Report Number: WUCSE-2006-14

2006-01-01

A Proposed Architecture for the GENI Backbone Platform

Jonathan Turner

The GENI Project (Global Environment for Network Innovation) is a major NSF-sponsored initiative that seeks to create a national research facility to enable experimental deployment of innovative new network architectures on a sufficient scale to enable realistic evaluation. One key component of the GENI system will be the GENI Backbone Platform (GBP) that provides the resources needed to allow multiple experimental networks to co-exist within the shared GENI infrastructure. This report reviews the objectives for the GBP, reviews the key issues that affect its design and develops a detailed reference architecture in order to provide a concrete example for... Read complete abstract on page 2.

A Proposed Architecture for the GENI Backbone Platform

Jonathan Turner

Complete Abstract:

The GENI Project (Global Environment for Network Innovation) is a major NSF-sponsored initiative that seeks to create a national research facility to enable experimental deployment of innovative new network architectures on a sufficient scale to enable realistic evaluation. One key component of the GENI system will be the GENI Backbone Platform (GBP) that provides the resources needed to allow multiple experimental networks to co-exist within the shared GENI infrastructure. This report reviews the objectives for the GBP, reviews the key issues that affect its design and develops a detailed reference architecture in order to provide a concrete example for how the objectives can be met.

2006-14

A Proposed Architecture for the GENI Backbone Platform

Authors: Jonathan Turner

Corresponding Author: jon.turner@wustl.edu

Abstract: The GENI Project (Global Environment for Network Innovation) is a major NSF-sponsored initiative that seeks to create a national research facility to enable experimental deployment of innovative new network architectures on a sufficient scale to enable realistic evaluation. One key component of the GENI system will be the GENI Backbone Platform (GBP) that provides the resources needed to allow multiple experimental networks to co-exist within the shared GENI infrastructure. This report reviews the objectives for the GBP, reviews the key issues that affect its design and develops a detailed reference architecture in order to provide a concrete example for how the objectives can be met.

Type of Report: Other

A Proposed Architecture for the GENI Backbone Platform[*]

Jonathan Turner
Washington University
jon.turner@wustl.edu

Abstract

The GENI Project (Global Environment for Network Innovation) is a major NSF-sponsored initiative that seeks to create a national research facility to enable experimental deployment of innovative new network architectures on a sufficient scale to enable realistic evaluation. One key component of the GENI system will be the *GENI Backbone Platform* (GBP) that provides the resources needed to allow multiple experimental networks to co-exist within the shared GENI infrastructure. This report reviews the objectives for the GBP, reviews the key issues that affect its design and develops a detailed reference architecture in order to provide a concrete example for how the objectives can be met.

1. Introduction

The GENI initiative [GE06] seeks to create an experimental facility that will enable networking researchers to develop and deploy novel network architectures that address important shortcomings of the current Internet architecture (including security, mobility, quality of service and multicast among others). The proposed facility will use virtualization to enable different groups to share the available resources, including physical link bandwidth and processing resources in the network nodes.

A key objective of GENI is to enable new network architectures to be deployed and evaluated *at scale*. This means, among other things, that at least some of the experimental networks must have the potential to provide service to large numbers of end users, where larger is taken to mean at least 100,000. This is considered important for two reasons. First, because many of the network characteristics that one would like to be able to evaluate, only become apparent as networks get large. Second, if new network architectures are to have an impact on commercial practice, network providers and equipment vendors must have a good reason to take them seriously. The most compelling evidence for the value of a new network architecture is the fact that large numbers of people use it.

[*] This work supported by the National Science Foundation (CNS 0435173).

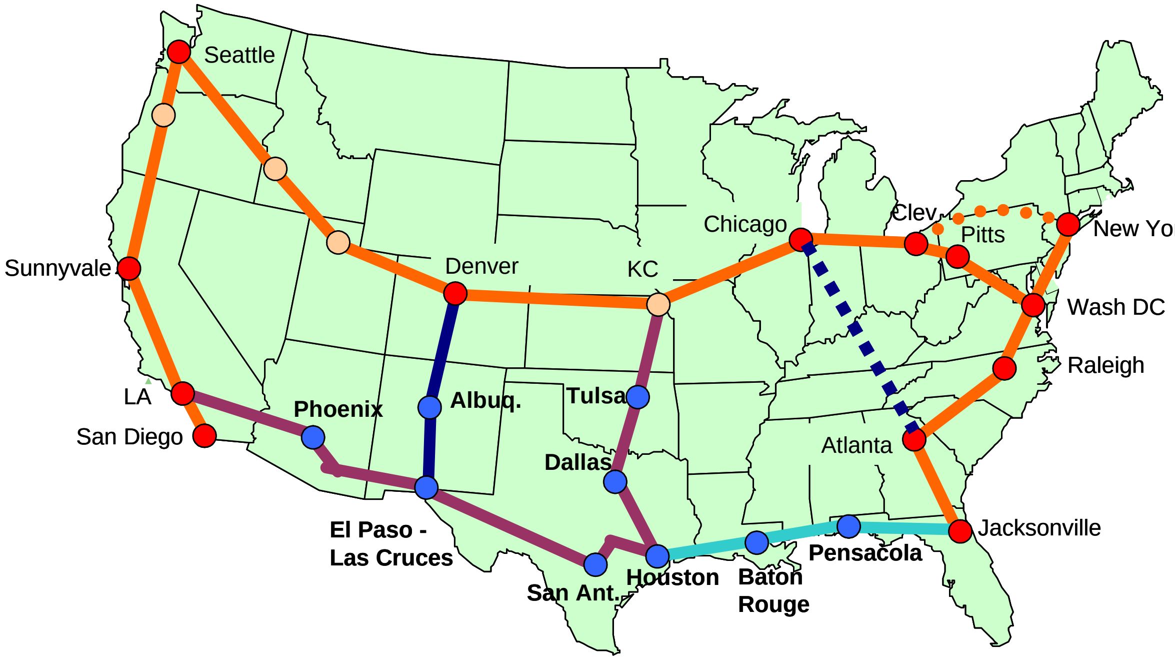


Figure 1. National Lambda Rail Network Topology

The *GENI Backbone Platform* (GBP) is one of the key components of the planned GENI facility. The GBP is envisioned as a flexible, high performance networking platform that provides the resources needed to allow multiple experimental networks to move large volumes of traffic across the country. It is expected that GBPs will be located at a few tens of sites around the country, with each site terminating a small number of fibers (typically 2-4), with possibly multiple wavelengths on each fiber. It is likely that the GENI backbone will use fiber facilities made available through the National Lambda Rail (NLR) [NLR]. The NLR network topology, shown in Figure 1, has 25 sites and has two major east-west routes. A GENI network with a single 10 Gb/s wavelength on each NLR fiber segment would be able to support up to 10,000 concurrent cross-country flows with an average bandwidth of 1 Mb/s, while maintaining an average link occupancy of 50%. Substantial increases in either the number of concurrent cross-country flows, or the average bandwidth per flow would require multiple wavelengths on the major cross-country routes. These observations provide a rough indication of the range of IO capacities that the GBP must provide.

In addition to IO, the GBP must provide flexible processing resources that can be allocated for use by the different experimental networks that are implemented in GENI. To provide maximum experimental flexibility, the GBP should provide sufficient processing resources to enable networks in which the ratio of processing to IO is substantially higher than in conventional routers. Since the GBP is intended as a general experimental platform, it is also important that it provide access to a variety of different types of resources, so that researchers can select the resources that best match the needs of their specific network architecture.

This report describes a reference design for the GBP. In Section 2, we identify the overall objectives for the design. In Section 3 we describe the key abstractions that are implemented by the the proposed reference design. Section 4 reviews a range of technology components that can be expected to play an important role in any GBP design. In Section 5, we identify and compare two major system architecture options for the GBP and argue that the *processing pool architecture* is the best choice for an experimental facility like GENI. Section 6 provides a detailed description of the proposed reference design, including descriptions of all the major subsystems. Finally, in Section 7, we describe a range of different system configurations and provide estimates of the cost of each configuration.

2. Objectives

The purpose of the GBP is to enable multiple, diverse *metanetworks* (aka slices) to co-exist within a common shared infrastructure. To do this, it must enable sharing of backbone links and node processing resources. We expect researchers to use GENI for a wide range of different experiments, with highly diverse requirements. To enable the GBP to serve the widest possible range of objectives, it should be highly flexible and should provide sufficient resources to avoid constraining the research agendas of its users.

A GBP will host multiple *metarouters* belonging to distinct *metanetworks* (we use the term metarouter and metanetwork rather than virtual router and virtual network, because the latter terms have been heavily overloaded and are more subject to misunderstanding). The GBP will provide resources that can be used by the different metarouters and the underlying mechanisms to allow each one to operate independently of the others, without interference. The term metarouter is used here in a very generic sense to mean any network component with multiple interfaces that forwards information through a network, while possibly processing it as it passes through. It can include components whose functionality is similar to that of an IP router, or components that switch TDM circuits, or components that operate like firewalls or media gateways. A given metanetwork may include metarouters of various types. It is left to the designers of individual metanetworks to define the precise functionality of their metarouters and to distinguish among different types of metarouters as they find appropriate.

The design of the GBP is distinctly different from the design of conventional routers and switches in that it must allow bandwidth to be allocated flexibly among multiple metarouters and must provide generic processing resources that can be allocated flexibly to different metarouters. It must allow them to use those resources without constraining them unnecessarily, while ensuring that different metarouters do not interfere with one another. The following paragraphs summarize some important high level objectives for the GBP.

- *Scalable performance*. The experimental networks developed for GENI will have a wide range of characteristics, leading to widely differing requirements for GBP resources. Metanets seeking to support high volume data transfers for e-science applications may require links with bandwidths above 10 Gb/s, while metanets designed to transfer text messages among pagers may have little use for links above 100 Mb/s. Different metarouters will also have very different processing needs. While IPv4 forwarding requires fewer than 20 instructions executed per data byte, some experimental networks may require hundreds of instructions executed per data byte. The GBP should enable its resources to be allocated flexibly among hundreds of different metarouters, and should support configurations suitable for a variety of performance ranges.
- *Stability and reliability*. If GENI is to provide a useful platform for experimentation and deployment of experimental network services, it must be sufficiently reliable and stable to allow researchers to work without interference from others. Because the experimental networks that run within GENI will be the subjects of on-going experimentation and modification, their stability will be highly variable. Nonetheless, the platform itself must be stable, so that researchers can focus on issues arising within their own experiments and not

be concerned with the stability of the underlying substrate. The isolation mechanisms for metarouters (discussed below) are one element of the overall strategy for achieving reliable operation. However, it is also important that the hardware components used to implement the GBP have high intrinsic reliability and that the GBP as a whole be easy to manage and maintain, so that outages due to operational errors are kept to a minimum.

- *Ease of use*. Academic researchers have limited resources that they can devote to development of experimental systems, making it important that it be as easy as possible for them to implement their metanetworks. There are some intrinsic challenges here, in that the technologies that yield the highest possible performance are often not the easiest to use. The GBP should enable use of high performance technologies, while minimizing the barriers that make them difficult to use. In particular, it should be possible for researchers to port applications developed in the PlanetLab environment to GENI without a great deal of effort. In addition, the GBP should facilitate sharing of common software and configurable logic modules among different research groups.
- *Technology diversity and adaptability*. The GBP should enable the construction of metanetworks using a variety of different underlying technologies, including general purpose processors, network processor subsystems and configurable logic subsystems. This will allow different researchers to pursue different strategies for meeting their research objectives and will provide the flexibility for the system to incorporate new implementation technologies, as they become available.
- *Flexible allocation of link bandwidth*. Link bandwidth is a key resource. The GBP should support flexible allocation of bandwidth to different metanetworks including both reserved and shared bandwidth models. It should also provide mechanisms for circuit-based management of link resources, allowing researchers to experiment with novel frame formats and time-domain switching techniques.
- *Isolation among metarouters*. The GBP must allow different metarouters to co-exist without interference. Ideally, each metarouter should have the illusion that it is operating within a dedicated environment, rather than a shared environment. This means that resources like memory and disk space must be free from modification by other metarouters and that metarouters have the ability to reserve dedicated processing capacity and link bandwidth.
- *Minimize constraints on metarouters*. The GBP should place as few constraints as possible on the metarouters it hosts. In particular, it should not place any constraints on data formats or limit the ways in which metanetworks provide various capabilities, or constrain the way in which a metarouter uses its underlying processing resources.

It should be understood that the above list is not comprehensive. It has been intentionally limited to fairly high level objectives. The remainder of this report will provide a much more detailed view of the GBP, but what we describe is one point in the design space of possibilities. There may well be other approaches to meeting the above objectives that are equally valid.

3. GBP Abstractions

A GBP will host multiple *metarouters* that terminate *metalinks* connecting to other metarouters and to end systems. The metarouter and metalink are key abstractions that are implemented by the GBP. These abstractions are described in the following subsections.

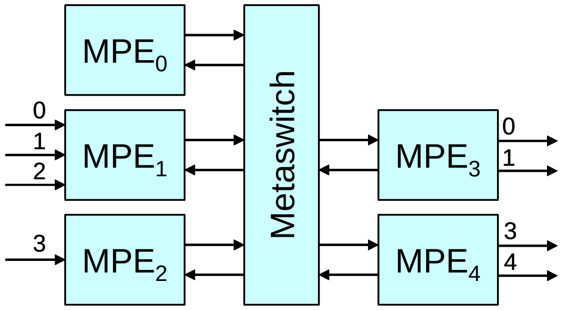


Figure 2. Example metarouter

3.1. Metalink Abstraction

The metalink abstraction implemented by the GBP is an abstraction of a point-to-point physical link. A metalink may have a guaranteed transmission bandwidth and a maximum transmission bandwidth, but is not required to have either. While the backbone links connecting GBPs will typically have provisioned bandwidths, we expect at least some links connecting GBPs to user sites will be implemented using unprovisioned tunnels. In addition, not all metanetworks require provisioned bandwidth. The configuration of a metanetwork will include a specification of its metalinks. For those metalinks that are specified with provisioned bandwidth, the metanetwork configuration software will need to choose an underlying implementation that can support this. A metalink may be unidirectional or bi-directional. A bi-directional metalink may have asymmetric bandwidth provisioning.

The metalink abstraction can be extended to support links with more than two endpoints. The primary motivation for supporting a multipoint metalink is to make use of the features of multiaccess subnets in the LAN environment. Since this report is concerned with a platform for backbone applications, we do not consider the multipoint case here.

3.2. Metarouter Abstraction

A metarouter is an abstraction of a conventional router, switch or other network component, which typically consists of three major components, *line cards*, a *switching fabric* and a *control processor*. The line cards terminate the physical links and implement specific processing functions that define a particular network. On input, this may include performing a table lookup of some sort, to determine where an arriving packet should be sent next and what special processing (if any) it should receive. Alternatively, it might involve identification of TDM frame boundaries, and time-division switching of timeslots within frames. On output, it might include scheduling the packet for transmission on the outgoing link. The switching fabric is responsible for transferring data from the line cards where they arrive, to the line cards for their outgoing links. Switching fabrics are typically designed to be *nonblocking*, meaning that they should handle arbitrary traffic patterns, without interference within the fabric. For circuit networks this means that any set of circuits that can be supported by the external links should be supported by the fabric. For packet networks, it also means that excessive traffic to a particular output line card should not interfere with traffic going to other line cards. The control processor in a conventional router implements various control and administrative functions (such as executing routing protocols and updating tables in the line cards). These functions are generally implemented in software running on a general-purpose microprocessor.

A metarouter has a similar structure. It consists of two types of components *metaprocessing Engines* (MPE) and a *metawitch* (MS). MPEs can be used to implement data path functions within a metarouter or higher level control functions and may be implemented using various types of underlying processing resources. Metalinks terminate at *meta-nterfaces* (MI) on MPEs and MPEs are connected to each other through the MS. Figure 2 shows an example of a metarouter. MIs are subject to maximum bandwidth constraints, as are the interfaces between MPEs and the MS.

Metarouters with limited performance needs may use a single MPE. In this case, there is no need for a metaswitch. Another common case is a metarouter with two MPEs, one that implements the normal data forwarding path, and another that implements control functions and handles exception cases. In this case, the MIs will typically all be associated with the data path MPE (implemented on a network processor, perhaps) which will have a logical connection directly to the control/exception MPE (implemented on a general purpose processor). In this case, the MS reduces to the single logical connection between the two MPEs.

4. Technology Components

This section reviews some key technology components that are expected play an important role in the GBP. The purpose of this review is to help quantify the capabilities of existing components and project how these capabilities are likely to improve over the next three to five years.

4.1. Network Processors

Network processors are special-purpose multiprocessor chips designed specifically to handle network traffic. Large router vendors, such as Cisco, are increasingly using proprietary network processors in their high performance systems to reduce development effort, while maximizing system flexibility. At the same time, component companies such as Intel, are producing chips that can be and have been incorporated into systems produced by a variety of system vendors.

The Intel IXP 2800 series is representative of the capabilities now available in network processor chips [IXP]. The 2800 (which has been available for over two years now), contains 16 special-purpose processors, called *Micro-Engines* (ME), for processing data. Each ME is a 1.4 GHz 32 bit RISC processor, with 256 general purpose registers, a program store that holds 4K instructions and 2.5K bytes of local memory, plus a number of features for enabling efficient communication with other MEs. The chip is managed by an Xscale control processor, which typically runs a general purpose operating system (e.g. embedded Linux) and which controls the MEs (starting and stopping individual MEs, loading programs, configuring control registers, etc). The chip also includes 3 banks of high performance RDRAM with an aggregate memory bandwidth of over 75 Gb/s, four QDR SRAM channels with an aggregate memory bandwidth of more than 25 Gb/s and a high performance IO interface, compliant with the SPI-4 interface standard, with a bandwidth of more than 12 Gb/s in each direction. The 2800 is designed for use in 10 Gb/s applications, with each port of a high performance router or switch using two chips, one for ingress traffic (from link to switch) and one for egress traffic.

Network processors are examples of a broad trend in computer architecture. As silicon resources have grown, systems with multiple processor cores have gained a growing performance advantage relative to single processor systems with more complex processors. Designers of special purpose processor chips (DSPs, graphics processors) have taken advantage of this trend for many years, making it natural to do the same thing with network processors.

This trend is now also affecting general-purpose processors, which have until recently been dominated by single core designs, in large part because marketing forces have put such a premium on single-threaded performance.

One challenging aspect of multiprocessors is that they are intrinsically more difficult to program than conventional processors, simply because parallel programming is more difficult than sequential programming, and because relatively few software developers have significant experience with parallel programming. Network processors pose additional challenges, since current products typically need to be programmed for peak performance in order to achieve the performance levels that system vendors require. While the challenges of programming these chips is significant, the situation is improving with the development of better tools. Newer components, with higher clock rates, instruction caches and larger numbers of processors can also be expected to help, since they will reduce the need for low level hand-optimized code in order to achieve desired performance objectives.

While NP programming poses a challenge, these devices do offer a compelling combination of flexibility and performance. In the GENI context, this flexibility is essential, since the GBP must be adaptable enough to support a wide variety of different network architectures, many of which can be expected to bear little resemblance to conventional architectures. While general purpose processor chips can provide similar flexibility, and are more familiar and easier to program, they cannot come close to matching the performance of NPs. Typical single core processors used in routing applications can typically sustain throughputs of no more than a few Gb/s, and these performance levels are only achieved for traffic with a high proportion of large packets. When subjected to traffic consisting entirely of minimum size packets, their performance deteriorates significantly, in part for intrinsic architectural reasons and in part because existing general purpose operating systems were never intended for this type of application and were never optimized to perform well in this setting. Network processors, on the other hand, are designed to support sustained high throughput, even when subjected to such extreme traffic conditions. Emerging multi-core processors can be expected to do much better, but only if programmed to take full advantage of the opportunities for parallelism that the hardware offers. In such applications, the programming challenges are much like the challenges that arise when programming NPs, so one should not expect continuing performance improvements in general-purpose chips to eliminate the challenges associated with programming network elements for high performance.

4.2. Field Programmable Gate Arrays

Field programmable gate arrays are configurable logic devices that can be used to implement arbitrary digital circuits. The basic building block of an FPGA is typically a logic cell that includes a flip flop and a lookup table (LUT) that can implement any logic function on up to four inputs. Reference [CH02] describes a flexible gigabit router that uses an FPGA at each port to do all of its packet processing (both ingress and egress). The components used in this design were fabricated using 180 nm technology and are now over five years old. They contain approximately 38,000 logic cells, have 80 KB of on-chip memory and are limited to clock rates of under 100 MHz. The newest components (e.g. Xilinx Virtex 5 family) are being fabricated in 65 nm technology and include logic cells that can implement any 6 input logic function, allowing them to implement complex functions with fewer levels of logic, leading to faster designs [VRTX]. The largest devices have over 200,000 logic cells (200,000 6 input LUTs and 200,000 flip flops), over 1 MB of memory and can be clocked at over 500 MHz. These raw numbers suggest that the new devices are more than 20 times as capable as the earlier generation. While this probably overstates the advantage, it is likely that the new devices offer at least 10 times the performance.

The combination of increasingly capable FPGAs and modern design tools based on hardware description languages (HDL) such as VHDL and Verilog, make the design of high performance network hardware roughly comparable in difficulty to the design of high performance software. HDLs allow circuit designers to operate at higher levels of abstraction, allowing them to be more productive and leading to greater opportunities for design re-use. Freshmen and sophomore computer science and computer engineering students now routinely learn to use these tools and are developing the skills needed to implement significant hardware systems. While HDLs make hardware development more like software development, there are significant differences. The most significant conceptual difference is that the semantics of HDLs are intrinsically different from the semantics of conventional programming languages, since HDLs define circuits, rather than instructions to be executed sequentially on some underlying processor. Parallelism is an intrinsic characteristic of the design of hardware systems and HDLs do not relieve the designer of the need to deal with parallelism. However, as noted above, trends in processor architecture are making it important for software developers to come to grips with parallelism as well.

One sometimes awkward aspect of the FPGA marketplace is that, because of the relatively limited competition in this market (two vendors dominate), prices for the highest performance components tend to be unreasonably high (thousands of dollars per chip). On the other hand, prices of mid-range components are fare more reasonable (hundreds of dollars per chip), and the mid-range components of today were the high end components of just a few years ago. In research systems, there is a natural (and appropriate) tendency to always use the highest performance devices available, and this is likely to be the case for the GBP. This implies that FPGAs may represent a relatively high cost element for the GBP. This is likely to be offset by the fact that FPGA vendors have long been very generous in providing components for research use, either for free or at very substantial discounts. It seems likely that vendors will show similar generosity in the context of an important and highly visible national project like GENI.

Over the next 3 to 5 years, we can expect to see continuing improvements in FPGA technology. Since the transition to 65 nm technology is just now taking place, it is probable that the next big improvement will be at least 3 years from now. In the meantime we can expect to see the kinds of capabilities now available in high end devices become available in mid-range devices, and we can expect to see versions of the high end devices that incorporate processor cores and higher performance IO (10 GHz differential signals). Such developments will give networking researchers the tools needed to implement high performance systems that implement novel network architectures.

4.3. Memory Components

Memory components are a crucial element of packet processing subsystems, and are typically one of the key determinants of overall system performance. In conventional routers, memory is used to store packets awaiting transmission on outgoing links and to store routing information. In metarouters, the uses of memory will depend on the specifics of the metanetwork architecture, service model and protocols.

There are three key parameters of memory subsystems: storage capacity, bandwidth and latency. DRAM modules provides the highest storage capacity (256 MB to 1 GB) and can deliver bandwidths of about 25 Gb/s, but their latency is quite high, in the range of 50 to 100 ns in typical system contexts. In a network processor, whose constituent processors completes an instruction in 0.7 ns, the time needed to retrieve data from DRAM is a serious issue, particularly since caches in NPs are typically quite small or non-existent. QDR SRAM chips provide bandwidths of 25 Gb/s and capacities of up to 8 MB. Their latencies are substantially better than DRAM, but the effective latency seen by a processor core in an NP, or a circuit module in an FPGA can still be in the 10-30 ns range.

TCAMs have become an important element of many packet processing systems, since they provide a flexible, high performance associative lookup capability that is particularly useful for packet classification in firewalls and content scanning. Modern TCAMs can support 256K words of 72 bits each, and can handle lookups on keys of up to eight words. In typical applications, they can support over 100,000 associative lookups per second. Their latencies generally fall between those of SRAM and DRAM. On the other hand, they can consume large amounts of power, making it necessary for applications using them to manage their use to minimize power consumption.

To get the highest possible performance, packet processing systems need to deal with the different performance characteristics of different memory technologies. This is one of the factors that make programming of NPs challenging, since programmers must deal with a non-uniform memory model. To make matters more complicated, the bandwidth of individual memory subsystems is limited, making it important for programmers to allocate data structures to specific memory subsystems based on the frequency of access. In conventional network settings, it's possible to do this, since the data access frequencies are predictable enough to make it feasible to map data structures based on worst-case assumptions. In the GENI context, the access patterns for some data structures may be much less predictable, making it necessary for programmers to devise other strategies for using the memory bandwidth effectively. On the other hand, because GENI is an experimental environment, it is less necessary to maintain the very highest performance levels. It's certainly not essential that metarouters sustain packet processing rates determined by minimum packet sizes, as is the standard practice for commercial routers. Furthermore, we expect a higher ratio of processing to IO in GENI networks than for conventional routers. This will also contribute to reducing the pressure on IO performance, making the programming task somewhat less difficult.

4.4. Switching Components

Switching components represent another key building block for the GBP. In recent years, there have been big strides in the performance and features of switching components, and these developments can be expected to have a direct impact on the GBP. In particular, vendors have recently begun sampling switch chips that implement complete 10 GE switches with 20 or more interfaces. These chips incorporate level 2 forwarding and support VLAN subnets with independent spanning trees. Using these chips, system vendors can implement complete high performance LAN switches by adding little more than the optical components needed to terminate the external links. Moreover, these chips can also be used as backplane switching devices, using the VLAN tags to distribute traffic load across highly parallel interconnection network topologies. An example of such a component is Fulcrum's FM2224 which has 24 bi-directional 10 GE interfaces, giving it a throughput of 240 Gb/s. Features include a 16K entry address table, support for the full complement of 4K VLAN tags and support for eight packet priorities with weighted round-robin scheduling [FLCM].

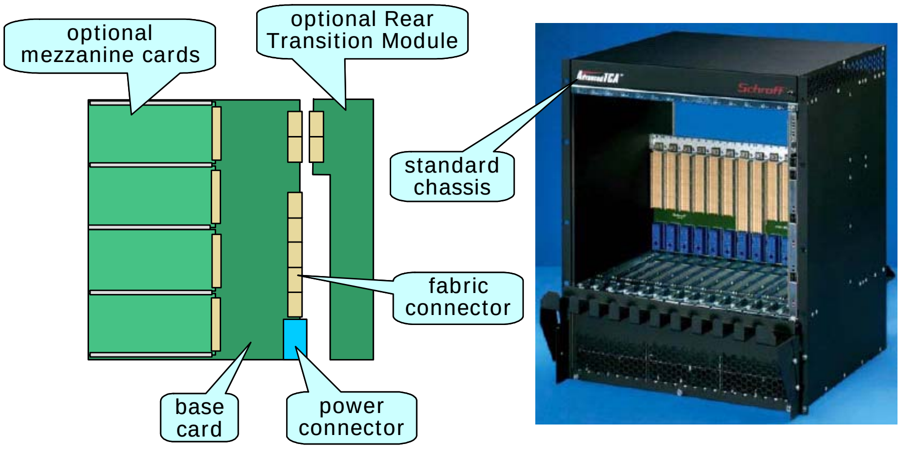


Figure 3. Selected ATCA Components

4.5. ATCA Standards and Components

The *Advanced Telecommunications Computing Architecture* (ATCA) is a rapidly developing set of standards designed to facilitate the development of carrier-class communications and computing systems [PCMG]. ATCA defines standard physical components and some standard patterns for how to use those components to construct high performance systems. It has attracted broad industry support and is expected to lead to the development of a range of inter-operable subsystems that will allow more cost effective and flexible development of new communication systems.

ATCA has important implications for the networking research community. Networking researchers interested in creating new network architectures and services have long had to content themselves with implementing experimental networks using commodity PCs. Commercial routers have been difficult to use in research contexts, because vendors have been unwilling to allow researchers to have access to the technical details needed to perform experiments and make changes. ATCA is creating an intermediate market for router subsystems that can be assembled into powerful, carrier-class communication systems. Subsystem vendors design their products to be highly flexible to enable their use by multiple system vendors. This is creating an unprecedented opportunity for the networking research community. We now have the tools to create high performance research systems that are built on a hardware platform that is directly comparable to the best commercial systems.

Figure 3 shows a standard 14 slot ATCA chassis with backplane, power distribution system and cooling fans. Such chasses are now available from several vendors. The backplane includes standard signals for clock distribution and low level system management. It also defines *fabric connections* that implement several interconnection topologies for high speed inter-board communication (differential signal pairs suitable for 2.5 Gb/s data rates). ATCA standardizes key aspects of the boards that are used with the chassis, including physical size, connector type and placement and the use of certain of the connector signals. It also defines standards for *mezzanine cards* that can be optionally used with an ATCA base card. In addition, it defines standards for optional *Rear Transition Modules* (RTM) which are small cards that are inserted into the back side of the chassis and are can be used for interconnecting multiple chasses in larger systems. These elements of the standard are also shown in Figure 3.

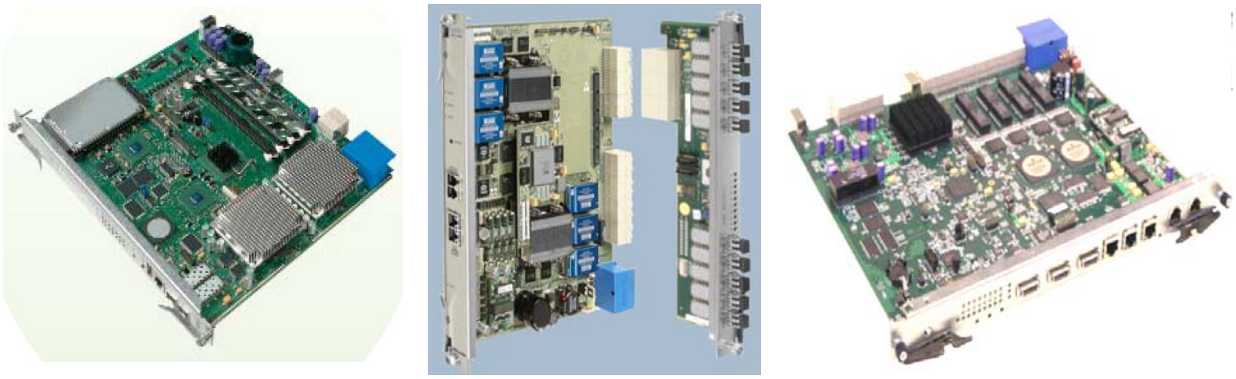


Figure 4. Sample ATCA Components (Intel MPCBL0001, Radisys 7010, Diversified Technology ATS2148)

Figure 4 shows examples of ATCA subsystems that are now starting to appear. The Intel MPCBL0001 is a compute blade that includes two Xeon processors that implement a shared memory multiprocessor, with an on-board disk. The Radisys 7010 is a network processing blade that contains two IXP 2800 network processors [RA05]. The two NPs also share access to a dual-port TCAM that can be used for packet classification and other applications requiring associative lookup. The Radisys board supports RTMs that provide external IO connections. The figure shows an RTM with 10 fiber gigabit Ethernet interfaces. Diversified Technologies' ATS2148 is a switch blade that includes an Infiniband switch with 10 Gb/s ports. In a typical application, two such switch blades would be used in a chassis to provide switching among twelve other cards. Other switch types are also available. In particular, switch boards that support 10 Gb Ethernet ports (with multi-spanning tree VLAN support) are expected to become available in the third quarter of 2006.

5. System Architecture Options and Issues

This section discusses two high level system architecture options for the GENI GBP and issues arising from consideration of these options.

5.1. Virtualized line card architecture

Consideration of a conventional router or switch architecture leads naturally to a GBP architecture in which line cards are replaced by a *virtualized line cards* that consist of a substrate portion and generic processing resources that can be assigned to different meta line cards (see Figure 5). The substrate is responsible for configuring the generic processing resources so that different meta line cards can co-exist without interference. On receiving data from the physical link, the substrate first determines which meta line card it should be sent to and delivers it. Meta line cards pass data back to the substrate, in order to forward it through the shared switch fabric, on input, or to the outgoing link, on output.

One issue with this architecture concerns how to provide generic processing resources at a line card, in a way that allows the resources to be shared by different meta line cards. Conventional line cards are often implemented using Network Processors (NP), programmable devices that include high performance IO and multiple processor cores to enable high throughput processing. It seems natural to take such a device and divide its internal processing resources among multiple meta line cards. For example, an NP with 16 processor cores could be used by up to 16 different meta line cards, by simply assigning processor cores. Unfortunately,

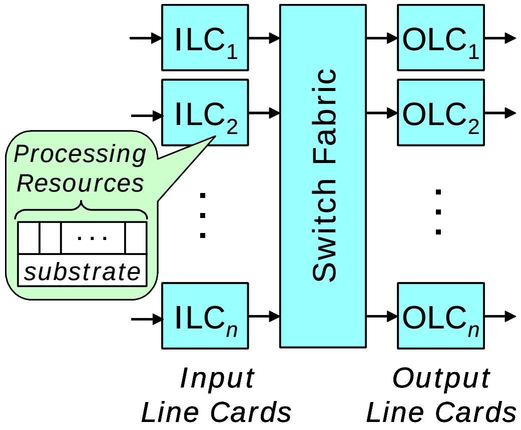


Figure 5. Virtualized Line Card Architecture

current NPs are not designed to be shared in this way. All processing cores share access to the same physical memory (there are no built-in mechanisms for memory protection), making it difficult to ensure that different meta line cards don't interfere with one another. Also, each processor core has a fairly small program store. This is not a serious constraint in conventional applications, since processing can be pipelined across the different cores, allowing each to store only the program it needs for its part of the processing. However, a processor core implementing an entire meta line card must store the programs to implement all the processing steps for that meta line card. The underlying issue raised by this discussion is that efficient implementation of an architecture based on virtualized line cards, requires components that support *fine-grained virtualization* and conventional NPs do not.

The virtualized line card approach is also problematic in other respects. Because it associates processing resources with a physical link, it lacks the flexibility needed to support metarouters with a wide range of different processing needs. Some metarouters may require more processing per unit IO bandwidth than NPs provide, and this is difficult to accommodate in a virtualized line card approach. The virtualized line card approach also does not easily accommodate alternate implementation approaches for metarouters (such as configurable logic).

5.2. Processing pool architecture

The processing pool architecture separates the processing resources used by the metarouters from the physical link interfaces. This allows a much more flexible allocation of processing resources and greatly reduces the need for fine-grained virtualization. This architecture, which is illustrated in Figure 6, provides a pool of *Processing Engines* (PE), that are accessed through the switch fabric. The line cards that terminate the physical links forward packets to PEs through the switch fabric, but do not do any processing that is specific to any particular metanetwork. There may be different types of PEs, including some implemented using network processors, others implemented using conventional microprocessors and still others implemented using FPGAs. The NP and FPGA based PEs are most appropriate for high throughput packet processing, the conventional processor is most appropriate for control functions that require more complex software or for metanetworks with a high ratio of processing to IO. A metarouter may be implemented using a single PE or multiple PEs. In the case of a single PE, data will pass through the physical switch fabric twice, once on input, once

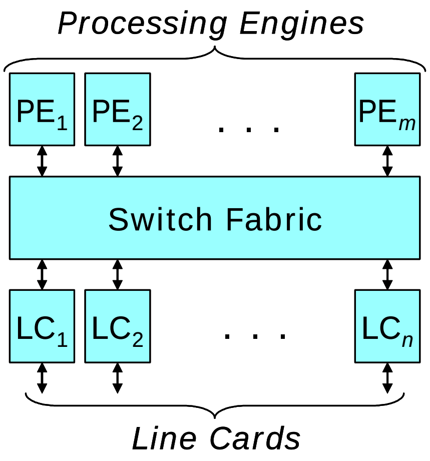


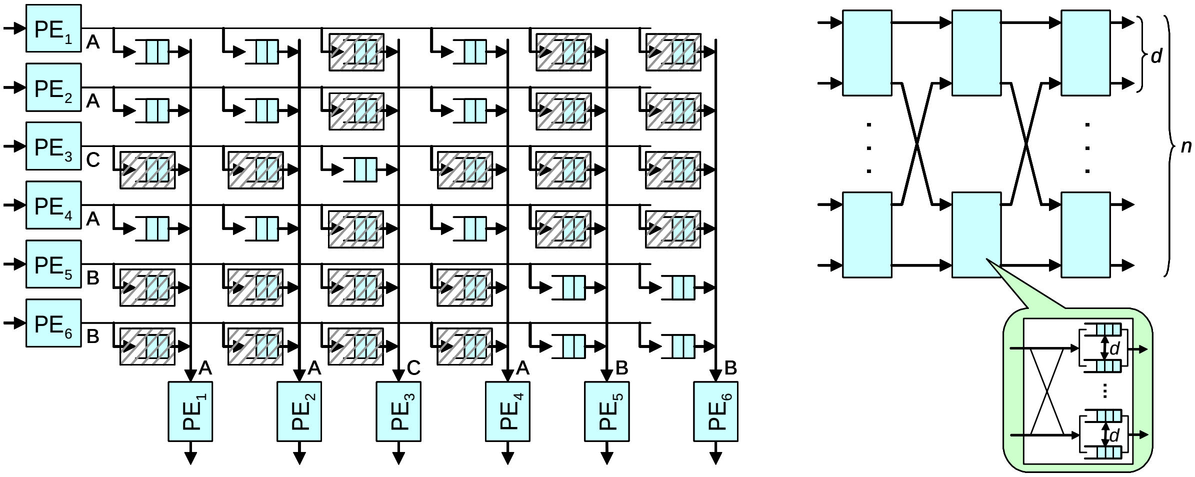
Figure 6. Processing pool architecture

on output. In a metarouter that uses multiple PEs to obtain higher performance, packets may have to pass through the switch fabric a third time.

The primary drawback of the *processing pool architecture* is that it requires multiple passes through the switch fabric, increasing the delay that packets are subjected to and increasing the switch fabric capacity needed to support a given total IO bandwidth. The increase in delay is not a serious concern in wide area network contexts, since switch fabric delays are typically 10 µs or less. The increase in the switch fabric capacity does add to system cost, but since a well-designed switch fabric represents a relatively small part of the cost of a conventional router (typically 10-20%), we can double, or even triple the switch fabric capacity without a proportionally large increase in the overall system cost. In the GENI context, the switch fabric bandwidth implications of the processing pool architecture are significantly reduced, since we expect the metarouters implemented within a GBP to have a relative high ratio of processing capacity to IO bandwidth, compared to conventional routers.

The great advantage of the processing pool architecture is that it greatly reduces the need for fine-grained virtualization within NP and FPGA-based subsystems, for which such virtualization is difficult. Because the processing pool architecture brings together the traffic for each individual metarouter, there is much less need for PEs to be shared among multiple metarouters. The one exception to this is metarouters with such limited processing needs that they cannot justify the use of even one complete PE. Such metarouters can still be accommodated by implementing them on a general purpose processor, running a conventional operating system that supports a virtual machine environment. We discuss below one approach that allows such metarouters to share an NP for *fast path forwarding*, while relying on a virtual machine running within a general purpose processor to handle exception cases.

Another advantage of the processing pool architecture is that it simplifies the sharing of the switch fabric. The switch traffic must maintain traffic isolation among the different metarouters. One way to ensure this is to constrain the traffic flows entering the switch fabric so as to eliminate the possibility of internal congestion. This is difficult to do in all cases. In particular, metarouters consisting of multiple PEs should be allowed to use their "share" of the switch fabric capacity in a flexible fashion, without having to constrain the pair wise traffic flows among the PEs. However allowing this flexibility makes it possible for several PEs in a given



Buffered Crossbar Switch

Buffered Multistage Switch

Figure 7. Traffic Isolation in Switch Fabrics

metarouter to forward traffic to another PE at a rate that exceeds the bandwidth of the interface between the switch fabric and the destination PE.

There is a straightforward solution to this problem in the processing pool architecture. To simplify the discussion, we separate the handling of traffic between line cards and PEs from the traffic among PEs in a common metarouter. In the first case, we can treat the traffic as a set of point-to-point streams that are rate-limited when they enter the fabric. Rate-limiting these flows follows naturally from the fact that they are logical extensions of traffic flows on the external links. Because the external link flows must be rate limited to provide traffic isolation on the external links, the internal flows within the switch fabric can be configured to eliminate the possibility of congestion.

For PE-to-PE traffic, we cannot simply limit the traffic entering the switch, since it's important to let PEs communicate freely with other PEs in their same metarouter, without constraint. However, because entire PEs are allocated to metarouters in the processing pool architecture, it's possible to obtain good traffic isolation in a straightforward way, for this case as well. We illustrate this in Figure 7 for two different switch fabric architectures. The first uses a buffered crossbar and divides the six PEs among three metarouters, identified by the letters, *A, B,* and *C*. Each crosspoint has a configurable *enable* bit that allows the PE in its row to send to the PE in its column. If these bits are configured to allow only the traffic flows among the desired sets of PEs, each of the metarouters can operate as though it has a dedicated crossbar of its own (in the diagram, the shaded boxes identify crosspoints that are disabled).

The second architecture uses a more scalable three stage network, with buffered switch elements, similar to those used in large, conventional routers, such as Cisco's CRS-1 [CSCO]. Traffic entering the switch fabric from a PE belonging to one metarouter can be sent only to PEs in the same metarouter. This can be easily enforced at the switch fabric input. The switch elements in the first stage distribute traffic evenly across the switch elements in the second stage to balance the load. Each of the second stage switch elements implements *d* separate queues at each of its output links, where *d* is the number of output ports of the third stage switch elements (typically 32 or 64). This allows the second stage switches to isolate the traffic flows going to different outputs of the overall network, so that they cannot interfere with one another. Since

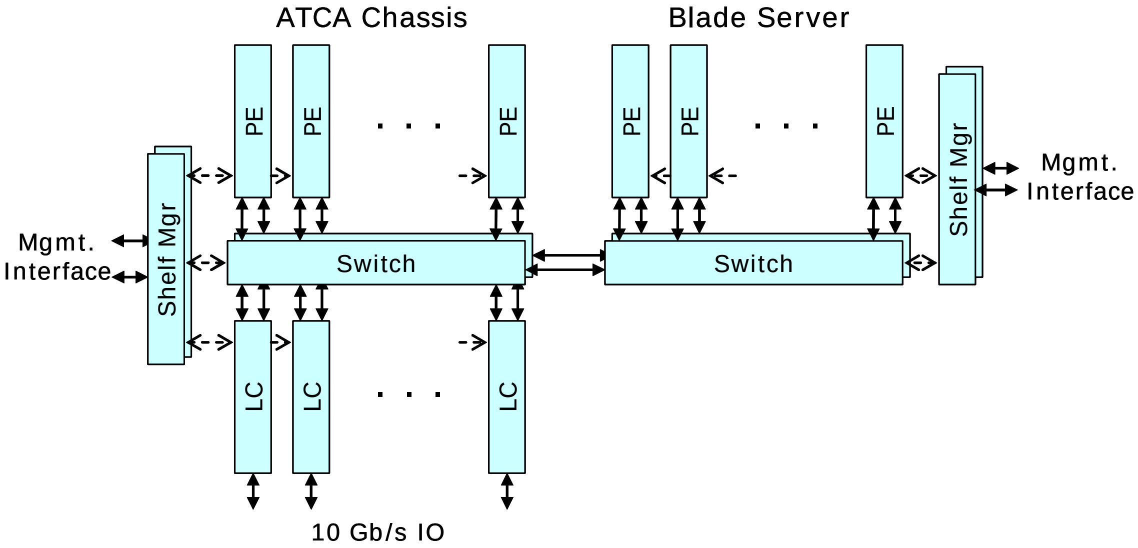


Figure 8. Baseline Configuration

PEs are assigned to specific metarouters, this level of traffic isolation is sufficient to ensure that no metarouter can interfere with the traffic for another metarouter. These two examples are actually just special cases of the more general observation that any switch fabric that is nonblocking on a port basis, can serve as a nonblocking metaswitch, so long as the routing of packets can be constrained to stay within the set of PEs belonging to each virtual router.

5.3. Fine-grained sharing

As mentioned above, the processing pool architecture greatly reduces the need for fine-grained sharing of NP and FPGA-based subsystems. The only metarouters that require fine-grained sharing are those that have relatively modest total processing requirements. As has been noted, these can served by virtual machines on general purpose processors. While this is a workable solution, it does prevent such metarouters from making use of the higher performance processing that an NP-based PE can provide, significantly reducing the overall scalability of systems that rely exclusively on general purpose processing to handle metarouters with limited processing needs. Since we expect a GENI GBP to host many metarouters that do have limited processing needs, it would be useful to find some way to enable such metarouters to share an NP-based PE.

While NPs provide no mechanisms to support virtualization, they can be effectively shared in a certain common special case. We expect that many metarouters developed for GENI will be naturally decomposable into a *fast path* and an *exception path*. The fast path processing is responsible for forwarding the vast majority of packets and requires relatively simple operations, while the exception path deals with packets that require more complex decision-making. For metarouters that can be decomposed into a fast path and an exception path, the fast path can be delegated to an NP. Using fairly simple, well-understood techniques, an NP can be shared by multiple fast paths, while maintaining the necessary isolation to keep the different metarouters from interfering with one another. A specific design for sharing an NP among multiple fast paths is discussed in Section 6.

Figure 9. IBM Blade Server

6. Reference Design

This section describes a reference design for a GENI backbone platform, that attempts to meet the objectives outlined in Section 2. Wherever possible, we have identified specific components and subsystems that can be used to implement various parts of the system. This is not meant to suggest that these are the only possible choices, but to make it clear that an effective GBP solution can be assembled largely from components that have been or are being developed for commercial use. While the integration of these subsystems into a comprehensive system is not a trivial effort, there is very little new hardware that must be developed, significantly reducing the risks associated with the GBP development.

6.1. System Overview

The reference design uses ATCA components to implement the processing pool architecture discussed in Section 5. To reduce costs, it also makes use of commercial blade servers where appropriate. Figure 8 shows a baseline system configuration that will be used as a reference throughout this section. This configuration consists of a single ATCA chassis plus a commercial blade server. Section 7 discusses several other specific system configurations, that demonstrate how the architecture can be scaled to both larger and smaller sizes using the same system building blocks.

The ATCA chassis contains several primary components. The redundant *Shelf Manager* (SM) monitors the operation of the system and controls power and cooling. It provides an Ethernet network interface through which the chassis can be monitored remotely and through which individual blades can be controlled (including hardware reset capability). The *Line Cards* (LC) terminate the external IO links and implement the substrate functions needed to multiplex/demultiplex different metalinks to/from shared physical links. The *Processing Engines* (PE) provide generic processing resources for use by different metarouters. The architecture supports multiple types of PEs, including PEs based on general-purpose processors, network processors and configurable logic chips. The *Switch Blades* provide high

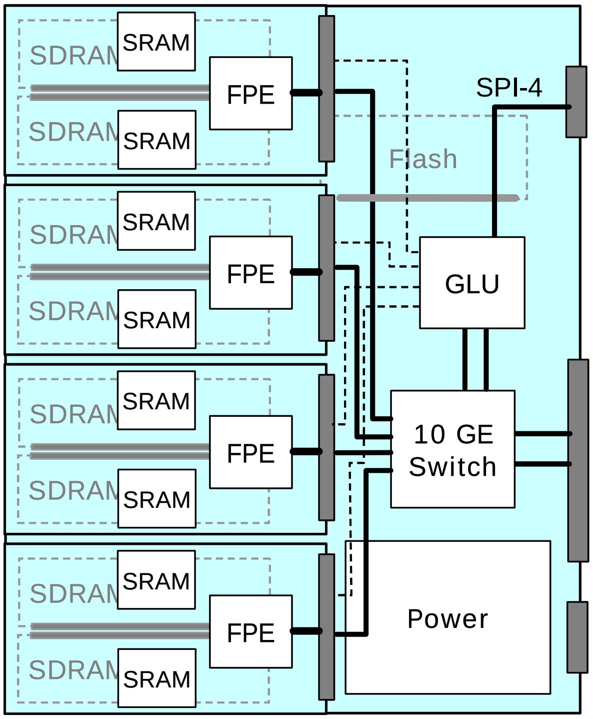


Figure 10. Configurable Logic Blade

bandwidth IO linking the LCs and PEs and each has four up-links, for connecting to other chasses, that can operate at either 1 Gb/s or 10 Gb/s. Each switch port provides a 10 Gb/s Ethernet interface with full VLAN support. The switch blades can be configured through the SM. The chassis has a total of 14 slots, two of which are reserved for the switch blades, leaving 12 for LCs and PEs (the SMs use separate, special-purpose slots). A typical GBP might use three of these for LCs and nine for PEs.

6.2. General Purpose Processing Blades

To reduce overall system costs, we propose to use a commercial blade server to host the general purpose PEs, rather than using ATCA blades for this. Because the ATCA standards are still relatively new, the cost of ATCA components is not yet as competitive as those for commercial blade servers. Also, the IO capabilities of the ATCA chassis far exceeds what conventional processor blades can use effectively. For this reason, it makes sense to reserve ATCA slots for PEs that can make greater use of its IO resources. Figure 9 shows an IBM Blade Center system which is typical of the class of systems that can be used to provide general purpose processing in the GBP. This system includes 14 processor blades, each with two 3.6 Ghz Xeon processors with two on-board 80 GB disks and up to 8 GB of memory. These are interconnected through redundant switch cards that plug into the rear side of the chassis and support redundant 1 Gb/s Ethernet connections to each slot. Each switch card has six 1 Gb/s up links that can be used to connect to the ATCA chassis. It is likely that switch cards with 10 Gb/s up links will be available soon.

6.3. NP Blades

Network Processors (NP) are high performance components with tens of processor cores and high performance IO. NP blades can be used in multiple contexts within the GBP. Specifically, they can be used both to implement line cards and PEs. The NP blades can be implemented

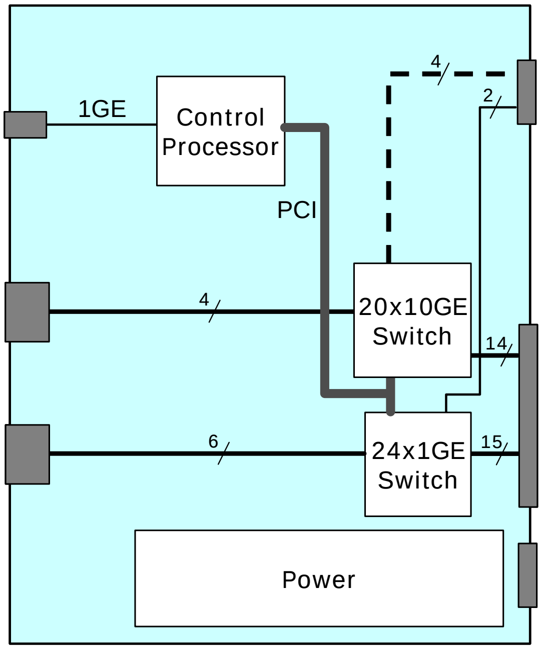


Figure 11. Simplified Block Diagram of Radisys ATCA 2210

using the Radisys ATCA 7010 (see Figure 4). These blades each contain two Intel IXP 2800 network processors [RA05]. Each NP has sixteen internal processor cores for high throughput data processing, plus an Xscale processor (running Linux) for control. Each NP has three banks of RDRAM providing 750 MB of storage and four banks of QDR SRAM. The two NPs also share access to a dual-port TCAM that can be used for packet classification and other applications requiring associative lookup. The Line Card provides external IO through a Rear Transition Module (RTM). The board has several network connections. Two 10 Gb/s Ethernet connections are provided to the backplane for high speed data transfers. These connections go to the redundant switch blades. In addition, there are two 1 Gb Ethernet connections from the Xscale to the backplane and two more that come out the front panel.

6.4. Configurable Logic Blades

Figure 10 shows a block diagram of a configurable logic blade that can be used to implement hardware-based PEs. This blade includes a carrier card with four mezzanine card slots, each of which hosts a large FPGA called the FPE (e.g. Xilinx Virtex 5 LX330 or Altera Stratix II EP2S60) plus two banks of SDRAM and two or more QDR SRAM chips. The carrier card includes an on-board 10 GE switch that has two ports connecting to the backplane (one to each switch blade) and one port for each of the mezzanine cards. The carrier card also includes a GLU chip that has two key functions. First, it provides the logic to program the FPEs on the mezzanine cards remotely. New bit files are sent to it through the on-board 10 GE switch, where they are stored in a local flash memory. From there, they can be transferred to the FPEs, which are then reset. The GLU chip also provides a SPI4 interface to the RTM connector. This is intended to allow RTMs that provide external IO connections to be used with the FPE blade. A blade configured with an RTM can be used to implement Line Card functions.

6.5. Switch Blades

Figure 11 is a block diagram of a switch blade that was under development by Radisys (ATCA 2210) at the time of this report was written and is expected to be available in the third quarter of

2006. This blade includes a 20 port 10 Gigabit Ethernet switch that provides one port to each of the 12 slots designated for PEs and LCs, plus two ports for connection to a redundant switch blade. It also provides four ports that can be connected either to the front panel, or the RTM connector. The 10 GE switch includes VLAN support, making it possible to constrain the routing of traffic from different metarouters. This is useful for providing the traffic isolation needed to keep metarouters from interfering with one another.

The board also includes a 24 port 1 GE switch intended for carrying control traffic and a Control Processor that configures the two switch components through an on-board PCI interface. The Control Processor has a front panel connection through which it can receive control messages and report status. Additional details can be found at www.radisys.com.

6.6. Line Cards

As noted earlier, the Line Cards can be implemented either using an NP blade or a configurable logic blade. However it is implemented, the LC must provide the substrate functionality needed to allow multiple metalinks to share the external physical links. On the ingress side, packets are demultiplexed and forwarded through the switch to the appropriate PEs. The LC can be configured to terminate IP and/or MPLS tunnels to facilitate reception of packets from remote sites that have no dedicated connections to the GBP. It must include a header mapping function to map arriving packets to a metarouter number, a meta-interface number and a physical destination within the GBP. Packets are labeled with their metarouter number and meta-interface number by the LC and forwarded through the switch to the specified destination. Packets going to the switch are sent through queues with a configured maximum rate, in order to prevent switch congestion.

On the egress side, packets are received from the switch, already labeled with their metarouter and meta-interface numbers. The LC uses these to map the packets to the proper outgoing queue, which is configured to provide the appropriate encapsulation (if necessary). The egress-side software also monitors the rate at which packets are received on each meta-interface, and raises an exception to the GBP control software, if the received rate exceeds the allowed rate for a given virtual interface. It is then up to the GBP control software to decide what action needs to be taken, if any.

6.7. Processing Engines

The reference design supports several types of PEs that can be used to implement metarouters. We refer to these as *General Purpose PEs* (GPE), *Network Processor PEs* (NPE) and *Field Programmable Gate Array PEs* (FPE). Users will specify the number of PEs of each type that are needed for their design and for each PE, they will specify its meta-interfaces and its interfaces to the metaswitch (see Figure 2).

The GPEs can be used in one of two modes. In *raw mode*, the entire GPE blade is under the complete control of its user. Users may run their own operating system on a GPE in raw mode and are fully responsible for its operation. There is no software to implement substrate functionality on a GPE in raw mode, making it necessary for the system to use the switch fabric and LCs to ensure the necessary isolation. In *cooked mode*, a GPE blade runs a standard GBP OS that provides substrate functionality and allows the blade to be shared by multiple metarouters. In this mode, users may reserve a portion of the blade's resources for their exclusive use, and the system will attempt to accommodate such requests by mapping MPEs to physical PEs where the needed resources are available.

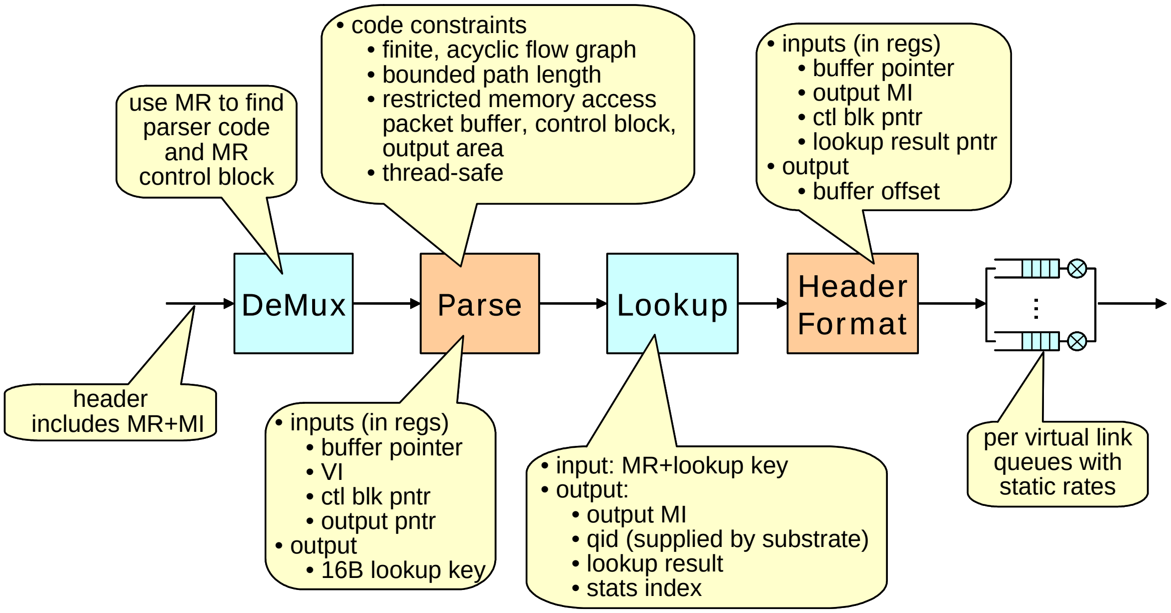


Figure 12. Cooked mode NPE Processing pipeline

NPEs and FPEs can also be used in raw mode. When an NPE is used in raw mode, the GBP control software boots it using a standard OS kernel, but then configures a user account on the NPs' control processors with permissions that allow the user to reconfigure the control processors' OS kernel as well as the software running on the micro-engines. Since an NPE in raw mode contains no substrate functionality, isolation is provided by the switch fabric and LCs. FPEs are handled similarly. In this case, the user specifies a configurable logic bit-file to be downloaded to their assigned FPE and this file is sent to the GLU component on the carrier card which programs the FPE and initializes it. Again, it is up to the switch fabric and LCs to provide isolation.

Since NPE and FPE blades have little built-in support to facilitate shared use, it is more difficult to share them in a fully general way. In the case of NPEs however, sharing is feasible if the scope of the metarouter-specific processing is limited. In the next subsection we describe a cooked mode for NPEs that we expect to be useful in certain cases that we expect to be quite common in the GBP.

6.8. Cooked Mode for NPEs

Because Network Processors lack the mechanisms to enable general shared use, it's not practical to try to provide shared usage of the NPEs, in a general sense. However, there is a particular way that NPEs can be shared among metarouters that can be useful in the GENI context. In particular, we expect many metarouters to be naturally decomposable into a *fast path* that handles routine forwarding of packets and an *exception path* that handles packets for which more complex processing is required. NPEs are well-suited to the fast-path processing and the fast-path processing can be organized into a generic framework that allows fast path processing for multiple metarouters to be implemented within a single IXP 2850 subsystem (half of an NPE blade).

The fast path can be viewed as a pipeline with five stages. In the *Demux* stage, packets are received from the switch fabric, with the *metarouter* number (MR) and *meta-nterface* number (MI) already inserted into the packet header (they are placed there by the ingress LC). The Demux stage uses the MR number to identify an MR-specific control block and a pointer to an MR-specific code segment that parses the packet header and returns an opaque *Lookup Key* for use by the next stage. The second stage is the *Lookup Stage* that combines the given Lookup Key with the MR number to perform a lookup in the TCAM. The first matching entry in the TCAM is returned as the lookup result, which includes an output MI and some MR-specific results. These are used in the next stage, the *Header Formatting* stage, which includes an MR-specific code segment that formats the header for the outgoing packet, which is then placed in a per MI queue. There is also a queue for exception packets, which are forwarded to a GPE for exception processing. The fast path processing pipeline is illustrated in Figure 12.

Note that the per-MR code segments in the Parsing and Header Formatting stages must be restricted to ensure that the different MRs can co-exist without interference. In particular, they are restricted in the memory they can access and they must be free of unbounded iteration or recursion. These restrictions can be enforced using a combination of static and dynamic checks. Alternatively, they can be enforced by requiring that users specify their code in a specially designed language that enforces the necessary constraints by construction. Since the purpose of these code segments is very limited, these restrictions pose no serious constraints on the MRs. Note that MRs that cannot live with the constraints imposed by the cooked mode always have the option of using a raw NPE blade.

7. System Configurations

The baseline configuration shown in Figure 8 comprises two chasses, an ATCA chassis and a general purpose blade server chassis. The ATCA chassis has 14 slots, two that are for the switch blades and 12 that can be used for either LCs or PEs. A typical blade server has a comparable number of slots that can each be equipped with general purpose processing blades, often with dual processors operating in a shared memory mode. The system is designed to be very flexible and can support different mixes of cards of the various types. For the GBP application however, we expect most of the slots in the ATCA chassis to be devoted to PEs of various types rather than LCs. This is to allow users to experiment with networks that do more extensive processing than is typically done in conventional routers, and to relieve researchers of the need to highly optimize their designs to get the maximum possible performance. With this in mind, we expect a typical configuration to include three times as many slots for PEs as for LCs, so the ATCA chassis might include 3 slots for LCs and 9 for PEs. Of the PE slots, we would expect most to be used for NPEs with perhaps 1 or 2 for FPEs. This reflects two things. First, we expect more researchers to be interested in using NPEs than FPEs, and second, each FPE blade contains four mezzanine cards that can be allocated independently, reducing the number of blades that are needed. We expect all users to require general purpose processing resources for control purposes, and many GENI users will likely use GPEs for data forwarding as well, since GPEs offer a more familiar development environment in which it is easier to develop and test experimental systems. Because the IO capability of GPEs is relatively limited (perhaps 1 Gb/s per blade), having a fully configured blade server to go along with the ATCA chassis makes sense.

7.1. Multi-Chassis Configurations with Direct Connections

The simplest way to scale up the baseline configuration to is to replicate it and connect the ATCA chasses to one another using direct connections, as illustrated in Figure 13. Here, we have three subsystems, each consisting of an ATCA chassis and a general purpose blade server.

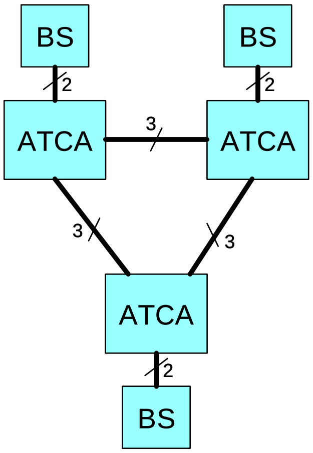


Figure 13. Multi-chassis Configuration with Direct Connections

In each pair, the ATCA chassis and blade server are connected by a pair of 10 Gb/s links, while each pair of ATCA chasses is connected by three 10 Gb/s links. This gives each chassis 60 Gb/s of inter-chassis bandwidth. While this is considerably less than the intra-chassis bandwidth (each ATCA chassis has an internal switching capacity of 240 Gb/s), it is sufficient so long as the PEs used by any single metarouter are clustered within the same chassis. In this case, inter-chassis bandwidth is only used to gain access to LCs that terminate physical links in other chasses. If each chassis has only 3 LCs, each terminating a 10 Gb/s link, 30 Gb/s of inter-chassis bandwidth is sufficient to handle the worst-case in this configuration. This does mean that no single metarouter can scale up to use more than 9 PEs, but since we expect the vast majority to use no more than one or two PEs (indeed many will use a fraction of a PE), this appears to be an acceptable limitation. Note that this limitation is entirely a function of the specific switch blades that have been proposed for the reference system. Switch blades with larger numbers of uplink ports would provide greater inter-chassis bandwidth, relaxing the constraints on the number of PEs in any single metarouter.

The direct connection approach can be used for systems with 2, 3, 4 or 7 chasses. In systems with 4 or 7 chasses, some inter-chassis traffic may require two hops, but the inter-chassis bandwidth is sufficient to accommodate this, so long as inter-chassis bandwidth is used only to reach LCs and so long as each chassis hosts at most 3 LCs. Note that a 7 chasses system has 84 slots in its ATCA chasses that can be used for LCs, NPEs or FPEs and 98 slots for GPEs.

7.2. Multi-Chassis Configurations with Multistage Switching

The direct connection approach while conceptually simple offers limited scalability. Figure 14 shows a multistage configuration that connects 20 subsystems, each containing an ATCA chassis and a blade server. Such a system has 240 slots for LCs, NPEs and FPEs plus 280 for GPEs. If each ATCA chassis has 3 LCs, the system as a whole, terminates 60 10 Gb/s links, providing 600 Gb/s of system IO capacity. The middle stage of switching is provided by switch cards that each have 20 external interfaces at 10 Gb/s each. While there are no existing ATCA cards that are configured in this way, the essential switching capability is no different than that provided by the Radisys 2210 cards. All that is needed is to equip such a card with 20 external interfaces, rather than connecting most of its 10 GE ports to the backplane. Hence, should it become necessary to scale the GBP to larger configurations, it should not be difficult to obtain cards with the requisite capability.

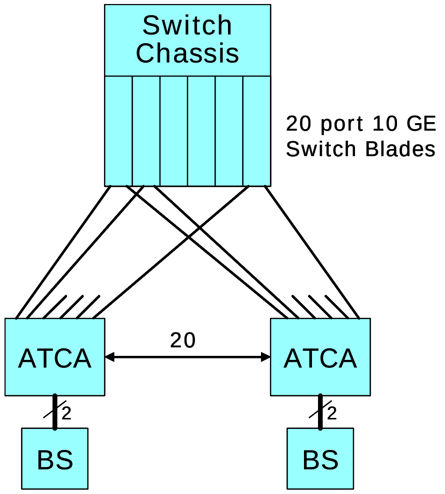


Figure 14. Large Configuration Using Multistage Switching

7.3. Smaller Configurations

While high performance systems will be needed for the GENI backbone, the GENI testbed will also require access routers at university sites, to act as gateways that feed traffic into the backbone. Smaller scale configurations of the GBP, perhaps with a different mix of PEs, can be useful in this context. A single ATCA chassis with one or two 10 Gb/s LCs, plus a mix of GPEs and NPEs could be suitable for this application. Smaller ATCA chasses (8 slot and 5 slot) with a single switch blade, rather than a redundant pair can also be used in such settings.

7.4. Cost Estimation

Figure 15 shows three tables, summarizing the estimated hardware cost of various system configurations, using three different sets of assumptions for component costs. The high estimate (top table) is based on current list prices, where these are known, and estimates of expected list prices, where list prices are not yet available. It does not take into account any discounts that might be obtained by negotiating with vendors, although it is certainly reasonable to expect that vendors will offer significant discounts for a high visibility national project of the size and importance of GENI. Hence, these numbers can reasonably be viewed as upper bounds on the costs of the various system configurations. The blade server pricing is based on the IBM Blade Center product line with the HS20 server blades configured with two processors and two disks. The NP blade prices are based on the Radisys 7010 product.

The medium estimate reflects prices that might be available in early-to-mid 2008, while the low estimate reflects such future prices accounting for significant discounts from vendors. It seems most likely that the actual costs to the GENI program would fall between the low and medium estimates, although if vendors choose to make major donations to GENI, some costs could be significantly lower.

Note, that while the spreadsheet shows a wide range of system of configurations, the most likely configurations for GENI backbone nodes are the single chassis pair configuration and the multichassis-3 configuration. The latter configuration would allow a backbone site with three incident fibers to terminate 2 wavelengths on each of its backbone fibers and have 30 Gb/s of bandwidth available to terminate access links. If we assume (conservatively) that 25 GENI backbone sites are equipped with the multichassis-3 configuration, the total hardware cost can

High Estimate		single chassis pair		multi-chassis 3		multi-chassis 5		multi-chassis 20	
component	unit cost	qty.	cost	qty.	cost	qty.	cost	qty.	cost
ATCA chassis	$8,000	1	$8,000	3	$24,000	5	$40,000	20	$160,000
ATCA shelf mgr.	$1,000	2	$2,000	6	$6,000	10	$10,000	40	$40,000
NP blade	$12,000	10	$120,000	30	$360,000	50	$600,000	200	$2,400,000
RTM with 10GE IO	$3,000	3	$9,000	9	$27,000	15	$45,000	60	$180,000
Config. logic blade	$10,000	2	$20,000	6	$60,000	10	$100,000	40	$400,000
Switch blade	$10,000	2	$20,000	6	$60,000	10	$100,000	40	$400,000
Blade server chassis	$3,000	1	$3,000	3	$9,000	5	$15,000	20	$60,000
Blade server mgr.	$600	2	$1,200	6	$3,600	10	$6,000	40	$24,000
Blade server switch	$4,000	2	$8,000	6	$24,000	10	$40,000	40	$160,000
Blade server (2 CPU, disk)	$6,000	14	$84,000	42	$252,000	70	$420,000	280	$1,680,000
Center stage chassis	$8,000	0	$0	0	$0	0	$0	1	$8,000
Center stage shelf mgr.	$1,000	0	$0	0	$0	0	$0	2	$2,000
Center stage blades	$15,000	0	$0	0	$0	0	$0	6	$90,000
Total			$275,200		$825,600		$1,376,000		$5,504,000

Medium Estimate		single chassis pair		multi-chassis 3		multi-chassis 5		multi-chassis 20	
component	unit cost	qty.	cost	qty.	cost	qty.	cost	qty.	cost
ATCA chassis	$6,000	1	$6,000	3	$18,000	5	$30,000	20	$120,000
ATCA shelf mgr.	$800	2	$1,600	6	$4,800	10	$8,000	40	$32,000
NP blade	$9,000	10	$90,000	30	$270,000	50	$450,000	200	$1,800,000
RTM with 10GE IO	$2,000	3	$6,000	9	$18,000	15	$30,000	60	$120,000
Config. logic blade	$8,000	2	$16,000	6	$48,000	10	$80,000	40	$320,000
Switch blade	$8,000	2	$16,000	6	$48,000	10	$80,000	40	$320,000
Blade server chassis	$2,000	1	$2,000	3	$6,000	5	$10,000	20	$40,000
Blade server mgr.	$400	2	$800	6	$2,400	10	$4,000	40	$16,000
Blade server switch	$3,000	2	$6,000	6	$18,000	10	$30,000	40	$120,000
Blade server (2 CPU, disk)	$4,000	14	$56,000	42	$168,000	70	$280,000	280	$1,120,000
Center stage chassis	$6,000	0	$0	0	$0	0	$0	1	$6,000
Center stage shelf mgr.	$800	0	$0	0	$0	0	$0	2	$1,600
Center stage blades	$10,000	0	$0	0	$0	0	$0	6	$60,000
Total			$200,400		$601,200		$1,002,000		$4,008,000

Low Estimate		single chassis pair		multi-chassis 3		multi-chassis 5		multi-chassis 20	
component	unit cost	qty.	cost	qty.	cost	qty.	cost	qty.	cost
ATCA chassis	$4,000	1	$4,000	3	$12,000	5	$20,000	20	$80,000
ATCA shelf mgr.	$600	2	$1,200	6	$3,600	10	$6,000	40	$24,000
NP blade	$6,000	10	$60,000	30	$180,000	50	$300,000	200	$1,200,000
RTM with 10GE IO	$1,500	3	$4,500	9	$13,500	15	$22,500	60	$90,000
Config. logic blade	$6,000	2	$12,000	6	$36,000	10	$60,000	40	$240,000
Switch blade	$6,000	2	$12,000	6	$36,000	10	$60,000	40	$240,000
Blade server chassis	$1,500	1	$1,500	3	$4,500	5	$7,500	20	$30,000
Blade server mgr.	$200	2	$400	6	$1,200	10	$2,000	40	$8,000
Blade server switch	$2,000	2	$4,000	6	$12,000	10	$20,000	40	$80,000
Blade server (2 CPU, disk)	$2,000	14	$28,000	42	$84,000	70	$140,000	280	$560,000
Center stage chassis	$4,000	0	$0	0	$0	0	$0	1	$4,000
Center stage shelf mgr.	$600	0	$0	0	$0	0	$0	2	$1,200
Center stage blades	$8,000	0	$0	0	$0	0	$0	6	$48,000
Total			$127,600		$382,800		$638,000		$2,552,000

Figure 15. Cost Estimates

be expected to fall between $10 million and $15 million dollars. Note however, that in order for a backbone site to terminate ten wavelengths on each of three backbone links it needs to have at least 30 LCs, implying a multichassis-10 configuration. The cost of one such system can be expected to fall between $1.2 million and $2 million.

8. Closing Remarks

The reference design described in this report represents just one of a number of possible system architectures for the GBP. The purpose in putting it forward is not to rule out other possibilities, but to provide one example of a design that is sufficiently specific and detailed that it can serve as a reference point for consideration of alternative approaches. It is quite likely that the proposed design will not serve the needs of all researchers who would like to use GENI. One of the purposes in putting the design on paper is to enable researchers to examine the system in detail, think about how they might use it and identify in what ways it may be deficient. The more feedback that researchers provide to those interested in designing and implementing the GBP, the more likely it is that the resulting system will meet the needs of the largest number of prospective users.

References

[AN05] Anderson, Tom, Larry Peterson Scott Shenker and Jonathan Turner. "Overcoming the Internet Impasse through Virtualization," *IEEE Computer Magazine*, 4/05.

[CH02] Choi, Sumi, John Dehart, Ralph Keller, Fred Kuhns, John Lockwood, Prashanth Pappu, Jyoti Parwatikar W. David Richard, Ed Spitznagel, David Taylor, Jonathan Turner and Ken Wong. "Design of a High Performance Dynamically Extensible Router," *Proceedings of the DARPA Active Networks Conference and Exposition*, 5/02.

[CH03] Chun, Brent, David Culler, Timothy Roscoe, Andy Bavier, Larry Peterson, Mike Wawrzoniak, and Mic Bowman. "PlanetLab: An Overlay Testbed for Broad-Coverage Services," *ACM Computer Communications Review*, vol. 33, no. 3, 7/03.

[FLCM] Fulcrum Microsystems 24-Port 10-Gigabit Ethernet Switch Chip. `http://www.fulcrummicro.com/fm2224.htm`.

[GE06] Global Environment for Network Innovations. http://www.geni.net/, 2006.

[IXP] Intel IXP 2xxx Product Line of Network Processors. `http://www.intel.com/design/network/products/npfamily/ixp2xxx.htm`.

[NLR] National Lambda Rail. `www.nlr.net`.

[NW05] Report of NSF Workshop on Overcoming Barriers to Disruptive Innovation in Networking. `www.arl.wustl.edu/netv/noBarriers_final_report.pdf`, 1/05.

[PCMG] PCI Industrial Computer Manufacturers Group. "AdvancedTCA Specifications for Next Generation Telecommunications Equipment ," available at `http://www.picmg.org/newinitiative.stm.`

[PE02] Peterson, Larry, Tom Anderson, David Culler and Timothy Roscoe. "A Blueprint for Introducing Disruptive Technology into the Internet," *Proceedings of ACM HotNets-I Workshop*, 10/02.

[RA05] Radisys Corporation. "Promentum™ ATCA-7010 Data Sheet," product brief, available at `http://www.radisys.com/files/ATCA-7010_07-1283-01_0505_datasheet.pdf`.

[VRTX] Virtex 5 Multiplatform FPGA. `http://www.xilinx.com/products/silicon_solutions/fpgas/virtex/virtex5/index.htm`.