



Engines for Global Connectivity

communicator

summer 1998

From the President

In developed countries, more than half of the traffic on the PSTN network today is non-voice. This ratio continues to increase. For instance, according to one estimate, in Sweden the non-voice traffic exceeded 50 percent in 1991. In 1996, the non-voice traffic was around 80 percent of all traffic on the network. Similar trends can be discerned in other developed countries.

The drivers behind the non-voice traffic are (a) Internet, (b) On-Line Services, (c) LAN-to-LAN Communication, and (d) some multimedia traffic.

To maintain revenue growth and profitability, the service providers want to migrate all non-voice traffic out of the end-users' private networks to their networks. The service providers want to offer Virtual Private Network Capability

continued on page 2

Programmability in VLSI Devices for Communications

The spring issue of *communicator* carried an article "VLSI Devices: Next Century Technology," which discussed motivation and architectural considerations for programmable devices. This article considers the benefits and challenges of programmability, both to TranSwitch and to our customers. It also outlines the architecture and development environment for the on-chip software which will be referred to as microcode.

As a product, a programmable device is always accompanied by software: microcode, development tools or both. Programmable devices and the associated microcode and development tools fundamentally change the picture for both TranSwitch and our customers, offering unique benefits to all.

Benefits

Customers enjoy considerable benefits even if the customer doesn't program the device. For the customer, these benefits include in-

creased versatility and increased life cycle of both the device and the customer's boards and subsystems that contain it.

When TranSwitch developed its first programmable device, the Quad EI Mapper, one option was to hard-code the microcode in on-chip ROM which takes up much less area than RAM and which would save customers the additional initialization step of loading the microcode. Upon asking, customers strongly favored the RAM option, since the relevant standards were likely to evolve over time, and didn't uniquely specify every field that the QE1M processes.

In the two years since the QE1M was initially released to customers, we have been able to fine-tune features and evolve the product to accommodate standards changes with a small number of microcode releases. Customers can upgrade their system products with the

continued on page 2

In this issue

Benefits of programmable VLSI devices

Product list

From the President...continued

integrating both voice and non-voice traffic very similar to what they have been able to achieve with voice traffic over the last few years.

With the non-voice traffic growth, the assumptions regarding the calling rate and call-holding times, which we all have come to accept, need to be revised. For instance, a study by a major hotel chain revealed that the guests traveling with lap-top PCs, using them for e-mail and Internet access, use the hotel telephone for an average of 90 minutes a night compared to 8 minutes a night of average use for voice services only.

The same phenomena has spread over the public networks of all the developed countries as subscribers are using more than 30 minutes on average to browse the Internet.

Connectivity Engines will provide answers for network flexibility

What is the net result of this? The result is that the service providers are re-thinking the assumptions behind the architecture of their current networks and developing scenarios for evolving this network to meet the new requirements. The equipment suppliers who are willing to participate in this re-thinking process and help their customers -the service providers -to meet the service requirements of this new breed of sophisticated end-users, have the best chance of surviving this massive dislocation in the PSTN market. In fact, it is quite likely that one or two of the data communication systems vendors might dislodge the entrenched telecom systems vendors and emerge as winners in this new game.

To meet the requirements of the new line of equipment needed to re-architect the network, TranSwitch is working feverishly to develop a set of programmable VLSI devices; we call them "Connectivity Engines."

To increase flexibility, the networks of the future will need to be reconfigurable with software. Programmable equipment platforms, which in turn must contain programmable component modules, will be needed to realize this capability. These core programmable modules will form the building blocks of the new flexible network.

As new programmable platforms are deployed across the full spectrum of the information infrastructures, software will be the key to implementing a more versatile and adaptable network. Software will enable these platforms to be configured for transport of any type of information, regardless of whether it takes the form of cells, packets, or continuous bit streams. Changes in standards can easily be accommo-

dated by reprogramming the "Connectivity Engines" within these platforms.

Broad deployment of "Connectivity Engines" should facilitate seamless integration and management of the networks, while maintaining more commonality in both hardware and software. This should drastically reduce the cost of infrastructure upgrades and maintenance. We are excited to be pioneering in a field of such promise! ❖

Santanu Das

Programmability...continued

latest microcode from our website at any time, without having to replace QE1M devices or their hardware.

Another feature of programmability that benefits both the customer and TranSwitch, is the lower risk of implementing complex, lengthy, algorithmic signal processing. For TranSwitch, programmability extends

the time for "getting it right" to beyond fabrication into testing of the silicon device, where issues that are not covered by simulation can arise. For the customer, the reduced risk of silicon respins means more reliable development schedules for fully working systems.

Programmability also allows configuration of a number of products for a single silicon device with different microcode. This can increase the number of applications, or the functions available on a device. Every new function, such as maintenance or management, added to a device, carries several benefits. For the customer, the code and cycle demands on the host (e.g., controller board) gets reduced. For TranSwitch, the microcode is executed on-chip and so is independent of, and thus compatible with, all system platforms.

Finally, customer programmability gives customers an added level of control over the device functions. They can add proprietary features: such as dedicated connections, specialized frame overhead field processing, or special queuing or management functions. The associated development tools will let customers prototype functions on a device, to help them decide how to architect their systems.

Challenges

Programmability also poses some challenges. A customer that does not program the device, still needs to provide for loading the executable microcode into the instruction memory for the device.

Customers who do program a device, will face an additional software development activity. They must design microcode to operate in a high-performance, complex environment. Although a C compiler will be available, designers may still need to use assembly language to 'meet timing and memory constraints. As described in the Microcode Development Environment

section below, TranSwitch plans to provide substantial library and reference design support to minimize the customer programming effort.

For TranSwitch, programmability adds several additional tasks to the development process. First, the architectural design that follows the completion of the functional specification now must include partitioning and interfacing of functions between hardware and microcode -the architecture team increases from a minimum of two members (systems and VLSI) adding a senior software member.

Next, in addition to product microcode, three other sets of software are needed: test microcode to help verify the hardware design, diagnostic microcode to support Alpha testing of the silicon, and optionally, device-specific features for our SPIM simulator. Finally, certification of the device requires testing the static and real-time debug interfaces on the device, plus integration, testing and debugging of the product microcode on the silicon prototypes.

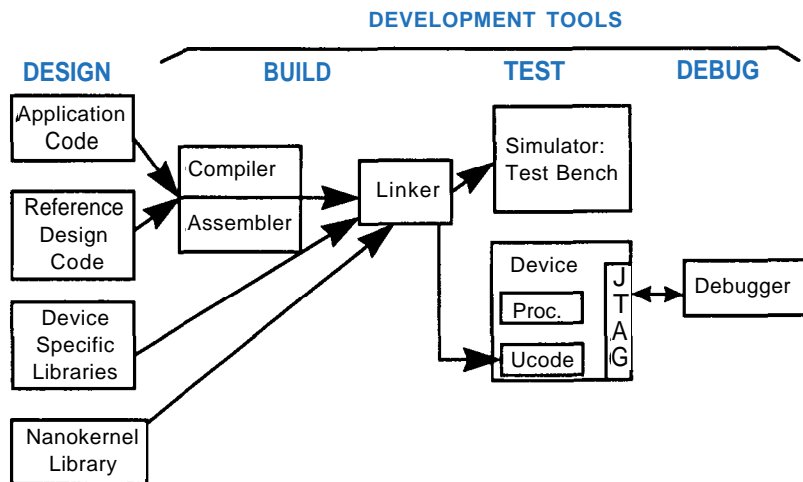
Microcode Development Environment

To meet these challenges, several resources are combined to support microcode development by TranSwitch and ultimately our customers (see figure). The first resource is a set of development tools. A C-compiler, assembler and linker are used to build the executable microcode. During the pre-silicon development phase, two simulators are available for the executable microcode:

- (1) an instruction-level simulator adapted to our processor's instruction set, and used for initial logic debugging, and
- (2) a test bench that exercises the hardware model for the device, and in turn executes the microcode.

When silicon prototypes are available, some are inserted into development boards with associated host

Microcode Development Environment



software; the combination serves a function similar to the test bench. Additional debug hardware and software access the device's scan interface port to provide standard debug functions. Since the same processor core is used in all new device designs, the build, simulation and debugging tools may be applied across all new programmable devices.

The next type of resources are components of the software architecture for the device. The Nanokernel is designed to be simple and compact, and supports non-pre-emptive or cooperative multitasking, in which each task runs to completion before returning to the kernel. Tasks can be foreground where most interrupts are disabled and background which can be interrupted. This provides a structure for both real-time-critical, event-driven and scheduled, non-real-time functions. The Nanokernel services which include task management, scheduling, inter-task communications and timing services, can be configured to meet the application needs of a device with a minimum of overhead.

Supplementing the generic Nanokernel library are device-specific libraries containing co-processor access routines and interrupt service

routines (ISR's). Many of these routines can be adapted from early test microcode used to verify processor and co-processor interfaces in the device VLSI design. The Nanokernel and the device-specific libraries provide working building blocks for the application designer.

The final resource helps bridge the gap between building blocks and a complete, coherent application design. A microcode reference design is derived from product microcode and supports basic features for an application area, plus "hooks" for adding new features. It embodies a set of design trade-offs and a level of testing and documentation that give designers a large head start on their specific designs. Making changes to a working design involves much less effort and uncertainty than doing a complete design from scratch.

The combination of performance-oriented architectures and microcode development resources will make it possible for both TranSwitch and our customers to develop versatile, high-performance, system-level devices and to extend their useful lifetimes with ongoing software enhancements. ❖

List of TranSwitch Products

<u>Product Name</u>	<u>Product Number</u>	<u>Product Description</u>
Asynchronous VLSI Devices		
ART/ARTE	TXC-02020/21	Advanced DS3/STS-1 Line Interface Device
DS3F	TXC-03401	DS3 Framer Device
DS3LIM-SN	TXC-20153	DS3/STS-1 Line Interface Module
E123MUX	TXC-03361	E1/E2/E3 Mux/Demux Device
E3LIM	TXC-20163	E3 Line Interface Module
E2/E3F	TXC-03701	8- 34-Mbit/s Framer Device
HDLC	TXC-05101	HDLC Controller Device
JT2F	TXC-03702	6-Mbit/s Framer Device
M13E	TXC-03303	DS3/DS1 Mux/Demux Device
MRT	TXC-02050	6- 8- 34-Mbit/s Line Interface Device
QDS1F	TXC-03102	Quad DS1 Framer Device
QE1F	TXC-03104	Quad E1 Framer Device
QTIF-Plus	TXC-03103	Quad T1 Framer-Plus Device
XBERT	TXC-06125	Bit Error Rate Generator/Receiver Device
SONET/SDH Synchronous VLSI Devices		
ADMA-EI	TXC-04002	2-Mbit/s to TU-12 Async Mapper-Desync Device
ADMA-T1/T1P	TXC-04001/11	1.544 Mbit/s to VT1.5/TU-11 Async Mapper-Desync Device
DS1MX7	TXC-04201	DS1 Mapper 7-Channel Device
L3M	TXC-03452	SDH/SONET Level 3 Mapper Device
L4M	TXC-03456	SDH/SONET Level 4 Mapper Device
QE1M	TXC-04252	Quad E1 Mapper Device with Microcode
QT1M	TXC-04251	Quad T1 Mapper Device
SOT-1/SOT-1E	TXC-03001/11	SONET STS-1 Overhead Terminator Device
SOT-3	TXC-03003	STM-1/STS-3/STS-3c Overhead Terminator Device
SYN155/155C	TXC-02301/02	155-Mbit/s Synchronizer Device
ATM (Asynchronous Transfer Mode) VLSI Devices		
ALI-25C/25T	TXC-07125/225	ATM 25-Mbit/s Line Interface Controller Device/Transceiver Device
CDB	TXC-05150	ATM Cell Delineation Block Device
CUBIT-Pro	TXC-05802	ATM CellBus Switch Device
SALI-25C	TXC-07625	Six ATM 25-Mbit/s Interface Controllers Device
SARA-R	TXC-05601	ATM/SMDS Reassembly Controller Device
SARA-S	TXC-05501	ATM/SMDS Segmentation Controller Device
SARA-Lite	TXC-05551/L	ATM Segmentation and Reassembly Device with Integrated OC-3 SONET/SDH Frames and AAL0/5 Microcode

Evaluation Boards

Contact TranSwitch Sales

Development Support Boards and Systems

Contact TranSwitch Sales