**QNX**

**QNX** ist ein [proprietäres](https://de.wikipedia.org/wiki/Propriet%C3%A4r) [POSIX](https://de.wikipedia.org/wiki/Portable_Operating_System_Interface) (eine für [Unix](https://de.wikipedia.org/wiki/Unix) entwickelte [standardisierte](https://de.wikipedia.org/wiki/Standard) [Programmierschnittstelle](https://de.wikipedia.org/wiki/Programmierschnittstelle))-fähiges [unixoides](https://de.wikipedia.org/wiki/Unixoides_System) [**Echtzeitbetriebssystem**](https://de.wikipedia.org/wiki/Echtzeitbetriebssystem), das primär auf den Embedded-System-Mark gerichtet ist.  
  
Das Produkt wurde ursprünglich in den frühen 1980er Jahren von der kanadischen Firma **Quantum Software Systems** entwickelt, später in **QNX** Software Systems umbenannt und letztendlich von BlackBerry im Jahr 2010 übernommen. Es war eines der ersten kommerziell erfolgreichen **Microkernel-Betriebssysteme** und wird in einer Vielzahl von Geräten wie **Autos** und **Mobiltelefonen** verwendet.

Als ein [**Mikrokernel**](https://de.wikipedia.org/wiki/Mikrokernel)**-**[**Betriebssystem**](https://de.wikipedia.org/wiki/Betriebssystem) **(**auch **µ-Kernel** oder **Mikrokern**, ist ein [Betriebssystemkern](https://de.wikipedia.org/wiki/Kernel_(Betriebssystem)), der im Gegensatz zu einem [monolithischen Kernel](https://de.wikipedia.org/wiki/Monolithischer_Kernel) nur grundlegende Funktionen erfüllt – in der Regel sind dies Speicher- und Prozessverwaltung**)** basiert QNX auf der Idee, den größten Teil des Systems in Form von Prozessen laufen zu lassen.

Dabei läuft jeder Prozess in einem eigenen, durch die **MMU (**Speicherverwaltungseinheit**)** des Prozessors geschützten Speicherbereich – egal ob es sich dabei um eine Applikation oder um einen Treiber handelt. Versucht **Prozess A** Daten oder Code von **Prozess B** zu überschreiben, wird der **QNX-Microkernel** über die **MMU** darüber informiert und beendet **Prozess A**, während **Prozess B** unberührt bleibt, das kann Entwicklern bei der Fehlersuche helfen. Diese Fähigkeit ist das Hauptunterscheidungsmerkmal im Vergleich mit traditionelleren [**monolithischen Kerneln**](https://de.wikipedia.org/wiki/Monolithischer_Kernel) (Kernel, wo auch [Treiber](https://de.wikipedia.org/wiki/Ger%C3%A4tetreiber) für die [Hardwarekomponenten](https://de.wikipedia.org/wiki/Hardware) und weitere Funktionen direkt eingebaut sind), wo das Betriebssystem ein großes Programm mit speziellen Fähigkeiten ist.

Bei QNX erlaubt der Microkernel dem Nutzer (Entwickler), alle von ihm nicht benötigte Funktionalität (Audio, Grafik und andere Systemteile …) wegzulassen, ohne den Kernel anfassen zu müssen.

Ein weiterer **Vorteil der Microkernelarchitektur** ist die Möglichkeit, selbst essentielle Systemtreiber während des laufenden Betriebes auszutauschen.

Bei behutsamer Vorgehensweise (Berücksichtigung der Abhängigkeiten) können auf diesem Wege sogar **Systemkomponenten ohne Neustart** des Gesamtsystems **ausgetauscht** werden. Ein System besteht aus vielen einzelnen **Modulen**, die jeweils dynamisch geladen werden. Um trotz des Speicherschutzes durch die MMU eine möglichst effiziente Kommunikation zwischen den Prozessen zu ermöglichen enthält QNX seit der ersten Version umfangreiche Mechanismen zur Prozesskommunikation ([IPC](https://de.wikipedia.org/wiki/Interprozesskommunikation) – verschiedene Verfahren des Informationsaustausches). Im Laufe der Entwicklung der verschiedenen Versionen wurden diese Mechanismen ausgebaut.