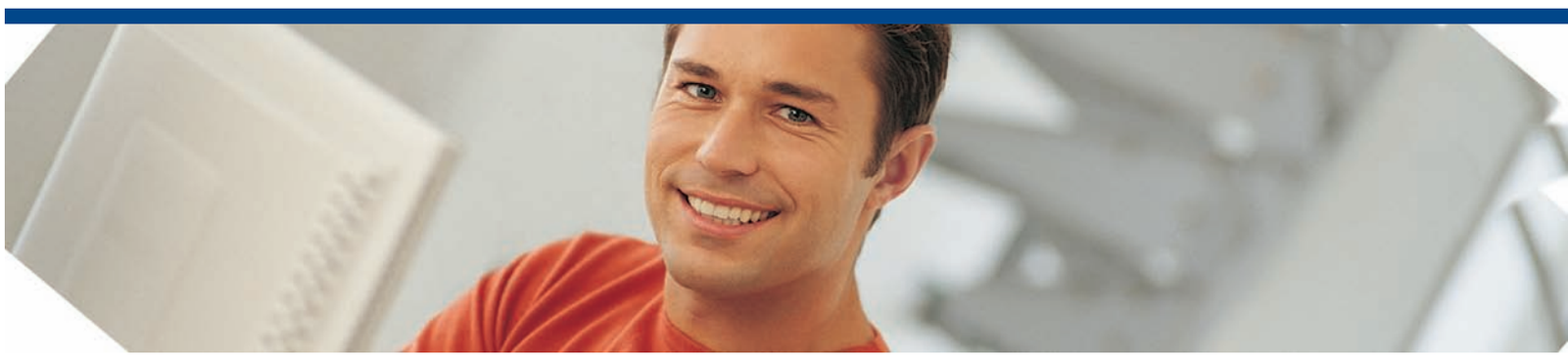
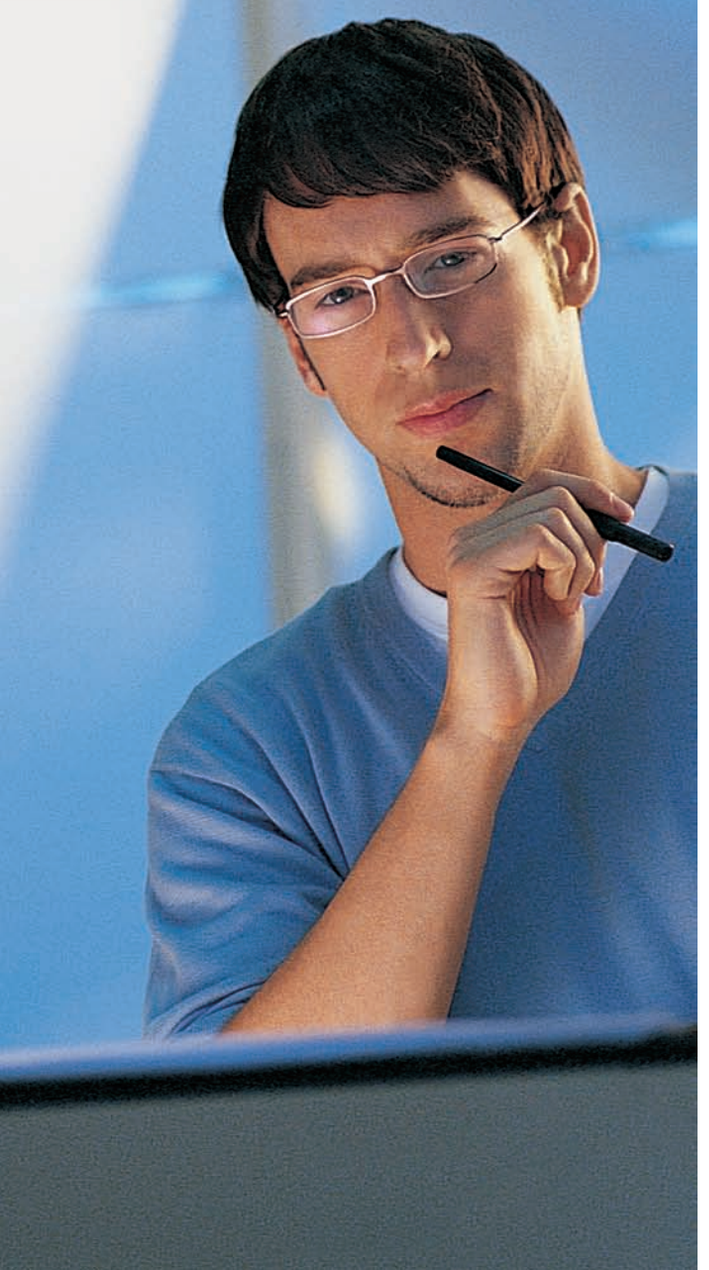
QNX® Momentics®

КОМПЛЕКТ РАЗРАБОТЧИКА



Гибкая IDE-среда на основе платформы Eclipse, позволяющая проводить глубокий анализ работы системы с помощью инновационных инструментов профилирования, отладки и оптимизации.

Уже более 30 лет технологии реального времени QNX помогают создавать самые надёжные в мире приложения: от комплексов сетевой маршрутизации, автомобильных телематических систем до широкомасштабных распределённых систем промышленного управления.



СОДЕРЖАНИЕ

Профессиональный комплект разработчика QNX Momentics® позволяет ускорить все этапы вашего проекта на основе QNX Neutrino: от встраивания на процессорную плату до системного анализа. Он включает в себя всё, что вам необходимо: интуитивно понятную среду разработки, полный набор инструментов, пакеты поддержки процессорных плат и множество готовых к использованию комплектов для разработки драйверов – и всё это в едином высокоинтегрированном комплекте. Он также обладает чрезвычайно широкой гибкостью, поскольку поддерживает множество языков программирования, множество инструментальных сред и множество целевых процессоров. Свяжитесь с нами прямо сейчас, чтобы узнать, как QNX Momentics® может помочь вам ускорить цикл разработки вашего проекта.

Прежде всего QNX Momentics® дает вам выбор. Вы можете применять разные языки программирования, инструментальные платформы и процессорные архитектуры. Писать программы на C, C++, Embedded C++. Вести разработку в среде Windows, Linux. Компилировать код для целевых процессоров ARM, MIPS, PowerPC, SH-4 или x86. И всё это в одной и той же среде разработки (IDE). Более того, вы можете работать даже с несколькими языками и архитектурами одновременно.

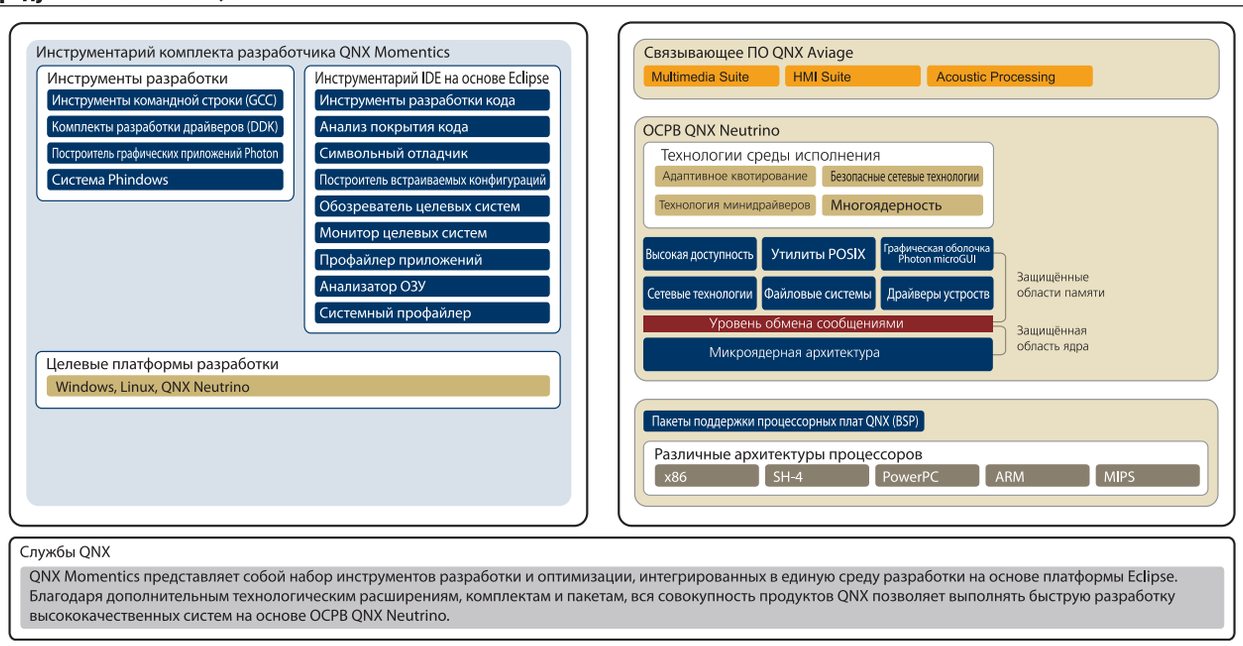
QNX Momentics® упрощает цикл разработки от начала до конца. Вам нужно "оживить" процессорную плату? Просто выберите пакет поддержки процессорных плат (BSP), импортируйте его файл описания в построитель встраиваемых систем и сгенерируйте целевой образ. Вы начинаете новый проект? Запустите специальный мастер, выберите нужные опции, и проект будет создан автоматически. Вы сразу же можете начать писать код с помощью редакторов, разработанных для C и C++.

Вы разрабатываете нестандартный драйвер? Вам помогут комплекты разработки драйверов (DDK), содержащие солидную базу исходных текстов и готовых шаблонов.

Вам нужно оптимизировать производительность? Профайлер приложений и анализатор ОЗУ тесно интегрированы с остальными инструментами среды разработки, что упрощает вашу работу. Вам нужна комплексная отладка и оптимизация системы на низком уровне? Системный профайлер и отладочная версия микроядра позволят визуально отслеживать проблемы синхронизации, взаимные блокировки и прочие часто возникающие ошибочные ситуации.

Профессиональный комплект разработчика QNX Momentics® содержит в себе всё необходимое для разработки и внедрения встраиваемых систем высшего качества.

Продуктовая линейка QNX



Продуктовый набор QNX обеспечивает быструю разработку качественных систем на основе ОСРВ QNX Neutrino.

Открытая платформа Eclipse

С комплектом разработчика QNX Momentics® вы свободны в выборе инструментов благодаря тому, что среда разработки основана на Eclipse – открытой платформе для интеграции инструментария, поддерживаемой большим и постоянно расширяющимся сообществом компаний-производителей инструментов.

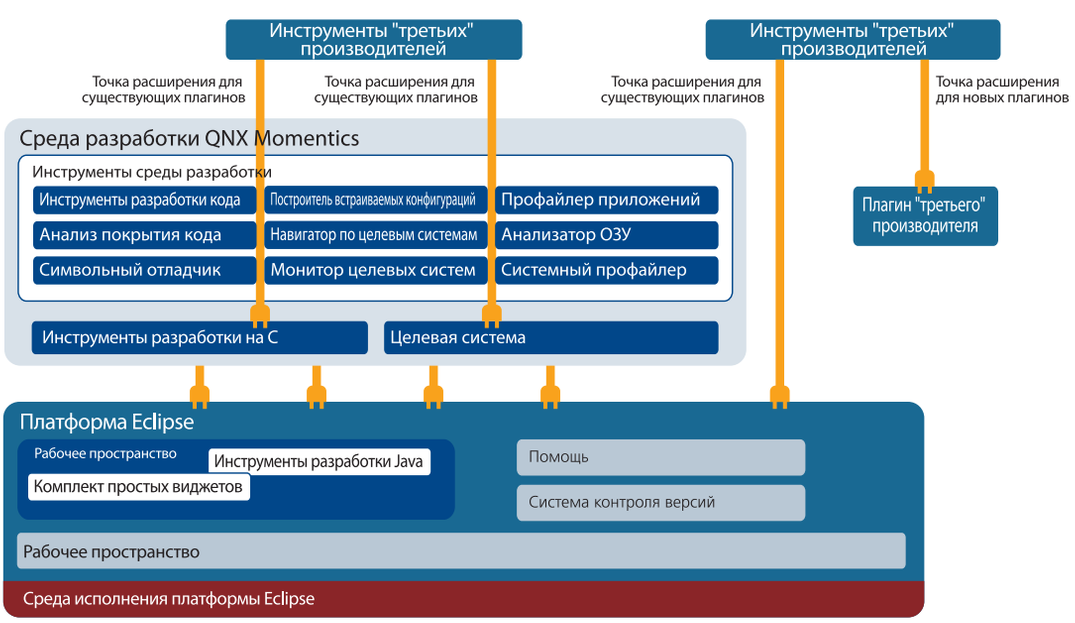
Платформа Eclipse обеспечивает интерфейсы (т. н. точки расширения) для прозрачной интеграции инструментов. В результате весь инструментарий QNX Momentics имеет единый графический стиль. Таким образом, вам не придётся заново изучать пользовательский интерфейс каждого отдельного инструмента или каждой инструментальной платформы.

Платформа Eclipse имеет расширяемую архитектуру подключаемых модулей (плагинов), позволяющую QNX Momentics работать практически с любым типом данных. Например, в состав QNX Momentics входит множество подключаемых модулей для навигации по исходным текстам на C/C++, анализа встраиваемых и загружаемых образов, анализа производительности систем в реальном времени. Вы также можете разрабатывать свои собственные подключаемые модули для работы с любыми другими объектами или использовать готовые модули "третьих" производителей – в обоих случаях новые подключаемые модули прозрачным образом полностью интегрируются с существующими инструментами.

Комплект разработчика QNX Momentics обеспечивает:

- полную интеграцию с Eclipse - открытой платформой для интеграции инструментов;
- поддержку набора функций Eclipse v3.5.x, что ускоряет загрузку IDE и плагинов;
- поддержку новейших возможностей платформы Eclipse C/C++ Development Tools (CDT) v6.x.

Расширяемая архитектура платформы Eclipse



Платформа Eclipse имеет расширяемую архитектуру и обеспечивает интерфейсы для прозрачной интеграции инструментов в единой графической среде.

Средства разработки кода

QNX Momentics® обеспечивает оптимизированную среду для разработки на C/C++ и Embedded C++. вы можете работать как в командной строке, так и в графической среде, в которой можно пользоваться целым рядом средств для повышения производительности (например, мастерами, редакторами кода, гибкими структурами make-файлов и др.).

Мастера проектов

Создавайте новые проекты с помощью всего нескольких щелчков мышью. В QNX Momentics вы можете применять встроенные мастера для автоматизации всего процесса.

Мастера позволяют:

- выбрать тип проекта (C, C++, сборочный файл (makefile), проект QNX);
- пройти все шаги по созданию проекта для любой комбинации целевых процессоров;
- создать для проекта каталоги построения, файлы зависимостей и заготовки исходных текстов;
- автоматически скомпилировать весь проект для создания готового приложения.

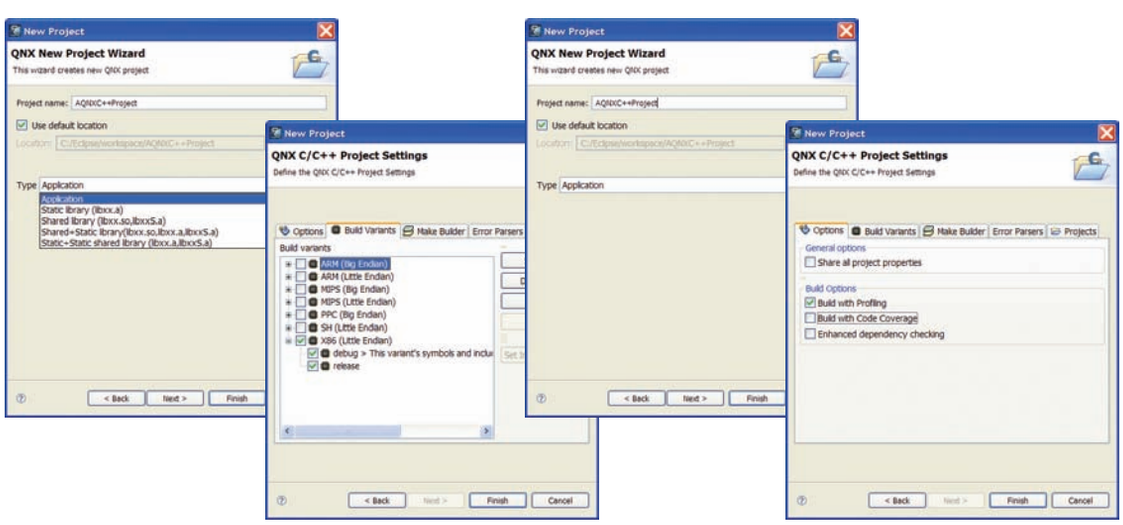
Редакторы кода

Пишите больше кода за меньшее время. Редакторы кода, входящие в состав QNX Momentics, предоставляют все необходимые возможности, включая операции "вырезать/вставить", отмену действий, добавление комментариев и формирование отступов, типичных для используемого языка. Кроме того, редакторы кода обеспечивают следующие функции:

- **Подсветка синтаксиса** – ключевые слова и синтаксис языков программирования автоматически выделяются цветом. Это применяется и к парным скобкам (что удобно при написании множества вложенных секций кода).

- **Контекстно-зависимая справка** – если подвести курсор мыши к имени функции, то редактор отобразит её описание, аргументы и все необходимые заголовки. Если нажать "горячую" клавишу при выбранной функции, то редактор автоматически вставит в текст все необходимые директивы #include.

Мастера проектов



Встроенные мастера будут сопровождать вас на каждом этапе создания проекта – от выбора процессора до окончательной компиляции кода.

- **Сворачивание кода** – разработчики могут скрыть/отобразить фрагмент кода, над которым они не работают в настоящее время. Функцию «Сворачивание кода» (Code folding) можно настроить на отображение отдельных участков кода, когда определенные участки отображаются даже если родительский процесс свернут.

- **Шаблоны кода** – просто нажмите "горячую" клавишу, чтобы вставить в текст типовые часто используемые блоки кода (циклы for, условные выражения if-then-else и т. п.). Каждый редактор содержит набор готовых шаблонов, которые вы можете редактировать или модифицировать.

- **Изменяемые настройки** – задавайте шрифты, цвета и прочие настройки, как вам удобно.

- **Расстановка маркеров в тексте** – каждый редактор отображает в тексте маркеры, связанные с текущим файлом, включая ошибки, указанные средствами построения. Вы также можете расставлять свои собственные маркеры в виде заданий или закладок.

- **Список задач, чтобы лучше организовывать работу** – представляет собой централизованный репозитарий для маркеров (точек останова, ошибок и т. п.) и заданий. Щёлкните на ошибке, и редактор автоматически перейдёт к нужной строке.

Навигатор по исходному тексту

Навигатор по исходному тексту поддерживает контекстно-зависимый поиск ссылок и определений, а также позволяет упростить разработку кода.

Навигатор по исходному тексту выполняет:

- обзорное отображение для быстрой навигации по исходному коду и заголовочным файлам проекта;

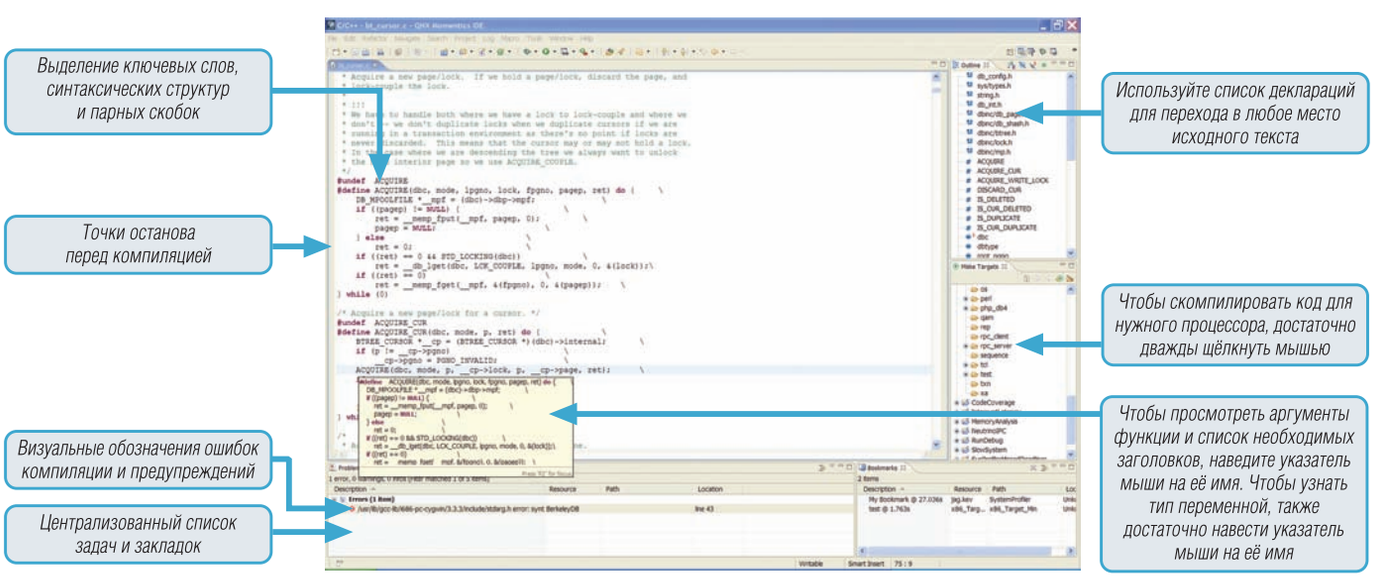
- быстрый переход к декларациям и прототипам;

- автозавершение при вводе имён функций.

Локальный журнал изменений

Вы можете легко разрабатывать прототипы систем с помощью мощного инструмента "локальный журнал изменений" без необходимости обращения к репозитарию исходных текстов. Вы можете вносить пошаговые улучшения в прототип, при этом будет выполняться автоматическое отслеживание изменений в файле, с которым вы работаете. Таким образом, вы можете легко производить тестирование системы и, если нужно, отменять любые изменения в коде. Локальные версии работают так же, как и версии в репозитарии исходных текстов, поэтому вы можете легко использовать их в утилитах графического сравнения и слияния файлов.

Редакторы кода



Редакторы кода имеют множество функций, включая дополнение кода, контекстно-зависимую справку и централизованный список задач, которые ускоряют и упрощают работу

Инструменты управления исходными текстами

Управляйте всей базой исходных текстов из одной среды. Среда разработки включает в себя встроенную поддержку протокола управления исходными текстами CVS, включая поддержку удалённого сервера и доступ к защищенным репозитариям посредством SSH. Также поддерживаются другие системы управлениями версиями, такие как ClearCase, Perforce, SDWI и т.д.

Поскольку управление версиями и конфигурациями встроено непосредственно в среду разработки, вам не нужно выходить из неё, чтобы управлять своими исходными текстами. Более того, вы можете работать в неоднородной среде управления версиями, используя разные протоколы для разных проектов, и даже для разных файлов в пределах одного и того же проекта.

С помощью инструментов управления исходными текстами вы можете:

- управлять версиями при обновлении своей базы исходных текстов;
- управлять версиями при коллективной разработке;
- просматривать журнал версий, чтобы узнать, кем и какие изменения были внесены;
- выполнять визуальное сравнение различных версий файлов;
- использовать интерактивное слияние версий, чтобы быстро разрешать конфликты между изменениями, внесёнными несколькими разработчиками в один и тот же файл.

Структуры make-файлов

Выбирайте структуру make-файла, оптимальную для вашего проекта. Когда вы создаете проект на C/C++ в QNX Momentics, вы можете применять следующие структуры make-файла:

- рекурсивная структура для многопроцессорных проектов позволяет быстро компилировать проект для одного или нескольких поддерживаемых целевых процессоров;
- ваша собственная структура файла построения и дерева каталогов позволяет импортировать существующие проекты на основе make-файлов или переносить проекты, использующие средства построения, отличные от make.

Если вы выбираете многопроцессорный вариант, вам не придётся создавать свои make-файлы вручную. С помощью диалогового окна вы просто выбираете мышью нужные вам настройки, и среда разработки формирует по ним соответствующий make-файл.

Инструменты построения

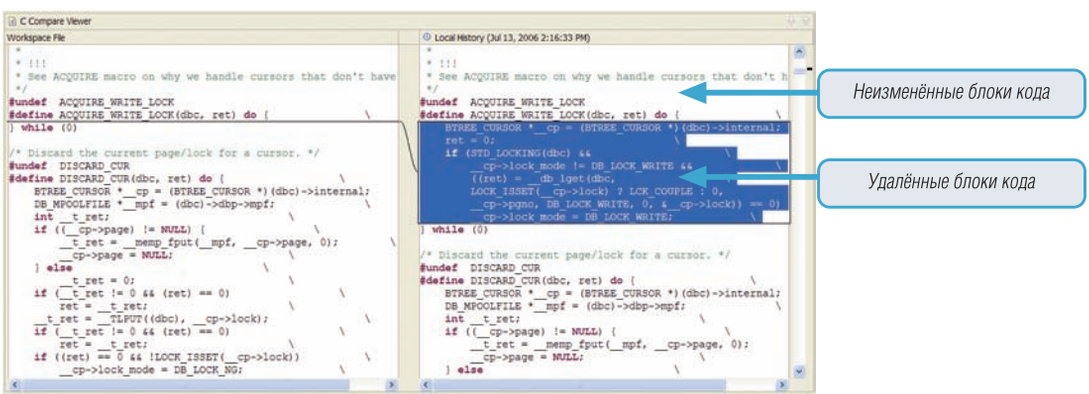
Используйте инструменты, которые вам давно знакомы, для разработки встраиваемой системы. Многие графические компоненты среды разработки основаны на командно-строковом инструментарии. Например, компиляторы и другие средства построения проекта (например, make) вызываются из среды разработки с нужными параметрами командной строки, после чего их вывод передаётся обратно в среду разработки. В результате вы можете работать либо с командно-строковыми инструментами, либо с инструментами IDE (либо и с теми, и с другими одновременно), и при этом получать абсолютно одинаковые двоичные модули и контекст.

Вы также можете управлять параметрами построения проекта, поскольку среда разработки позволяет выбирать опции компилятора и компоновщика, списки подключаемых заголовков, списки библиотек и др. В процессе построения исходного кода проекта среда разработки вызывает стандартные компиляторы, помечает обнаруженные ошибки и предупреждения с помощью маркеров в исходном тексте.

Мастер конфигурации запуска

С легкостью компилируйте, запускайте на исполнение и отлаживайте код посредством одного щелчка мышью. Используя мастер конфигурации запуска, вы можете быстро определять, какие программы запускать, на каких целевых системах и с какими отладочными опциями. Эти установки сохраняются в среде разработки, поэтому вы можете быстро запускать заданные конфигурации в последующих сессиях.

Визуальное сравнение файлов



Утилита визуального сравнения файлов быстро находит отличия между двумя версиями файла и выделяет их цветом.

Средства отладки

QNX Momentics® обеспечивает мощную унифицированную среду отладки для всех поддерживаемых языков программирования. Интуитивно понятный интерфейс отладчика полностью интегрирован с остальными инструментами IDE и даёт максимальную гибкость для решения возникающих задач.

Экранные панели любого другого инструмента среды разработки можно открыть прямо в символьном отладчике, что расширяет возможности представления информации о состоянии и данных приложения.

Символьный отладчик позволяет:

- одновременно отлаживать несколько приложений, написанных на C и C++;
- отлаживать многопоточные приложения, контролируя каждый поток по отдельности и отслеживая передачу управления между ними;
- отлаживать несколько процессов, выполняемых одновременно на разных процессорах, и отслеживать их распределение по процессорам;
- динамически подключаться отладчиком к любому выполняющемуся процессу;
- проводить "посмертный" анализ дамп-файлов.

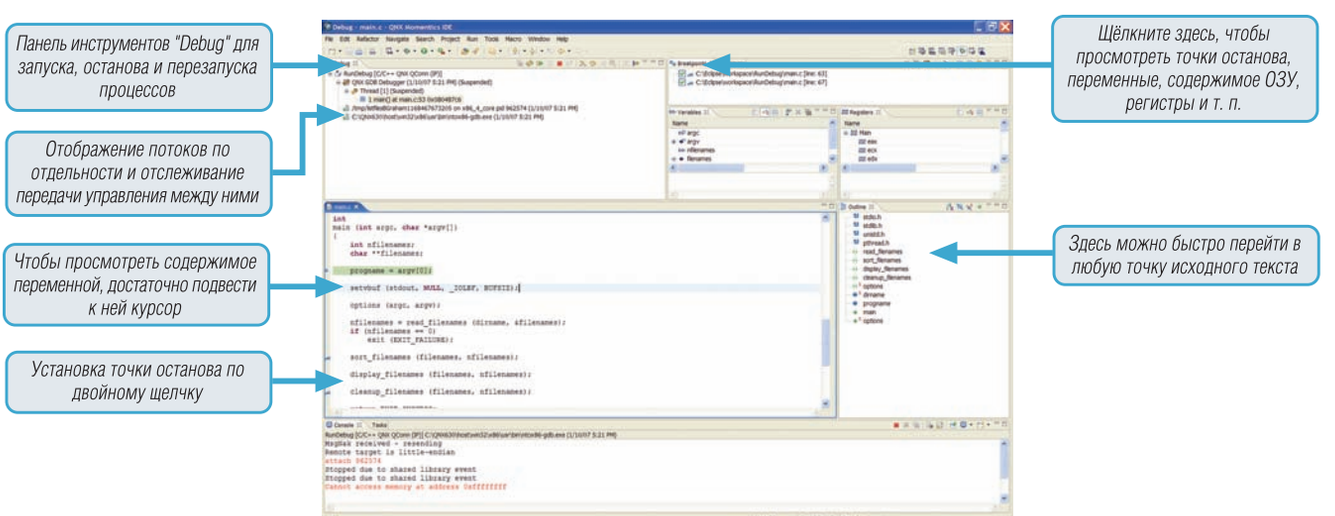
Выбирайте нужные типы точек останова

Тип точки останова	Описание
Условная	Активизируется в соответствии со значением заданного выражения.
Счётная	Активизируется или вычисляет условное выражение по достижении определённого числа итераций.
Пошаговое выполнение	Позволяет выполнять код в пошаговом режиме, а также либо входить в вызываемые функции, либо выполнять их целиком за один шаг. Кроме того, вы можете устанавливать точки останова на момент возврата из функции.
Выполнение до курсора	Позволяет создать временную точку останова, установленную на другой точке в той же функции. Программа будет выполняться до указанной точки.
Сторожевая	Останавливает выполнение программы при изменении значения переменной.

Используйте дополнительные панели отображения

Экранная панель	Описание
Переменные	Вы можете просматривать переменные, приводя их к любому типу (символьный, строковой, целый и т. д.). Указатели можно разыменовывать или представлять в виде массивов. Вы также можете выделить глобальные переменные и поместить их в панель отображения переменных.
Выражения	Эта панель отображает результаты вычисления любых выражений, в том числе переменных, констант, указателей.
ОЗУ	Позволяет редактировать любой блок памяти в пределах адресного пространства программы.
Стек	Отображает состояние стека для любого выбранного потока. На этой же панели можно для сравнения отобразить все выполняемые процессы.
Регистры	Позволяет просмотреть текущее состояние регистров процессора для любого потока.
Точки останова	Отображает все типы точек останова: условные, счётные, пошаговые или с выполнением до курсора.

Символьный отладчик



Символьный отладчик позволяет отлаживать несколько процессов и потоков одновременно, даже если эти процессы написаны на разных языках программирования.

Системный профайлер

Системный профайлер QNX Momentics® обладает значительно более широкими возможностями, чем обычные инструменты отладки и анализа кода, и позволяет анализировать взаимодействие между всеми компонентами сложной системы реального времени.

С помощью системного профайлера вы можете разрешать конфликты синхронизации, обнаруживать ситуации взаимных блокировок, выявлять корни семантических ошибок, находить скрытые неполадки в программном и аппаратном обеспечении и оптимизировать производительность вашего приложения, причём как для однопроцессорных, так и для многопроцессорных целевых систем.

Системный профайлер позволяет выполнять следующие задачи:

- получать визуальное отображение взаимодействий между компонентами целевой системы, извлекая данные из отладочной версии микроядра;
- определять момент возникновения события, связанные с ним программные модули и действия, выполненные ими, а также интерпретировать возникшее событие;
- получать информацию о событиях на целевой системе, применять динамические фильтры событий, а также получать визуальное отображение и интерпретацию данных о процессорной активности и статистики по событиям;
- отображать сводные данные, в т. ч. относящиеся к многоядерной обработке (например, миграцию потоков между процессорами и обмен сообщениями между ядрами);
- применять специализированные инструменты для определения текущего статуса потоков и причин их работы;
- создавать несколько оконных панелей для одновременного отображения информации о разных аспектах трассировки системы; эти панели можно синхронизовать по масштабу времени.

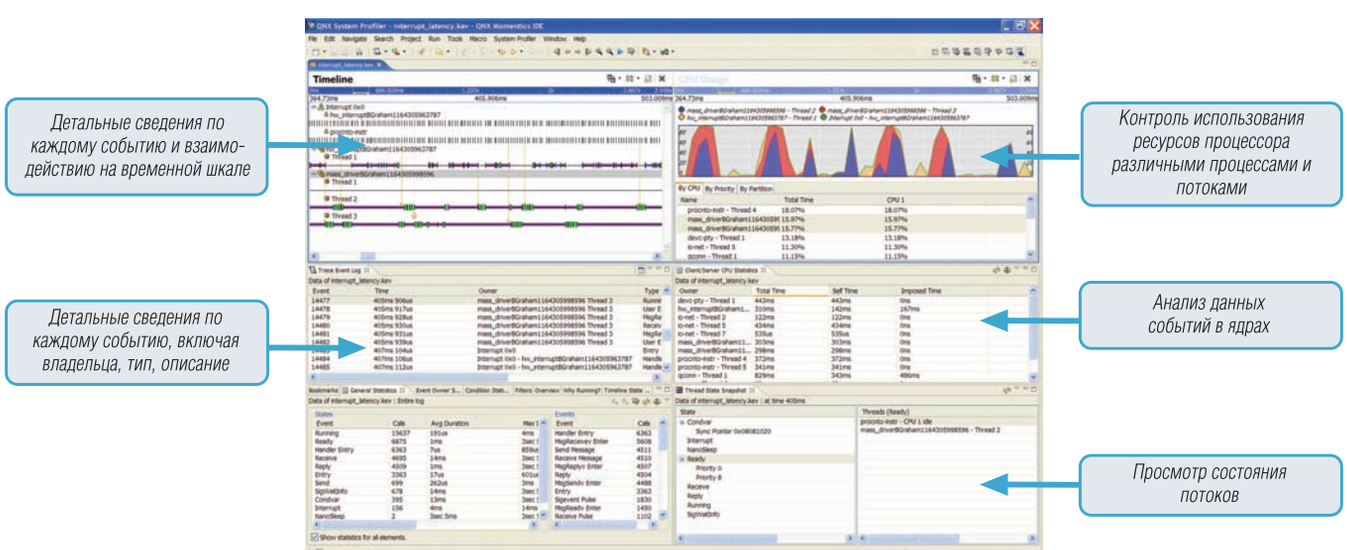
Визуальное наблюдение событий для быстрого выявления проблем

Системный профайлер может отображать огромные объёмы информации, включая информацию о вызовах ядра, аппаратных прерываниях, состоянии потоков, обмене сообщениями и событиях планировщика. Кроме этого, он даёт вам чёткий контроль над тем, какие события записывать и когда, чтобы вы могли сосредоточиться именно на том, что вам нужно. Он также позволяет применять фильтры событий и различные опции отображения, чтобы детально анализировать подозрительные фрагменты кода и легко прослеживать сложные взаимодействия.

Отладка и оптимизация многоядерных систем

Системный профайлер работает на основе QNX Neutrino, первой ОСРВ, поддерживающей симметричную многопроцессорность (SMP). Он обладает данными обо всех потоках, запущенных на различных

Системный профайлер



С помощью системного профайлера вы можете разрешать конфликты синхронизации, определять ситуации взаимных блокировок, выявлять корни семантических ошибок и другие причины снижения производительности системы.

процессорах, и помечает их различными цветами, что существенно облегчает восприятие. Например, системный профайлер можно использовать для обнаружения состязаний за обладание ресурсами или неправильной работы кода, перенесённого с одноядерных устройств. Сводка миграций потоков выводит сведения о перемещении отдельно взятого потока между ядрами, которые запланированы планировщиком работы ядер, – помогает выявлять проблемы пробуксовки или кэширования. Сводка обмена сообщениями между ядрами показывает частоту обмена сообщениями между ядрами в системе – её удобно просматривать для выявления плотно взаимодействующих потоков. В каждом из случаев системный профайлер может вывести информацию по каждому процессору в отдельности.

Полная информация о событиях в системе

При просмотре истории событий в системе часто задаётся вопрос: "Почему этот поток запущен именно сейчас?" Системный профайлер снабжён утилитой "причина запуска", которая записывает события, приведшие поток к данному состоянию, и при необходимости прослеживает серию событий до самого начала следа. Также системный профайлер может предоставить статистику клиента/сервера, включающую точное количество времени, требуемое клиенту, и что требуется от серверов, которые он использует. Утилита просмотра мгновенного состояния потока в системном профайлере позволяет просмотреть сводку по каждому потоку и его состояние в выбранный момент времени – вы можете узнать текущее состояние любого процесса в системе в любой выбранный вами момент времени.

Собственные фильтры событий

Если штатные фильтры событий, предоставляемые отладочной версией микроядра, не соответствуют ситуации, которую вы хотели бы проанализировать, это не проблема: ядро поддерживает систему динамических фильтров, определяемых пользователем, чтобы эффективно отслеживать сложные комбинации условий. Таким образом, вы можете сосредоточиться на отдельных событиях и уменьшить объём данных, выводимых для анализа.

Разделы системного профайлера	Описание
Журнал истории событий	Предоставляет подробную информацию по каждому событию в истории ядра. Журнал событий находится в постоянной синхронизации с выбранной временной шкалой.
Закладки	Позволяет вам отмечать отдельные диапазоны мест и событий, отображаемых в редакторе системного профайлера, и в последствии просматривать отмеченные события в разделе закладок.
Статистика клиента/сервера	Представляет более подробную информацию о работе системы, измеряя время, отведённое на каждый служебный поток клиентам.
Выборочная статистика	Выводит статистический отчёт в виде таблицы событий в системе, которые отвечают заданным пользователем условиям. Статистика может быть основана на данных всего журнала событий или же только на выбранном диапазоне.
Миграция между процессорами	Планирует переход потоков с одного процессора на другой (миграция), что необходимо при работе ОСРВ QNX Neutrino в режиме симметричной многопроцессорности для обеспечения полного использования ресурсов процессоров. В данном разделе представляются данные о степени миграции потоков в системе.
Статистика принадлежности событий	Выводит статистический отчёт в виде таблицы отдельных событий на основе принадлежности процесса/потока. Статистика может быть основана на данных всего журнала событий или же только на выбранном диапазоне.
Общая статистика	Выводит общий статистический отчёт в виде таблицы истории событий в ядре. Статистика может быть основана на данных всего журнала событий или же только на выбранном диапазоне.
Обзор	Выводит графическое представление использования процессора и распределения событий для всей истории событий в ядре в независимости от текущего выбранного диапазона.
Сводка по декомпозиции	Выводит сводку всего журнала событий, делая акцент на технологии адаптивной декомпозиции ОСРВ QNX Neutrino. Для каждой отдельной конфигурации групп, найденных в журнале событий, выводится распределение используемых группой процессорных ресурсов вместе с таблицей, отображающей распределение процессорных ресурсов по каждому событию внутри каждой группы.
Мгновенное состояние потока	Отображает количество потоков в каждом из возможных состояний на выбранный момент времени истории. В любом из состояний можно выделить отдельные потоки.
Причина запуска	Работает совместно с панелью редактора временной шкалы системного профайлера и при наведении курсора на какой-либо поток и нажатии кнопки мыши отвечает на вопрос "Почему этот поток запущен?".

Анализатор ОЗУ

Анализатор ОЗУ совмещает в себе расширенные возможности визуализации с инновационным методом записи данных об использовании памяти.

Анализатор ОЗУ помогает визуально отобразить использование памяти вашими программами и может быстро выявлять переполнения буферов, некорректные освобождения памяти и множество других типовых ошибочных ситуаций. Анализатор ОЗУ позволяет получить:

- уникальные инструменты для визуализации, предоставляющие графический вид профилей памяти;
- информацию на уровне процесса, с помощью которой можно быстро оценить карту памяти программы; закрашенные области представляют собой стек, код, библиотеки и "кучу";
- специализированную статистику распределения памяти для выявления возможных проблем утечки памяти; статистика содержит суммарное количество свободных, распределённых и используемых байтов и блоков;
- динамический журнал использования памяти для оценки изменений.

Выявление ошибок распределения памяти при помощи штатной библиотеки

Комплект разработчика QNX Momentics содержит библиотеку распределения памяти, в которой реализовано большинство типовых операций над строками и памятью. Эти функции перед выполнением операции проверяют корректность использования указанной области памяти, позволяя разработчикам выявлять ошибки переполнения, выборки из пустого буфера, некорректного использования памяти и повторного освобождения одной и той же области. При обнаружении ошибки разработчик автоматически переводится при помощи обратной трассировки событий в редактор кода на нужную строку кода. Эти ошибки можно быстро исправить и провести тестирование заново.

Анализатор ОЗУ

Анализатор ОЗУ обеспечивает визуализацию использования памяти для понимания того, где и как в системе используется память.

Выбирайте наиболее подходящий способ обработки ошибок

Анализатор ОЗУ обладает "интеллектуальным" механизмом отслеживания ошибок работы с памятью. При возникновении ошибки, соответствующий фрагмент исходного текста автоматически помечается предупреждением. В этом случае разработчик может:

- продолжить выполнение программы;
- завершить программу и сохранить её образ в дамп-файле;
- остановить программу и сразу переключиться в отладчик, где в вашем распоряжении будут все необходимые средства для локализации проблемы.

В любом случае, анализатор ОЗУ обнаруживает проблему и отслеживает по истории стека вызовов путь к причине возникновения ошибки. Производится быстрое исправление ошибок и повторное тестирование.

Профиль использования памяти для более глубокого понимания

Понимание профиля памяти системы является ключевым моментом в оптимизации ресурсов памяти. Анализатор ОЗУ трассирует все распределения и освобождения памяти и отображает их упорядоченными по времени, чтобы помочь понять где, когда и как используется память.

Оптимизация ресурсов памяти

При помощи технологии профилирования памяти анализатор ОЗУ помогает оценить возможности оптимизации использования памяти. Например, профилирование памяти может обнаружить интенсивное использование памяти в определённом объёме в конкретном временном промежутке. Профиль памяти отображает время распределения памяти, место в коде этого распределения, как часто это происходит и в каких временных промежутках. Нежелательные распределения памяти можно легко обнаружить в коде, где разработчики могут применить более эффективную схему.

Тип ошибки памяти	Описание
Нелегальное освобождение памяти	Происходит при попытке вызова функции free() библиотеки C с пустым указателем, указателем на стек или статическую память, или с указателем на тип памяти, который не указывает на начало распределяемого блока, или при попытке произвести повторное освобождение.
Разыменование пустых указателей	Регистрирует ошибки работы символической памяти и строк, вызываемых в программе.
Переполнение буфера	Выдаётся при нечаянной записи программой в область памяти, которой не хватает для указанного буфера.
Использование освобождённой памяти	Выдаёт ошибку памяти, когда производится попытка прочитать или записать в память, которая была только что освобождена. Например, программа A вызывает функцию free() для одного блока, но продолжает после этого его использовать, тем самым создавая проблему повторного использования при произведении запроса malloc().
Чтение неинициализированной памяти	Выдаёт ошибку памяти при попытке прочитать или записать в память, которая была освобождена по причине отсутствия инициализации.
Утечка памяти	Происходит, когда программа распределяет память, но не освобождает её в последствии. Например, функция A возвращается, когда локальная переменная является указателем на распределённую память, которая не была освобождена.

Профайлер приложений

Профайлер приложений комплекта разработчика QNX Momentics® позволяет оценивать общую производительность программ независимо от их сложности или объёма и без необходимости построчной обработки.

С помощью профайлера приложений вы можете быстро обнаруживать чрезмерно интенсивно используемые секции кода и применять к ним отладку, анализ производительности и оптимизацию. Например, профайлер приложений позволяет выполнять следующие задачи:

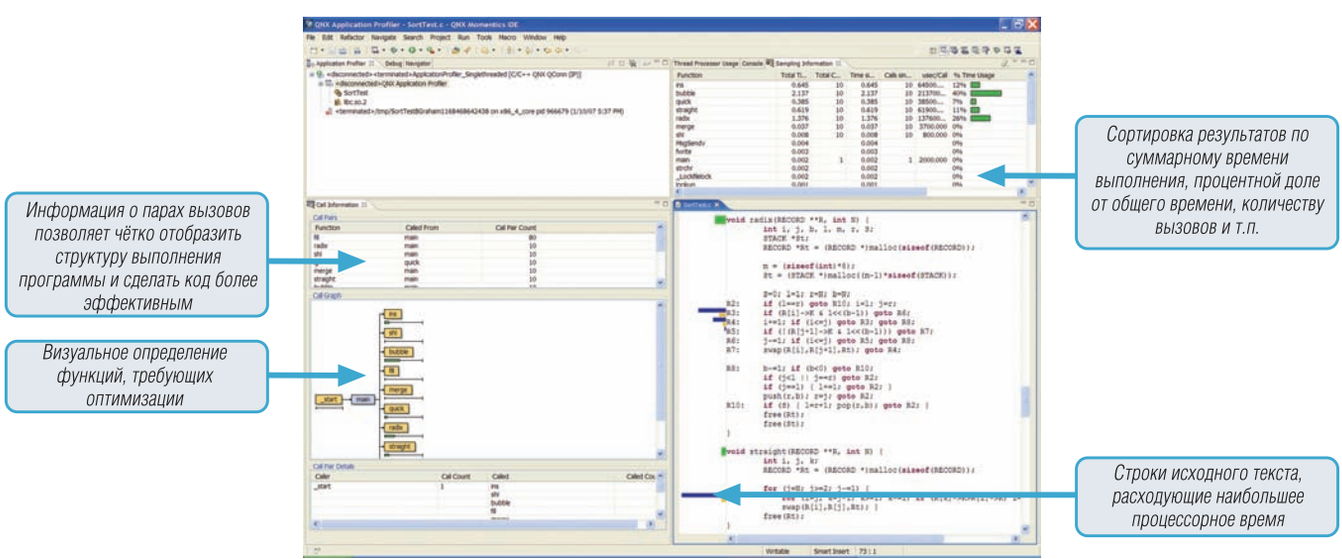
- в реальном времени собирать информацию, подключив профайлер к программе, выполняющейся на целевой системе;
- на уровне исходного текста выявлять строки, потребляющие наибольшие процессорные ресурсы;
- анализировать использование процессорного времени по множеству процессов и целевых систем, а также разделяемых библиотек;
- получать полную информацию о выполнении с помощью компилирования и запуска новой копии программы на целевой системе;
- производить "посмертное" профилирование и анализ посредством загрузки файлов статистики в профайлер.

Статистическое профилирование для неагрессивного контроля

Сканируя выполнение программ через фиксированные временные интервалы, профайлер приложений строит точную картину того, в каких участках кода система тратит больше всего времени. Для этого не требуется специальный инструментарий, изменение кода или особый режим компиляции. Неагрессивное профилирование также гарантирует, что профайлер не исказит собираемую информацию.

Профайлер приложений предоставляет информацию об использовании процессорного времени каждым потоком и отображает её как в абсолютных значениях, так и в процентах от общего времени, после чего её можно сортировать удобным образом. Профайлер приложений также позволяет начать процесс профилирования заново без перезапуска самого приложения. В итоге, вы можете сравнить результаты для одного и того же приложения при различной нагрузке. Более того, профайлер может анализировать динамически загружаемые разделяемые библиотеки, что позволяет определить, где кроется причина снижения производительности – в коде приложения или в вызываемой им библиотеке.

Профайлер приложений



Профайлер приложений позволяет анализировать использование приложением процессорных ресурсов и разделяемых библиотек на различных целевых системах. Профилирование может быть как "посмертным", так и в реальном времени.

Внедрение диагностических тегов для точных измерений

Чтобы получить дополнительную информацию о выполнении кода, вы можете дать указание компилятору внедрить в результирующий код диагностические теги для профилирования. Код с диагностическими тегами будет выдавать информацию о вызовах функций и о парах вызовов (т. е. откуда и куда был вызов). С помощью информации о вызовах и статистики выполнения вы можете локализовать "узкие места", а затем перейти к визуальной диаграмме вызовов, чтобы определить место функции в цепочке вызовов. Это весьма удобный способ поиска фрагментов кода, требующих оптимизации.

Выявление неэффективных фрагментов с точностью до строки исходного текста

Для оптимизации производительности часто недостаточно знать, какие именно функции потребляют наибольшее процессорное время. Чтобы получить более детальные сведения, профайлер приложений может спуститься до уровня исходного текста и показать, какие именно строки исходного текста потребляют наибольший процессорный ресурс. Это происходит с помощью графического аннотирования исходного текста в редакторе кода на C/C++ и отображения потребляемой доли процессорного времени – как для отдельных функций, так и для отдельных строчек кода. Вы можете визуально сравнить эффективность различных стратегий оптимизации, выявить неэффективные алгоритмы и сосредоточиться на оптимизации нужных фрагментов кода, просто просматривая исходный текст.

Широкие возможности реализации параллелизма

Новейшая многоядерная технология QNX в сочетании с поддержкой симметричной многопроцессорности (SMP) позволяет реализовать истинный параллелизм во встраиваемых системах. Профайлер приложений помогает быстро ответить на вопрос “В каких точках можно оптимизировать систему для работы в многоядерном режиме?” посредством выявления тех частей кода, которые могут подходить для реализации параллелизма. После первоначального переноса кода на многоядерную систему разработчики могут быстро выделить код для параллелизации на основе анализа, выполненного с помощью профайлера приложений. Например, высокоактивную подпрограмму обработки сигналов можно разбить на несколько многопотоковых подпрограмм для параллельного исполнения на SMP-ядре (например, ОСРВ QNX Neutrino).

Анализ покрытия кода

Ускорьте этапы оптимизации, тестирования и контроля качества программного продукта в процессе разработки. Инструмент анализа покрытия кода, входящий в состав профессионального комплекта разработчика QNX Momentics®, реализует специальную методологию для обеспечения качества программного кода и целостности продуктов.

Данный инструмент позволяет выделить ветви исходного кода, не прошедшие тестирование, и при необходимости, модифицировать или удалить этот сегмент.

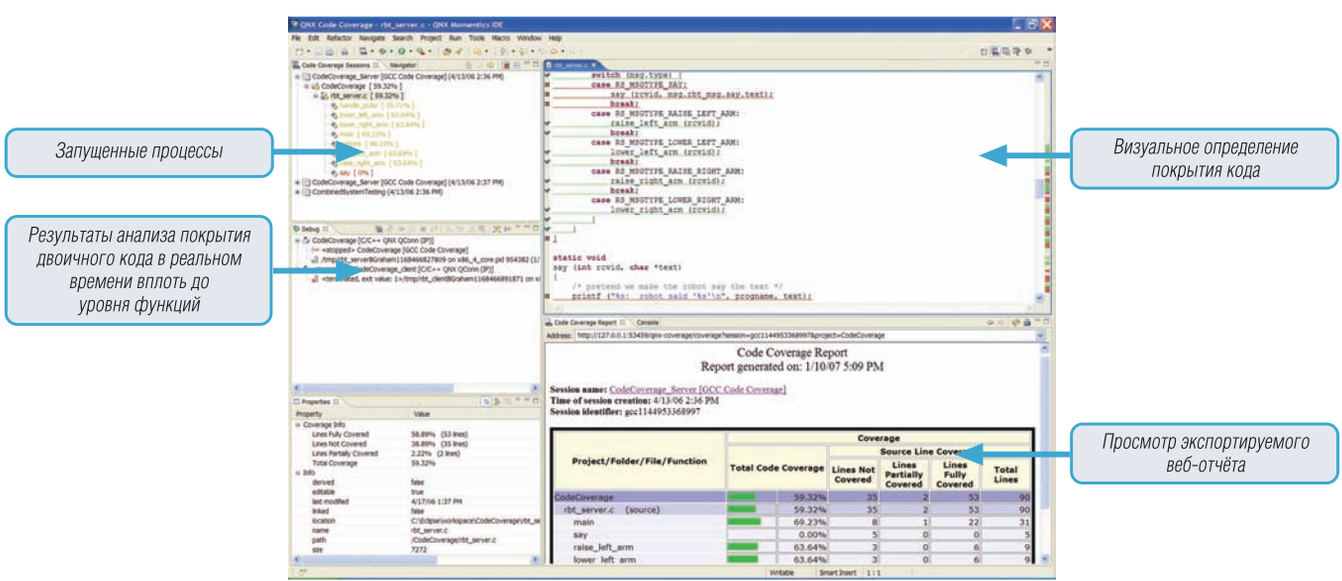
Благодаря полной интеграции этого инструмента со средой разработки, оптимизация, тестирование и контроль качества программного кода становятся значительно проще.

Анализ покрытия кода – важнейший инструмент в тех случаях, когда тестирование, исправление ошибок и сопровождение программного продукта осуществляется разными группами инженеров, которые могли и не участвовать в разработке кода.

Инструмент анализа покрытия кода позволяет:

- запускать сессию анализа покрытия кода и непосредственно наблюдать работу приложения в процессе его выполнения;
- получать результаты анализа покрытия двоичного кода в реальном времени вплоть до уровня основных блоков (ветвей кода);
- запускать редактор кода для быстрого определения, какие строки кода были покрыты и какие нет;
- использовать все инструменты навигации и комментирования, имеющиеся в среде разработки;
- наблюдать за покрытием кода одновременно нескольких приложений;
- генерировать отчёты для последующего анализа.

Анализ покрытия кода



Анализатор кода QNX Momentics позволяет проводить точное и полное тестирование программного обеспечения.

Средства работы с целевой системой

QNX Momentics® предоставляет полный набор инструментов для начальной загрузки и взаимодействия с целевым оборудованием. В этот инструментарий входят пакеты поддержки для широкого спектра процессорных плат, построитель встраиваемых систем, позволяющий быстро формировать и настраивать целевые образы, и уникальный расширяемый целевой агент, обеспечивающий одновременное выполнение множества задач для настройки целевой системы, в том числе, отладки, профилирования и сбора системной информации.

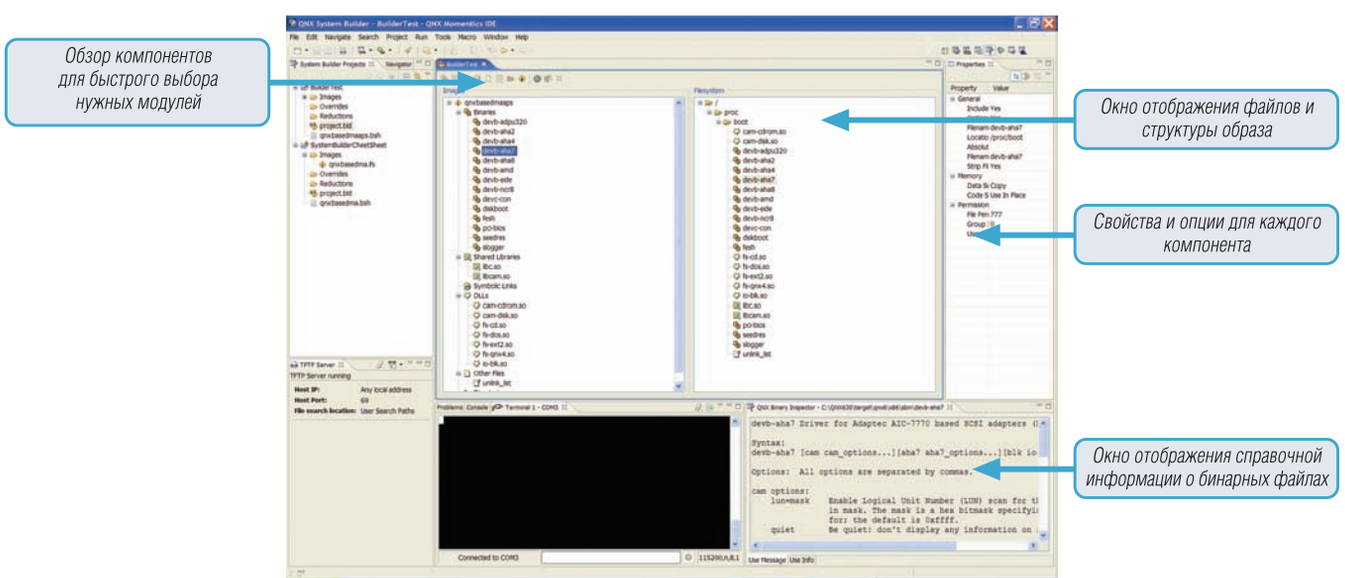
Построитель встраиваемых систем

Построитель встраиваемых систем позволяет существенно сэкономить время при создании загрузочных образов и флеш-образов встраиваемых файловых систем для ваших целевых систем. Для каждого создаваемого образа построитель встраиваемых систем позволяет выполнять следующие задачи:

- автоматизировать создание новых BSP-проектов в среде разработки;
- импортировать существующее описание образа из BSP-пакета или создать свой собственный файл описания;
- использовать браузер компонентов образа для быстрого выбора нужных двоичных модулей, динамически подключаемых и других библиотек;
- проверять взаимные зависимости библиотек и получать предупреждения об отсутствующих компонентах;
- гарантировать применение последней версии проекта в образе;
- сокращать объём памяти, занимаемый приложением, с помощью исключения ненужных функций из разделяемых библиотек.

Для переноса полученного образа на целевую систему в построителе встраиваемых систем имеется встроенный последовательный терминал, который может взаимодействовать с удалёнными мониторами ПЗУ. Он также может пересылать образы по TFTP или BOOTP. Запустив образ на целевой системе, вы сможете внести дополнительные изменения и переслать файлы при помощи любого из доступных механизмов. Например, вы можете редактировать файлы прямо на целевой системе, используя редактор среды разработки.

Построитель встраиваемых систем



Построитель встраиваемых систем значительно сокращает время создания, оптимизации и передачи целевых образов.

Агент целевой системы

Агент целевой системы предоставляет среду разработки с расширяемым механизмом взаимодействия с одной или несколькими целевыми системами. Каждый инструмент в составе среды разработки может взаимодействовать с целевой системой через агента, запуская на ней нужный модуль при запросе соответствующей службы с инструментальной машины. Когда служба больше не требуется, модуль можно выгрузить, чтобы освободить ресурсы. Как и любой другой драйвер в QNX, агент целевой системы может быть при необходимости запущен или выгружен в любой момент.

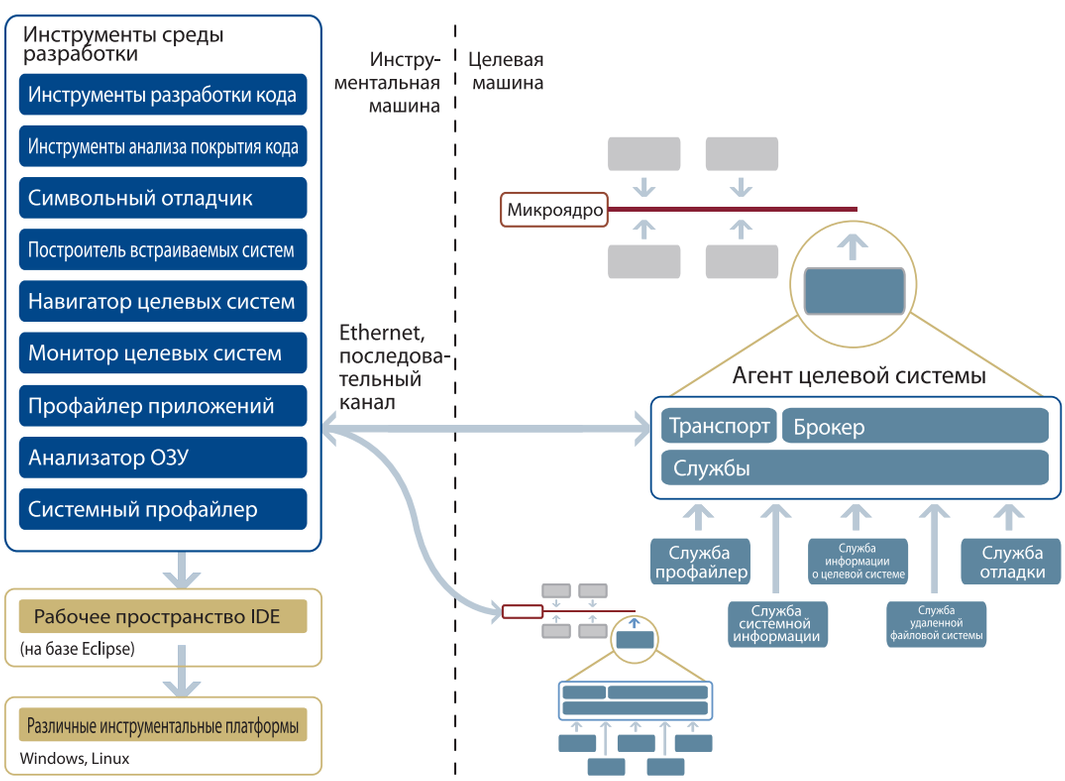
Навигатор целевых систем

Навигатор целевых систем позволяет всем инструментам среды разработки согласованно взаимодействовать с целевыми системами. Вы можете использовать этот навигатор для определения целевых систем и подключения к ним. С помощью навигатора целевых систем вы также можете связать текущий проект (например, образ системы) с той или иной целевой системой. Кроме того, навигатор целевых систем позволяет отобразить доступные аппаратные компоненты и устройства целевой системы.

После определения целевой системы навигатор целевых систем позволяет выбрать целевой компонент для применения последующих операций, например запуска telnet-сессии.

Новые инструменты могут использовать отображаемый целевой компонент как точку расширения, чтобы добавить свои операции в контекстное меню целевых систем. Например, такими операциями могут быть отправка сигнала процессу, выполняемого на целевой системе, подключение отладчика к выполняемому процессу.

Агент целевой системы



Агент целевой системы работает как брокер соединений, позволяя навигатору целевых систем (и другим инструментам среды разработки) взаимодействовать с одной или несколькими целевыми системами.

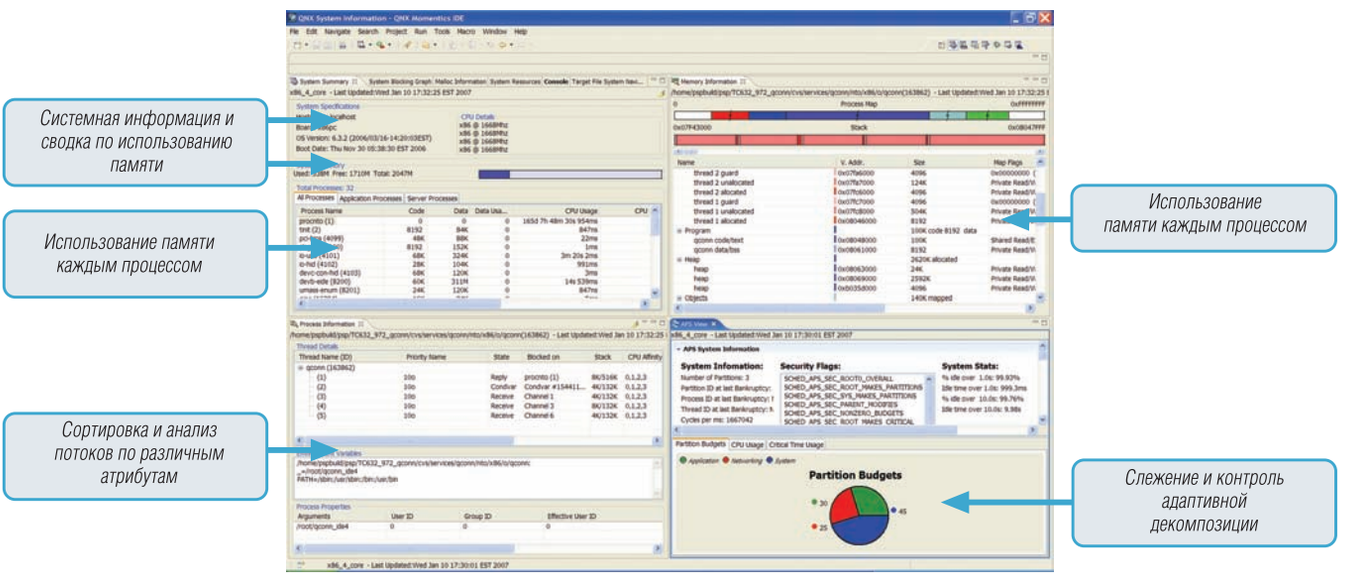
Монитор целевых систем

Монитор целевых систем предоставляет огромный объём информации о системах и процессах и генерирует отчёты (как в реальном времени, так и в "посмертном" варианте) для выбранной целевой системы. При выборе целевой системы сразу же происходит обновление всей информации о текущих процессах в ней. Вы можете начать с общего обзора системы, просмотрев список выполняющихся процессов и их аргументы. Вы также можете наблюдать за использованием процессора и памяти, как для системы в целом, так и для каждого процесса в отдельности. Затем вы можете получить более детальный анализ с помощью инструментов, позволяющих отследить атрибуты потоков (например, состояние, дисциплину планирования, использование процессора, размер стека), состояния сигналов, карты памяти программы, файловые дескрипторы и т. д.

Монитор целевых систем позволяет:

- отслеживать интенсивность использования ресурсов (например, расход памяти и ресурсов процессора, количество открытых файлов), в реальном времени наблюдая за процессами и потоками;
- выявлять потенциальные ситуации взаимных блокировок с помощью графической диаграммы отношений блокирования между процессами;
- управлять файлами на удалённой целевой системе напрямую с инструментальной машины;
- легко копировать файлы на удалённую целевую систему и редактировать их при помощи редактора напрямую из среды разработки;
- расширять возможности загрузчика программ для запуска исполняемых модулей на целевой системе простым двойным щелчком мышью.

Монитор целевых систем



Монитор целевых систем собирает полную информацию об атрибутах потоков, использованию процессора, открытых файловых дескрипторах, блокировках и множестве других параметров.

Photon Application Builder (PhAB)

С помощью Photon Application Builder (графического инструмента разработки приложений для оконной системы QNX Photon® microGUI®) вы можете легко, одной мышью создавать полнофункциональные пользовательские интерфейсы. Фактически, вам не приходится писать никакого кода – построитель приложений может автоматически сгенерировать полностью работающий прототип графического интерфейса вашего приложения.

С помощью Photon Application Builder вы можете:

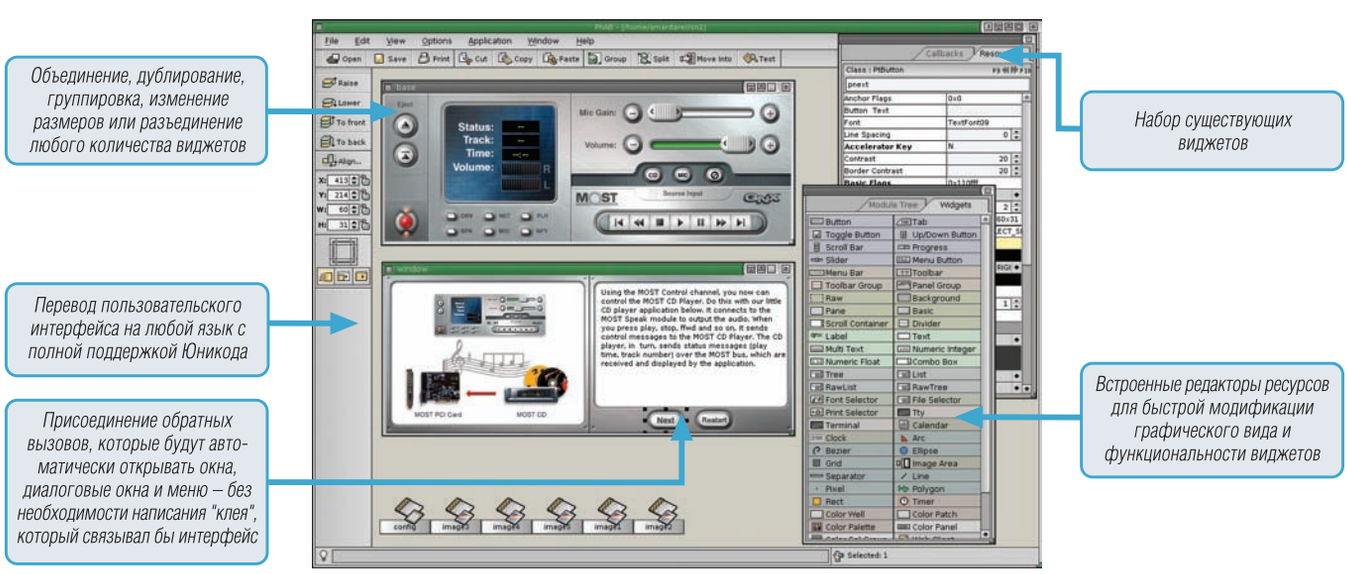
- быстро начать разработку, взяв за основу существующие шаблоны, либо строить пользовательский интерфейс из обширной палитры готовых элементов управления (виджетов);
- привязывать диалоговые окна и меню непосредственно к виджетам или добавлять вызовы функций;
- динамически генерировать копии ранее созданных виджетов в любом необходимом для приложения количестве;
- переводить пользовательский интерфейс на различные языки без перекомпиляции и перекомпоновки кода; версии для всех языков могут использовать один и тот же исполняемый модуль.

Photon Application Builder предоставляет компоненты пользовательского интерфейса для самых различных приложений: от карманных устройств до операторских терминалов.

Базовые компоненты	Дополнительные компоненты
Кнопки	HTML
Контейнеры	Терминал
Измерительные шкалы	Выбор шрифта
Векторная графика	Выбор принтера
Изображения	Flash
Списки	Контейнер видео
Деревья	
Меню	
Ярлыки и текст	

Примечание: построитель приложений поддерживается только на инструментальной платформе QNX Neutrino.

Построитель приложений



С помощью построителя приложений вы можете создавать полнофункциональные графические интерфейсы для ваших приложений, не написав ни одной строчки кода.

Комплекты разработки драйверов

С помощью комплектов разработки драйверов (DDK) вы можете быстро создавать драйверы для нестандартного оборудования – аудио-, графических и сетевых адаптеров, устройств ввода, принтеров, символьных и USB-устройств.

Комплекты содержат готовый программный каркас для написания администраторов ресурсов и классов драйверов устройств, а также включают в себя детальную документацию и исходные тексты. Программный каркас драйверов реализует весь высокоуровневый аппаратно-независимый код в виде библиотек, поэтому вам остаётся сосредоточиться только на аппаратно-зависимом коде для микросхемы, используемой в вашем устройстве. Даже если вам понадобится разработать драйвер для устройства нового типа, для которого не существует DDK, вы сможете использовать этот каркас как основу для быстрого старта.

Отлаживайте драйверы в исходном тексте с помощью обычных инструментов IDE

Поскольку в QNX драйверы выполняются как обычные пользовательские процессы, их можно отлаживать и оптимизировать при помощи того же интегрированного инструментария, который в QNX Momentics служит для отладки обычных приложений. Нет никакой необходимости применять отладчики на уровне ядра, так как это может застопорить работу всей целевой системы и, в результате, скрыть ошибки в коде.

Более того, микроядерная архитектура QNX Neutrino позволяет тестировать изменения в коде драйверов без перезагрузки системы и даже без перезапуска сеанса отладки – просто перекомпилируйте и перезапустите драйвер.

Отлаживайте драйверы прямо на своём компьютере

Если вы предпочитаете резидентную модель разработки и программируете непосредственно в среде QNX Neutrino, вы можете спокойно тестировать и отлаживать драйверы прямо на своей инструментальной машине. Драйверы выполняются в защищенной области памяти, поэтому вы можете применять стандартные инструменты отладки исходного кода на той же самой машине. Более того, исходный код драйверов является совместимым, поэтому в него не требуется вносить изменения, чтобы компилировать для разных процессорных архитектур.

Архитектура DDK-комплектов

Комплекты разработки драйверов сокращают процедуру написания драйверов до минимума – во многих случаях большинство вашей работы уже сделано.

Операционная система реального времени QNX Neutrino

Основой профессионального комплекта разработчика QNX Momentics® является ОС реального времени QNX Neutrino, которая обладает уникальной репутацией, подтверждённой годами бесперебойной работы – 24 часа в сутки, 365 дней в году, без остановки. Что делает ОС QNX Neutrino столь надёжной? Ответ прост: это настоящая операционная система на основе микроядра.

В QNX Neutrino ядро обрабатывает только базовые примитивы ОС. Все остальные компоненты – драйверы, файловые системы, стеки протоколов, пользовательские приложения – выполняются вне пределов ядра как отдельные процессы, каждый в своём защищённом адресном пространстве. Такой подход автоматически обеспечивает системам на основе QNX Neutrino "встроенную" отказоустойчивость.

Не менее важной особенностью является и то, что все компоненты QNX Neutrino взаимодействуют друг с другом через единый, чётко детерминированный механизм – синхронный обмен сообщениями. Этот механизм образует между компонентами системы виртуальную "программную шину", позволяющую подключать к ней или, наоборот, отключать любой компонент "на лету". Более того, сообщения могут свободно передаваться между узлами вычислительной сети, предоставляя прозрачный доступ к любому ресурсу, где бы он ни находился.

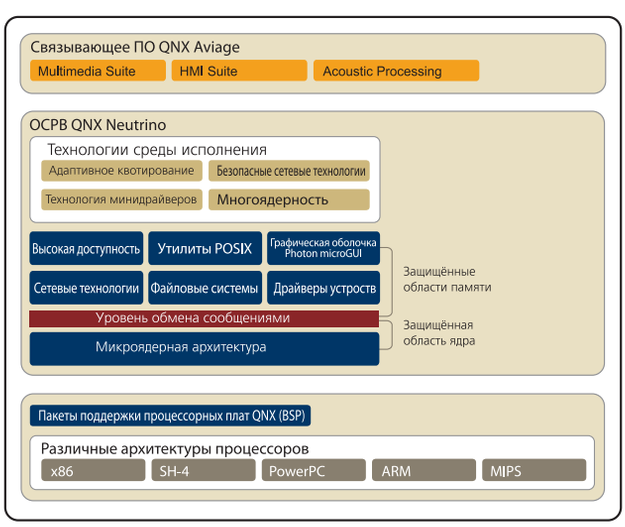
Используя QNX Neutrino, вы можете осуществлять следующие задачи.

- **Создавать самовосстанавливающиеся системы.** В QNX Neutrino любой компонент в случае отказа может быть перезапущен динамически, не нарушая работу микроядра и других компонентов. Например, если драйвер пытается обратиться к памяти за пределами своего адресного пространства (что для большинства ОС является фатальной ошибкой), QNX Neutrino корректно завершит этот драйвер и освободит все занятые им ресурсы. Вы можете даже автоматически перезапустить этот драйвер, используя монитор ключевых процессов QNX Neutrino.

- **Применять одну и ту же ОС во всей своей линейке продуктов.** Благодаря исключительной модульности QNX Neutrino, любые уже испытанные и проверенные компоненты – драйверы, приложения, дополнительные сервисы ОС – можно использовать повторно в других продуктах. Фактически, вы можете создать универсальный набор бинарных модулей и затем применять его либо в однопроцессорном устройстве, либо в SMP-системе, либо в вычислительном кластере. Независимо от масштаба и сложности разрабатываемой системы вы сможете использовать одну и ту же ОС, один и тот же программный интерфейс приложения (API) и один и тот же инструментарий разработчика.

- **Производить модернизацию систем, не останавливая их.** Поскольку практически любой компонент в QNX Neutrino может быть добавлен или удалён динамически, ваша система может продолжать работу даже во время замены или добавления в неё новых приложений, драйверов или стеков протоколов.

ОСРВ QNX Neutrino



Благодаря микроядерной архитектуре с защитой памяти, ОСРВ QNX Neutrino обеспечивает надёжную основу для создания систем, способных автоматически обнаруживать ошибки и самовосстанавливаться, выполнять динамическую модернизацию и обладать широкой масштабируемостью.

Краткий обзор QNX Momentics

От инструментов встраивания на процессорную плату до инструментов системного анализа – QNX Momentics включает в себя всё, что вам необходимо:

ОС реального времени QNX Neutrino

- Высоконадёжная архитектура на основе микроядра
- Прозрачные распределённые вычисления
- Симметричная многопроцессорность
- Отладочная версия микроядра
- Оконная система QNX Photon microGUI
- Сотни утилит POSIX, UNIX и QNX

Полностью интегрированная Среда разработки

- Платформа Eclipse как основа
- Средства разработки кода на C, C++, embedded C++
- Средства управления версиями
- Анализ покрытия кода
- Символьный отладчик
- Построитель встраиваемых систем
- Монитор целевых систем
- Агент целевой системы
- Профайлер приложений
- Анализатор ОЗУ
- Системный профайлер

Различные инструментальные платформы

- QNX Neutrino 6.5
- Windows 2000, XP, Vista, 7
- Red Hat Enterprise Linux Workstation 5.4
- Red Hat Fedora 12
- Ubuntu Workstation 9.10
- openSUSE 11.2

Библиотеки и инструментарий GNU

- ANSI C
- Dinkum C++, Embedded C++
- Оптимизирующие компиляторы GCC v4.4
- Компилятор Intel C++ Compiler
- GDB 6.7

Построитель приложений

- Инструмент графической разработки для создания полнофункциональных пользовательских интерфейсов
- Готовые шаблоны и виджеты для разработки пользовательских интерфейсов
- Перевод пользовательских интерфейсов на различные языки без перекомпиляции и перекомпоновки кода

Пакеты поддержки процессорных плат (двоичная форма)

- Для популярных процессорных плат на основе ARM, MIPS, PowerPC, SH-4, x86

Комплекты разработки драйверов

- Для аудио-, графических и сетевых адаптеров, устройств ввода, принтеров, символьных и USB-устройств

Документация

- Контекстно-ориентированная справка и тысячи страниц онлайновой документации

Системные требования

- Pentium IV 2ГГц и выше
- не менее 512 Мб ОЗУ
- не менее 3 Гб дискового пространства
- разрешение экрана не менее 1024x768

О компании QNX Software Systems

QNX Software Systems является одной из дочерних компаний Research in Motionи занимает лидирующую позицию в области технологий операционных систем реального времени. Такие ведущие компании мира, как Cisco, DaimlerChrysler, General Electric, Lockheed Martin и Logitechs, применяют технологии QNX для создания комплексов сетевой маршрутизации, медицинского оборудования, транспортных телематических систем, систем безопасности и обороны, промышленной робототехники и других приложений критического назначения и жизнеобеспечения. Компания QNX Software Systems была основана в 1980 году. Её штаб-квартира расположена в Оттаве (Канада), а продукция компании распространяется более чем в 100 странах мира.

SWD Software Ltd.
Платиновый дистрибьютор компании QNX Software Systems
196210, Санкт-Петербург, ул. Внуковская, 2,
БЦ "Пулково Скай", офис С-507.
Тел: (812) 611-07-51, 611-07-59, факс: (812) 611-07-58
info@swd.ru ▪ www.swd.ru

QNX® Momentics®

КОМПЛЕКТ РАЗРАБОТЧИКА