

## QNX 4.25 Evaluation Executive Summary

During the summer of 1998, Real-Time Consult officially started an RTOS evaluation program. First, Windows NT and the real-time extensions to Windows NT were studied. The evaluation report on QNX 4.25 from QNX Software Systems Ltd. was part of this evaluation program.

These evaluation reports, as well as comparison reports highlighting decision-critical information, are available on our Web site at <http://shop.realinter.net/rtshop/>.

### INTRODUCTION

During the summer of 1998, Real-Time Consult officially started an RTOS evaluation program. First, Windows NT and the real-time extensions to Windows NT were studied. Evaluation reports for the following products are currently available:

- RTX 4.2 from VenturCom, Inc.
- INtime 1.20 from Radisys Corporation Ltd.
- Hyperkernel 4.3 from Imagination Systems, Inc.
- VxWorks/x86 5.3.1 from WindRiver Systems Inc.
- pSOSystem/x86 2.2.6 from Integrated Systems Inc.
- QNX 4.25 from QNX Software Systems Ltd.

These evaluation reports, as well as comparison reports highlighting decision-critical information, are available on our Web site.

### ARCHITECTURE

QNX4.25 has a client-server architecture consisting of a lean microkernel that implements only core services and optional cooperating processes. The microkernel itself is never scheduled. Its code is executed only as the result of a kernel call, the occurrence of a hardware interrupt or a processor exception.

QNX 4.25 is a message-based operating system. Message passing is the fundamental means of Inter-Process Communication (IPC). The message passing service is based on the client-server model: the client (e.g. an application process) sends a message to a

server (e.g. device manager), which replies with the result.

A client-server architecture has many advantages, of which robustness is one. Each manager (except for the process manager) and device driver runs in its own address space, resulting in a robust, reliable system. The price paid for this is performance: execution of system calls require a few context switches (with an overhead produced by memory protection), resulting in somewhat lower performance.

Due to its architecture and the deep integration of message passing and network messaging, QNX qualifies as a fully distributed operating system.

#### Tasks

QNX 4.25 is a multi-process system. QNX 4.25 does have threads, but they are implemented in an unconventional way that is substantially different from POSIX threads.

A QNX 4.25 thread behaves more like a child process spawned by a parent process than like an actual thread. When a QNX thread is created by a process, the thread will use the same code and data segment as the process, as is the case with conventional threads. However, certain objects like timers and file handles created by the parent process cannot be accessed by the thread.

#### Memory

In QNX 4.25, every process has its own virtual memo-

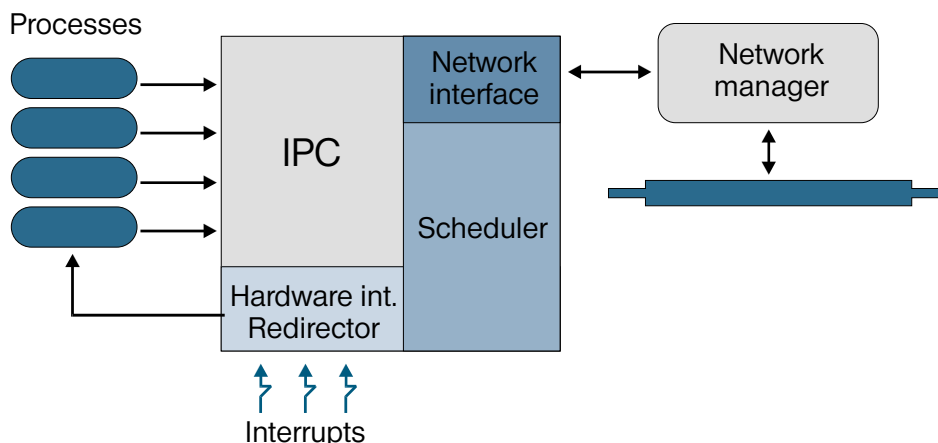


Figure 1. QNX 4.25 Microkernel Architecture.

# EXECUTIVE SUMMARY

QNX 4.25	
Model	Processes only
Priority levels	32 (0->31)
Max. number of tasks	2,000 (default is 500)
Scheduling policies	<ul style="list-style-type: none"> <li>• Prioritized FIFO scheduling</li> <li>• Round-robin scheduling</li> <li>• Adaptive scheduling</li> </ul>
Number of documented states	9

Table 1. QNX 4.25 Task Handling Properties.

ry, code and data segment, and consequently its own Local Descriptor Table (LDT). This virtual memory is provided by the paging mechanism in the Intel processor (the processor is used in protected mode).

Every process has its own code and data segment, so when a process is deleted, these segments will be deleted too. Hence it is of the utmost importance that fixed-size segments are used. If variable-size segments were used, memory space would become fragmented due to the constant creation and deletion of variable-size memory blocks.

## 2.3 Interrupts

Interrupts are not disabled during the execution of an interrupt handler, so the processor is always able to receive the interrupt signal from the PIC (Programmable Interrupt Controller). However, inter-

QNX 4.25	
MMU	Required
Physical page size	4KB
Paging/Swapping	Yes/No
Memory protection models	Each process runs in its own virtual address space.
Virtual memory	Yes

Table 2. QNX 4.25 Memory Management Properties.

rupts with the same or lower priority remain pending in the PIC for however long interrupt handler execution takes. This can only be pre-empted by interrupt handlers from a higher priority interrupt source.

Interrupt-to-task communication is limited. Only objects

QNX 4.25	
Handling	Nested, prioritized
Interrupt-to-task communication	Limited. Only proxies.
Stack	Kernel stack
Context	Context of the process that attached the handler
Minimum RAM	?

Table 3. QNX 4.25 Interrupt Handling Properties.

called proxies can be used. In situations where a thread has to be scheduled from an interrupt handler, these proxies are less powerful than semaphores.

## API Richness

To assess API richness, we created a list of features for the most common system calls and compared it with the available system calls in QNX 4.25. Table 4 gives an overview of all the categories and the scores (in percentage points) that were obtained. For a breakdown of the categories into individual features and system calls, readers should refer to the evaluation report.

This table should not be misconstrued. QNX 4.25 has system calls that are not in our list, and therefore do not feature in Table 4.

An average percentage of 40 % was obtained. This average percentage does not include any weighting factors, it is simply the mean of the score in each category. Table 4 shows that the QNX 4.25 API is lacking

Mechanism	Richness
Thread Management	50%
Clock	86%
Interval Timer	83%
Fixed block size memory partition	0%
Non-fixed block size memory pool	55%
Interrupt Handling	75%
Counting Semaphore	60%
Binary Semaphore	0%
Mutex	0%
Conditional Variable	0%
Event Flags	0%
POSIX Signals	89%
Message Queue	63%
Mailbox	0%
<b>AVERAGE PERCENTAGE</b>	<b>40%</b>

Table 4. API Richness.

some basic primitives like mutexes and event flags.

pSOSystem/x86 2.2.6 from Integrated Systems Inc., for example, scored an average percentage of 44%.

## PERFORMANCE TESTS

### Interrupt latencies

For this test, we measured two latencies:

- Interrupt Latency (task to interrupt handler): The time that elapsed between the execution of the last instruction in the interrupted thread and the first instruction in the interrupt handler.
- Interrupt Dispatch Latency (interrupt handler to task): The time needed to move from the last instruction in the interrupt handler to the next task scheduled

# EXECUTIVE SUMMARY

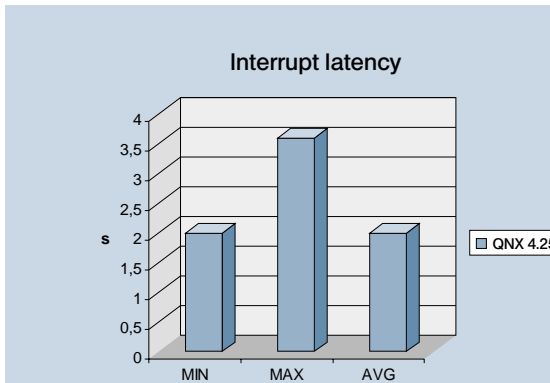


Figure 2. Interrupt Latency - QNX 4.25.

to run.

Figures 2 and 3 display the minimum, maximum and average values for both interrupt latency and interrupt dispatch latency.

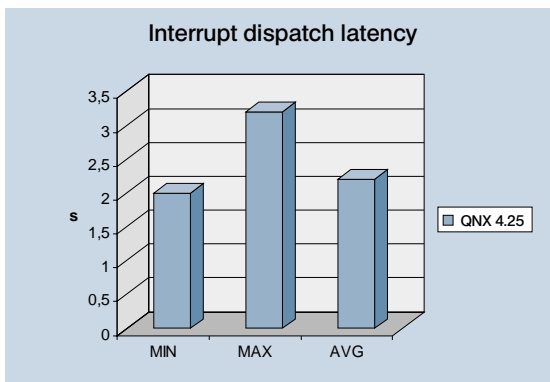


Figure 3. Interrupt Dispatch Latency - QNX 4.25.

## Priority inheritance

The priority inheritance test is usually executed using a mutex. QNX 4.25 however does not provide a mutex, although one can be simulated with threads (or processes) by using message-passing primitives. Due to the client-driven priority mechanism used in message passing, a priority inversion safe mutex can be simulated. For more details on how this test is implemented and executed, readers should refer to the full evaluation report.

In this test, we measured the time it takes for the highest priority thread to acquire the simulated mutex. This

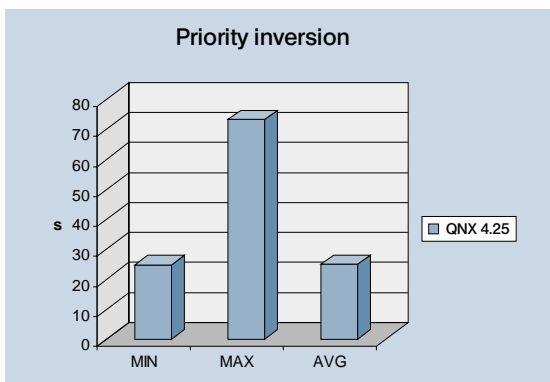


Figure 4. Priority Inheritance.

period includes the time it takes to boost the priority of the lowest-priority thread, have it release the mutex, and switch back to the highest-priority thread so it can acquire the mutex.

The test results are shown in Figure 4. Slower results than those for competing products that use mutex objects were obtained. Simulating a mutex with a thread is slower because a context switch is required every time the mutex is manipulated.

## TOOLS & DEVELOPMENT METHOD

QNX 4.25 uses the Watcom tool chain, which contains the most commonly-used tools. However these tools are not integrated in a single IDE (Integrated Development Environment) encompassing the whole development process. Tools are only available from the command line.

QNX originally used the host = target approach only (i.e., host and target are the same machine). The advantage of this approach is that the user has the option to do it all on one machine: the application can be tested on the same machine as it was developed on, debugging can be done locally, etc.

Cross-development is also supported now, making it possible for the developer to compile and debug code from a MS Windows-based host. Cross-development was not tested during our evaluation.

## DOCUMENTATION & SUPPORT

The documentation is clear and easy to read, but sometimes lacks in-depth information. The description of API calls is inadequate in some places.

The quality of the technical support we received was mediocre. We contacted technical support various times via e-mail, and on several occasions were disappointed with both the time it took to respond and the content of the responses.

## CONCLUSION

QNX 4.25 has a modern client-server architecture and is completely message-based. The system is robust. QNX also qualifies as a fault-tolerant, fully-distributed operating system.

QNX 4.25 is a multi-process system, and there are no conventional POSIX threads. QNX does have objects called threads, but they have an unusual implementation and behave more like processes. The documentation is unclear on the specifications for these threads. QNX 4.25 does not provide mutexes. Priority-inversion safe mutexes can be simulated by means of threads and message passing services, but at the expense of performance and flexibility. Mutex objects should be provided.

The most commonly-used tools are provided but are not integrated into one IDE. Tools are available from the command line instead.

## OTHER PUBLICATIONS AND SERVICES

As mentioned at the outset, evaluation reports for Windows NT 4.0, RTX 4.2, Hyperkernel 4.3 and INtime

1.20 have been on sale from our Web site since the beginning of 1999.

Real-Time Consult has also evaluated VxWorks/x86 5.3.1, QNX 4.25 and pSOSystem/x86 2.2.6. Evaluation reports for these products have been available since April 20, 1999.

The evaluation reports are intended for anyone who is somehow or other involved with dedicated systems technology. This obviously includes systems design engineers and application developers, who need to have a detailed understanding of how the product behaves in a real-time environment, but the intended audience also includes managers and project leaders who need to make strategic decisions like which RTOS to use, and need to ascertain how it will affect project execution overall.

Finally, Real-Time Consult also performs feasibility studies and product validations at the request of customers. Please contact our offices for additional information ■

---

*Dr. Martin Timmerman has a degree in Telecommunications Engineering from the Royal Military Academy (RMA) Brussels and received a Doctorate in Applied Science from the Gent State University (1982) in Belgium. In 1983 he transferred to Computer Engineering and set up the System Development Centre (SDC) at RMA. He gives general courses on Computer Platforms and more specific courses on System Development Methodologies. He is a consultant to the Joint Staff of the Belgian Armed Forces in areas concerning Information System Methodologies and CASE tools and he is the Belgian representative in some NATO technical commissions. Outside the RMA, Martin is known for his audits, reviews and seminars, and for his company Dedicated Systems Experts, where he makes use of his considerable knowledge of the Dedicated Systems world. Dedicated Systems Experts is the publishing house responsible for Dedicated Systems Magazine, an International magazine about Dedicated Systems development. Dedicated Systems Experts provides hardware and software support services and is involved in project engineering for Dedicated Systems.*

*Bart Van Beneden has been with Dedicated Systems Experts since 1998 where he is involved in the RTOS evaluation program of Dedicated Systems Magazine as a project manager. He received his degree in computer science at the Free University of Brussels. Before joining Dedicated Systems Experts, he designed multimedia applications at LaserMedia Inc.*



## Dedicated Systems Magazine

### EDITORIAL CALENDAR

- 1 Q00 NETWORKS  
(incl. RTOS Section)
- 2 Q00 INSTRUMENTATION & AUTOMATION
- 3 Q00 BUS TECHNOLOGY
- 4 Q00 EMBEDDED INTERNET

previews on the web at:  
<http://www.dedicated-systems.com>

### ADVERTISEMENT INDEX

<b>QNX SOFTWARE SYSTEMS</b>	<b>2</b>
<b>MPL</b>	<b>17</b>
<b>POLYHEDRA</b>	<b>29</b>
<b>ESMERTEC</b>	<b>33</b>
<b>FORCE COMPUTERS</b>	<b>43</b>
<b>IMAGE MEDIA</b>	<b>47</b>
<b>PHARLAP</b>	<b>55</b>
<b>VMIC</b>	<b>69</b>
<b>WORLD COMPUTER LTD.</b>	<b>73</b>
<b>NATIONAL INSTRUMENTS</b>	<b>85</b>
<b>ACT EUROPE</b>	<b>99</b>
<b>VMETRO</b>	<b>100</b>

For more information on advertisement opportunities, call 32-2-520.55.77 or send an e-mail to [Info@dedicated-systems.com](mailto:Info@dedicated-systems.com)