QNX® NEUTRINO® RTOS V6.3

Disclaimer

Although all care has been taken to obtain correct information and accurate test results, Dedicated Systems Experts and Dedicated Systems Magazine cannot be liable for any incidental or consequential damages (including damages for loss of business, profits or the like) arising out of the use of the information provided in this report, even if Dedicated Systems Experts and Dedicated Systems Magazine have been advised of the possibility of such damages.

http://www.dedicated-systems.com

E-mail: info@dedicated-systems.com

EVALUATION REPORT LICENSE

This is a legal agreement between you (the downloader of this document) and/or your company and the company DEDICATED SYSTEMS EXPERTS NV, Bergensesteenweg 421 B12, B-1600 St-Pieters-Leeuw, Belgium.
It is not possible to download this document without registering and accepting this agreement on-line.

1. **GRANT**: Subject to the provisions contained herein, Dedicated Systems Experts hereby grants you a non-exclusive license to use its accompanying proprietary evaluation report for projects where you or your company are involved as major contractor or subcontractor. You are not entitled to support or telephone assistance in connection with this license.
2. **PRODUCT**: Dedicated Systems Experts shall furnish the evaluation report to you electronically via Internet. This license does not grant you any right to any enhancement or update to the document.
3. **TITLE**: Title, ownership rights, and intellectual property rights in and to the document shall remain in Dedicated Systems Experts and/or its suppliers or evaluated product manufacturers. The copyright laws of Belgium and all international copyright treaties protect the documents.
4. **CONTENT**: Title, ownership rights, and an intellectual property right in and to the content accessed through the document is the property of the applicable content owner and may be protected by applicable copyright or other law. This License gives you no rights to such content.
5. **YOU CAN NOT**:
   - You can not, make (or allow anyone else make) copies, whether digital, printed, photographic or others, except for backup reasons. The number of copies should be limited to 2. The copies should be exact replicates of the original (in paper or electronic format) with all copyright notices and logos.
   - You can not, place (or allow anyone else place) the evaluation report on an electronic board or other form of on line service without authorisation.
6. **INDEMNIFICATION**: You agree to indemnify and hold harmless Dedicated Systems Experts against any damages or liability of any kind arising from any use of this product other than the permitted uses specified in this agreement.
7. **DISCLAIMER OF WARRANTY**: All documents published by Dedicated Systems Experts on the World Wide Web Server or by any other means are provided "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT. This disclaimer of warranty constitutes an essential part of the agreement.
8. **LIMITATION OF LIABILITY**: Neither Dedicated Systems Experts nor any of its directors, employees, partners or agents shall, under any circumstances, be liable to any person for any special, incidental, indirect or consequential damages, including, without limitation, damages resulting from use of OR RELIANCE ON the INFORMATION presented, loss of profits or revenues or costs of replacement goods, even if informed in advance of the possibility of such damages.
9. **ACCURACY OF INFORMATION**: Every effort has been made to ensure the accuracy of the information presented herein. However Dedicated Systems Experts assumes no responsibility for the accuracy of the information. Product information is subject to change without notice. Changes, if any, will be incorporated in new editions of these publications. Dedicated Systems Experts may make improvements and/or changes in the products and/or the programs described in these publications at any time without notice. Mention of non-Dedicated Systems Experts products or services is for information purposes only and constitutes neither an endorsement nor a recommendation.
10. **JURISDICTION**: In case of any problems, the court of BRUSSELS-BELGIUM will have exclusive jurisdiction.

Agreed by downloading the document via the internet.

http://www.dedicated-systems.com
email: info@dedicated-systems.com

DOCUMENT CHANGE LOG

Issue No.	Revised Issue Date	Para's / Pages Affected	Reason for Change
1.00	November 17, 2004	All	Initial Issue

1 概要

1.1 目的と評価範囲

この文書は、QNX Neutrino v6.3 オペテーティング システムの品質に関する評価結果をまとめたものである。ウェブサイトではまた、同オペレーティング システムの PPC プラットフォームにおけるテスト結果も掲載している。 [Doc. 5].

この報告書のレイアウトと内容については、「The evaluation test report definition （評価テスト報告の定義）」 [Doc. 3] および 「The OS evaluation template （OS 評価テンプレート）」 [Doc. 4]. に従っている。これらの文書の詳細については、セクション 1.3 を参照のこと。したがって、これらの文書は、本報告書の一部を成すものである。

これら文書は緊密に関わり合っているため、「OS 評価テンプレート」のフレームワーク バージョンと、本評価報告のフレームワーク バージョン（2.9）は合致する必要がある。文書のバージョン、テスト、および両方の関係についての詳細については、本書セクション 1.3 内の「The evaluation framework （評価のフレームワーク）」 [Doc. 1] を参照のこと。

1.2 文書バージョン：2.9 フレームワーク

この文書は、評価フレームワーク 2.9 の適用範囲における結果を示している。

1.3 関連する文書

以下、本報告書と関連の深い文書を上げる。これらの文書は、以下のリンクからダウンロードすることができる。
http://www.dedicated-systems.com/encyc/buyersguide/rtos/evaluations

Doc. 1 The evaluation framework （評価のフレームワーク）
この文書は、評価フレームワークについて述べている。また、どの文書が入手可能であるか記載し、命名規則、番号とバージョンの関連についても説明している。この文書は、評価フレームワークのベースとなる文書である。
EVA-2.9-GEN-01 Issue: 1 Date: April 19, 2004

Doc. 2 What is a good RTOS? （優れた RTOS とは何か？）
この文書は、Dedicated Systems Experts がオペレーティング システムを「リアルタイム」と呼ぶ際に使用する基準について述べている。評価テストは、この文書に定義する基準に基づき行われる。
EVA-2.9-GEN-02 Issue: TBD Date: TBD

Doc. 3 The evaluation test report definition （評価テスト報告の定義）
この文書には、本報告書において行われた各種テストおよび各テストのフローチャート、汎用擬似コードを記載している。テスト ラベルは、すべてこの文書で定義されている。
EVA-2.9-GEN-03 Issue: 1 Date: April 19,2004

Doc. 4 The OS evaluation template （OS 評価テンプレート）
この文書には、一定のフレームワークにおけるすべての報告書で使用されるレイアウトが記載されている。
EVA-2.9-GEN-04 Issue: 1 Date: April 19, 2004

Doc. 5 OSE 4.5.1 on a PPC platform （PPC プラットフォームにおける OSE 4.5.1）

EVA-2.9-OS-OSE-PPC-01 Issue: 1 Date: April 19, 2004

2 結果の要約

2.1 評価された製品

QNX Software Systems Ltd による QNX NEUTRINO RTOS v6.3.0

2.2 結論

2.2.1 ポジティブな点

- 堅牢な分散システムのための優れたアーキテクチャ
- 非常に高速で確実に予測できるパフォーマンス
- すべてのドライバ、ボードサポート パッケージは、ソースコードの形で提供される（プロフェッショナル エディションのみ）
- 平均以上の質を持つマニュアル

2.2.2 ネガティブなポイント

- 統合開発環境が遅いので、ユーザー フレンドリーとはいえない

2.2.3 評定

評定の詳細については、 [Doc. 3] を参照

RTOS アーキテクチャ	0	9	10
OS マニュアル	0	8	10
OS コンフィグレーション	0	8	10
インターネット コンポーネント	0	8	10
開発ツール	0	7	10

3 概要

3.1 評価製品

3.1.1 RTOS

QNX Software Systems Ltd の QNX NEUTRINO RTOS v6.3.0

3.2 概要

QNX Software Systems Ltd は、1980 年の設立以来、一貫して組込み市場向けのソリューション提供を行ってきた。

QNX は、長い間、x86 関連プロセッサのみサポートしてきた。1988 年に QNX Neutrino アーキテクチャを発表して以来、この方針は変更され、現在ではほとんどの CPU をサポートしている。

QNX Neutrino がほかの RTOS と異なる重要な点の一つは、POSIX API スタンダードに基づく OS であるという点である。しかし、誤解してはいけない。この OS は UNIX のようなシステムでは全くなく、真の意味でのリアルタイム オペレーティング システムである。

QNX Neutrino は、メッセージ ベースのプロセス間通信を行う、真のマイクロカーネル アーキテクチャに基づいている。

3.3 対応 CPU

QNX Neutrino v6.3.0 は、以下のプロセッサをサポートしている。

- x86
- ARM
- Xscale (=ARM コア)
- PowerPC
- MIPS
- SH-4

4 技術的評価

これは、品質的なアプローチの評価であり、このため、評点は評価製品における経験にもとづいて決定され、客観的な基準に基くものではない。したがって、評価報告のうちこの部分は、３つの批准ロゴの識別には使用されない。

註：この報告では、技術的な評価が行われているが、一定プラットフォームにおける同オペレーティングシステムの量的なテスト結果は、他の報告書にて記載されている。

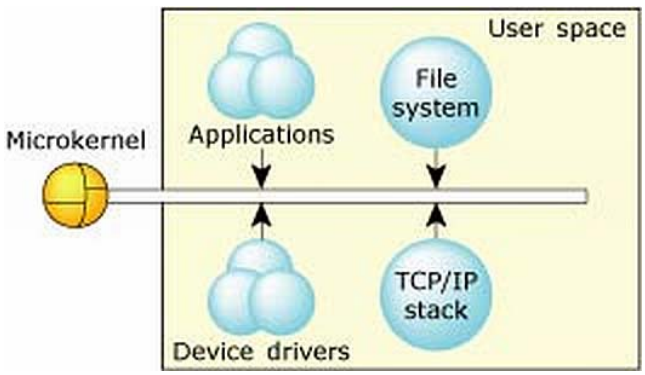
4.1 OS アーキテクチャ

RTOS アーキテクチャ 0 9 10

完全バーチャルメモリ保護を提供する真のクライアントサーバー アーキテクチャ。QNX Neutrino はメッセージ ベースの OS であり、複数のノードを解してシームレスに分散することができる。同 RTOS は、SMP をサポートし、複数の HA（ハイ アベイラビリティ）機能を実装している。

QNX NEUTRINO RTOS v6.3 は、マイクロカーネルとオプショナルの協力プロセスから成るクライアント サーバー アーキテクチャになっている。マイクロカーネルは、スレッド、プロセス、シグナル、メッセージ パッシング、同期、スケジューリング、タイマ サービスなど、コアサービスのみを実装する。マイクロカーネルそのものは、けしてスケジューリングされない。マイクロカーネルのコードは、カーネルコール、ハードウエア割り込み、またはプロセッサ例外の結果としてのみ実行される。

その他の機能は、協力するプロセス群として実装される。これらのプロセスは、サーバー プロセスとして動作し、クライアント プロセス（例：アプリケーション プロセス）のリクエストに応答する。こうしたサーバー プロセスの例として、ファイルシステム マネージャ、プロセス マネージャ、デバイス マネージャ、ネットワーク マネージャなどが上げられる。これは以下の図に示されている。



マイクロカーネル アーキテクチャ

CPU において、カーネルは特権レベルで実行されるが、ほかのプロセスはすべてユーザー レベルで実行される。したがって、マネージャとデバイス ドライバも、ユーザー レベルで実行される。しかし、これ

らのプロセスは、OS にスレッド I/O 特権をリクエストするこができ、これで I/O にアクセス、割り込みハンドラをインストールすることができる（アプリケーションがルート権限を持って実行されている場合のみ可能）。使用される CPU タイプに応じて、スレッドに I/O 特権を与えるシステムコールは、実行スレッドの CPU 権限レベルを変更する場合がある。

I/O 権限をリクエストしないアプリケーション プロセスは、常にユーザー レベルで実行される。

デバイス ドライバとサーバー プロセスがカーネルの一部ではないため、デバッグが簡単（アプリケーションと同様）であり、デバイス ドライバの不具合によりカーネルがクラッシュすることがない！このため、QNX Neutrino は、非常に安全で信頼性の高い RTOS となっている。

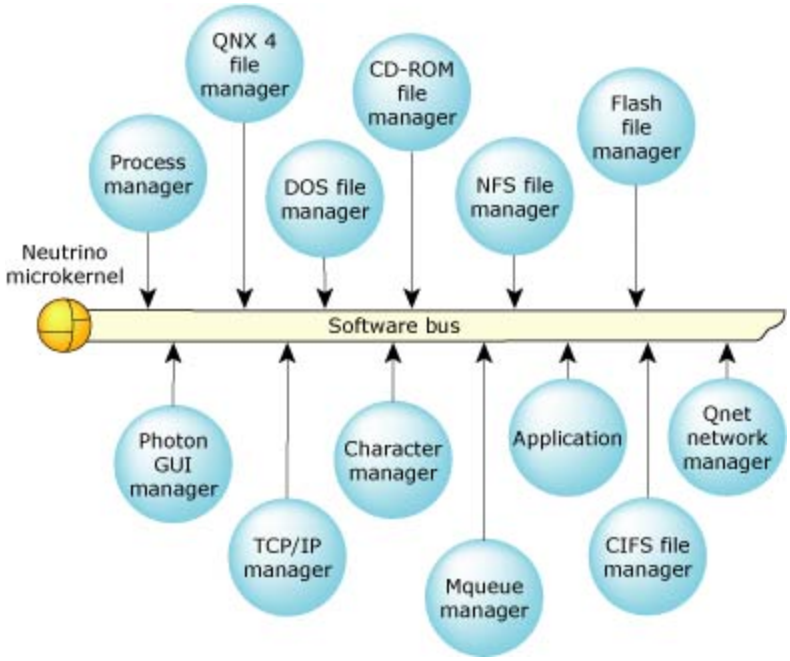
QNX NEUTRINO RTOS v6.3 は、メッセージ ベースのオペレーティング システムである。メッセージ パッシングは、この RTOS において、プロセス間通信の基本的な方法である。メッセージ パッシングは、クライアント サーバー モデルに基づいている。クライアント（例：アプリケーション プロセス）がサーバー（例：デバイス マネージャ）にメッセージを送信し、サーバーは結果を返す。QNX NEUTRINO RTOS API コールの多くは、メッセージ パッシングのメカニズムを使用している。たとえば、あるアプリケーション プロセスがファイルをオープンしたい場合、システム コールがファイル システム マネージャに送られるメッセージに翻訳される。ファイル マネージャは、デバイス ドライバを介してディスクにアクセスした後、ファイルのハンドルを返す。このメッセージ パッシング メカニズムは、ネットワーク透過である。つまり、アプリケーション コードに変更を加えることなく、システムを複数のノードにわたってシームレスに分散することができるのである。

同期 IPC メカニズムの場合、サーバー プロセスは、優先度順にメッセージを受信する。サーバー内のスレッドがリクエストを受信すると、送信元スレッドの優先度を継承する（スケジューリング アルゴリズムは継承しない）。その結果、サーバーの作業をリクエストしているスレッドの相対的な優先度が保持され、サーバーの作業は、優先度に従って実行される。このメッセージ駆動の優先度継承により、優先度の逆転という問題を回避できる。メッセージで待機（ブロック中）のスレッドがない場合、ビジースレッドの優先度は最も優先度が高い待機中メッセージの優先度に引き上げられる。

この IPC メカニズムは、メッセージ キューを使用しないため、プロセス間での余分なデータコピーを回避できる。メッセージ駆動の優先度システムでは、クライアントからはメッセージが見えない形になる。つまり、同一のスレッドから手続きをコールする場合と同様になる（メッセージが LAN を介して渡されるのではない場合）。

当然ながら、QNX Neutrino は、プロセス間でメッセージを送受信している間も、完全にプリエンプトが可能である。プリエンプトが終了すると、中断された場所からメッセージ パッシングが再開される。

QNX では、アーキテクチャがソフトウエア バスのようになっている。つまり、追加の機能が必要になったら、新規ソフトウエア モジュールを、オンデマンドでプラグインできる。以下の図を参照：



ソフトウエア バスの概念

このクライアント サーバー アーキテクチャには、大きな利点があり、代表的なものが堅牢性である。それぞれのマネージャ（プロセス マネージャを除く）とドライバが独自のバーチャル メモリ アドレス空間で実行されるので、堅牢で信頼性の高いシステムが実現する。ここで、パフォーマンスに対する懸念が発生する。システムコールの実行には複数のコンテキスト スイッチが伴い、これは、メモリ保護と引き換えに発生するオーバーヘッドである。理論上では、このためパフォーマンスがいくばくか低下するはずである。しかし、テスト結果によると、パフォーマンス上のトレードオフは、事実上目に見えないといっていい。クライアント サーバーのアーキテクチャによりコーディングが簡素化され、パフォーマンスを改善しているのである！

4.1.1 タスク処理メソッド

4.1.1.1 プロセスとスレッド

QNX NEUTRINO RTOS v6.3 はプロセスとスレッドを使用する。プロセスはスレッドが実行されるアドレス空間を定義し、常に少なくとも一つのスレッドを有している。これは、この報告書全体を通じて使用される定義である。

4.1.1.2 スレッドの優先度

我々の報告書では、QNX Neutrino の以前のバージョンの欠点として、ユーザー優先度に 63 のレベルしかないことを上げている。これは、特に、RMA（Rate Monotonic Analysis）などのテクニックを使用して設計される大規模なリアルタイム アプリケーションで問題となった。こうしたアプリケーションでは、格スレッドで独自の優先度レベルが必要になるためである。

優先度は、順番に番号がついている。例えば、最も低い優先度は 0 であり、アイドル スレッド用にリザーブされている。

QNX は、我々のアドバイスに従い、255 のユーザー優先度レベルを可能にした。しかし、QNX では、特別なやりかたでこれを行った。これらの優先度を、デフォルトでいくつかのセクションに分けたのである。

- 優先度 0: アイドル スレッド用にリザーブ
- 通常の優先度 1 – 63: 任意のスレッドで使用可能
- 特権優先度 64 - 255: ルート プロセスでのみ使用可能（unix 系のオペレーティング システムにおけるルート。つまり、ユーザー id が 0 のプロセス）

通常優先度と特権優先度の境界は、改変することができる。この境界は、プロセス マネージャ プロセスの起動パラメータである。

このメカニズムにより、QNX では同時に次の二つの問題を解決した。

- 優先度の数を 255 に引き上げた。
- 高い優先度は、特権プロセスによってのみ使用される。

特権プロセスのコンセプトにより、サードパーティ開発者がリアルタイム アプリケーションの上に非リアルタイム アプリケーションを開発できるので、便利である。リアルタイムの部分は特権モードで実行し、リアルタイム性が必要ない部分は、権限が無いので、より高い優先度のスレッドに割り込むことができない。

4.1.1.3 スケジューリング メカニズム

QNX Neutrino は、リアルタイム オペレーティング システムなので、常に優先度ベースのスケジューリングを使用する。同じ優先度レベルに属するスレッド群のみ、異なるスケジューリング オプションを使用できる。

- FIFO: スレッドは、自発的に制御を放棄する（通常、ブロッキング システム コールによる）まで、キューの先頭に保持される。
- ラウンドロビン：FIFO と同様であるが、タイムスライス機能が適用され、一定のタイムスライスが消費されるとキューの末尾にスレッドを配置する。
- スポラディック スケジューリング：このタイプの適応スケジューリングでは、スレッドに二つの優先度がある。一定のタイムスライス内に CPU の消費が多すぎると、スレッド優先度が下がる。低優先度で一定の時間が経過した後（補充期間）、そのスレッドは元の優先度に戻る。

これらのパラメータはすべて、各スレッドに適応させることが可能である。

スポラディック スケジューリングは、レート モノトニック分析（RMA）の後に許可されている時間より長い間スレッドが実行されることを回避ために使用することができる。

4.1.1.4 プロセス管理

QNX Neutrino システムでのプロセスおよびメモリ管理は、プロセス マネージャによって行われる。プロセス マネージャは、マイクロカーネルがその中に含まれ、同じアドレス空間を所有しているという点で、特別なプロセスである。カーネルもプロセス マネージャも、すべてのプロセスにおけるメモリにアクセスが必要なので、これは必然的なことである。

ただし、プロセス マネージャは、特権モードでは実行されず、マイクロカーネルを発動する場合には、特権スイッチを行わなければならない。これは、プロセス マネージャもシステム内のほかのプロセスと同様に動作するということを意味している。

	評価対象 OS
対応メモリモデル	複数のスレッドを有するプロセス群が、それぞれ保護されている。
スレッド/プロセス優先度レベル	256 レベル（特権優先度レベルを含む）。
プロセスの最大数	sysconf() コールで設定される_SC_CHILD_MAX。ユーザーid によるプロセス数。
スレッドの最大数	プロセス内のスレッド最大数は、利用可能なシステム リソースによってのみ、制限される。
スケジューリング ポリシー	同じ優先度に属さないスレッドに対する優先度ベースのスケジューリング。 同じ優先度のスレッド：FIFO またはラウンドロビン（スレッドのタイムスライスは、常に 4 OS クロック ティックとなる）。 スレッドは、ダイナミックに切り替わる二つの優先度を持つことができる。これは、スポラディック スケジューリングと呼ばれる。
マニュアルに記載されているスレッド状態の数	レディ（Ready）、実行中（Running）およびブロック（blocked）。QNX Neutrino は、ブロックの理由によって異なる状態名を使用する。ブロック状態の詳細を含めると、マニュアルに記載されている状態の数はトータルで 21 となる。
マニュアルに記載されていないスレッド状態	該当なし。

4.1.2 メモリ アーキテクチャ

QNX Neutrino のアーキテクチャでは、すべてのドライバ、アプリケーション、プロトコルスタック、ファイルシステムが、カーネルの外で、安全なメモリ保護ユーザー スペースで実行される。

対照的に、Solaris、Linux、Windows 2000 など、従来的な OS のプロセス メモリ保護は、アプリケーション レベルに限られる。

各プロセスはまた、独自のプライベート バーチャル メモリ スペースを持つ。各プロセスは、使用するプロセッサによって、2 から 3.5 GB のバーチャル メモリを持つことができる。

4.1.2.1 ブート ROM

コードがブート ROM にある場合、RAM コピーを取らずに実行することができる。

4.1.2.2 ページング

ページングは、QNX Neutrino のデフォルトでは使用されない。しかし、swapctl ユーティリティによって、スワップ ファイルを使用するオプションは存在するようである。しかし、QNX Neutrino のマニュアルでは、この件に関して非常に曖昧である。ただし、マニュアルにまったく記述が無いというわけではない。これは、QNX Neutrino ワークステーションで開発スイートも実行している場合にのみ使用される。

4.1.2.3 ヒープ

ヒープについては、従来的な OS で使用されるものと同様のスタンダード ヒープである。ソフトウエアのデバッグを行う場合は、なんらかのチェック機能とあわせてヒープを使用することが可能である。このヒープ ライブラリ バージョンを使用する場合は、メモリ リークを検出することが可能である。

当然ながら、これはオーバーヘッドになるので、最終的な製品バージョンでは使用されるべきではない。

	評価対象 OS
MMU サポート	あり
物理ページ サイズ	プロセッサにより、1Kb または 4Kb
スワッピング／デマンド ページング	通常、使用不可
バーチャル メモリ	あり（MMU がある場合）
メモリ保護モデル	☺ 完全バーチャル メモリ保護。各プロセスが、独自のバーチャルメモリ スペースで実行される。

4.1.3 割り込み処理

マイクロカーネルの割り込みリダイレクタが、初期段階の割り込み処理をする。このリダイレクタが現在実行中スレッドのコンテキストを保存し、ISR がコードおよび、ISR を含むスレッド（ISR をアタッチするスレッド）の部分データにアクセスが与えられるような形にプロセッサのコンテキストを設定する。

QNX NEUTRINO RTOS v6.3 は、割り込み共有をサポートしている。割り込みが発生すると、ハードウエア割り込みにアタッチされている各割り込みハンドラが順番に実行される。ユーザーは、割り込みを共有している割り込みハンドラが実行される順番を憶測することはできない。

☺ 割り込みハンドラの実行中は、割り込みはディスエーブルにされない。したがって、割り込みはネストすることができる。アンマスクされた割り込みは、割り込みハンドラの実行中にサービスすることができる。

割り込みサービス ルーチンは、NULL を返す（アクティブ化するものがない）か、マイクロカーネルに対するイベントを指すポインタを返すことができる。イベントを使用する場合は、ISR をアタッチしたスレッドは、そのイベントの発生を待機することができる。

☺ 割り込み処理のコンセプトでは、デバイス ドライバがほかのユーザー プロセスと同様、カーネルとリンクされないようになっている（このため、我々は「真の」マイクロカーネル アーキテクチャと呼んでいる）。デバイス ドライバは、デバイス メモリにアクセスするのに十分な特権のみ必要となる。

ISR はまた、アプリケーションで定義された構造を介して、これをアタッチしたプロセスにデータを渡すことができる（ISR は、実際にはこれをアタッチしたプロセス空間で実行されるため）。ISR 内でのセマフォは、意図的にブロックしてある。これは、メッセージ ベースのアーキテクチャでは、あまり優雅なソリューションとはいえないためである。

	評価する OS
ハンドリング	☺ ネスト可、優先度順
コンテキスト	ISR は、その割り込みハンドラをアタッチしたプロセスのコンテキストで実行される
スタック	ISR は、特別な制限つきスタックを持つ（通常、200 バイト程度）
割り込みとタスクの通信	ISR の内部から IST にシグナルを行うには、イベントのみ使用することができる（その後、イベントへのポインタが ISR から返される）

4.1.4 システム タイマ

オペレーティング システムのクロック タイマ（クロック ティックとも言う）は、40MHz よりも早いプロセッサでは 1 ミリ秒、それよりも遅いプロセッサでは、10 ミリ秒とデフォルトで設定されている。

しかしながら、システム コールでこれを変更することは可能である。

すべてのシステム タイマは、すべてのオペレーティング システムと同様、クロック周期よりも正確になることはない。

4.1.4.1 高分解能タイマ

QNX Neutrino はまた、使用されるハードウエアによっては、高分解能タイマもサポートしている。これらのタイマは、ブート時にゼロで始まる、ナノ秒単位の一般的な 64 ビット タイマを使用している。こうしたハードウエアが存在しない場合は、カーネルがこれをエミュレートする。

SMP システムにおいては、これらの CPU タイマは、現在スレッドが実行されている CPU に依存する。

4.1.4.2 タイムアウト タイマ

QNX Neutrino v6.3 では、ブロックを行うシステム コールでタイムアウトをする際の特別なメカニズムがある。ブロックコールを行う前に、一定の時間が経過したらタイムアウトするよう、スレッドがカーネルにリクエストできるのである。これにより、カーネルそのものが、システム コールでタイマをスタートするので、利点となる。ブロックを行うシステム コールの前に、スレッドがタイマを開始する場合は、実際にシステム コールを呼び出す前にプリエンプトされる可能性がある。

4.1.5 同期メカニズム

QNX Neutrino は同期プロセス間通信メッセージング システムに基づいて構築されている。これは、プロセス間通信のメカニズムとして最適化されており、頻繁に使用されている。このシステムは、実際にはほかのプロセスで実行されるライブラリ関数コールと似た働きをするが、コールは処理が完了するまでリターンしない。

QNX Neutrino は、POSIX スタンダードに基づいた、非同期メッセージング キューもサポートしている。これは、メッセージ内容を特に認識しなくてもよいシステム設計では、便利なソリューションである（出しっぱなしソリューション）。

4.1.5.1 保護メカニズム

QNX Neutrino v6.3 は、さまざまな POSIX スタンダードで見通されているすべての保護メカニズムを提供している。スレッド間でのみ使用できるものもあれば、プロセス間、リモートプロセス間（ネットワーク）で使用できるものもある。

以下は、一般的な保護メカニズムである。（POSIX 参照）

– ミューテックス：我々の評価用語で定義されているものと同様の保護メカニズム。常に優先度の継承を行い、優先度の逆転を防ぐ。
これは、通常スレッド間のみで使用される。ミューテックスが、二つのプロセスの共有領域にある場合、プロセス間で使用できる。

ミューテックス コールが、呼び出し元の状態を変更するか、スレッドをリリースするまで、カーネルはミューテックス処理には関わらない点に注意。

– セマフォ：これは優先度継承を行わず、容易にプロセス間で使用することができる。QNX Neutrino では、名前つきセマフォを提供しており、ネットワーク ノード間で使用することができる。

これらの一般的な保護メカニズムのほかに、QNX Neutrino では、以下の POSIX メカニズムも提供している。

– 条件変数：「真」になるまで、クリティカル セクション内で待機する。
– バリア：協力関係にあるスレッド群に「陣形」を取らせる同期メカニズム。ひとつのスレッドが先に進む前にすべてのスレッドが完了するよう、特定のポイントで待機を強制する。
– スリープオン ロック：特殊な条件変数
– リーダ／ライタ ロック：これらのロックは、データ構造へのアクセスが、多数のスレッドがデータを読み込み、（多くとも）一つのスレッドがデータを書き込む、というパターンの場合に使用される。これらのロックは、ミューテックスよりもコスト高になる。

これらの機能の詳細については、POSIX スタンダードおよびオンライン QNX 資料を参照とする。

使用する同期プリミティブについて、多くの選択肢があるという点は重要である。しかし、通常は、ミューテックスとセマフォで十分である。

本評価報告のフレームワークにおいては、ミューテックスの優先度継承メカニズムが最も重要な機能となる。

4.1.5.2 インターロック メカニズム

QNX Neutrino は、以下のアトミック システム コールをサポートしている。

– 値の追加
– 値の減算
– ビットのクリア
– ビットの設定
– （相互補完）ビットのトグル

4.1.5.3 通信メカニズム

ここでも、QNX Neutrino は POSIX スタンダードの中で見通されているプロセス間通信のうち、最も多くのメカニズムを提供している。

4.1.5.4 同期メッセージ パッシング

最も重要なのは、異なるシステム コールで使用されるメッセージ パッシング インターフェイスである。このメッセージ パッシングは、同期メッセージ ベース優先度でメッセージのサーバー処理を行い、システムコールがほかのプロセスで処理されても、同じスレッドで処理（同じ優先度を使用）されても透過的になるようにする。

このメッセージ パッシング システムは、メッセージの送信にチャネルを使用する。したがって、メッセージは直接スレッドに送られるのではなく、チャネルに送られる。

4.1.5.5 パルス

パルスは、異なるプロセス間でメッセージを送るための同期システムである。しかし、小さなペイロードでしか使用できない（8 ビットコードおよび 32 ビットデータ）。

4.1.5.6 イベント

パルスと POSIX シグナルは、全て QNX Neutrino 内部イベント処理システムに基づいている。

4.1.5.7 POSIX シグナル

これに関しては、我々は POSIX を参照している。多くの開発者は、UNIX アプリケーション構築の経験から、このメカニズムを知っている。

4.1.5.8 非同期メッセージ パッシング

同期メッセージ パッシングのほかに、QNX Neutrino は POSIX の非同期メッセージ キューもサポートしている。これは、オプショナルのリソース マネージャ、mqueue によって行われる。

QNX Neutrino の同期メッセージと、POSIX メッセージ キューには、根本的な違いがある。同期メッセージは、ブロックを行い、メッセージを送るプロセス間で直接データをコピーする。一方、POSIX メッセージ キューは、「保存して転送」形式を実装し、送信側はブロックの必要がなく、待ち行列に待機中のメッセージをいくつも入れることができる。同期メッセージはまた、メッセージ送信元の優先度を継承するが、これは POSIX メッセージ キューでは行われない。

POSIX に従う非同期メッセージ パッシングは、ファイルシステムに類似したシステムを使用する。キューをオープンするパス、およびそのパスで送受信するデータの読み込み／書き込み、というシステムである。

4.2 API の豊富さ

QNX NEUTRINO RTOS v6.3 は、POSIX 準拠 API と独自 API の両方を提供する。API はメッセージ ベースのシステムに合わせてあり、QNX のシステム アーキテクチャにマッチしている。ただし、固定ブロック サイズのメモリ パーティションには不足が見られる。

以下の表は、カーネル関連の API に関してのみ評価を行っている点に注意。

以下の表では、POSIX および OS 独自のインターフェイスに実装されている、基本的なリアルタイム オブジェクトおよび機能の評価をまとめている。

結果の解釈を行う際には、以下の表では厳密に定義されたシステム コールのセットのみカバーしていることを読者は留意するべきである。異なる OS に対する異なる API を比較することは困難なため、このセクションでは評点をつけていない。このセクションは、読者が自分のアプリケーションで必要とする API コールが使用可能であるかどうかを調べる目的で使用するべきである。

一般的に、使用できるシステムコールが多いほど、プログラマが自分のアプリケーションで使用する適切なコールがわかりやすく、インスタンス用コールの独自ライブラリを記述する必要がない。

4.2.1 タスク管理

スレッド管理	あり
スタック サイズ取得	✓
スタック サイズ取得設定	✓
スタック アドレス取得	✓
スタック アドレス設定	✓
スレッド状態取得	✓
スレッド状態設定	✓
TCB 取得	✓
TCB 設定	-
優先度取得	✓
優先度設定	✓
スレッド ID 取得	✓
スレッド状態変更ハンドラ	-
現在のスタック ポインタを取得	✓
スレッドの CPU 使用を設定	-
スケジューリング メカニズムを設定	✓

スレッド管理	あり
メモリ内でスレッドをロック	✓[1]
スケジューリングを無効にする	-

4.2.2 クロックとタイマ

クロック	あり
時刻の取得	✓
時刻の設定	✓
分解能の取得	✓
分解能の設定	✓
時間の調整	✓
リードカウンタ レジスタ	✓
自動的に時間を調整	✓

インターバル タイマ	あり
絶対期日でタイマが失効	✓
相対期日でタイマが失効	✓
周期的にタイマが失効	✓
残り時間を取得	✓
オーバーランの数を取得	✓
ユーザー ルーチンを接続	✓

4.2.3 メモリ管理

固定ブロック サイズ パーティション	なし
パーティション サイズの設定	-
パーティション サイズの取得	-
メモリ ブロック サイズの設定	-
メモリ ブロック サイズの取得	-
パーティション位置の指定	-

[1] スワッピング メカニズムがないため、スレッドは常にメモリ内にロックされる。

固定ブロック サイズ パーティション	なし
メモリブロックの取得 – ブロッキング	-
メモリブロックの取得 – ノンブロッキング	-
メモリブロックの取得 – タイムアウトつき	-
メモリ ブロックのリリース	-
パーティションの拡張	-
空きメモリ ブロックの数を取得	-
メモリ内のパーティションをロック／ロック解除	-

非固定ブロック サイズ プール	あり
プールサイズ設定	✓
プールサイズ取得	✓
新規プール作成	-
メモリブロック サイズ取得	-
メモリ ブロック取得－ブロッキング	✓
メモリ ブロック取得－ノンブロッキング	-
メモリ ブロック取得－タイムアウトつき	-
メモリブロック リリース	✓
プール拡張	-
ブロック拡張	✓
残りの空きバイトを取得	-
メモリ内のプールをロック／ロック解除	✓
メモリ内のブロックをロック／ロック解除	✓

4.2.4 割り込み処理

割り込み処理	あり
割り込みハンドラのアタッチ	✓
割り込みハンドラのデタッチ	✓
割り込み待機 – ブロッキング	✓
割り込み待機－タイムアウトつき	✓
割り込み開始	-
ハードウエア割り込みのディスエーブル／イネーブル	✓
ハードウエア割り込みのマスク／マスク解除	✓
割り込み共有	✓

4.2.5 オブジェクトの同期と除外

セマフォのカウンティング	あり
最大カウントの取得	-
最大カウントの設定	-
初期値の設定	✓
プロセッサ間で共有	✓
待機 – ブロッキング	✓
待機 – ノンブロッキング	✓
待機 – タイムアウトつき	✓
ポスト	✓
ポストーブロードキャスト	-
状態の取得（値）	✓

バイナリ セマフォ	なし
初期値の設定	-
プロセッサ間で共有	-
待機 – ブロッキング	-
待機 – ノンブロッキング	-

バイナリ セマフォ	なし
待機 – タイムアウトつき	-
ポスト	-
状態の取得	-

ミューテックス	あり
初期値の設定	✓
プロセッサ間で共有	✓
優先度の逆転を回避するメカニズム	✓
再帰的取得	✓
スレッド削除の安全性	-
待機 – ブロッキング	✓
待機 – ノンブロッキング	✓
待機 – タイムアウトつき	✓
リリース	✓
状態の取得	-
オーナーのスレッド ID を取得	-
ブロックされたスレッドの ID を取得	-

条件変数	あり
ノンブロッキングでペンディング	✓
タイムアウトつきでペンディング	✓
FIFO／優先度順でペンディング	-
ブロードキャスト	✓
優先度の逆転	-

イベント フラグ	なし
一件づつ設定	-
複数を設定	-
一件でペンディング	-
複数でペンディング	-

イベント フラグ	なし
OR 条件でペンディング	-
AND 条件でペンディング	-
AND および OR 条件でペンディング	-
タイムアウトつきペンディング	-

POSIX シグナル	あり[2]
シグナル ハンドラのインストール	✓
シグナル ハンドラのデタッチ	✓
シグナルのマスク／マスク解除	✓
送信元の検出	✓
宛て先 ID の設定	✓
シグナル ID の設定	✓
シグナル ID の取得	✓
スレッドのシグナル	✓
キューに入れられたシグナル	✓

4.2.6 通信とメッセージ パッシング オブジェクト

キュー	あり
メッセージの最大サイズ設定	✓
メッセージの最大サイズ取得	✓
キューのサイズを設定	✓
キューのサイズを取得	✓
キューの中のメッセージ数を取得	✓
プロセス間で共有	✓
受信－ブロッキング	✓
受信－ノンブロッキング	✓
受信－タイムアウトつき	✓
送信－ACK つき	✓

[2] これらのシグナルの他に、QNX Neutrino は拡張POSIXリアルタイム シグナルのすべての対応している。

キュー	あり
送信－優先度つき	✓
送信－ OOB （帯域外）	-
送信－タイムアウトつき	-
送信－ブロードキャスト	-
タイムスタンプ	-
通知	✓

メールボックス	なし[3]
メッセージの最大サイズ設定	-
メッセージの最大サイズ取得	-
プロセス間で共有	-
送信－ACK つき	-
送信－タイムアウトつき	-
送信－ブロードキャスト	-
受信－ブロッキング	-
受信－ノンブロッキング	-
受信－タイムアウトつき	-
状態の取得	-

[3] メールボックスとは、すなわち一件以上のメッセージを保存できないメッセージ キューのことである。これは、完全性を目指す目的のみで存在しているが、多くのオペレーティング システムでは、現在では明示的にサポートが行われていない。

4.3 マニュアル

OS マニュアル 0 8 10

マニュアルは、以前のバージョンに較べて改善された。また、BSP のセクションも改善された。市場の中で、もっとも優れたマニュアルを持つRTOS である。

☺ API は完全にマニュアル化され、6.2 リリースよりもさらに改善された。すべてのパラメータの説明が記載され、また、パラメータが構造体と関連している場合は、構造体の説明へのリンクが提供されている。

以前と同じく、マニュアルでのシステムとアーキテクチャ概要紹介は、かなりしっかり記述されている。QNX Neutrino を始めて使用するユーザーには、システム アーキテクチャ マニュアルを最初に読むことを推奨する。

独自 BSP 作成のマニュアルは改善された。BSP 使用のサンプルを掲載するセクションが追加されたので、BSP マニュアルについては、多くの競合他社よりも優れた内容となっている。多大な設定オプションについての情報を提供するのは簡単ではないと言及しておきたい。

また、各種ドライバのソースコードについては、あまりよく記述されていない。まったく言及されていないドライバもある。しかし、大部分のソースコードは再使用が可能である。

QNX ソフトウエアは、顧客に拡張有償サポート オプションを提供している（スタンダード サポートとプライオリティ サポート）。

4.4 OS コンフィグレーション

OS コンフィグレーション 0 8 10

BSP が QNX から提供される場合、インストールは簡単でシンプルである。カスタム QNX イメージについては、IDE またはテキスト ベースのビルドファイルで行われる。しかし、新規 BSP の構築は容易ではない。いずれにしても、すべての BSP とドライバはソース コードが提供され、ユーザーのプラットフォームに移植することができる。カーネルとライブラリのみ、ソースコードが提供されない。

4.4.1 OS ブート オプション

QNX Neutrino では、ターゲットのブートは 3 つのステップから成る。

- 初期プログラム ローダー（PL）。これは、以下の働きをする。
  - すべてのソフトウエアを実行できるよう、重要なハードウエアをセットアップする（例：DRAM タイミング）。
  - OS（とアプリケーション）のロードに使用されるデバイスをセットアップする。
  - メモリに OS をロードする。
  - ロードされた OS の最初のアドレス（スタートアップ プログラム）にジャンプする。
- スタートアップ プログラム。これは、以下の働きをする。
  - カーネルが必要とするコンフィグレーションを定義済みの構成でセットアップする。
  - ブート可能コンフィグレーション（複数をインストールすることが可能）を検索する。
  - ブート ファイル システムをマウント（ダウンロードされた OS の場合は、メモリにマップされる）し、カーネルをスタートする。
- その後、カーネルがスタートアップ スクリプトを実行し、対応デバイスがロードされ、アプリケーションがスタートされる。

サンプル IPL は、ソースコードで提供されている（各対応ボードに対して）。これを利用して、独自の IPL を記述することができる。しかし、ハードウエアの設定は容易ではなく、ボード メーカーによるドキュメンテーションが必要である。

IPL は、QNX 製のものである必要はなく、ハードウエアを設定、バイナリをダウンロードおよび起動するコードならば、どのようなものでも使用できる。

スタートアップ メカニズムを理解するのは、ほかの RTOS の場合を同じく、非常に難しい。

4.4.2 OS コンフィグレーション

QNX Neutrino のカスタム ターゲット イメージ作成は、ビルドファイルで行われる。モジュールの追加、削除、設定は、これらテキストベースのファイルを編集して行うことができる。マニュアルには、ビルドファイルの例が豊富に記載されている。

加えて、IDE にはシステム ビルダ ツールが含まれており、これで QNX イメージの管理ができる。ビルド ファイルの代わりにグラフィカルなツールを使用し、イメージ（ブート イメージとフラッシュ イメージの両方）を作成、既存ビルドファイルのインポートができる。システム ビルダ ツールには依存性分析機能があり、どのライブラリが足りないか検出することができる。また、同ツールには「ダイエティシャン」機能もあり、使用する共用ライブラリのうち、必要な機能のみを集めたバージョンを作成することができる。

☹ QNX Momentics® 開発スイート v6.3 IDE の起動には非常に時間がかかる（古い PC システムの場合は数分にも及ぶ）が、システム ビルダ ツールは比較的速く動作し、使用法が簡単である。最初は、どこにモジュールが格納されているか理解するのに時間がかかる（たとえば、特定のデバイスは DLL に含まれている）。これは、選択プロセスの中にコメント コラムを追加することで改善されるはずである。システム ビルダにモジュールが入れられると、IDE にコメント セクションが表示され、ライブラリの使用法とオプションの説明を見ることができる。非常に便利である。システム ビルダはまた、ターゲット システム コンフィグレーションに追加されたモジュールで必要となる共有ライブラリを自動的に追加する。

システム ビルダは、既存の（シンプル）ビルド ファイルで初期化するのが一番良い。こうすると、基本コンフィグレーションが適切に設定される。

☺ 「ダイエット」機能は非常に便利であり、ほかのプラットフォーム ビルダには見られない機能である。もちろん、共有ライブラリの「ダイエット」は非常に時間のかかる作業である（すべての依存性をチェックするため）。しかし、これは、開発サイクルの最後の方でのみ必要な作業である。

4.4.3 独自 BSP の構築

☺ QNX Momentics® 開発スイート v6.3 をインストールする際、ほとんどすべてのものがソースコードで提供される（ライセンスによる）ことに気がつくだろう。Linux 的な「オープン ソース」運動そのものは、とくにこの製品に大きな利点を提供するものではない。

ソースコードで提供されない主なものは、プロセス マネージャ（procnto）、つまり、QNX Neutrino のマイクロカーネルである。カーネル（およびライブラリ）以外のすべてはソース コードで提供されるため、カーネルによって、特定のプロセッサがサポートされるかどうかを決定される。

☺ 前述のように、すべての BSP とデバイス ドライバは、QNX Neutrino v6.3 Momentics プロフェッショナル エディションをインストールする際にソース コードで提供される。ここで、すべての対応ターゲット プロセッサをインストールすることが肝要である。すべてのドライバがすべてのターゲット プロセッサに対してコンパイルされるわけではないからである。

QNX Neutrino では、ドライバがひとつのターゲットで対応していれば、ほかのターゲットでも簡単に使用できることがわかった（適切なコンパイラで再コンパイルすればよい）。

もちろん、独自のデバイス ドライバを開発することもできる。しかし、これは別の問題となるので、この文書では議論の対象外とする。

4.5 インターネット コンポーネント

インターネット コンポーネント 0 8 10

QNX Momentics® 開発スイート v6.3 には、以下の製品とツールが含まれている。

- 組込みウェブ サーバー。この組込みウェブ サーバーは CGI 1.1、 HTTP 1.1、およびダイナミック HTML （サーバー サイド インクルード コマンドを介したサポート）に対応している。
  また、データ サーバーを介して SSI を処理することもできる。データ サーバーでは、複数のスレッドによるデータの共有が可能である。プロセスがハードウエアの状態についてデータ サーバーをアップデートし、この間、別個に組込みサーバーが安全にその状態にアクセスできる。
- メモリ要件が小さい情報表示用のウェブ ブラウザ。HTML 3.2 完全サポート、フレーム、javascript 、クッキーなどのサポート。
- QNX ソフトウエア システムズでは、ACCESS システム アメリカとパートナーシップを結び、ACCESS の NetFront HTML 4 組込みブラウザを提供するようになった（このブラウザは、Pocket PC 誌の 2004 年度ベスト ソフトウエア賞を獲得している）。QNX Voyager 2™ 組込みブラウザと、ACCESS NetFront 技術は、小型ディスプレイ レンダリング技術、XHTML 1.1（モバイル プロファイル）を含む HTML 4.01、SSL 3.0、JavaScript、WAP、WML などの機能を提供し、CPU リソースが限られているハイ パフォーマンス システムで理想的である。ブラウザ ソリューションも、別途コストにて提供される。
- 組込みシステムでのインターネット対応アプリケーション開発用に、ソースも入手が可能である。
- ブロード ネットワーキングおよびプロトコル サポート。詳細については、QNX ウェブサイト (http://www.qnx.com) を参照。これについては、「拡張ネットワーク技術開発キット」を使用することができる。

4.6 開発ツール

開発ツール	0	7	10

QNX NEUTRINO RTOS v6.3 では、各種ツールを使用することができる。 拡張可能な eclipse フレームワークに基づく QNX Momentics® 開発スイート v6.3 IDE をテストした。

eclipse フレームワークは、主に RAM サイドで大量のリソースを消費する。リソース消費量は、350MB にも及ぶ。

eclipse フレームワークには、大きな欠点が一つある。それは、ユーザー フレンドリーではないという点である。IDE に求められる直感的な使いやすさに欠けているが、許容できる部分もある。

我々は、テスト システムの開発に eclipse フレームワークを使用した。セルフホストおよびクロス開発の両方のツールが使用できる。これらのツールキットは、一般的によく使われるツールを備えている。

eclipse フレームワークのほかに、QNX は GNU ライブラリおよびコマンドライン ツールをサポートしており、GCC ANSI C、POSIX、Dinkum 製 ISO 準拠 フル C++、GCC 3.3.1、GDB 5.2.1 が含まれる。

この IDE の主な欠点は、リソース消費が大量になる点である（CPU とメモリ）。このため、古いハードウエア プラットフォームでは、ほとんど役に立たないと言っていい。また、IDE を立ち上げるのには非常に時間がかかる。QNX が要求する最低限のシステム（Pentium III 700MHz、512MB RAM）では、約 30 秒かかる。IDE が起動された後は、スピードは遅いが、動きはする。間違って IDE のウィンドウを閉じてしまわないよう、気をつけなければならない...これは、新製品でより頻繁に見られる現象である。プログラマは、制限された環境でコーディングをすることに慣れず、効率の悪いやり方でプログラムのコーディングをするようになってきている！ これは、とくに Java 環境で顕著な現象である。

ウィンドウズ環境で IDE を実行する場合、300MB もの RAM が消費される。これに加えてマニュアルを開くと、さらに 50MB が消費される！

システム ビルダは非常に使い方が簡単である。

しかし、IDE を使用してアプリケーションを開発する場合、非常に細かいことを検索する必要が出てくる。このため、どこに何があるのかわからなくなってしまうが、インターフェイスはまったく直感的とは言い難い。既存のコードから新規プロジェクトを作成しようとしても、既存ソースファイルの追加すら不可能に思えてくる。また、一貫性に関しても、多くの問題がある。設定を変更する際、その設定が保存されていないようなのである！プロジェクトで設定の変更（たとえば、インクルード ディレクトリの追加）がアクセプトされる前に、IDE を再起動しなければならなかった。最終的に、我々は各アプリケーションに対する設定ファイルを手作業で変更することにした。この方が、ユーザー インターフェイスを使うよりも手っ取り早かったのである。

以下の表に記載するツールのほかに、以下の追加ツールも使用することができる。

- Photon インターフェイスを開発するためのアプリケーション ビルダ
- コア キット技術のアプリケーションを開発展開する、または、ソフトウエアを改変してカスタム製品を作成するための技術開発キット。QNX では、以下のキットを提供している。
  - クリティカル プロセス モニタリング（プロセスを監視し、障害から回復するシステム）

http://www.dedicated-systems.com
email: info@dedicated-systems.com

- 拡張ネットワーキング（ネットワーク プロトコルの大きなコレクション）
- MOST （Media Oriented Systems Transport:）マルチメディア通信の自動車用スタンダード
- 3D グラフィックス
- フラッシュ ファイルシステム
- 非対称型マルチプロセシング（SMP）
- ウェブ ブラウザ
- マルチメディア

– システム機能を拡張、拡張機能を追加、カスタム ハードウエア プラットフォームにポーティングするためのソース キット。透過分散処理、プラットフォーム コア ソース、カーネル ソースのキットが入手可能であり、ソースコード、マニュアル、サンプル アプリケーションを含む。

–

	有無	IDE で統合	スタンドアロン	コマンドベース	GUI ベース
エディタ					
カラー ハイライト	有	✓			✓
統合ヘルプ	有	✓			✓
自動コード レイアウト	有	✓			✓
コンパイラ					
C	有	✓	✓	✓	✓
C++	有	✓	✓	✓	✓
Java	有	✓	✓	✓	✓
Ada	無				
アセンブラ	有	✓	✓	✓	✓
リンカ					
インクリメンタル	有	✓	✓	✓	✓
記号テーブル生成	有	✓	✓	✓	✓
開発ツール					
プロファイラ	有	✓	✓	✓	✓
プロジェクト管理	有	✓			✓
ソースコード制御[4]	有	✓	✓	✓	✓
リビジョン制御	有	✓	✓	✓	✓
デバッガ					
シンボリック デバッガ	有	✓	✓	✓	✓
スレッド センシティブ	有	✓	✓	✓	✓

[4] Momentics IDE は、CVSおよび、サードパーティのプラグイン製品（例：Clearcase）をサポートしている。ソースコード制御システムがあってもなくても、Momenticsソース エディタには自動履歴保存機能があり、以前に保存されたバージョンのソースファイルが保持される。任意の時点で、自分のファイルと以前のファイルを比較が可能（グラフィカル ツールを使用）、以前のバージョンに戻ることもできる。履歴を保存する日数は調整できる。

	有無	IDE で統合	スタンドアロン	コマンドベース	GUI ベース
スレッド センシティブなデバッグ	有	✓	✓	✓	✓
混合ソースと逆アセンブリ	有	✓			✓
変数検査	有	✓	✓	✓	✓
構造検査	有	✓	✓	✓	✓
メモリ検査[5]	有	✓	✓	✓	✓
ターゲット接続					
JTAG	有			✓	
BDM	有			✓	
シリアル（ROM モニタまたは app?）	有	✓		✓	✓
ネットワーク（ROM モニタまたは app?）	有			✓	
システム分析ツール					
トレーシング	有	✓	✓	✓	✓
スレッド情報	有	✓	✓	✓	✓
割り込み情報	有	✓	✓	✓	✓
ローダー					
TFTP ブート ローダー	有	✓	✓	✓	✓
シリアル ブート ローダー	有	✓	✓	✓	✓
別個モジュールをロード	有	✓			

[5] malloc ライブラリのインスツルメンテッド バージョンに対して、デバッグ用 malloc ツールを使用し、アプリケーションを実行することができる。これで、メモリ割り当てをトレースし、一般的なメモリ関連のエラーを検出することができる。

http://www.dedicated-systems.com
email: info@dedicated-systems.com

5 付録 A: ベンダによるコメント

6 付録 B: 略語

略語	意味
API	Application Programmers Interface （アプリケーション プログラマーズ インターフェイス）
BSP	Board Support Package （ボード サポート パッケージ）：特定のボードで OS の実行に必要なすべてのコードとデバイス ドライバ
DSP	Digital Signal Processor （デジタル シグナル プロセッサ）
FIFO	First In First Out （ファーストイン ファーストアウト）
GPOS	General Purpose Operating System （汎用オペレーティング システム）
GUI	Graphical User Interface （グラフィカル ユーザー インターフェイス）
IPC	Inter-Process Communication （プロセス間通信）
IDE	Integrated Development Environment （統合開発環境）：アプリケーションの開発とデバッグに使用される GUI ツール
IRQ	Interrupt Request （割り込みリクエスト）
ISR	Interrupt Servicing Routine （割り込みサービス ルーチン）
MMU	Memory Management Unit （メモリ管理ユニット）
OS	Operating System （オペレーティング システム）
PCI	Peripheral Component Interconnect （周辺機器コンポーネント相互接続）
PIC	Programmable Interrupt Controller （プログラマブル割り込みコントローラ）
PMC	PCI Mezzanine Card （PCI メザニン カード）
PrPMC	Processor PMC （プロセッサ PMC）：プロセッサつき
RTOS	Real-Time Operating System （リアルタイム オペレーティング システム）
SDK	Software Development Kit （ソフトウエア開発キット）
SoC	System on a Chip （システム オン チップ）

http://www.dedicated-systems.com
email: info@dedicated-systems.com

7 付録 C: 文書更新履歴

7.1 更新番号 1.0 (November 17, 2004)

初期バージョン