

# Proof-of-Concept Demonstrates High Level of Density for Radio Network Controller (RNC) Applications

Utilizing Intel® Processor-based AdvancedTCA\* Blades and TietoEnator® RAN Signaling Plane Software

## White Paper

Telecom  
Wireless Infrastructure



# Executive Summary

AdvancedTCA\* architecture, coupled with Intel® processors, offers tremendous performance benefits to the RNC developer. While there are no industry standard benchmarks that can be used to derive overall performance for a Radio Network Controller (RNC), Intel Corporation and TietoEnator AB worked together to develop and test a proof-of-concept RNC utilizing an evolved, completely modular system architecture based on open hardware and software standards. This process led to validation of both performance and interoperability. Both companies provided modular building blocks to which system manufacturers can install additional software to create entire systems that may be qualified and brought to market.

This proof-of-concept RNC, based on Intel processors, is designed with three basic types of AdvancedTCA blades: Gigabit Ethernet backplane switches, signaling blades based on Intel processors, and user plane blades based on Intel® IXP2805 network processors. Mezzanine cards are then fit to the user plane blades to host line interface, cryptographic, and management middleware functionality, as required. Both user plane and control/signaling plane software is completely modular with well-defined, high-performance application program interfaces (APIs) between all modules and between the user, control, and signaling planes themselves. System capabilities, therefore, can be expanded by simply adding additional user and/or control plane blades and software.

Proof-of-concept systems help developers validate solutions, outline design strategies and eventually lead to innovative and deployable products. Testing of the combined user and signaling planes verified that approximately six Intel® Pentium® M processor-based signaling plane boards will support a 500K subscriber RNC, while additional blades increase capacity by a scaling factor of 0.7 per blade. Additional blades can also contribute to a system's high availability and redundancy.

This paper focuses on the system architecture and performance measurements of this proof-of-concept, Intel®-based RNC signaling plane sub-system and the transport/user and signaling planes. Although neither a full RNC application (sometimes termed "Radio Resource Control" [RRC]) nor complete Operations, Administration and Maintenance (OA&M) functionality was developed and tested, it is clear from this study that a very large RNC can be built using only a single rack of three AdvancedTCA chassis (36 blade slots, excluding six for backplane switches) and that a 500K subscriber RNC would require only 20 blades for user and signaling planes. The RNC application itself would require additional blades. Modeling and anecdotal customer information indicates that control plane blades, based on Dual-Core Intel® Xeon® processors LV 5138, could allow a large RNC to be built using far fewer signaling and application plane blades.

## Table of Contents

Executive Summary . . . . .	2
System Architecture for Intel® Processor-based RNC . . . . .	3
Performance Testing Details . . . . .	6
Results for Combined User and Signaling Plane Tests . . . . .	6
Dimensioning the Signaling Plane . . . . .	7
Projections for Intel® Core Microarchitecture-based Processors . . . . .	7
References . . . . .	8

# System Architecture for Intel® Processor-based RNC

## System Architecture

**Figure 1** shows the high-level architecture and data flow within the reference Universal Mobile Telecommunications System Release 99/Release 5 (UMTS R99/R5) RNC (the Iur line card was omitted from this diagram because it was not used during the control plane-only and combined tests). The upper portion of the figure shows the control plane (signaling) software, which would normally be horizontally distributed across multiple AdvancedTCA single board computers (SBCs) based on Intel processors. Intel® network processors, used in the line cards and radio network layer (RNL) blades, are shown in the lower part of the diagram. User plane software runs in the microengines of the network processors, while interfaces to management middleware and the control plane sub-systems utilize the Intel XScale® technology to run in each network processor. Line cards are connected to down-stream and up-stream equipment. RNL cards do not have external interfaces. Modeling has previously shown that the dedicated RNL (RNL-d) card presents the most challenging processing load because the radio link control (RLC) software and Kasumi ciphering algorithms run on that card. The Intel-based user plane RNL-d cards use an FPGA to perform the

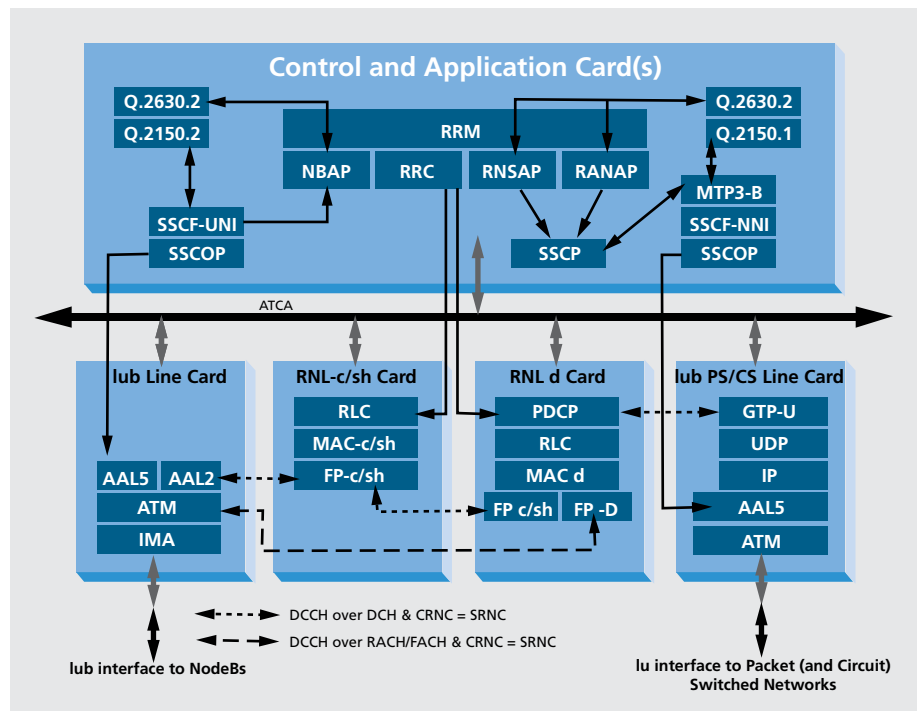
actual ciphering, which frees up several microengines to handle the heavy-weight RLC, media access control and frame protocol algorithms.

## Signaling Software Architecture

Modularity is the key to RNC signaling software for scalability and cost effectiveness. For this reason, the TietoEnator® signaling product allows protocols to be combined into almost any type of signaling stack. This means the signaling product can be easily ported to different hardware environments because all hardware dependencies are specified in a single file that is linked in when signaling modules are compiled.

Protocol modules are optimized for message processing through the protocol, which means the internal design of the modules varies depending on specific protocol requirements. Most modules, however, implement a number of state/event machines that handle the protocol-specific tasks and are designed to facilitate protocol management. The only dependencies between modules are those enforced by the standard specifications themselves.

**Figure 1:** Functional mapping and data flow within the proof-of-concept RNC based on Intel® processors



There are strict APIs between the protocol modules and, as much as possible, these APIs are kept very thin and without logic. An API can be used to access any of the protocol modules in the stack at the same time that normal signaling traffic is running over the same API.

This highly modular software architecture optimizes system manageability and also provides a number of support modules for configuration, alarms and other Operations and Maintenance (O&M) tasks. O&M is separated from the protocol modules because it is often one of the major adaptations needed when integrating the signaling product into a customer platform. Although the TietoEnator software architecture permits specific O&M additions or adaptations to the protocol modules themselves, additions or changes are seldom required. Usually the required O&M functionality is already provided for in the configuration options for the protocol modules.

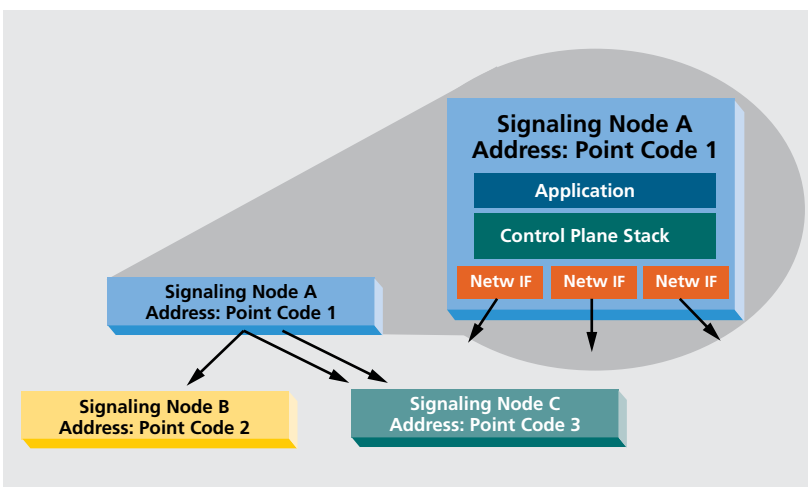
### Monolithic and Distributed Software Architectures

TietoEnator chose to design software for horizontal distribution across a number of blades, or a number of cores within a single processor. A traditional, non-distributed architecture is simpler than a distributed architecture because messages are only pushed up or down a single stack instance. However, non-distributed architectures typically do not scale well and active redundancy schemes can be less robust. Horizontally distributed signaling software architecture, on the other hand, offers high-end scalable performance and an N-way active redundancy scheme. A distributed architecture can also be ported into processor environments that do not support threads.

**Figure 2** shows a theoretical node in a telecom network, implemented in a non-distributed architecture. The node runs a service logic application and a control plane signaling stack that has signaling paths (links) to the rest of the network. In this type of non-distributed signaling stack architecture, the entire node has a single network address; all other nodes communicate with it using that single address or “point code”. The control plane stack handles all dynamics in the network in real-time, since any event may or may not have consequences for routing or handling of common resources.

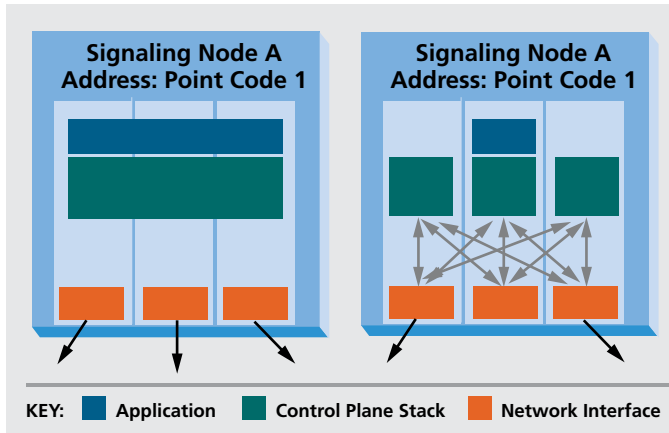
Such individual, monolithic signaling stack architectures can have higher performance per unit, but sacrifice flexibility and extensibility. **Figure 3** shows the same theoretical network node implemented in a blade system with a horizontally distributed signaling architecture. The logical architecture is shown on the left, while the diagram on the right shows how functions could be allocated to physical blades.

The horizontal distribution solution allows the control plane signaling stack to be distributed over many different blades, each of which can handle the complete signaling stack logic. Dynamic network-related traffic is distributed across all blades, whereas application traffic is routed only to the specific blade handling the application dialog services. New, incoming application-related traffic and non-dialog mode application traffic can be routed over the different blades in a number of different schemes. Each scheme is inherently more complex than a non-distributed design, and is more flexible since control plane signaling performance can be scaled to match traffic requirements simply by adding more blades. Because all blades have failover capability, it also provides a natural redundancy scheme.



**Figure 2: Non-distributed signaling software architecture**

**Figure 3:** Horizontally distributed signaling architecture



**Figure 4** shows horizontally distributed signaling and RNC application processing as it would be implemented in a production RNC. Multiple blades handle signaling (middle boxes) and the RNC application (upper boxes). There are also multiple user plane cards of each type.

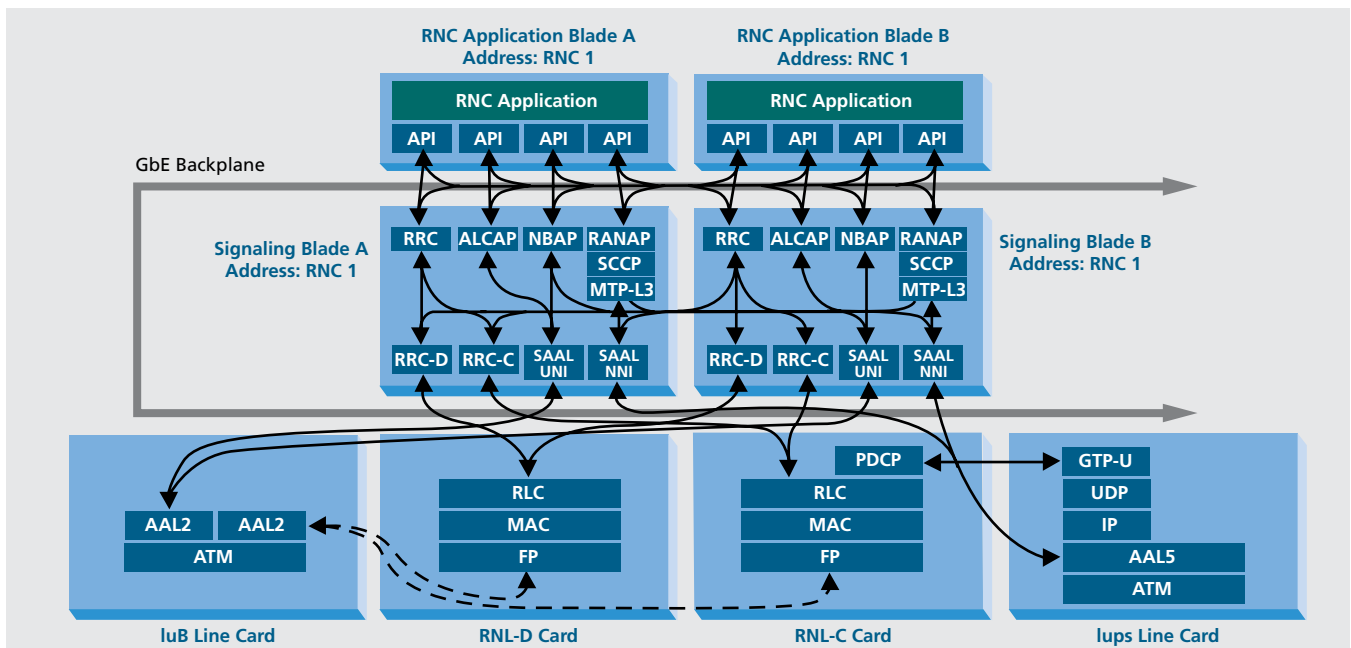
Note that performance testing of the proof-of-concept RNC only required that a single signaling blade be tested, as long as the software/hardware architecture allowed for scaling beyond the capabilities of one card. Therefore, the actual test RNC used a blade with a software load consisting of Linux\*, the signaling stacks and parts of the RNC application. TietoEnator signaling software, however, has been designed for horizontal distribution, and is currently deployed in this way within production systems from major equipment manufacturers.

### Common Modules and Platform-Specific Adaptations

The TietoEnator signaling product is developed and tested for conformance to the protocol specifications in an open hardware platform environment. This means the base signaling solution code is developed in isolation from any adaptations for specific customers or specific hardware. The product's general releases are also handled under a generic, open hardware, customer-agnostic platform model. Using a common software base is both a requirement from customers and a natural step, because versioning control would be extremely difficult if a separate version of each protocol had to be developed and released for each customer's specific hardware platform. TietoEnator's common software base strategy also means that the cost to develop and maintain the basic protocols is shared among all customers, so costs for individual customers can be reduced.

Platform adaptation projects are then used to tailor stacks for each customer's specific hardware environment and to add any required customization. The major platform-specific work is usually to add an O&M solution that fits with the rest of the manufacturer's platforms. Platform adaptation work may also include enabling or disabling different configuration options available in the generalized protocol modules. In some cases a different organization within TietoEnator handles the platform-specific project. This enables TietoEnator to preserve customer-specific intellectual property.

**Figure 4:** Horizontally distributed signaling implementation in the proof-of-concept RNC based on Intel® processors



# Performance Testing Details

Complete details of the full range of performance testing on the Intel-based, proof-of-concept RNC can be found in an article published by Embedded Computing Design titled *Benchmarking Wireless Telecommunications Infrastructure Equipment* (<http://www.embedded-computing.com/articles/carlston2/>). The article includes complete hardware and software configuration for

the RNC, the traffic model used during the testing phase, and a description of how the user plane and signaling plane sub-systems were tested separately. It also describes how the signaling plane was integrated with the user/transport planes, and provides overall system performance figures. The balance of this paper summarizes the results of the combined user and signaling tests detailed in the article.

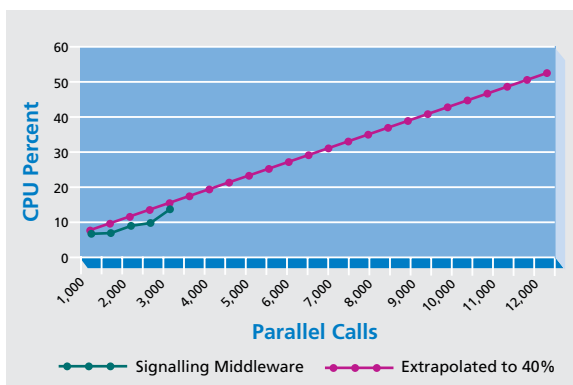
## Results for Combined User and Signaling Plane Tests

Measurements showed that the user plane, while running, had almost no effect on control plane performance. This is largely because the user and signaling planes utilize completely separate hardware and software modules, with clear separation between the planes. Another factor was the high backplane capacity of the proof-of-concept RNC, as noted in the article referenced above.

**Figure 5** shows the number of parallel calls the TietoEnator reference signaling software can handle at various processor loads. Measured data is shown along with extrapolated data. The data indicates that a single Intel Pentium M processor-based blade can process about 9,250 parallel calls at a 40% processor load.

The totals in Figure 5, however, must be de-rated by 50% due to the assumption that about half of the control messages actually present in an operational RNC were not implemented in the load generators. For example, none of the soft or softer handoff control messages, or those between RNCs on the Iur Interface, were implemented. The cell/URA location area update and paging control sequences also were not implemented. All of these sequences are controlled by the RNC application software known as radio resource management (RRM), which was not implemented. Implementing and testing sequences would not have indicated more about the signaling plane's performance since RRM is very processing-intensive and is typically run on dedicated blades, rather than on the signaling blades.

**Figure 5: Extrapolated control plane results**



De-rating 9,250 parallel calls by 50% led to the conclusion that a single Intel Pentium M processor-based blade running at 40% processor load can realistically handle 4,125 parallel calls. This equals a Busy Hour Call Attempt (BHCA) of approximately 109K or 34 calls per second.

The number of Erlangs per blade would be approximately 4,216, using call setup and release times of three seconds and one second, respectively. The Embedded Computing Design article details how Erlangs-per-signaling blade were derived.

# Dimensioning the Signaling Plane

As mentioned, TietoEnator’s signaling software is designed to be distributed across a number of blades. TietoEnator uses a software scaling factor of 0.7 to calculate how processing capacity scales as more signaling boards are added to a system.

**Table 1** shows the estimated dimensioning matrix for a set of six (half of an AdvancedTCA shelf) Intel Pentium M processor-based signaling boards. The table indicates that six signaling boards would support about 18,560 simultaneous calls.

The calculation to the right assumes a BHCA per subscriber of one, which means the six-slot signaling plane is configured to handle the contingency of each subscriber making one call during the busiest hour of the day. The traffic model assumes the calls will be about 30% packet switched, 70% circuit switched, and that the average duration of a voice and data call is 90 and 120 seconds, respectively.

Total Boards in System	Board 1	Board 2	Board 3	Board 4	Board 5	Board 6	Total Parallel Calls
0.7 Software Scaling Factor							
1	4125						4125
2	4125	2887					7012
3	4125	2887	2887				9899
4	4125	2887	2887	2887			12786
5	4125	2887	2887	2887	2887		15673
6	4125	2887	2887	2887	2887	2887	18560

**Table 1: Scaling the signaling plane: estimated performance of Intel® Pentium® M processor-based signaling plane blades with a 40% processor load**

Calculating the number of subscribers, rather than parallel calls, is normally done using the following formula:

$$\text{Total Subscribers} = (\text{seconds/hour} \times \text{active sessions}) / (\text{activity period} \times \text{BHCA per subscriber})$$

Therefore, the total number of subscribers that can be supported by six boards is captured below:

$$\text{Total Subscribers} = (3600 \times 18,560 / 120 \times 1) = 556,800 \text{ circuit switched} + \text{packet switched users}$$

## Projections for Intel® Core™ Microarchitecture-based Processors

Anecdotal information from customers indicates that the performance of protocols quite similar to UMTS signaling approximately doubles when moving from single board computers containing one Intel Pentium M processor to those containing two Dual-Core Intel Xeon processors LV 5138, even without software changes to take advantage of the second core.

Several enhancements to the Intel® Core™ microarchitecture appear to be involved. These include a super-scalar pipeline that can execute four instructions simultaneously (versus three with the Intel Pentium M processor), 4MB of shared L2 cache,

and improved memory prefetch and disambiguation algorithms. If these indications are validated for UMTS signaling stacks, single-threaded applications could perform at rates greater than three times those reported in this article if they were re-written to utilize core affinity or the SMP capabilities of Linux on two dual-core processors. In all likelihood, therefore, multiple processing cores in the Dual-Core Intel Xeon processors will enable truly significant improvement in performance density for this type of intensive signaling plane processing.

# References

## **AdvancedTCA:**

<http://www.picmg.org/newinitiative.stm>

<http://www.intel.com/go/atca>

## **Embedded Computing Design, Benchmarking Wireless Telecommunications Infrastructure Equipment**

<http://www.embedded-computing.com/articles/carlston2/>

## **Optimal RNC Data Plane Configuration for Hosting 500K Subscribers**

<http://www.intel.com/design/telecom/applnots/9679.htm>

## **Intel NetStructure® IXB2805 3G Boards:**

<http://www.intel.com/design/telecom/products/boards/rnc/8304/overview.htm>

## **Intel® Computer Boards and Platforms for AdvancedTCA**

<http://www.intel.com/design/network/products/cbp/atca/index.htm>

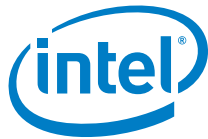
## **Networking and Communications Processors**

<http://www.intel.com/design/network/products/processors.htm?iid=nc+processors>

## **TietoEnator® Signaling Solutions**

<http://www.tietoenator.com/default.asp?path=1;93;16080;124;16837;17193;16242>

<http://www.signaling.tietoenator.com>



## **[www.intel.com](http://www.intel.com)**

Performance tests and ratings are measured using specific computer systems and/or components and reflect the approximate performance of Intel® products as measured by those tests. Any difference in system hardware or software design or configuration may affect actual performance. Buyers should consult other sources of information to evaluate the performance of systems or components they are considering purchasing. For more information on performance tests and on the performance of Intel products, visit Intel Performance Benchmark Limitations (<http://www.intel.com/performance/resources/limits.htm>).

Results have been simulated and are provided for informational purposes only. Any difference in system hardware or software design or configuration may affect actual performance.

Intel does not control or audit the design or implementation of third party benchmarks or websites referenced in this document. Intel encourages all of its customers to visit the referenced websites or others where similar performance benchmarks are reported and confirm whether the referenced benchmarks are accurate and reflect performance of systems available for purchase.

Copyright © 2007 TietoEnator R&D Services AB

Copyright © 2007 Intel Corporation. All rights reserved.

Intel, the Intel logo, Intel NetStructure, Pentium, Xeon, Intel Core and Intel XScale are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

TietoEnator and the TietoEnator logo are trademarks or registered trademarks of TietoEnator AB.

\*Other names and brands may be claimed as the property of others.