well as advanced network control functions, e.g., distribution of network status information. In particular, plaNET heavily relies on the previously mentioned well-known spanning tree, to rapidly inform all networks nodes of significant changes in the loading of any link. The timely distribution and availability of such information is critical to many network control functions such as route computation, call admission, etc. [11, 14, 23], and is readily achieved by using the TREE routing mode.

3.2 Delay and Loss Priority

Another aspect of the heterogeneity of the transport architecture is the support for different delay and loss priority classes within the plaNET network [15, 20, 22, 35, 36]. plaNET currently defines three delay classes and two loss priority classes within each delay class. Delay classes are served using a non-preemptive exhaustive service rule, i.e., a given delay class is only served when no packets from a higher priority class are waiting, while a threshold mechanism is used to discriminate between loss priorities. The use of different delay classes allows plaNET to provide different QOS to connections with different requirements, e.g., real-time traffic such as voice or video is typically assigned to the highest priority delay class, while some types of data traffic can be adequately supported by the lowest priority delay class.

Note, however, that with Gigabit/sec links it may not be necessary to provide distinct delay classes as even the worst case delay is likely to be acceptable[1]. The distinct delay classes are still valuable as they permit us to dimension the buffering space available for each class differently.

As discussed in section 6.2 (see also [23] for details), the network resources allocated to a connection determine the amount of traffic it is entitled to send. It can, however, be allowed to send in excess of its allocation, provided the additional packets do not interfere with regular traffic. The distinct loss priorities currently provided within each delay class, allow the network to accommodate this excess traffic without impacting the regular traffic. Specifically, excess traffic is sent at low loss priority and is discarded whenever the backlog in a network queue exceeds a given threshold. This guarantees a certain level of protection to regular traffic, irrespective of the intensity or arrival patterns of excess traffic [26].

4.0 THE BACKBONE SWITCH

The wide area backbone network is composed of interconnected plaNET switches as depicted in Figure 1. A key aspect of plaNET is that all intermediate packet handling functions are performed "on the fly" in hardware without software involvement. This enables operation at the full link speed irrespective of the packet size. In this section the hardware structure of the backbone switches is reviewed. Each switch consists of a backplane to which the link adaptors are attached as shown in Figure 4. The physical routing of packets is carried out through the backplane, from their input adaptor to their destination output adaptor(s). The adaptors themselves are responsible for all the packet handling functions requested by the different routing modes, and for the provision of a number of additional control functions, e.g., routing tables updates, link bandwidth management, etc.

4.1 The Switch Fabric

From a performance analysis perspective, the switch is an output queueing switch [37]. It is implemented by a high-speed 64-bit bus interconnecting all the adaptors. The bus has a total throughput of approximately 6 Gbps. While the electronics in our adapters can support link speeds of up to 1.2 Gbps, the links currently supported are SONET OC-12 (at 622 Mbps) and dark fiber at approximately 800 Mbps. The bus speed is, therefore, such that it exceeds the aggregate speed of the eight input links. This together with a simple arbitration scheme that approximates a First Come First Served (FCFS) service policy across the adaptors, ensures that the input queues are of bounded size and that the bulk of the queueing takes place in the output buffers [20]. The arbitration relies on a bus wide cycle which gates the packets re-

[1] The worst case delay on a 10-hop path with 64kbytes buffer at each hop and 1Gbps links is about 5msec, ignoring propagation delay

ceived in each adaptor, so that only packets received within the same cycle are transmitted whenever an adaptor gains access to the bus. Bus access is passed from adaptor to adaptor in a round-robin fashion. Packet transmission is, therefore, round-robin within a cycle and FCFS between cycles. The cycle duration was chosen small enough so that the policy closely approximates a global FCFS service.

Because of the high speed at which the bus operates, implementation constraints led to an active bus structure with a ring topology. In this configuration, the adaptor with current bus access puts 8 bytes of data on the bus[2] in every clock cycle, which are then delivered to the next adaptor on the ring. The signal is regenerated from adaptor to adaptor until it returns to the origin adaptor, at which point it is removed from the ring. Adaptors always transmit full packets, where packet sizes can vary from a few bytes to over 2 kbytes. The arbitration is pipelined with the data transmission so that packets can be transmitted back-to-back on the backplane. The capability of the backplane to handle such a range of packet sizes implies that both fixed size ATM cells and variable size packets can be switched equally efficiently. In addition to allowing for signal regeneration, the ring structure also has the advantage that each adaptor sees all the packets going through the switch. Functions such as broadcast, multicast, and the previously mentioned COPY mechanism are, therefore, readily implemented. The decision of whether a packet should be received (copied) by a given output is distributed in the adaptors, and their implementation is now described.

4.2 Link Adaptors

As shown in Figure 4, the link adaptor consists of a number of components which are involved in the flow of packets to and from the link and across the bus. Going from left to right, these components can be divided into *bus interface, processing element, packet buffering, packet handling (Receive (RHP) and Transmit (THP) Header Processors),* and ultimately a link specific interface not shown in the figure. The functions to be performed by the link specific interface depend on its characteristics and are detailed in the next section. They typically involve both a mapping of the plaNET packet formats onto the transmission format used on the link, and an interface to the appropriate physical layer. The components involved in the routing of packets are essentially the bus interface and the RHP and THP. The bus interface is responsible for the decision of whether a packet should be received or not, while the RHP and THP perform the required packet manipulation functions, e.g., ANR stripping, label swapping, etc. The processing element (Intel i960™ processor[3]) is not involved in the data path. It is only responsible for system initialization and configuration, network control functions such as management of bandwidth on its outgoing link, and for the maintenance and update of "routing tables" used to control the bus interface.

The bus interface is the contact point between the link adaptor and the backplane. It intercepts all packets flowing on the bus, identifies the routing mode used, extracts the corresponding routing field, and decides

2 In case the adaptor has no complete packet waiting for transmission, bus access is immediately handed to the next adaptor.

3 i960 is a registered trademark of the Intel Corp.

if the packet should be copied by the adaptor or simply forwarded to the next one on the ring. The bus interface is also responsible for recognizing and removing from the bus packets that originated from that adaptor. A copy of a packet is received by the adaptor if the address present in the routing field is recognized. Recognized addresses are stored in local tables[4] (SRAM), which are accessed using both the routing mode and the routing field information. In the case of the ANR routing mode, the table is quite small with static entries limited to the the link label(s) associated with the adaptor. The TREE and label swapping routing mode require significantly larger tables, which must be updated, i.e., the appropriate entry must be modified, each time a multicast or label swapping connection is added on or removed from the link. An interesting finding from the implementation of the bus interface, is that the support of several routing modes, e.g., ANR, TREE, ATM, etc., adds little complexity compared to a system targeting only a single routing mode. Most of the routing logic is shared, and only the appropriate routing tables need to be provided. Note, however, that the control overhead to update these tables remains, but it is not part of the data path.

Once the decision has been made to receive a packet, it is copied into the transmit buffer where it is queued for transmission. The buffer provides the rate adaptation and the synchronization functions between the bus and the link which operate at different and asynchronous clocks. The buffer is also the place where the previously mentioned service and admission policies are applied, e.g., discarding of low priority packets in case of congestion or packet scheduling to enforce delay classes. Transmission out of the buffer is typically onto the link, but the packet can also be forwarded to the local processor (i960™). Conversely, the local processor is capable of injecting packets into the queue of packets awaiting transmission on the bus.

The bulk of the packet manipulation functions are performed on the packet headers by the Transmit and Receive Header Processors. The required manipulations vary according to the routing mode, but they again follow similar logical steps. The two major functions performed by the THP include error checking (in all routing modes the header is protected by a Cyclic Redundancy Check - CRC), and routing field modification (i.e., ANR stripping, tree/label swapping). Similarly, the RHP also checks the header of all packets for errors, and if the reverse path accumulation feature is enabled, it updates the reverse path field by adding the ANR of the incoming link (this will be the desired outgoing link on the return path.) A special purpose VLSI chip capable of processing at the full link speed, implements these functions for all the routing modes. Therefore, packet processing not only does not introduce any bottleneck in the system throughput, but more importantly, a single component can support all the routing modes. As mentioned above, this is due to the similarity between the required steps in each mode. This similarity in the packet handling functions required by each routing mode is illustrated in Figure 6, which further reemphasizes the fact that it is possible to support several routing modes for only a small increment in hardware cost.

---

4 For implementation purposes, a single shared SRAM is actually used.

4.3 SONET Link Interface

As shown in Figure 2, the link adaptors of the plaNET switch can interface to a variety of transmission facilities, ranging from dark fibers to standard SONET links. The *SONET Converter* box provides the appropriate interface necessary to map plaNET packets into the standard SONET payload, and conversely extract plaNET packets from received SONET frames [41]. The SONET interface allows plaNET to directly connect to SONET OC-12 links and this capability will be tested in the Aurora field trial [28, 29]. In addition, the SONET interface can also be used to provide direct connectivity to ATM networks as the interface can support the simple bit shuffling (see section 3.1 and [33] for details) that allows handling and routing of ATM cells in the plaNET network. This shuffling will not be needed in future releases as the ATM packet format will natively supported.

4.4 Physical Package

The switch and the link adaptors are packaged in a 19" rack. Each link adaptor is in fact composed of two cards with the 64 bit wide active bus actually a “mid-plane” located between the two portions of the link adaptor. The first card contains the bus interface circuitry, while the second one has the link specific components. One of the slots on the bus is specifically designated as the “control” slot, and the adaptor located in that slot is enabled to perform clock distribution and the previously mentioned bus arbitration. The physical layout of the ring-shaped packet bus on the backplane card is by connecting the various link adaptor cards in the way described in Figure 5. This arrangement eliminates the need for a long wrap-around path to connect the adaptors at opposite edges of the backplane.

In addition to the backplane and the adaptors, the rack also contains power supplies and cooling fans, as well as a rack-mounted RS/6000 which will typically serve as the default control point. This RS/6000 is connected to the link adaptor in slot four through an ORBIT loop, which, however, can be extended to attach other stations as well. This avoids having to dedicate a switch port solely to the control point. In addition as mentioned earlier, in case of failure of the default control point, another RS/6000 attached to the switch through an ORBIT loop can take over the control point function.

4.5 The Next Generation

We expect to grow the capabilities of the switch significantly in the near future, and the design work for the next generation plaNET switches is already under way. The switch will grow to a size of 32×32, with port speeds slightly above 1Gbps. Except for the output queueing feature, this switch differs from the existing one in both structure and technology. It is designed as a shared output buffer switch and relies on a shared memory to perform the switching function. It will also support variable and fixed size packets, and provide a native multicast capability.

5.0 THE ORBIT GIGABIT LAN

As mentioned earlier, ORBIT is a LAN/MAN based on a buffer insertion structure with spatial reuse and a fairness algorithm proposed in METARING [27]. ORBIT was designed to be a native and fully compatible part of the plaNET network. In particular, aspects such as support of plaNET routing modes and control functions, traffic integration, initialization and configuration procedures have been designed to be "seamlessly" compatible with plaNET. These aspects are described in this section, together with the structure and operation of the ORBIT Gigabit LAN.

5.1 ORBIT Topologies and Configuration Process

The ORBIT topologies initially supported in the plaNET network are shown in Figure 7, which also illustrates the connectivity to a plaNET switch. There are three possible initial topologies, unidirectional ring, dual ring and dual bus, all capable of directly attaching to a port of a plaNET switch. The connectivity to the switch port is provided through a station, which can send/receive packets to/from the switch, let ORBIT packets pass through, and even receive packets. In order to allow such routing decisions, the station typically uses two routing IDs. One allows local reception of packets, while the second identifies outbound traffic for the switch. Details on the routing mechanisms are provided later, and we now outline the process used to initialize and configure ORBIT.

The purpose of the ORBIT initialization and configuration process is to provide a simple procedure, that satisfies the following requirements:

- Automatic "discovery" of ORBIT topology,
- Dynamic assignment of local station labels or IDs,
- Error recovery and graceful degradation in case of failures.

This is achieved through a topology discovery algorithm and a labelling algorithm. The topology discovery algorithm provides a real-time non-disruptive tracking of topology changes and is capable of identifying the initial configuration, as well as detecting link failures and stations ordering, addition, and removal. When detecting failures, the algorithm attempts to gracefully degrade the LAN operation. For example, a link failure in a dual ring ORBIT triggers a configuration change into a dual bus topology.

The topology algorithm is complemented by a labelling process. During this process, local IDs or labels are assigned to active stations using their "well-known" Global ID (GID), e.g., a 48-bit IEEE address, and one station is selected as *leader*. The local IDs provide a more compact addressing format, and the leader is responsible for handling various error scenarios, e.g., removal of looping packets in a ring configuration.

ORBIT is considered operational once a leader has been elected and all active ORBIT stations have been provided with complete knowledge of the current configuration and topology, i.e., location and ID of other active stations. This information can be dynamically updated each time a station is added or removed from ORBIT, without the need to disrupt operation or restart the initialization process.

5.2 Integration with plaNET

In order to allow for a seamless integration of ORBIT in the backbone plaNET network, identical packet formats and routing modes are used. The buffer insertion structure of ORBIT can handle both variable and fixed size packets, and full compatibility with the plaNET packet formats is further ensured by selecting the same structure and syntax for routing and control fields. The key aspect, though, is the mapping onto ORBIT of all the plaNET routing modes. Specifically, ORBIT will support the ANR source routing mode, the label swapping routing mode (including ATM), and the TREE routing mode. While all the plaNET routing modes are natively provided, several "interpretations" specific to ORBIT have been made in order to better accommodate the requirements of Gigabit/sec LANs.

One such interpretation is in the usage of ANR source routing labels, which in the backbone network are assigned for each hop, i.e., they point to the next outgoing link. In the LAN environment, both the simpler topologies and the potential for large number of hops (one for each station), called for some modifications of the use of ANR labels. Specifically, ORBIT ANR labels correspond to the previously mentioned local IDs, and identify attached stations rather than links. Because of this, only the ANR label of the destination needs to be specified when routing a packet within ORBIT, i.e., the labels of intermediate nodes need not be included in the routing header. Intermediate nodes simply forward the packet when they do not recognize their own ID. Only the destination receives the packet and removes it from ORBIT upon identifying its own ID. In a sense, the ORBIT local loop is viewed as equivalent to a single hop between the network node and any attached station.

We now illustrate the above discussion through a few examples. When referring to Figure 8, the routing of a packet from station 1 to station 6 in ORBIT 1 only requires that the ANR label <6> of the destination be specified in the routing header. Next, we consider the routing of a packet between station 1 on ORBIT 1 and station 4 on ORBIT 2. It uses the following string of ANR labels: $< 5', 6, 8, 2, 4 >$, where the first ANR label (5') indicates that the packet must be forwarded to the attached switch (node 2). As mentioned earlier, stations attaching ORBIT to a plaNET switch have two labels which allow them to either receive packets or forward them to the switch. Packets whose first ANR label is neither of these two labels are simply passed on to the next ORBIT station. In the case of station 5 in ORBIT 1, the label 5 identifies packets destined to the station, while the label 5' corresponds to packets that must be forwarded to the switch. Packets forwarded to the switch are stripped of their current first label, e.g., the previous packet is forwarded to the switch with labels $< 6 , 8 , 2 , 4 >$. Routing in the plaNET network is then carried out as previously defined, and the packet flows through nodes 2, 5, and 6, before reaching ORBIT 2 where it is finally received by station 4.

One addition to the plaNET routing modes that is specific to ORBIT, is the “neighbor” routing mode [27]. This routing mode is based on the ANR routing mode, i.e., it uses a dedicated ANR value, but the packet forwarding is always limited to a single hop. This routing mode is useful during initialization, when ORBIT stations need not yet know each others' IDs. The neighbor routing mode also provides a simple mechanism to support routing based on GIDs. Packets are forwarded from station to station, until they reach their final destination as indicated by the carried GID.

The support of the label swapping and tree routing modes is essentially a direct mapping of how they are handled in plaNET. ORBIT nodes typically[5] forward to the next ORBIT node any packet whose label or tree ID they do not recognize. When using the label swapping routing mode, the destination is again responsible for receiving and removing the packet from ORBIT. Note that this may correspond to forwarding the packet to an attached switch. The case of the tree routing mode is somewhat different as any node recognizing the tree ID receives a copy of the packet, but the elected leader is the one removing tree packets. This is necessary to ensure that all potential ORBIT destinations get a chance to “see” all tree packets.

5.3 ORBIT Media Access Mechanism

The next key aspect to seamless operations between plaNET and ORBIT, is to ensure that the ORBIT media access mechanism is sufficiently flexible to support the functionality required by plaNET. In particular, plaNET differentiates between several traffic classes, e.g., reserved and non-reserved (see section 6.2), and it is necessary that this distinction be preserved in ORBIT.

The ORBIT media access protocol not only ensures that the different plaNET traffic classes can be adequately supported, it also leverages the approach used in plaNET to better provide full compatibility with the plaNET network control flows and bandwidth allocation procedures. Specifically, ORBIT stations maintain for each of their outgoing links a measure of the current amount of bandwidth allocated to reserved traffic. This quantity is updated as connections are removed or added using the plaNET setup and bandwidth accounting processes. These procedures are outlined in a later section, and provide for a simple distributed mechanism to determine existing link loads and decide when to accept or reject new connections.

As an example, if we assume the previous connection example between station 1 on ORBIT 1 and station 4 on ORBIT 2, the setup process verifies that enough bandwidth is available not only on network links

[5] An exception is the elected leader, which may optionally decide to “kill” a packet if an error condition is detected, e.g., the packet is endlessly circulating.

but also on the links connecting ORBIT stations, e.g., links between stations 1 and 2, 2 and 3, etc. on ORBIT 1. In case there is not enough bandwidth on any of the links on the path, the connection will be rejected. Note that this approach preserves the advantage of spatial reuse for reserved bandwidth connections on ORBIT. For example, a connection between stations 1 and 4 on ORBIT 1 only reserves bandwidth on the ORBIT links between the two stations. Bandwidth on the other ORBIT links, i.e., links between stations 4 and 1, remains available to other connections.

The above approach limits the maximum reserved traffic loading permissible on ORBIT links (as well as plaNET links), which ensures proper operation in the absence of non-reserved traffic. Non-reserved traffic is, however, a major component in the LAN environment, where many stations wish to communicate without requiring the overhead of a connection setup. It is, therefore, important to provide this traffic with rapid access to ORBIT, while limiting the impact it can have on reserved bandwidth connections. In ORBIT, this is achieved by combining the SAT-based fairness mechanism [27], with a simple local priority scheme. This preserves the distributed nature of the media access control, and adequately guarantees resources availability to reserved traffic while allowing immediate ORBIT access to non-reserved traffic whenever possible.

Specifically, each ORBIT station distinguishes between its local reserved and non-reserved traffic and separates them into different access buffers. Reserved traffic, because its volume is controlled through the bandwidth allocation procedure, is allowed to access ORBIT whenever possible, i.e., the insertion buffer is empty. On the other hand, non-reserved traffic only accesses the ring when no reserved traffic is waiting, enough transmission quotas are available, and the insertion buffer is empty. Transmission quotas are granted using the previously mentioned SAT mechanism, which ensures a "fair" distribution of ring access between non-reserved traffic from competing ORBIT stations[6]. The SAT control signal propagates directly through the hardware, and therefore goes from station to station with minimal latency. Whenever it is received by a station, the SAT signal grants a new transmission quota[7]. Starved stations, i.e., active stations which have not been able to send their full quota, are allowed to hold the SAT control signal until they are "satisfied." Holding the SAT eventually exhausts transmission quotas for non-reserved traffic in ORBIT, and therefore ensures that a starved station is ultimately satisfied.

The SAT control is also held by a station if its local reserved traffic is not granted ORBIT access consistent with its bandwidth reservation. This can occur because of excess non-reserved traffic from upstream stations, i.e., upstream stations do not coordinate their non-reserved traffic transmissions, which can, therefore, exceed the residual link capacity left by reserved connections. A station decides that its reserved traffic has not been granted sufficient access to ORBIT, based on a simple delay threshold mechanism. Specifically, each packet forwarded to the reserved traffic buffer is time stamped with the value $t_{\max}$ of its maximum acceptable transmission time ($t_{\max} = t + \Delta$, where t is the current time and Δ is the maximum acceptable delay in the access buffer.) A station decides to hold the SAT, whenever the time stamp of the packet at the head of the reserved traffic buffer exceeds the current time. This scheme

---

6 See [27] for a detailed description of the SAT quota allocation mechanism.

7 In the dual ring topology, the SAT signal propagating on one ring grants quotas for transmission on the other ring.

allows stations to control the volume of upstream non-reserved traffic to ensure that their reserved traffic is granted adequate ORBIT resources. In addition, it has been found to be sufficiently responsive, i.e., the worst case access delay for reserved traffic can be kept small enough, except possibly for very large systems. In very large systems, the potentially long evacuation time[8] of non-reserved traffic can result in large worst case delays. Enhanced mechanisms which overcome this limitation at the cost of minimum additional complexity, are currently being investigated.

5.4 ORBIT Adaptor Structure

In this section we describe the structure of an ORBIT adaptor that implements the different functions mentioned in the previous sections. This adaptor allows microchannel based hosts, i.e., RS/6000s and PS/2s, to attach into ORBIT. As mentioned earlier, the attached hosts can be traffic sinks and sources themselves or serve as gateways (routers) between the plaNET/ORBIT network and other external networks, e.g., IP. The overall structure of the ORBIT adaptor is shown in Figure 9, and we now briefly review its components and their functionality.

The top portion of the figure shows the host bus (microchannel) and the associated interface logic. This provides connectivity with both the host system and others adaptors sitting on the host bus, which can themselves provide connectivity to other networks. The bus interface is followed by a large data store in which packets received from ORBIT or waiting to be transmitted on ORBIT, can wait. Packets received from ORBIT must wait for the host bus and/or the destination adaptor to become available. Packets destined to ORBIT must wait to be scheduled for transmission, i.e., forwarded to the appropriate access buffer. In the case of reserved traffic, this scheduling depends on the access control function (Leaky Bucket) implemented in the local processor (i960™). This function and its implementation are described in another section. Non-reserved traffic is forwarded to its access buffer on the basis of available SAT quotas and space in that buffer. Note that in addition to the data path through the microchannel, the ORBIT adapter allows data to be received from and sent to ORBIT through an auxiliary or external port. This external port is used to provide connectivity between ORBIT and high speed devices. For example, it is used for the transmission of video signals in our multimedia workstation (see section 8.2).

The next component in the adaptor is the local processor (i960™), which in addition to the previously mentioned access control is also responsible for the overall operation of the adaptor. In particular, it maintains and updates status information, and performs basic packet handling functions. For example, in the case of packet transfers to and from other adaptors, it "prepends" plaNET/ORBIT routing headers to packets received from the host bus, and removes any residual header for packets received from ORBIT. The processor is also responsible for supervising the attached outgoing ORBIT link. In particular, it executes the initialization and configuration processes and participates in the topology updates necessary to

8 This is the time required to complete transmission of all currently allocated SAT quotas.

detect and recover from failures. It also handles all connection setup requests and manages the link bandwidth. For performance purposes, the processor is, however, not responsible for actual data movement.

Packets received from ORBIT or scheduled for transmission on ORBIT are temporarily stored in high speed FIFOs, which provide the required rate adaptation between the Gbps links and the data store. As mentioned earlier, distinct access buffers are provided for reserved and non-reserved (SAT) packets. In addition, a third logical buffer is available, that allows the direct attachment of an external high speed device, e.g., video coder. Conversely, received packets can also be directly forwarded to the external port.

The next layer of components provides some simple packet handling and control functions. The Transmit and Receive End-point Data Processors (TEDP and REDP respectively) operate at a clock rate compatible with the link speed, and are capable of selectively computing and checking, respectively, a CRC on the data payload of each packet. This provides for end-to-end error checking if desired. The REDP is also capable of determining which packets must be forwarded to the external port. The Rx and Tx control logic control when packets must be received and transmitted, respectively. For example, the Tx control logic enforces the local priority of reserved traffic over non-reserved traffic, and determines the handling of the SAT control mechanism.

The last set of components of the adaptor are those directly involved on the ORBIT data path, i.e., they handle all the ORBIT traffic. They include the Rx and Tx logic which provide the optical to electronic and electronic to optical interfaces between the link and the adaptor, the link or insertion buffer which allows the temporary storage[9] of an ORBIT packet that arrives while a local packet is being transmitted and also serves as a "flexible buffer" which absorbs clock rate fluctuations, and the Header Processor (HP.) This component is essentially identical to the Transmit Header Processor present in the plaNET link adaptors. It performs basic header manipulation and error checking functions, i.e., routing mode identification, routing decision and routing field modification, etc., and has a throughput compatible with the link speed.

6.0 plaNET CONTROL FUNCTIONS

In this section we briefly review the plaNET control functions, their functionality, their operation, and how they are provided. The interested reader is referred to [14, 22, 23, 25] for additional details and references. The control functions are provided by the plaNET Control Point, which consists of an RS/6000 attached through an ORBIT loop. As mentioned earlier, for reliability purposes there can be more than one RS/6000 capable of becoming the CP. Section 7.0 provides additional information on the mechanism used to select which RS/6000 is to take on the CP function.

[9] ORBIT packets typically cut-through this buffer, so that transmission delay is kept minimal.

The key aspect of the plaNET control is that it is distributed but uses global information about the network status. This global information is provided through a replicated database maintained in each network node. We first describe the mechanisms used to maintain these databases, and then focus on the network control functions themselves and how they interact with the database maintenance process. The relations and interactions between the different control functions are also identified.

6.1 Network Information Maintenance and Distribution

A major challenge faced by control functions in high-speed networks is to keep up with the processing needed to handle the very large number of connections these networks will support. A centralized approach where a single entity maintains all the required status information has the advantage of simplicity, but is likely to become a severe bottleneck. plaNET has, therefore, chosen a distributed approach where nodes are capable of making their own control decisions, based on information kept in local replicas of the network database.

The purpose of replicating the network database is to provide all nodes with a common view of the overall network status, without the need for a central entity responsible for handling all control queries. This, however, raises the question of how to keep this information up-to-date. This is particularly critical as not only static information such as link and node names, link characteristics, etc., are needed, but most control functions (e.g., route computation, call admission, etc.) typically require access to dynamic information such as actual link loads. The efficient and timely distribution of updates to maintain consistency between the content of the databases and the current network state is, therefore, a key issue. This is achieved through the combination of distributed, dynamic algorithms and the built-in hardware support for multicasting.

Topology updates are sent over a *Spanning Tree* that connects all the network control points (the *CP Spanning Tree*). This spanning tree uses the plaNET multicast routing mode to rapidly distribute updates to all the nodes. Specifically, any packet put on the CP spanning tree is routed by the hardware through all the nodes, which copy it without slowing down its delivery to other nodes. In parallel with the use of the CP spanning tree to rapidly disseminate topology updates through the network, a distributed tree maintenance algorithm is continuously run in all nodes [14]. The purpose of this algorithm is to guarantee the consistency of the CP spanning tree even in the case of link failures, addition or removal of nodes, etc. This algorithm interacts with the topology updates, i.e., uses the status information they provide, and determines how to evolve the multicast tree so that it always correctly spans the entire network.

Using the above mechanisms, each node in the plaNET network is, therefore, capable of maintaining a consistent replica of the network database. As mentioned earlier, the contents of this database can be divided into static configuration information, and dynamic load information. As can be expected, most of the topology updates are triggered by load information changes. In particular, each node monitors the load levels on all its links, and informs other nodes of any "significant" changes (see [23, 35, 36] for additional details.) Link loads change as a result of connections being added or removed. The establishment and release of connections is carried out as part of the resource allocation and management functions, which are now detailed.

6.2 Resource Allocation and Management

In this section we briefly sketch the operations and interactions of the resource allocation and management functions, and the reader is referred to [11, 14, 23, 25, 34-36] for details. As mentioned above, a key aspect is the control of connection establishment and release. This process must take into account not only connection characteristics and requirements, but also the current network status as known from the content of the network database. The goal is to satisfy connection requirements as well as possible, while ensuring proper network operation. This is achieved through the control cycle shown in Figure 10, which describes the different control processes involved in the establishment[10] of a connection.

The first step in the connection establishment process is the forwarding of the connection request to the network. The details of this process are outlined later, but it essentially provides the network with the desired destination(s) and amount of resources (e.g., bandwidth) requested by the connection, and possibly other relevant Quality-Of-Service (QOS) parameters. Once this information is available to the network, the next step is to determine the best possible route to the destination that can accommodate the new connection. This route is chosen to optimize the usage of network resources, while ensuring that the connection requirements can be met, i.e., enough bandwidth is available. Once a route has been selected (assuming one was found), it is then necessary to secure the resources needed along the route. This is the purpose of the bandwidth reservation phase, which prevents over-allocation of resources because of potential discrepancies between the information stored in the network database and the actual status of network resources. Note that such discrepancies are unavoidable because of the non-zero time needed to propagate updates throughout the network.

The bandwidth reservation is carried out through a *setup* message, which is sent using the ANR routing mode with the COPY function enabled. This ensures rapid delivery of the message to all intermediate nodes (see the earlier description of the plaNET routing modes.) Upon receipt of such a message, each node checks if enough bandwidth is available on its targeted link. This function is performed using a simple and yet flexible bandwidth accounting function described in [23, 34]. As new connections are being added or removed, each node tracks the corresponding changes in link loads. Whenever load/utilization level on a link changes "sufficiently," a topology update is multicast on the CP spanning tree to notify the network nodes of this new status [14, 23]. This new information is then used upon receipt of the next connection request.

---

[10] The functions involved when releasing a connection are essentially similar, with the exception of route computation which is not needed.

In parallel to the network control cycle, each connection is itself subject to a setup cycle as illustrated in Figure 11. This setup cycle shares several components with the network control cycle, but it also provides for more specific connection-oriented functions. In particular, it is responsible for translating the connection request into bandwidth requirements which can be used as input to the route computation procedure. The setup control cycle also ensures that positive acknowledgements have been received from all intermediate nodes before the source is allowed to start data transmission, i.e., the bandwidth has been effectively secured on all the links in the path. In addition, since acknowledgements from intermediate nodes can allocate less bandwidth than initially requested, it may be necessary to appropriately adjust the parameters of the access control regulating the connection. The purpose of the access control is to ensure that connections cannot exceed the amount of bandwidth they have been awarded while still allowing for some flexibility in how packets can be forwarded to the network. See [35, 36] for additional details on the plaNET rate control mechanism, and the approach used to relate it to the bandwidth allocation procedure.

The last step of the setup cycle takes place after the connection has been established. It consists of monitoring the traffic generated by the connection. As discussed in [23, 35, 36], this monitoring serves several purposes. It helps identify long term changes in connection characteristics, which may warrant an adjustment in the amount of allocated bandwidth. It is also an effective means to capture the impact of traffic statistics, and tune the connection representation accordingly. This is particularly useful for connections, which either cannot characterize their traffic accurately or exhibit complex behavior. Finally, traffic monitoring is also a powerful tool to identify misbehaving users, which can then be prevented from further abusing the network. Specifically, despite the presence of the resource allocation and access control mechanisms outlined above, it is still necessary for the network to maintain some buffer and bandwidth "margins" to protect well-behaved users against possible performance degradation caused by misbehaving ones. As shown in [35, 36], these margins can be significantly reduced, when traffic monitoring is used in addition to the access control mechanism.

6.3 Other Network Services

In parallel to its core control functions, the network also provides a number of additional services, that assist in the efficient operation of control functions. Two of the most important such services are *Directory Services* and *Multicast Set Management*.

Directory Services are necessary to allow users to easily establish connections across the network. Specifically, directory services provide the necessary mapping between resources' names and the corresponding network addresses. The resource can be external to the plaNET network, i.e., an IP network attached to plaNET through an access agent, or internal, i.e., a plaNET native agent such as a workstation attached to an ORBIT ring.

Directory services are provided through a distributed database whose content is spread over all the network nodes in a fashion similar to that of [16]. The portion of the database in each node is controlled by a directory agent, and all such directory agents are connected to each other over the network. The directory database in each node contains information about its local resources. In addition, it also maintains a small "cache" for network addresses of remote resources, that have been the target of recent requests.

A query for the resolution of the transport address of a given resource is typically triggered when a connection request is received. It is then necessary to map the name of the requested destination into the corresponding plaNET network address. If the destination is local or the information is already stored in the local cache, the requested information is readily available. In cases where the queried resource is not known locally, directory databases in other nodes must then be searched. This is achieved by broadcasting the query onto a spanning tree connecting all the directory agents[11]. The message is sent with the "Reverse Path Accumulation" option enabled (see the section on routing modes), to facilitate the return of the requested information. The node with which the queried resource is registered, will generate a reply with the information contained in the corresponding entry of its local directory database. Upon receipt, this information may be stored by the requesting node in its local directory cache, and forwarded to the initiator of the query.

Another important service provided by the network is *Multicast Set Management,* which provides support for the establishment and control of multicast connections. In other words, the combination of the native plaNET multicast routing mode and the multicast set manager control functions, allows the network to provide a full multicast service to users. The service (see [13] for details) relies on groups comprised of users that can be involved in a multicast connection. Each group is assigned its own ID and has a *leader* responsible for managing the group membership, and possibly also the trees supporting connections in which members of the group are involved. In addition, one of the nodes in the network is also responsible for maintaining a list of existing groups with their IDs and leaders. The group structure provides readily available information on the location of users that are the target of a new multicast connection request. This is particularly important to allow the rapid identification or computation of the multicast tree to be used. (Note that that multiple connections can share the same multicast tree.) In addition, the use of groups and leaders provides a simple mechanism to recover from failures, i.e., groups which have been split because of a network failure can rejoin when connectivity is reestablished.

7.0 Control Point Election and Configuration

In this section, we outline the mechanism used to decide which of the possibly several contending RS/6000s attached to a plaNET switch through ORBIT loops, is to become the Control Point for this network node. This selection is the responsibility of the *Control point Configuration Process* (CCP). Once elected, the CP must, in addition to performing the previously described network control functions, also keep a complete "picture" of the entire network node. This includes information about link adaptors and how they are configured, as well as information about all the ORBIT stations attached to the node.

[11] Note that the spanning tree may be protocol specific for a class of directory agents.

The process in charge of maintaining and distributing this information to other processes, e.g., network management or various network control functions (see section 6.0), is called the *Local Configuration Process* (LCP). We now briefly describe the functionality and operation of both the CCP and the LCP.

7.1 Control Point Configuration Process

The goal of the CCP is to allow automatic selection of a CP among the contending[12] ORBIT stations at node initialization, i.e., when the plaNET node is first turned on, and in the event of the failure of the current CP. The CCP is started after the previously described ORBIT configuration and initialization process has been completed (see section 5.1), so that information about the local topology and labels is available in the ORBIT stations.

The CCP is based on a simple *polling* scheme, which relies on the use of two basic polling message types. The polls are sent using a predefined multicast TREE address, so that they are received by all ORBIT stations attached to the same plaNET node (messages are, however, not propagated on links connected to other plaNET nodes.) The first type of polling message is sent when an ORBIT station fails to detect the presence of a CP on its plaNET node, e.g., using some timeout mechanism. This message contains the ID of the station and indicates its contention to become the CP of the plaNET node. The second type of polls is used to notify (periodically) all ORBIT stations of the presence and identity of the elected CP. When several stations are simultaneously contending to become the CP, the Global ID (see section 5.1) of the stations are used to resolve the contention, e.g., the station with the largest GID wins.

The above description, although superficial, outlines the mechanisms used to dynamically select a CP in a plaNET node. The approach is both simple (despite the many omitted details) and robust to station failures. It also allows new ORBIT stations to easily locate and contact the existing CP to obtain various network services, when they are first brought on-line. In particular the polling messages are sent with the reverse path option enabled, which directly provides receiving ORBIT stations with the return path to the CP.

7.2 Local Configuration Process

The Local Configuration Process is instantiated only in the station elected as the CP, and it is responsible for maintaining up-to-date information on the status of the network node and the attached ORBIT stations, and for making this information available to other processes. The information maintained by LCP about

[12] For reasons of both security and processing capability, not all ORBIT stations may be CP “eligible.”

the link adaptors includes link adaptor status, e.g., link state, buffer threshold and statistic counters values, etc., and a copy of the routing tables in each adaptor. LCP exchanges information with the each of the link adaptors through separate RTP connections. RTP is the transport layer protocol used by all the plaNET control functions, and is detailed in section 8.1.2.

Similarly, LCP also keeps information about all the ORBIT stations attached to the plaNET node and interacts with the ORBIT Configuration Process (OCP, see section 5.1) as well as the previously discussed CCP. The OCP in a designated ORBIT station initiates an RTP connection, which it uses to communicate with forward all its available configuration information to the LCP. This OCP is also responsible for informing the LCP of any configuration change it detects.

LCP data is kept in shared memory where it is available to potential client processes through a simple Shared Memory Interface (SMI, see section 8.1.1 for more details.) A particularly important client process is the Network Management Agent which provides the ability for a network operator at a network management console to view (and in some cases alter) the status of the network. The Simple Network Management Protocol (SNMP) [17] is the protocol used between the console and the agent in the CP. The agent in the CP obtains information from LCP and stores it in the form of an SNMP Management Information Base (MIB). This permits access to the information through the SNMP protocol. The SNMP protocol can also be used to alter some pieces of information. These requests will be received by the agent and then sent to LCP (possibly for subsequent relaying to the link adaptors). Examples of such alterable attributes include the reporting period of statistics counters, the states for which the LCP should generate alerts to the SNMP Agent, etc..

8.0 STRUCTURE AND OPERATION OF plaNET END-POINTS

In this section we describe the structure and components of plaNET end-points, i.e., traffic sources and sinks that attach to the plaNET/ORBIT network. For purposes of simplicity we assume a workstation based environment. This workstation can either provide native access to the network, e.g., it runs applications that directly interface to network services, or it can serve as an access agent that provides connectivity to and from other networks, i.e., IP. In addition to RS/6000s, we currently permit PS/2s to attach to the ORBIT network. The PS/2 is primarily used for its multimedia capabilities. In a later section, we describe briefly a multimedia videoconferencing workstation designed around the PS/2 with an ORBIT network interface. In general, an end-node can serve both purposes, i.e., provide both native access and access agent functionality. Figure 12 shows the overall structure of such an end-point, that supports both local applications and provides connectivity to an attached IP network (connected over a T3 link.)

8.1 Software Components

We first concentrate on the software components that enable applications (end-users) to efficiently access and communicate across the plaNET/ORBIT network. Details on the access agent functionality are provided later. The main components involved in providing connectivity across the plaNET/ORBIT network include, in addition to the resident portion of the *plaNET Control Software,* the *Shared Memory Interface* (SMI), the *ORBIT Device Driver* together with its high performance *Fastpath,* and the *Rapid Transport Protocol.*

8.1.1 Shared Memory Interface and ORBIT Fastpath

The Shared Memory Interface is a set of functions and definitions that allow a number of processes to efficiently communicate and coordinate their activities through a region of shared memory. The only implementation requirement is that processes have read/write access to a common region of memory. Communications between processes are carried out through a flexible and extensible set of *signaling* primitives, that let processes select how they prefer to be notified of a pending communication. In brief, SMI consists of a buffer pool and queues, together with methods to put, get, allocate, free, etc. buffers to queues and the necessary notification/signaling procedures.

The ORBIT Fastpath is the process responsible for data movement between the processor memory and ORBIT using SMI signaling. It is intended, when combined with the SMI, to facilitate inter-process communications and provide a high performance data path between user applications and the plaNET/ORBIT network. This is achieved through pinned memory pages which are shared by the user processes, the device driver, and the ORBIT adaptor. Data can then be efficiently transferred between user space and ORBIT, by minimizing the amount of unnecessary data movements. On the transmit side (user to ORBIT), user virtual memory pages are mapped onto the device (ORBIT) "register." On the receive side (from ORBIT), an interrupt is generated to the device driver and the packet header itself carries a "channel ID," that directly points to the appropriate SMI queue.

8.1.2 Rapid Transport Protocol

Another key component of a plaNET end-point not shown in Figure 12, is the transport protocol used to provide and efficient and reliable transport service. This transport protocol, named the *Rapid Transport Protocol* or RTP, provides a connection-oriented service with reliable, full-duplex delivery of arbitrary length messages. This transport protocol does impose some additional overhead when compared to a basic connectionless service. However, in addition to providing significantly greater functionality, it is also optimized for the plaNET/ORBIT environment. In other words, it makes a number of assumptions based on the features of high speed networks, which enable a high performance implementation (see [2] for a related although somewhat different approach.)

While the detailed description of RTP is beyond the scope of this document, it is important to mention some of its key principles. RTP can be characterized as an *optimistic* protocol, i.e., it assumes that the network is "mostly" reliable and that applications are typically ready and willing to communicate. This underlying optimism leads, for example, to a simplified connection establishment process, that allows much faster setup under the assumption that the receiving party is ready to receive the call, i.e., data is already included in the initial setup message. Similarly, a large "first" message will be segmented and all the segments *streamed* to the destination without further delay. In addition to this optimism, RTP also attempts to minimize handshaking between end-points and provides for some simplified acknowledgment procedures, e.g., a "no retry" option is available, which allows RTP to also provide connectionless or datagram service efficiently. As illustrated in Figure 13, which gives a more detailed picture of the end-point software components, RTP because of its efficiency is also used pervasively for communications between network control functions.

8.1.3 Access Control

In the plaNET/ORBIT network, the access control is distributed to the end-points, and this traffic regulation function is typically performed in the adaptor attached to the ORBIT LAN (see Figure 9.) This access control consists of a generalized Leaky Bucket (LB) [12, 15, 23, 35, 36], which determines when the packet waiting to be transmitted can be scheduled to access ORBIT. This mechanism is implemented in software, which provides for both flexibility and the ability to handle a large number of connections.

The software implementation, rather than emulating the traditional leaky bucket hardware operations, e.g., periodic token generation, etc., computes for every packet at the head of its transmission queue, the earliest time at which it can be scheduled. That is, instead of simply deciding **if** the packet can be sent, the software computes **when** the packet should actually be sent and updates the LB variables accordingly. It is this "forward-looking" characteristic, which ensures a simple and yet flexible implementation. This simplicity is highlighted in Figure 14, which shows the few basic steps that need to be carried out when scheduling a packet. Note that the figure shows the "marking" feature of the LB implementation, which lets applications select to have some packets sent as *red* instead of waiting for enough (green) transmission tokens to accumulate. Red packets are given a lower loss priority inside the network (see section 3.2.) This is provided without any significant impact to the overall complexity of the implementation.

8.2 Multimedia Workstation

The objective of the multimedia workstation is to display and generate a mixture of different media and to be able to transmit and receive this information over the plaNET/ORBIT network. For example, in a collaborative work mode, two or more users may be interacting with each other across the network. The interaction may involve a videoconference within a window on the screen with full-motion pictures of each participant and high quality composite audio, another video window containing a clip retrieved from a remote video server, another window containing a shared workspace or electronic "black-board," and perhaps another window containing a document with imbedded images that is being collaboratively edited.

As mentioned earlier, a PS/2 (see Figure 15) is used as the base for our initial multimedia workstation. The system is described in detail in [18]. Briefly, it consists of a PS/2 with an ORBIT card acting as the network interface. Data information received from the ORBIT card is passed to the microchannel as described in section 5.4. Image or video information received (to be sent) from (to) the network is passed directly from (to) the ORBIT card to (from) a card known as the MMT [18]. This is done through the auxiliary or external data port of the ORBIT adapter mentioned in section 5.4. The MMT card performs compression and decompression (using the JPEG standard), and video "mixing" of multiple incoming video streams into a single composite stream. It also performs the transformation from the packet domain into the bit stream domain, compensating for factors such as variable packet delays, etc. After going through the MMT card, the video stream is passed (again through an "auxiliary port") to a display card which places it within a window on the PS/2 screen.

The basic advantage of the design is that the video path is not across the microchannel. While the PS/2 can set up connections, determine placement of video windows, set compression and other parameters, it does not have to be involved in the actual movement of video information. This low level movement is handled by hardware. This design choice reduces communication and processing loads within the workstation, thereby permitting it to run more sophisticated applications, and to use higher quality (and higher bit-rate) video.

8.3 Connectivity Across plaNET/ORBIT

We now outline how a user requests the establishment of a connection through the plaNET/ORBIT network, and point to the different processes and components involved. An application wishing to establish a connection across ORBIT and plaNET, must first forward the request to a *Connection Agent* (CA), which provides all the necessary interfaces to the appropriate network control functions. Communications between the application and its CA are typically carried out through the SMI. As mentioned earlier, the request specifies both the desired destination and connection characteristics. The first step when handling such a request, is the gathering of local routing information, i.e., "tail" information that identifies the path from the end-user to plaNET, and the resolution of the destination's network address. This is achieved by invoking the Directory Services (DS) component.

Two cases must be distinguished depending on whether the destination has been locally cached or not. In the first case, the local Directory Services can identify the destination without any further interaction with the network. In case the requested destination is not contained in the local cache, it is necessary to initiate a query to the plaNET Directory Services. (Note that in the case of a stand-alone ORBIT, the query will be sent locally on ORBIT.) This is again carried out by the local DS acting as a “stub” to the associated function in the network control point. The plaNET network DS returns the desired destination network address and appropriate tail information.

Once network addresses are available for both the origin and destination, the CA passes this information together with the desired connection characteristics to the local Route Selection (RS) process. This component, acting again as a stub to the corresponding network control function, asks for a path to the destination. This path is then computed by the (Topology and) Route Selection (TRS) process in the Control Point, and the required information, i.e., routing headers, is returned to the local RS stub which forwards it to the CA. The CA then initiates the setup process through the network. Once this process has been successfully completed, the access control parameters are initialized and transmission is ready to start. As mentioned earlier, all the communications between the different control functions in both the end-point and the network node, are carried out through RTP. In addition, for those applications that elect to use it, RTP also takes care of formatting application data units into network packets and attaching the routing header. Conversely, on the receiving side RTP regenerates application data units from network packets.

8.3 Access Agent Functionality

plaNET end-points in addition to allowing local applications to access the plaNET/ORBIT network, can also serve as access agents for end-users attached to other networks. Two important examples are IP and SMDS networks. Attachments of both networks are supported through RS/6000-based router platforms, with a similar overall structure as illustrated in Figure 16. Essentially, the per packet routing functions are distributed in the adaptors and limited to simple table lookups. The initialization/maintenance of these tables is performed by an associated “routing daemon” in the RS/6000. For purposes of illustration, we now outline the forwarding of an IP packet onto the plaNET/ORBIT network.

Upon receipt of an IP packet over the attached link, the adaptor first extracts the IP address from the routing header and uses it to access a local routing table. The associated table entry identifies the local destination adaptor, i.e., the ORBIT adaptor, to which this packet should be routed. All the adaptors in the RS/6000 routers are interconnected through the microchannel, which provides the required switching capability between adaptors. After being transferred through the microchannel, the packet reaches its destination adaptor where the appropriate plaNET/ORBIT routing header is attached. This again only involves a simple lookup in a table where plaNET headers are stored, and which is accessed using the IP address. The packet is then ready for transmission in the network.

It should be noted that the transfer of a packet through the microchannel can take place in either one of two ways. A direct card-to-card transfer can be used, or the system can serve as an intermediate relay for the packet. The former alternative is clearly preferable from a throughput and latency point of view, but it requires additional hardware and intelligence on the adaptors, i.e., the dotted line components in the ORBIT adaptor. In the case of adaptors without such capabilities, the system provides the intelligence needed to route packets. Specifically, the device drivers are responsible for providing the necessary routing information and for forwarding the packet to the appropriate adaptor.

A key aspect of the router operation is the population and update of the tables, where the header information needed to route packets through the plaNET network is stored. In the case of IP packets this essentially amounts to providing the ARP function [30]. In the plaNET/ORBIT network, the availability of services such as Directory Services, native multicast, and reverse path accumulation, allows us to easily and dynamically populate these tables with the required routing information.

For example, upon receipt of an IP packet with a destination address for which no path is currently known across the plaNET network, i.e., there is no corresponding local cache entry, the IP DS agent is invoked and sends a query. Upon receipt of the reply to the query, Route Computation is invoked and returns the appropriate path information, which is then stored in the previously mentioned cache table. A simple aging process is used to eliminate entries that correspond to destinations that have become inactive. In addition to this aging procedure, a "liveness" mechanism is used to detect path failures and purge them form the cache table. Note also that bandwidth can be secured between resources using the plaNET/ORBIT bandwidth reservation facilities. IP users can, therefore, be provided with guaranteed bandwidth across the plaNET wide area network.

9.0 CONCLUSION

In this paper, we have presented the various components on which the plaNET/ORBIT network is built, and described how they interact and support the various functions that the network provides. The paper should also provide useful pointers to the many other papers that have been published on the different aspects of the plaNET/ORBIT network, and allow readers to easily identify where to find additional information.

Finally, as mentioned earlier, the ideas upon which plaNET/ORBIT is built are to be tested in a number of forthcoming field trials [28, 29, 38, 39]. We believe such "real-life" tests are key to a better understanding of the efficiency, feasibility and applicability of the proposed approaches, and provide a direct feedback from actual users. The future success of high-speed networks depends not only on technical feats and innovations, but also on how well they will serve the needs of existing and emerging applications. The plaNET/ORBIT network was designed with flexibility and functionality in mind[13], so that applications can select the approach they deem most appropriate to meet their own requirements. Field trials and testbeds will enable us to both improve the technology and evolve the network to better satisfy user needs.

13 The network supports multiple packet formats and routing modes.

ACKNOWLEDGEMENTS

The authors would like to acknowledge the many people who have contributed to different aspects of the plaNET/ORBIT network. This work is the product of the efforts and dedication of many people, and the authors are merely the reporters of this achievement. While it impossible to list all those who contributed, a special mention needs to be made of all the members of the design and development team without whom the plaNET/ORBIT network would be just another "paper design."

REFERENCES

[1] "Framework for Providing Additional Packet Mode Bearer Services," *CCITT Subworking Party XVIII/1-2*, no. CCITT Recommendation I.122, 1988.

[2] "XTP Protocol Definition, 3.4," *Protocol Engines Inc., 1900 State Street, Suite D, Santa Barbara, CA 93101*, 1989.

[3] "Draft - General B-ISDN Aspects," *CCITT Study Group XVIII, Report R 34*, June 1990.

[4] "Special Issue on Teletraffic Analysis of Communications Systems," *IEEE J. Select. Areas Commun.*, vol. 9, no. 2, February 1991.

[5] "Special Issue on Teletraffic Analysis of ATM Systems," *IEEE J. Select. Areas Commun.*, vol. 9, no. 3, April 1991.

[6] "Special Issue on Congestion Control in High-Speed Packet Switched Networks," *IEEE J. Select. Areas Commun.*, vol. 9, no. 7, September 1991.

[7] "Special Issue on Architectures and Protocols for Integrated Broadband Switching," *IEEE J. Select. Areas Commun.*, vol. 9, no. 9, December 1991.

[8] "Special Issue on B-ISDN: High Performance Transport," *IEEE Commun. Mag.*, vol. 29, no. 9, September 1991.

[9] "Special Issue on Congestion Control in High-Speed Networks," *IEEE Commun. Mag.*, vol. 29, no. 10, October 1991.

[10] "Special Issue on Gigabit Networks," *IEEE Commun. Mag.*, vol. 30, no. 4, April 1992.

[11] H. Ahmadi, J. S.-C. Chen and R. Guérin, "Dynamic Routing and Call Control in High-Speed Integrated Networks," *Proc. Workshop Sys. Eng. Traf. Eng., ITC'13*, pp. 397-403, Copenhagen, Denmark, 1991.

[12] H. Ahmadi, R. Guérin and K. Sohraby, "Analysis of a Rate-Based Access Control Mechanism for High-Speed Networks," *IEEE Trans. Commun*, vol. 41, no. 6, pp. 940-950, June 1993. (See also Proc. GLOBECOM'90)

[13] J. Auerbach, M. Gopal, M. Kaplan and S. Kutten, "Multicast Group Membership Management in High Speed Wide Area Networks," *Proc. 11th Intl. Conf. Distrib. Comput. Sys.*, pp. 231-238, Arlington, Texas, 1991.

[14] B. Awerbuch, I. Cidon, I. Gopal, M. Kaplan and S. Kutten, "Distributed Control for PARIS," *Proc. 9th Annual ACM Symp. on Principles of Distributed Computing*, pp. 145-160, 1990.

[15] K. Bala, I. Cidon and K. Sohraby, "Congestion Control for High-Speed Packet Switched Networks," *Proc. INFOCOM'90*, San Francisco, CA, 1990.

[16] A. E. Baratz, J. P. Gray, P. E. Green, Jr., J. M. Jaffe and D. P. Podzefsky, "SNA Networks of Small Systems," *IEEE J. Select. Areas Commun.*, vol. SAC-3, no. 3, pp. 416-426, May 1985.

[17] J. D. Case, M. S. Fedor, M. L. Schoffstall and J. R. Davin, "A Simple Network Management Protocol," *Request for Comments 1157*, DDN Network Information Center, SRI International, May 1990.

[18] M.-S. Chen, Z.-Y. Shae, D. D. Kandlur, T. P. Barzilai and H. M. Vin, "A Multimedia Desktop Collaboration System," *IBM Research Report*, no. RC 17951, May 1992.

[19] I. Cidon and I. S. Gopal, "PARIS: An Approach to Integrated High-Speed Private Networks," *Int'l. J. Digital and Analog Cabled Sys.*, vol. 1, no. 2, pp. 77-86, April-June 1988.

[20] I. Cidon, I. S. Gopal, G. Grover and M. Sidi, "Real-Time Packet Switching: A Performance Analysis," *IEEE J. Sel. Areas Commun.*, vol. SAC-6, no. 9, pp. 1576-1586, December 1988.

[21] I. Cidon, I. S. Gopal and S. Kutten, "New Models and Algorithms for Future Networks," *Proc. 7th Annual ACM Symp. on Principles of Distr. Comp.*, pp. 75-89, Toronto, Ontario, Canada, August 1988.

[22] I. Cidon and I. Gopal, "Control Mechanisms for High-Speed Networks," *Proc. ICC'90*, pp. 269-273, 1990.

[23] I. Cidon, I. Gopal and R. Guérin, "Bandwidth Management and Congestion Control in plaNET," *IEEE Commun. Mag.*, vol. 29, no. 10, pp. 54-63, October 1991.

[24] I. Cidon, I. Gopal and S. Kutten, "Optimal Computation of Global Sensitive Functions in Fast Networks," in J. Van Leeuwen and N. Santoro, editor, *Distributed Algorithms, Proc. 4th Int'l Workshop on Distributed Algorithms, Bari, Italy*, pp. 185-191, Springer Verlag, September 1990.

[25] I. Cidon, I. Gopal and A. Segall, "Connection Establishment in High-Speed Networks," *IEEE/ACM Trans. Networking*, vol. 1, no. 4, pp. 469-481, August 1993. (See also Proc. SIGCOMM'90)

[26] I. Cidon, R. Guérin and A. Khamisy, "Protective Buffer Management Policies," *Proc. INFOCOM'93*, pp. 1051-1058, San Francisco, CA, March 1993.

[27] I. Cidon and Y. Ofek, "Metaring - A Full Duplex Ring with Fairness and Spatial Reuse," *IEEE Trans. Commun.*, vol. 41, no. 1, pp. 110-120, January 1993. (See also Proc. INFOCOM'90)

[28] D. Clark, B. Davie, D. Farber, I. Gopal, B. Kadaba, D. Sincoskie, J. Smith and D. Tennenhouse, "An Overview of the AURORA Gigabit Testbed," *Proc. INFOCOM'92*, pp. 569-581, Florence, Italy, 1992.

[29] D. Clark, B. Davie, D. Farber, I. Gopal, B. Kadaba, D. Sincoskie, J. Smith and D. Tennenhouse, "The AURORA Gigabit Testbed," *Computer Networks and ISDN Systems*, vol. 25, no. 1, pp. 599-621, January 1993.

[30] D. E. Comer, *Internetworking with TCP/IP, 2nd Edition. Vol. I, Principles, Protocols, and Architecture*, Englewood Cliffs, NJ, Prentice-Hall, 1991.

[31] A. S. Gopal, I. S. Gopal and S. Kutten, "Broadcast in Fast Networks," *Proc. INFOCOM'90*, San Francisco, CA, June 1990.

[32] I. Gopal and R. Guérin, "Network Transparency: The plaNET Approach," *Proc. INFOCOM'92*, pp. 590-601, Florence, Italy, May 1992.

[33] I. Gopal, R. Guérin, J. Janniello and V. Theoharakis, "ATM Support in a Transparent Network," *IEEE Networks Mag.*, vol. 6, no. 6, pp. 62-68, November 1992. (See also Proc. GLOBECOM'92)

[34] R. Guérin, H. Ahmadi and M. Naghshineh, "Equivalent Capacity and Its Application to Bandwidth Allocation in High-Speed Networks," *IEEE J. Select. Areas Commun.*, vol. SAC-9, no. 7, pp. 968-981, September 1991.

[35] R. Guérin and L. Gün, "A Unified Approach to Bandwidth Allocation and Access Control in Fast Packet-Switched Networks," *Proc. INFOCOM'92*, pp. 1-12, Florence, Italy, May 1992.

[36] L. Gün and R. Guérin, "Bandwidth Management and Congestion Control Framework of the Broadband Network Architecture," *Computer Networks and ISDN Systems*, vol. 26, no. 1, pp. 61-78, September 1993.

[37] M. J. Karol, M. C. Hluchyj and S. P. Morgan, "Input vs. Output Queueing on a Space-Division Packet Switch," *IEEE Trans. Commun.*, vol. COM-35, no. 12, pp. 1347-1356, December 1987.

[38] Press Release. Wall Street Journal, January 29, 1991.

[39] Press Release. Wall Street Journal, October 7th, 1991.

[40] A. Segall, T. P. Barzilai and Y. Ofek, "Reliable Multiuser Tree Setup with Local Identifiers," *IEEE J. Select. Areas Commun.*, vol. 9, no. 9, pp. 1427-1439, December 1991.

[41] V. Theoharakis and R. Guérin, "SONET OC-12 Interface for Variable Length Packets," *Proc. IC3N'93*, San Diego, CA, June 1993.

[42] H. T. Wu, Y. Ofek and K. Sohraby, "Integration of Synchronous and Asynchronous Traffic on the Metaring Architecture and its Analysis," *Proc. ICC'92*, 1992.

FIGURES

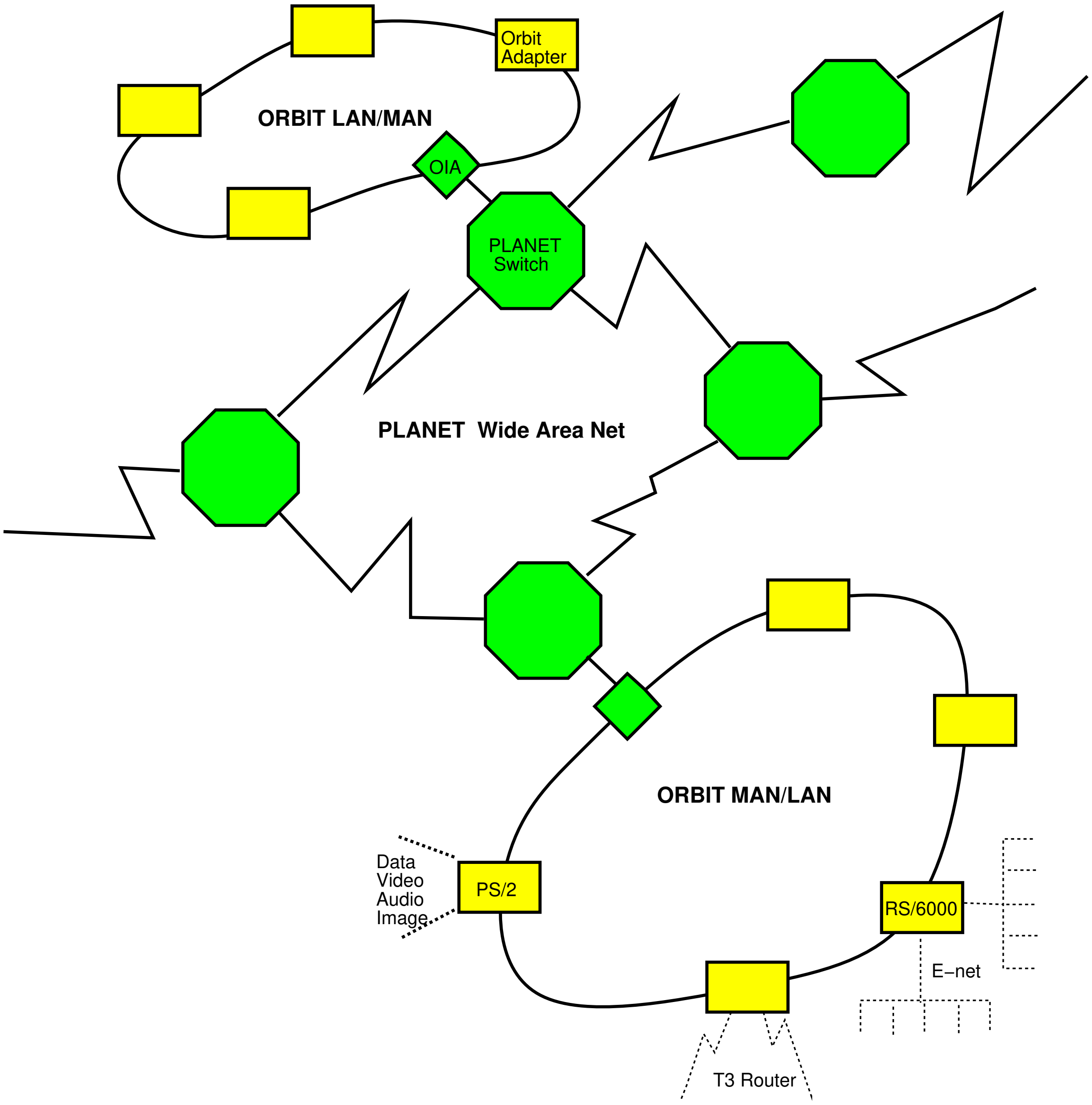

Figure 1. planET/ORBIT Network Overview

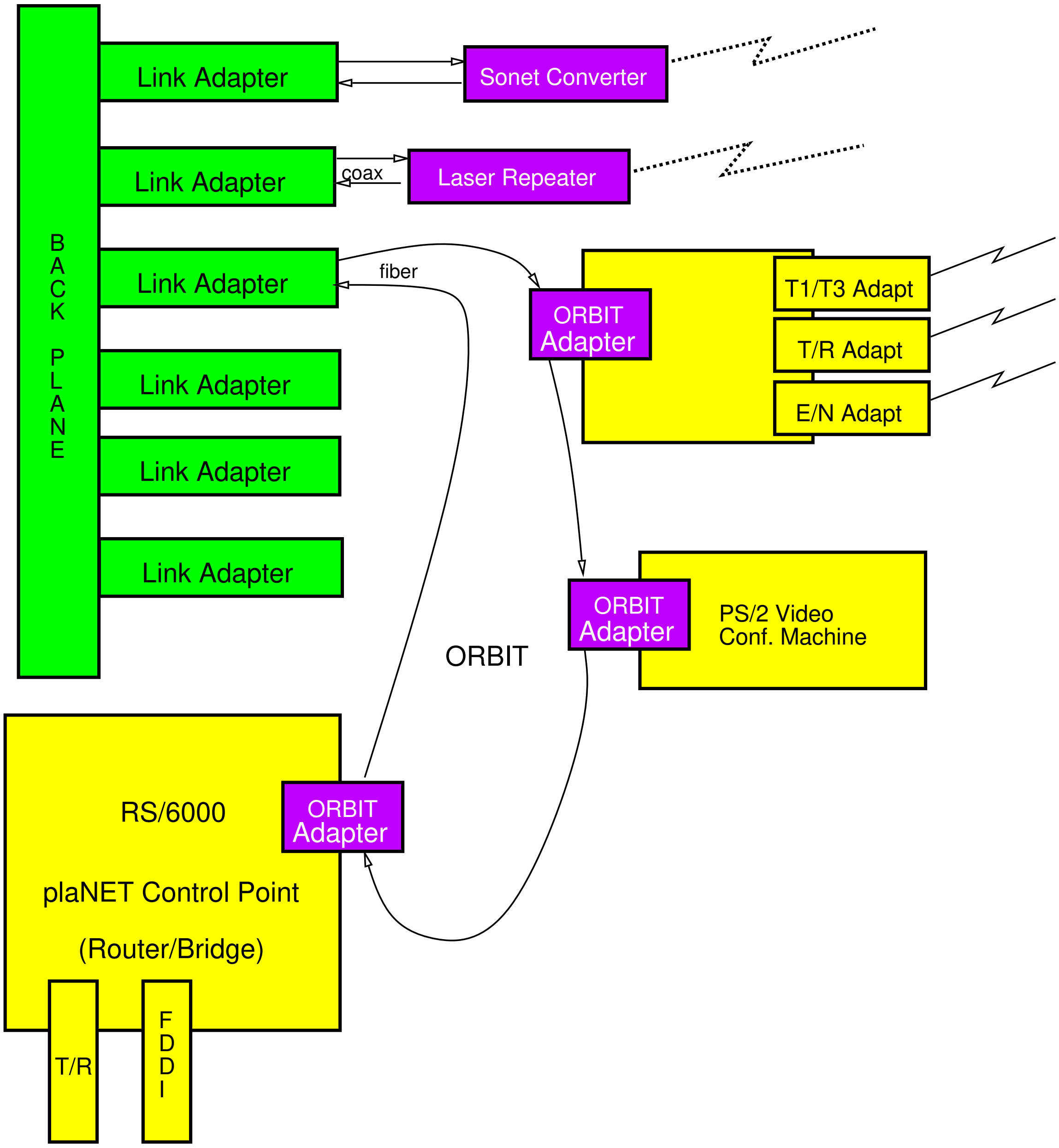

Figure 2. plaNET Node Structure

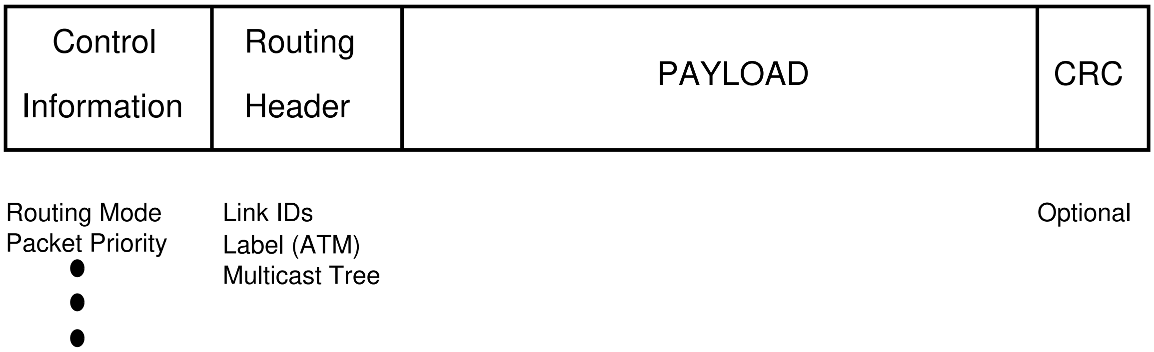

Figure 3. plaNET Packet Format

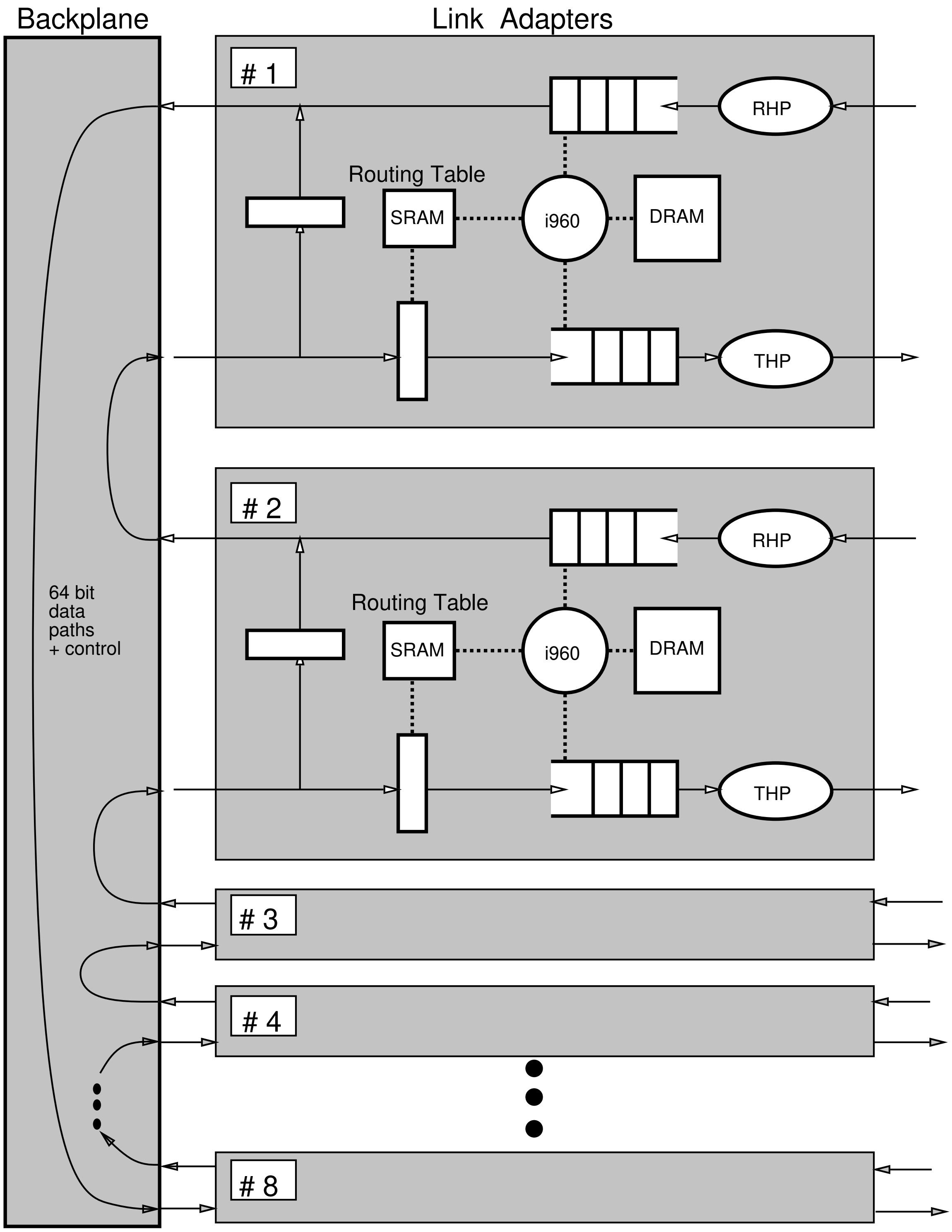

Figure 4. plaNET Switching System

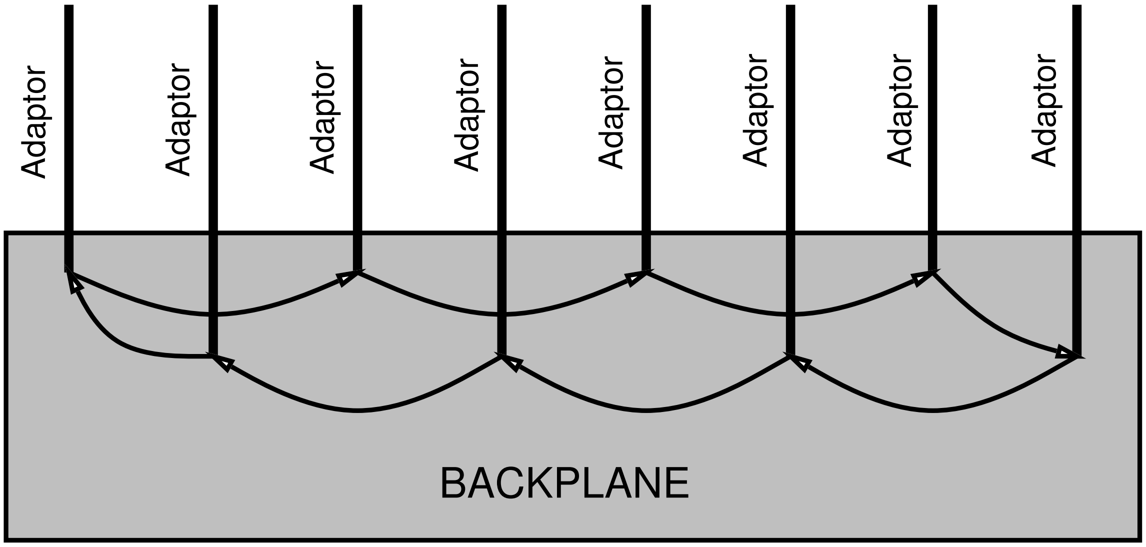

Figure 5. plaNET Switch Backplane Layout

Routing Mode	Routing Decision	Modify Routing Field
ANR	ANR label Match y/n?	1st ANR label $\longrightarrow \emptyset$
Tree	Tree ID match y/n?	Tree ID1 $\longrightarrow$ Tree ID?
Label Swap	Label match y/n?	Label1 $\longrightarrow$ Label2

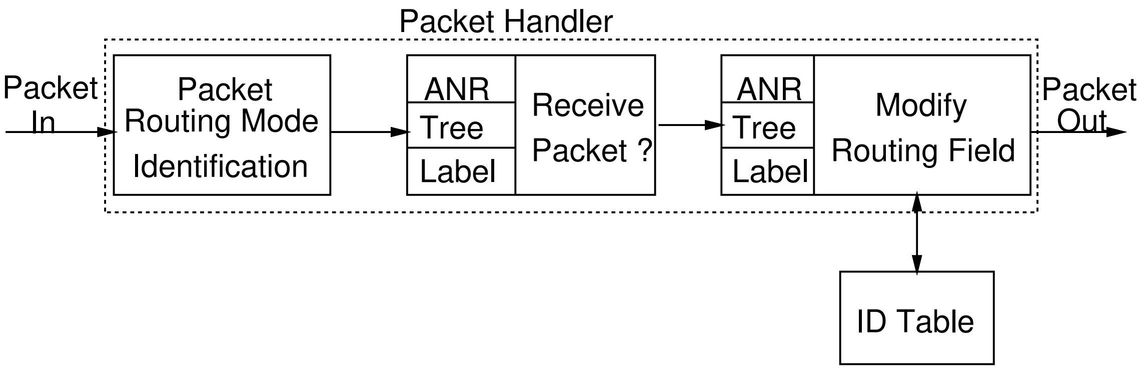

Figure 6. plaNET Packet Handling Functions per Routing Mode

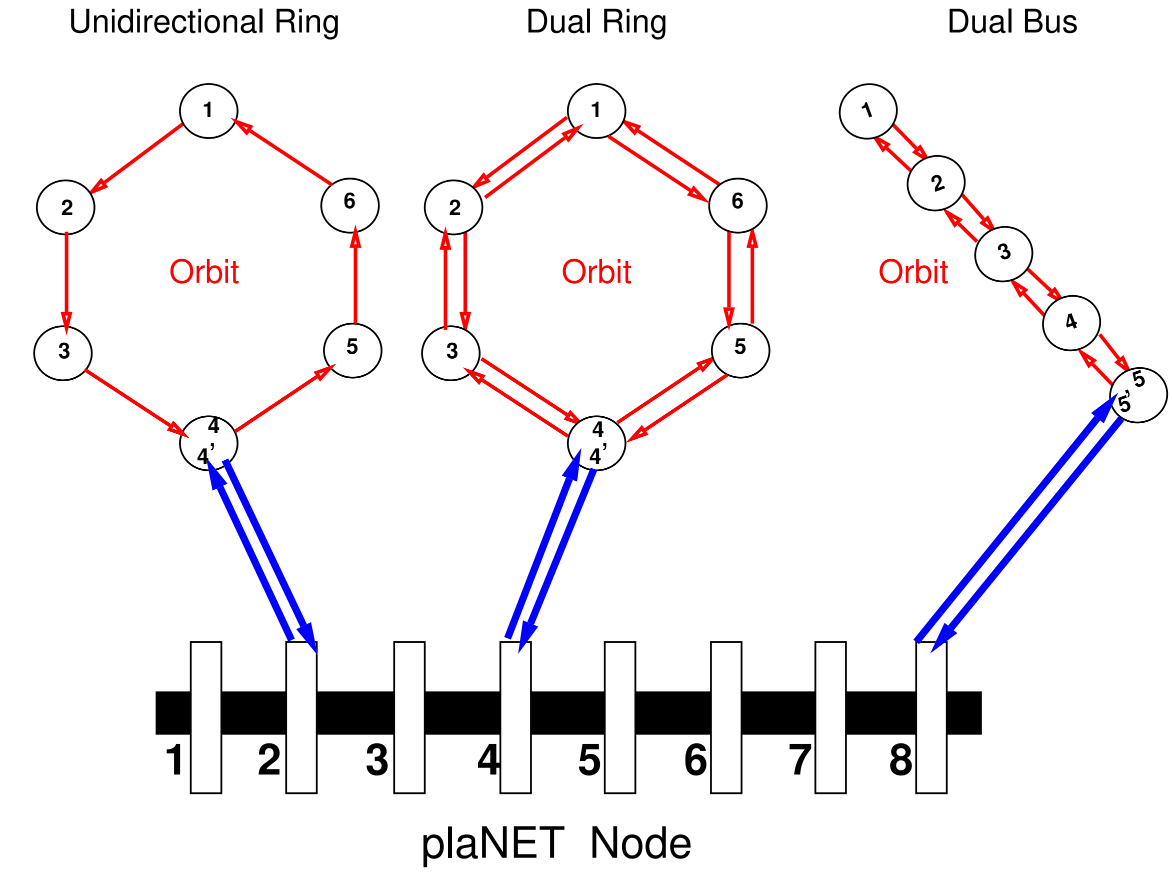

Figure 7. ORBIT Configurations and Attachment to plaNET

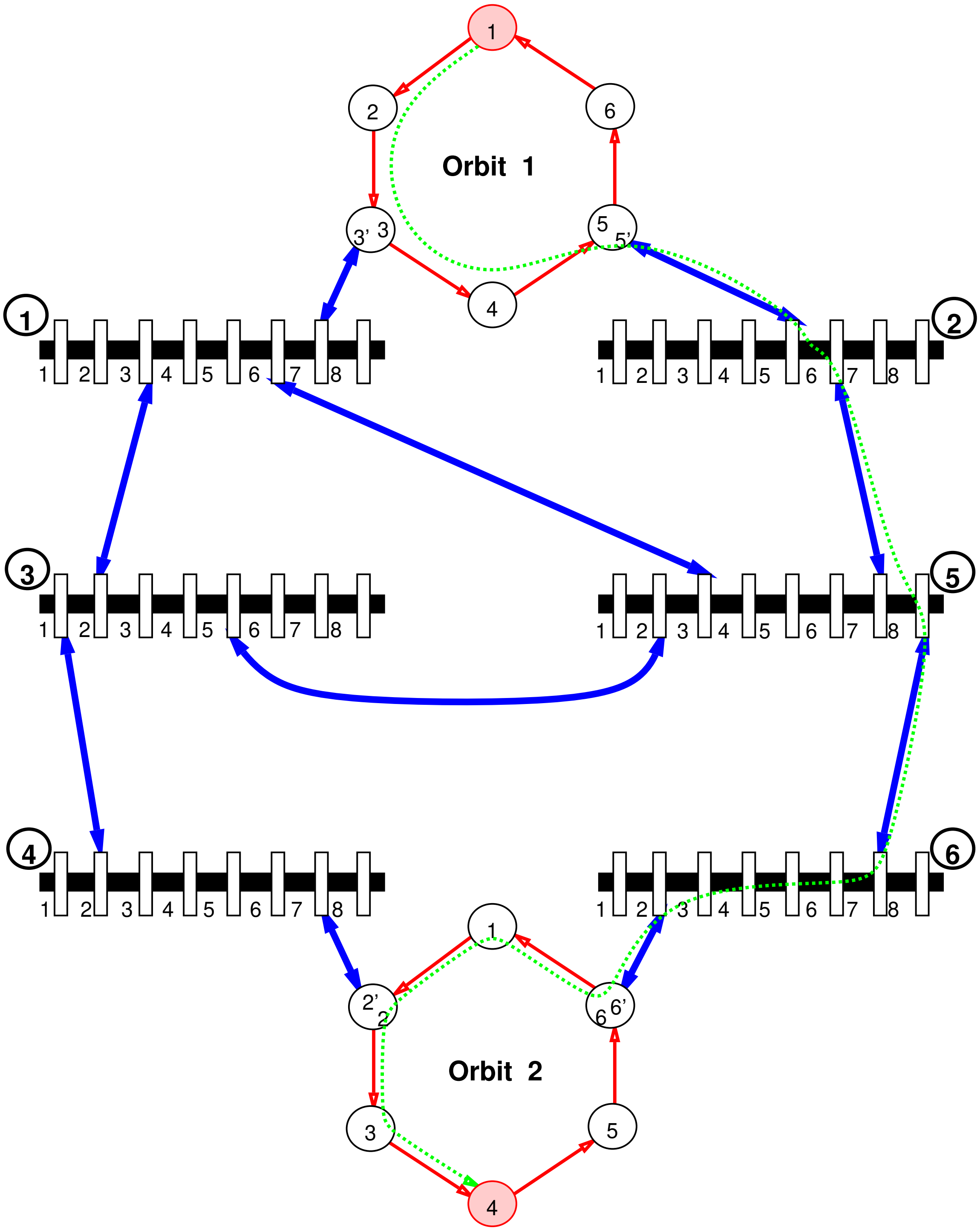

Figure 8. Connectivity across ORBIT and plaNET Networks

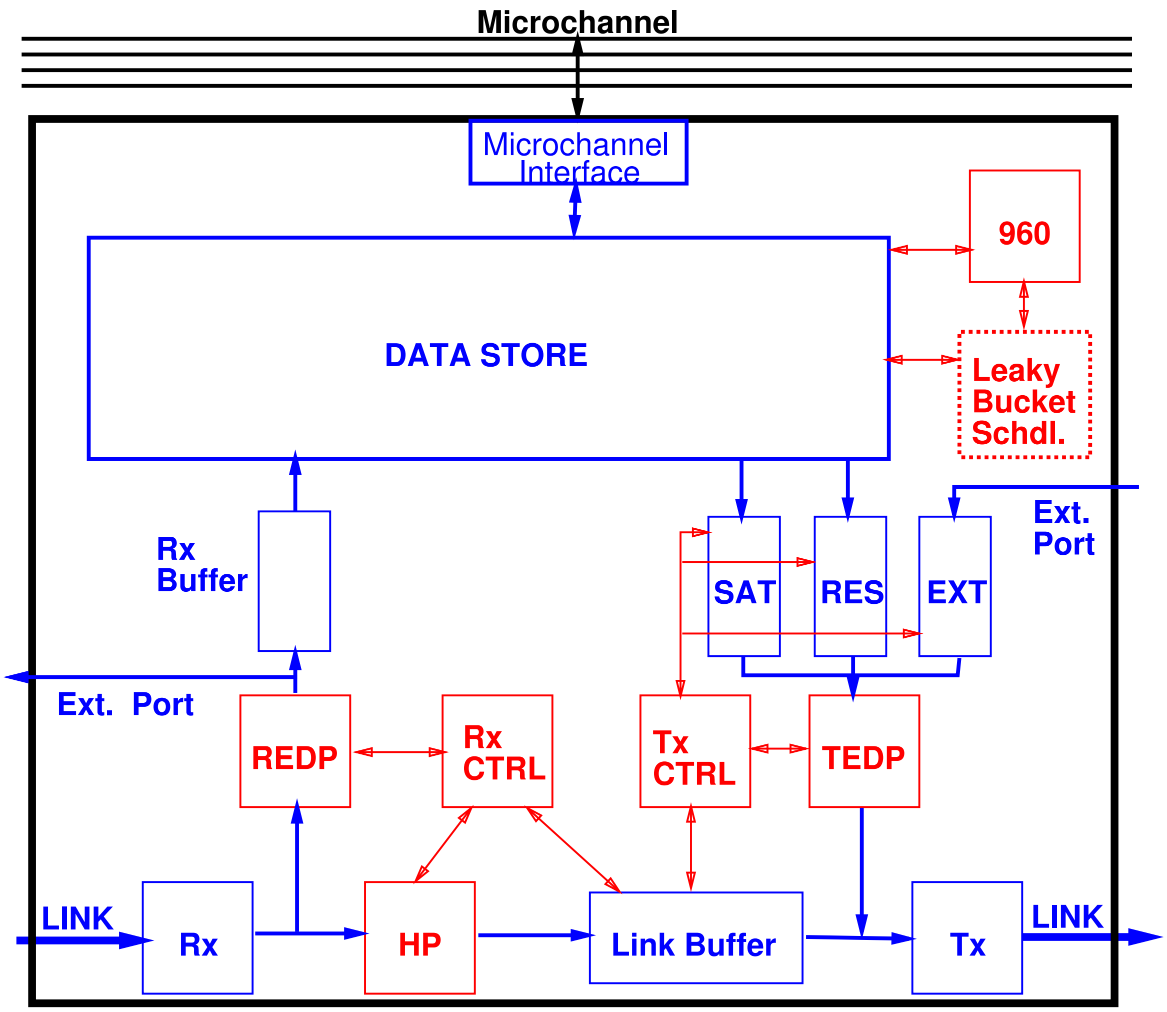

Figure 9. ORBIT Adapter

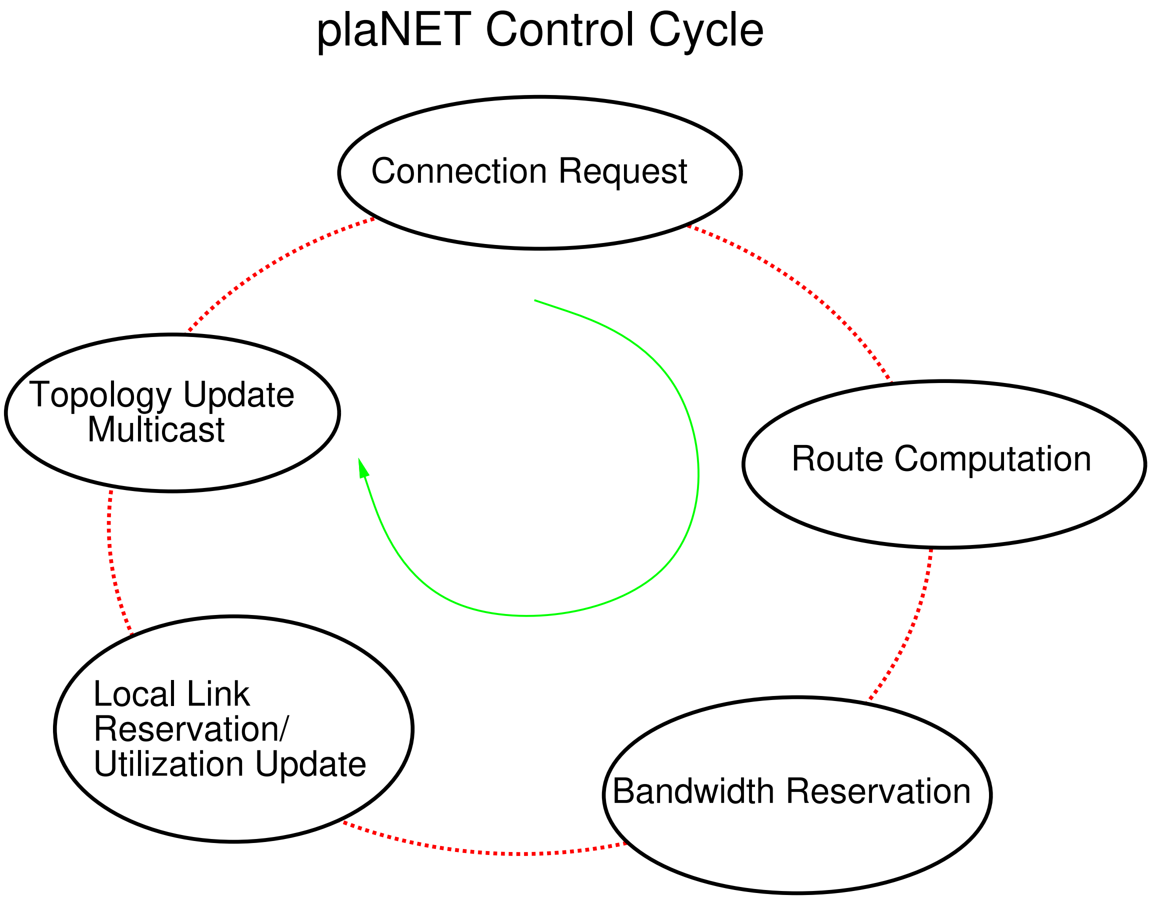

Figure 10. Overview of Network Control Cycle

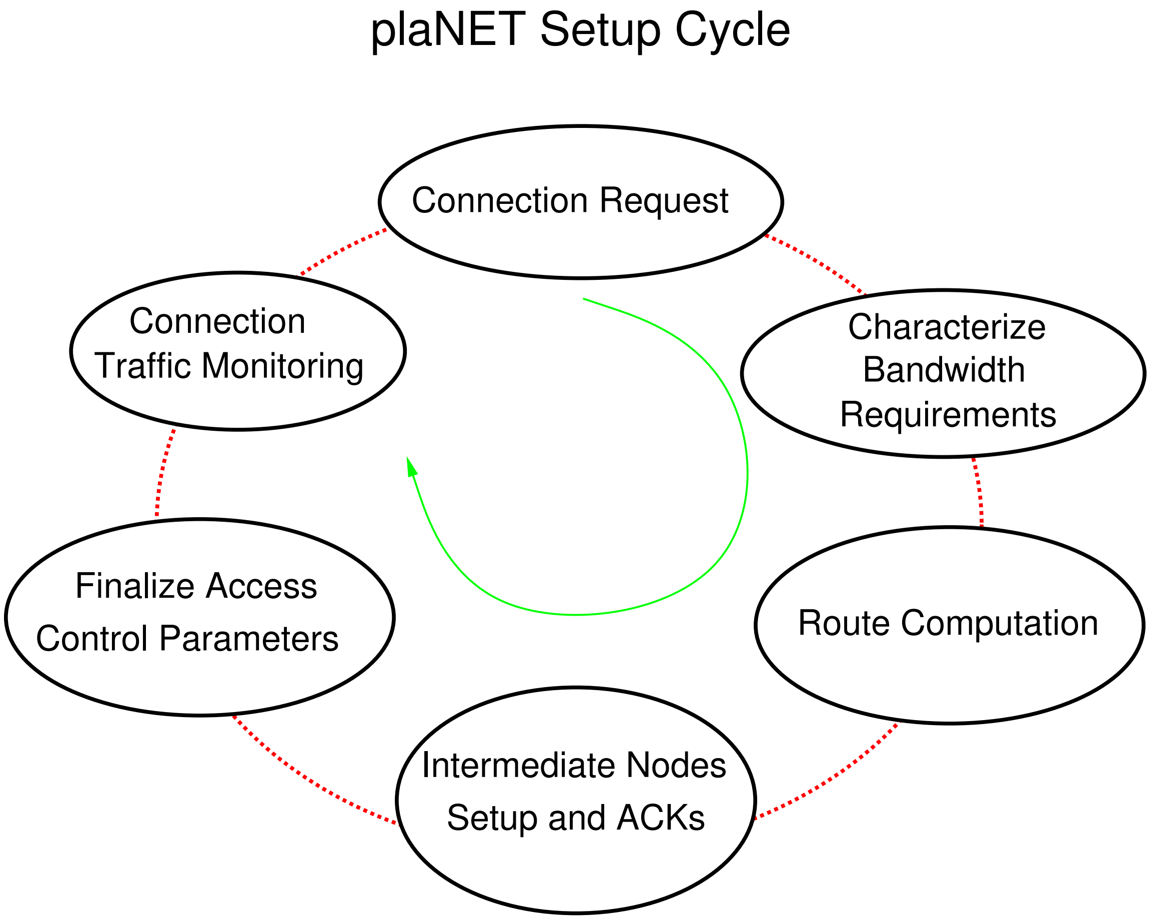

Figure 11. Overview of Connection Control Cycle

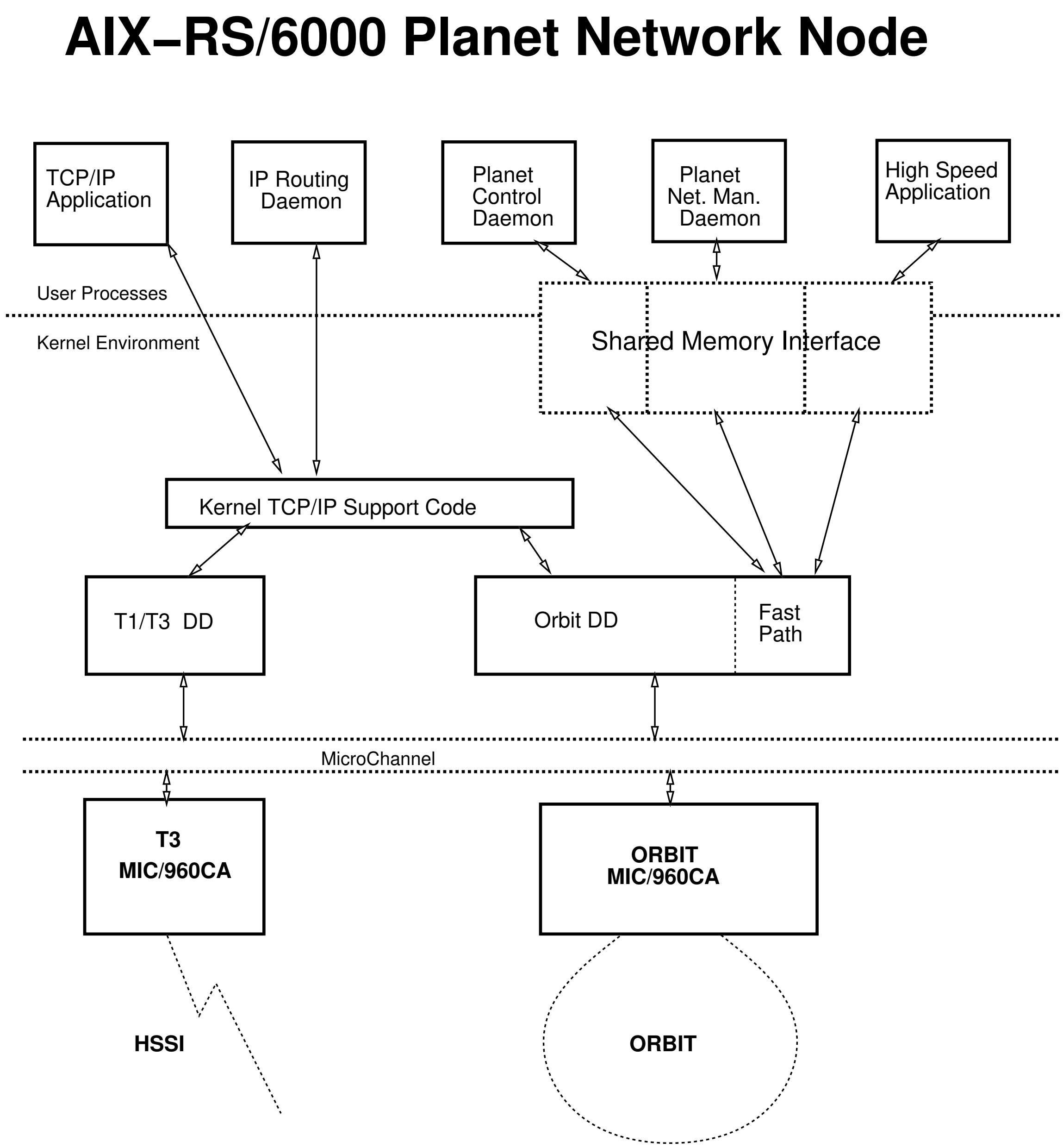

Figure 12. Structure of plaNET End-Point

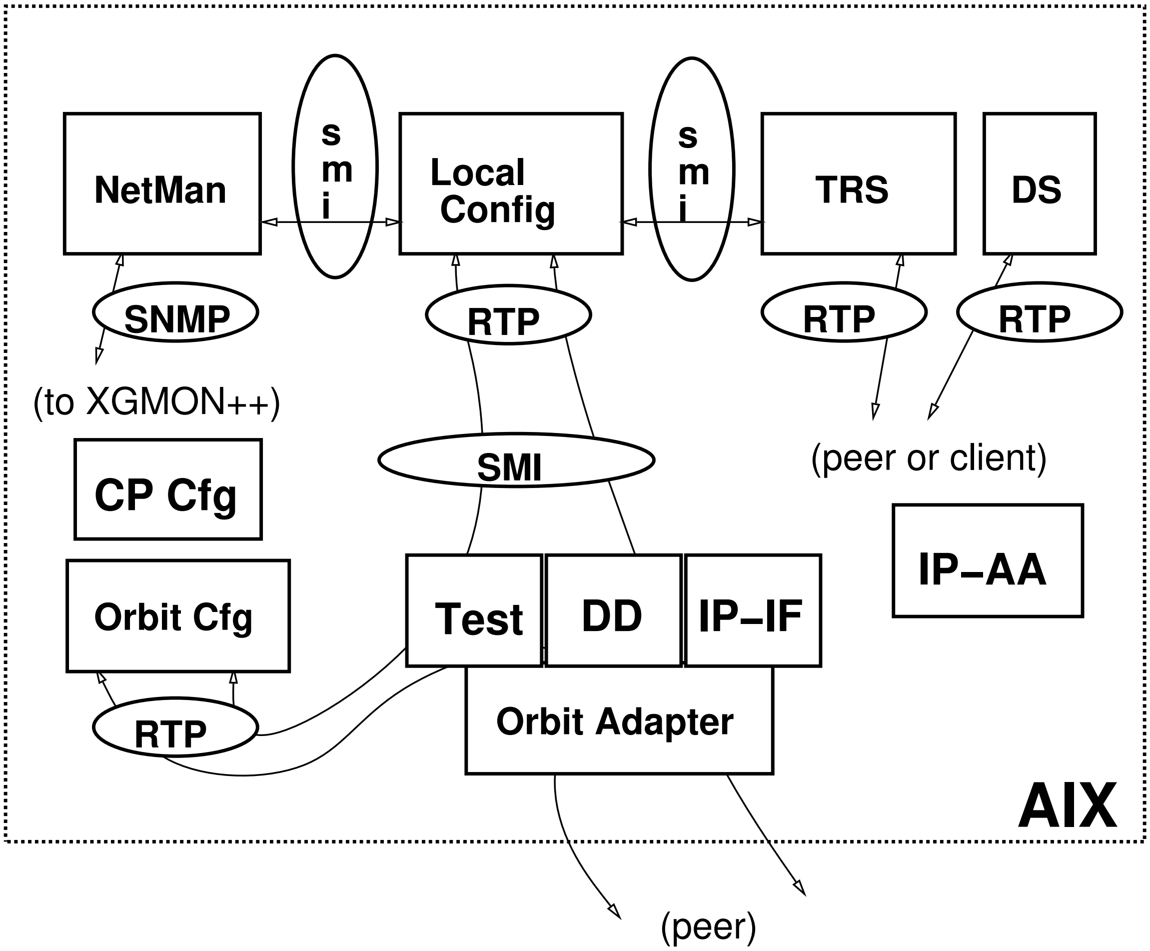

Figure 13. End-Point Software Components

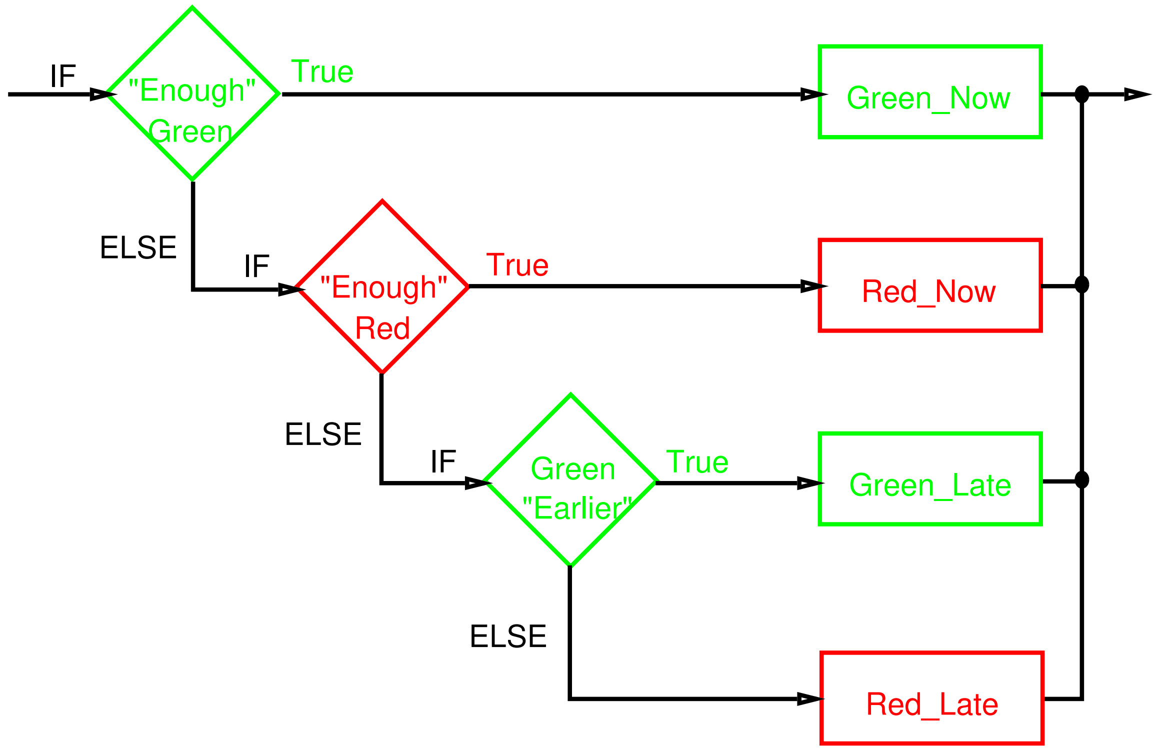

Figure 14. Flow Chart of Leaky Bucket Software Implementation

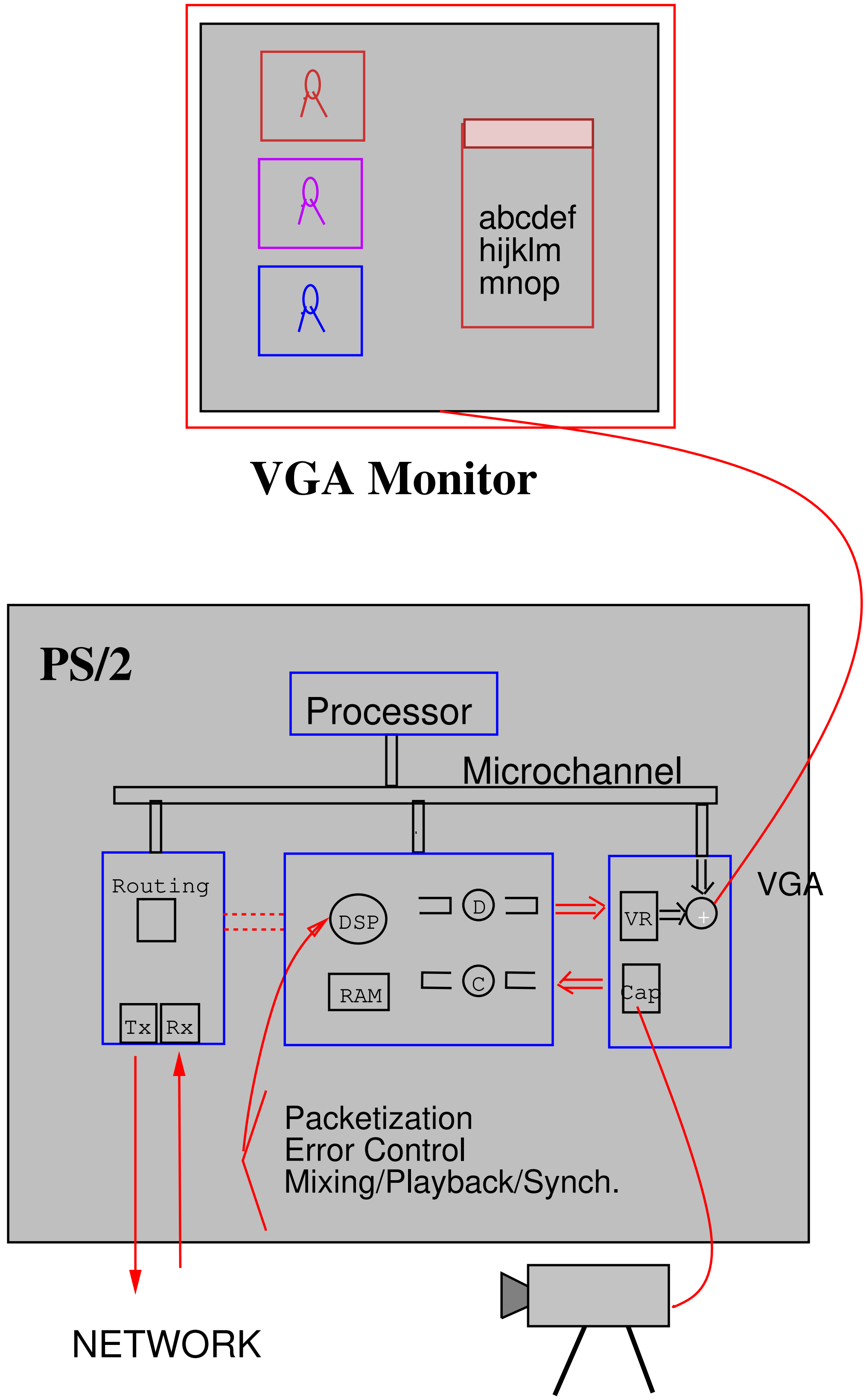

Figure 15. Overview of PS/2-Based Multimedia Workstation

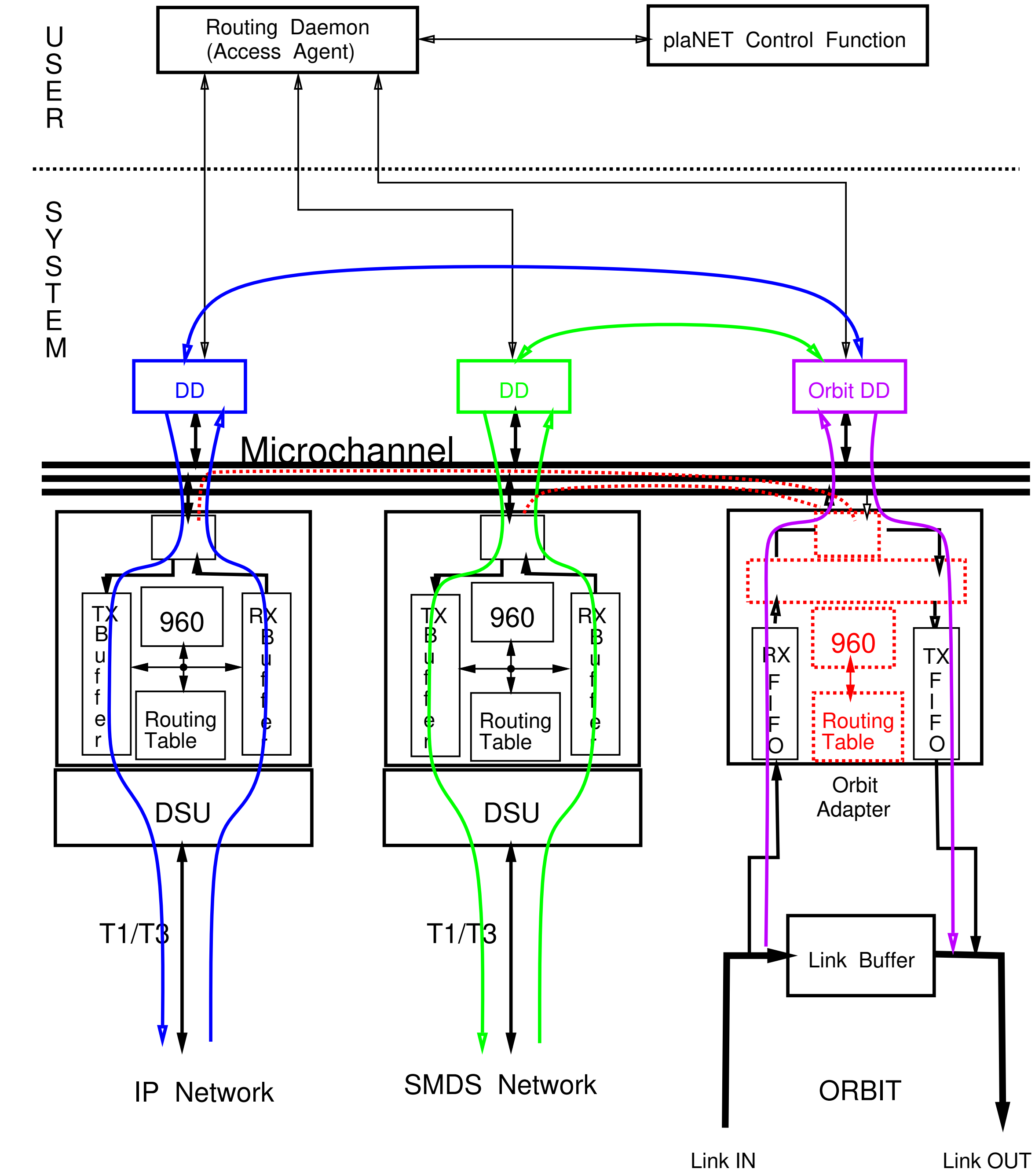

Figure 16. plaNET/ORBIT Access Agent