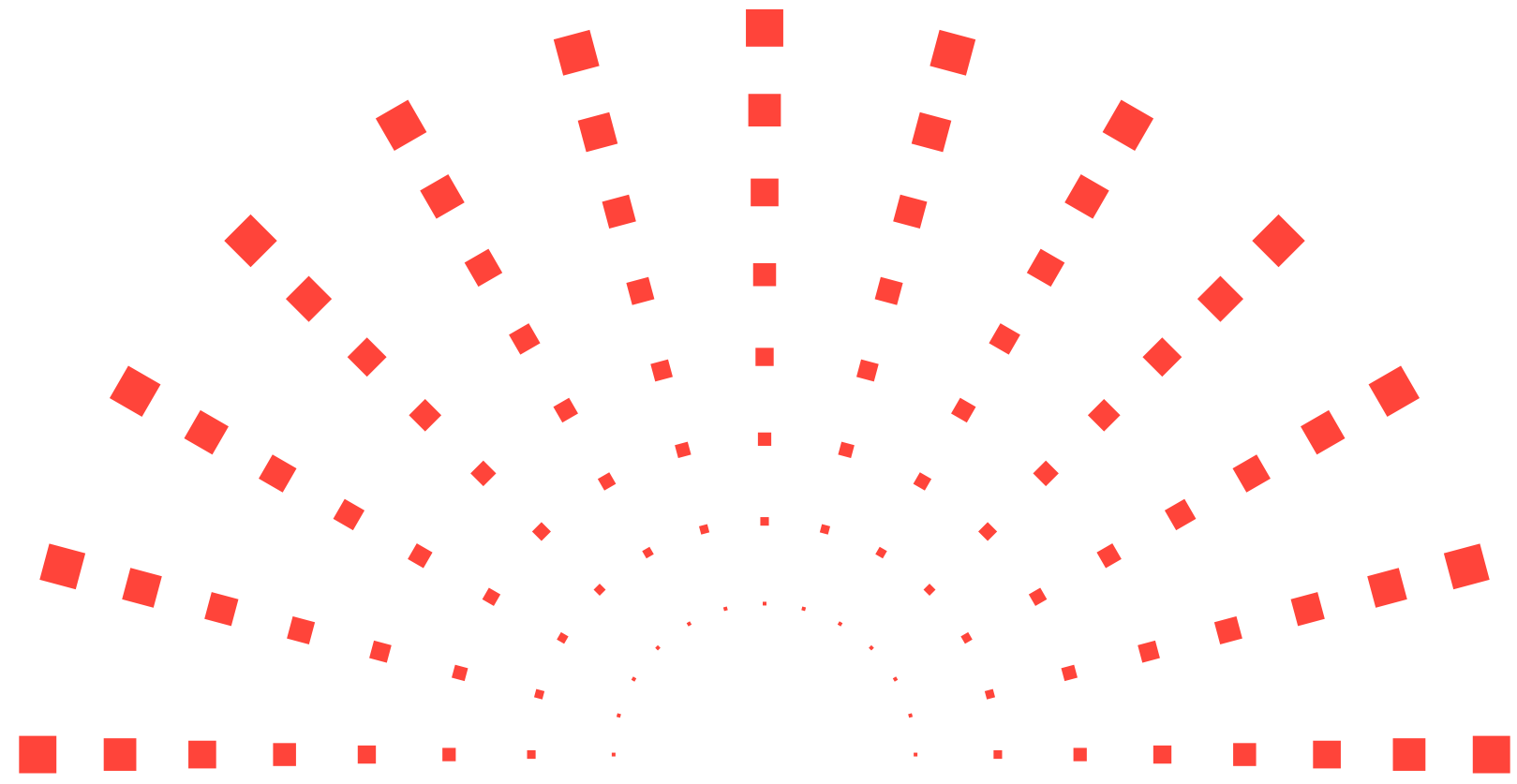


White Paper

# Designing Safe, Secure Medical Devices:

Building with NXP i.MX 9 Applications  
Processors and QNX OS for Safety





Imagine a surgeon carefully dissecting tissue near a critical nerve during a minimally invasive procedure. Every movement of the surgeon's hand at the console must be translated into precise motion at the robotic instrument. At the same time, the surgeon is relying on high-definition video, instrument position data, foot-pedal inputs, force limits, safety monitoring and system alerts, all working together as a single, cohesive system.

From the surgeon's perspective, the robot simply needs to respond exactly as expected. There can be no hesitation, no unexpected behavior, and no uncertainty about what the system is doing. When the surgeon moves a control, the instrument must move when expected, by the amount expected, every time.

Delivering that experience is as much a software challenge as it is a hardware challenge. Modern surgical robots depend on thousands of software functions operating simultaneously while maintaining deterministic real-time performance. Motion control, imaging, user interfaces, safety monitoring, communications and data logging must all execute predictably without interfering with the control system responsible for the robot's movement.

In many robotic systems, this means maintaining control loops running at very high frequencies. Every microsecond, the system must process inputs, calculate desired motion, enforce safety constraints and issue commands to the robotic arms. Missing deadlines or introducing unpredictable delays may not be noticeable in a conventional computing application, but in a robotic surgery system, predictable timing is a fundamental design requirement.

As medical systems become increasingly connected, intelligent and software-defined, manufacturers must deliver more capability while preserving the deterministic behavior that clinicians depend on. Safety, cybersecurity, and real-time performance are therefore not independent engineering objectives. They are interconnected requirements that must be designed into the foundation of the system from the earliest stages of development.



## Connected Medical Devices Raise The Architectural Stakes

Today's medical devices and surgical robots are no longer isolated machines. They have the ability to integrate advanced imaging systems, real-time analytics, hospital networks, remote service capabilities, electronic medical records, and increasingly sophisticated software applications. These capabilities help clinicians make better-informed decisions, improve workflows, and create opportunities for future innovations.

But every new capability introduces additional complexity. More software means more interactions between applications. Richer visualization, AI-assisted analysis, cybersecurity services and network communications all compete for processing resources. Left unmanaged, these workloads can introduce variability into the system and affect the timing predictability required for robotic control.

The challenge for device manufacturers is not simply adding features. It is ensuring that critical functions remain isolated and deterministic regardless of what else the system is doing. A software update, a cybersecurity scan, a network transaction or a non-critical user-interface process must never compromise the performance of the motion-control system responsible for executing the surgeon's commands.

This is why architecture matters. The underlying platform must provide strong isolation between applications, deterministic scheduling of critical workloads, and the ability to detect and contain faults before they can impact essential system functions. These same architectural foundations also support cybersecurity objectives by limiting the ability of a compromised component to affect the rest of the system.

Regulatory compliance remains an important part of the process, but compliance is not the only goal. The overarching objective is to ensure patient safety and maintain the clinician's trust. Every procedure depends on the assumption that the system will behave consistently, predictably and safely, regardless of workload, operating conditions or unexpected events. Achieving that level of confidence requires more than innovative hardware. It requires software architecture designed from the ground up for reliability, determinism, and resilience.

# Start With Risk, Not Technology

When developing a medical device, the first question should not be which processor, operating system or security technology to use. The first question should be: what could go wrong, and what would the consequences be for the patient, clinician, and procedure?

Take a surgical robot for example. It is a complex interaction of robotic arms, control software, imaging systems, user interfaces, communications networks, safety mechanisms, and human operators. Each component introduces potential hazards, and many of the most significant risks emerge from the interactions between these systems rather than from any individual component alone.

What happens if a robotic arm receives an incorrect command? What if imaging data is delayed? What if a software fault impacts the control loop? What if a cybersecurity event occurs during a procedure? What if a clinician is presented with incomplete or outdated information? These questions help define the system requirements long before architectural decisions are made.

Risk-based development therefore follows a practical progression:

- Understand the clinical workflow and how clinicians interact with the system.
- Identify hazards across the complete device architecture, including hardware, software, networking, security, and human-machine interactions.
- Assess the potential impact of those hazards on patient safety and system operation.
- Translate the resulting risks into technical requirements and safety mechanisms.
- Select technologies and components that can demonstrably meet those requirements.

- Verify, validate, and continuously monitor the system throughout its lifecycle.

Recognized standards provide structure for this process. IEC 62304 defines medical device software lifecycle activities; ISO 14971 establishes the framework for risk management, and ISO 13485 governs quality management systems. Product-specific standards such as IEC 60601-1 provide additional guidance for medical electrical equipment.

Ultimately, however, standards are only part of the story. The manufacturer must have a complete understanding of the system architecture and the dependencies between all components and the consequences when things do not behave as expected. Only then can manufacturers make informed decisions about partitioning workloads, isolating critical functions, implementing cybersecurity protections, and ensuring deterministic real-time performance.

The device manufacturer remains accountable for the safety and effectiveness of the final product. However, carefully selecting hardware and software suppliers that provide rigorous development processes, certification evidence and supporting artifacts can significantly reduce development effort and program risk. Pre-certified components do not certify the complete medical device, but they do provide trusted building blocks that help manufacturers concentrate their engineering effort on the unique clinical capabilities that differentiate their system.

# Key Medical-device Standards and Their Roles

## Software Lifecycle



### IEC 62304

Defines processes, activities, and tasks for safe medical device software development

## Risk Management



### ISO 14971

Establishes hazard identification, risk analysis, and risk control practices

## Quality Management



### ISO 13485

Governs organizational processes for medical device quality systems

## Product Safety



### IEC 60601-1 (pt safety) / 61010-1 (IVD)

Defines safety requirements for medical electrical equipment



# Functional Safety Requires Cybersecurity

Understanding the hazards within a medical system is only part of the challenge. Once engineers identify the safety functions required to protect patients and clinicians, they must also ensure those functions cannot be compromised, intentionally or otherwise.

This is where functional safety and cybersecurity converge.

Functional safety focuses on maintaining or restoring safe operation when a fault or malfunction occurs. In a surgical robot, that might include enforcing motion limits, stopping movement when abnormal conditions are detected, maintaining safe system states, or preventing a fault in one subsystem from affecting another.

Cybersecurity addresses a different class of risk. Its purpose is to protect the system from unauthorized access, manipulation, and other external threats that could alter the intended behavior of the device.

Traditionally, these disciplines were often treated separately. In today's connected medical devices, they are increasingly inseparable.

Consider an infusion pump designed to detect an occlusion, stop medication delivery and alert a clinician. From a functional safety perspective, the system has behaved correctly when faced with an internal fault condition. However, if a security vulnerability allows an attacker to disable the alarm, modify operating parameters or override motor-control functions, the very mechanisms intended to keep the patient safe can no longer be trusted. The lesson applies equally across the medical-device landscape. A device cannot be considered truly safe if its safety mechanisms can be bypassed, disabled or manipulated.

For manufacturers, this means cybersecurity must be designed into the architecture from the beginning rather than added as a final compliance activity before product release. Security considerations now extend across the entire platform, including secure boot, authentication, authorization, software and data integrity, access control, secure software updates, vulnerability management and a full understanding of all software components.

Just as importantly, cybersecurity is not a point-in-time exercise. Many medical devices remain in service for years, sometimes decades. During that time, new vulnerabilities are discovered, software components evolve and threat landscapes change. Manufacturers must therefore view cybersecurity as an ongoing lifecycle responsibility, continuously assessing risk and maintaining protections long after the device has entered clinical use.

Ultimately, functional safety and cybersecurity share the same objective: ensuring the device continues to behave as intended, even when things go wrong. One protects against accidental failures. The other protects against deliberate ones. Modern connected medical devices require both.

# From Fault Detection To A Safe State

Identifying hazards and implementing safeguards is only part of the equation. A medical device must also know how to respond when something goes wrong.

In medical systems, functional safety is ultimately about three capabilities: continuously monitoring the system, detecting abnormal conditions, and transitioning the device into a predefined safe state within a known period. Consider a robotic procedure where the surgeon is controlling multiple robotic instruments while viewing a high-definition video feed. The system is constantly monitoring hundreds of conditions, including motion-control performance, communications between subsystems, sensor health, actuator status, user inputs, and software execution timing.

If a fault is detected, the question is no longer whether the system recognizes the problem. The critical question becomes: what happens next?

For example, if a critical component within the motion-control chain stops responding, the system may need to immediately prevent additional instrument movement, hold the robotic arms in a controlled state and clearly alert the surgeon that a fault has occurred. If communication between a robotic arm and its controller is lost, the appropriate response may be different. If a non-critical user interface process fails, the system may be able to continue operating safely while notifying the operator and activating a redundant function.

The correct response depends on the device architecture, intended use, and underlying risk assessment. What matters is that the response has been defined in advance, thoroughly validated, and can be executed predictably whenever required.

As noted above, this safe state response is part of functional safety design and applies across all medical devices.

For engineers developing safety-critical medical systems, response timing is equally important as fault detection itself. Detecting a fault seconds after it occurs may be too late. The system must identify abnormal conditions, initiate the appropriate response, and reach a safe state within the fault-tolerant time defined by the risk analysis. A medical device does not simply need to detect failures. It must react predictably, within known timing boundaries, and transition the system to a safe state before the fault can create risk for the patient or clinician. That ability to monitor, respond, and recover deterministically is a foundational requirement of modern medical-device design.

These principles only matter if the underlying platform can deliver them. Below, we'll explore how processor and operating-system architecture turn these requirements into reality.

# Translating Safety Requirements Into System Architecture

Once hazards have been identified, safe states have been defined, and system requirements have been established, the next challenge is deciding where those functions execute and how they are protected.

In a medical device, not every workload carries the same level of criticality. High-definition imaging, user interfaces, data recording, connectivity services, analytics and device control all place different demands on the system. Some functions are performance oriented. Others are safety critical. The architecture must ensure that lower-priority or non-critical activities cannot interfere with functions that are responsible for patient safety.

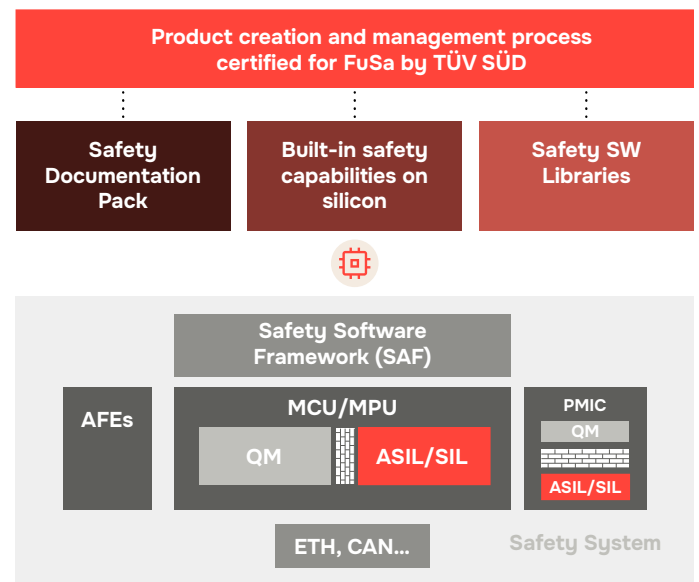
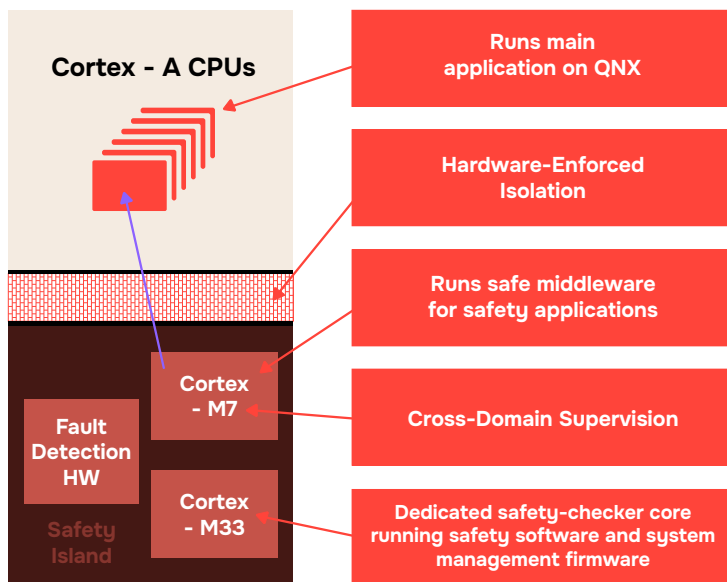
This is one reason manufacturers increasingly look for processing platforms designed to support both advanced applications and safety-oriented workloads.

The NXP i.MX 95 processor family provides a heterogeneous architecture that combines high-performance application processing with dedicated real-time and safety-oriented processing domains. Application workloads such as imaging, visualization and user interfaces can execute on the Cortex-A cores, while independent Cortex-M processing resources can be assigned to safety monitoring, supervision and system-management functions.

This architectural separation enables safety-critical functions to supervise and verify system behavior without depending entirely on the subsystem being monitored. Rather than relying on a single computing domain, manufacturers can implement independent monitoring paths capable of detecting faults, triggering recovery actions, and helping the system transition to a predefined safe state when required.

The processor also serves as a central hub for the broader medical-device workflow, connecting sensors, cameras, displays, communications interfaces, and control systems. With EdgeLock Secure Enclave integrated security capabilities, including secure boot, cryptographic services and hardware-protected key management, help establish the trusted foundation required for modern connected medical devices.

Of course, no processor alone creates a safe medical device. Safety and security are system-level properties. The hardware must provide a foundation upon which those capabilities can be built.



## Why The Operating System Matters

If the processor provides the foundation, the operating system determines how the system behaves when multiple software components operate simultaneously and, more importantly, how it behaves when something goes wrong.

This becomes particularly important in medical devices, where control, imaging, safety monitoring, cybersecurity services, and user-interface software may all be running concurrently. The challenge is not simply executing these functions. It is ensuring they execute predictably and do not negatively impact one another.

QNX OS for Safety 8.0 was designed with this challenge in mind.

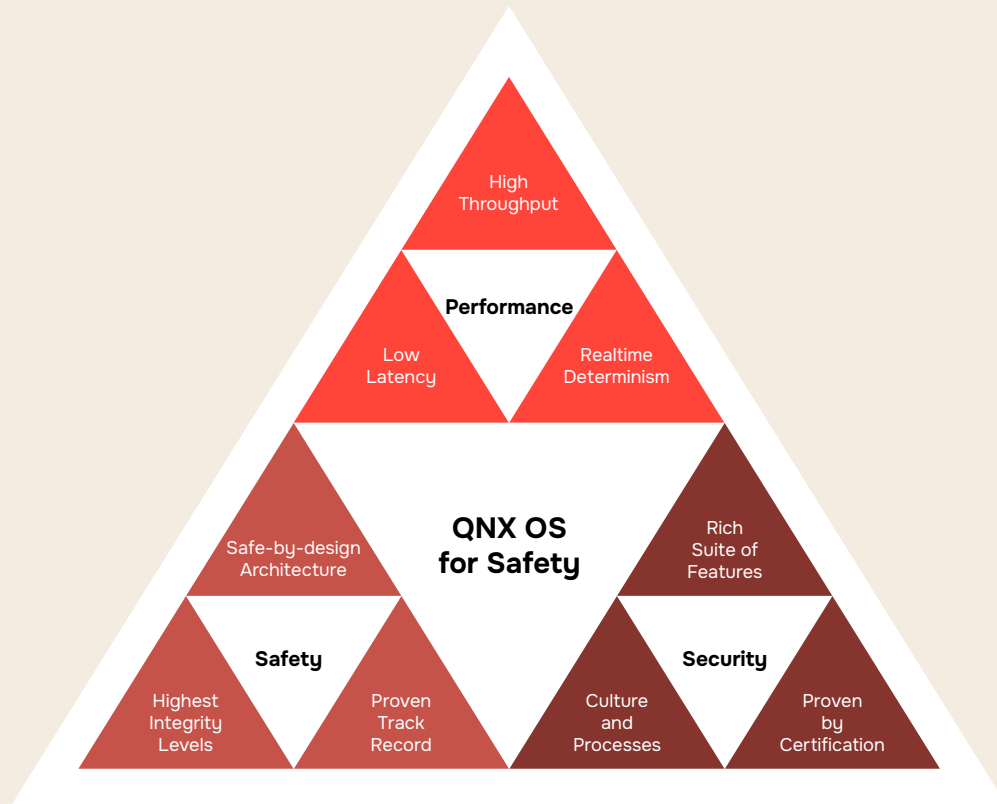
Unlike traditional monolithic operating systems, QNX uses a microkernel architecture that isolates drivers, file systems, networking stacks, and applications into separate protected processes. This architecture helps prevent failures in one component from propagating throughout the system. If a non-critical service encounters a fault, it can often be detected, contained, and restarted without affecting unrelated functions that continue to operate.

For medical-device developers, this architecture provides three particularly important benefits:

- Isolation helps prevent software components from unintentionally corrupting one another's memory or resources.
- Deterministic scheduling helps ensure that time-critical workloads, such as motion control and safety monitoring, receive processing resources when required.
- Fault containment and recovery help the system detect failures, isolate impacted services, and execute predefined recovery actions before safety is compromised.

These capabilities align with the requirements of medical systems, where maintaining predictable behavior is often just as important as delivering high performance.

As systems become more sophisticated, manufacturers are increasingly seeking platforms such as NXP and QNX that not only provide technical capabilities, but also include supporting safety documentation, development processes, certification evidence and long-term lifecycle support. This helps reduce development effort and allows engineering teams to focus their resources on clinical innovation rather than rebuilding foundational infrastructure.



## Build for The Complete Product Lifecycle

Modern medical devices are expected to do more than ever before. They process larger volumes of data, support richer user experiences, connect to external systems, and increasingly incorporate advanced automation and intelligence.

Yet despite this growing complexity, the fundamental requirement remains unchanged: the device must continue to behave safely, securely, and predictably.

This is where platforms such as NXP i.MX 9 series applications processors and QNX OS for Safety 8.0 become compelling. Together, they provide the architectural foundations needed to support deterministic operation, workload isolation, fault containment, and cybersecurity while allowing manufacturers to scale system capabilities over time.

Ultimately, though, technology selection should always trace back to managing risks.

Can the platform separate critical and non-critical functions? Can it continuously monitor safety-related activities? Can it detect faults and transition to a safe state within the required timeframe? Can suppliers provide the evidence, documentation, and long-term support needed to satisfy regulatory expectations and maintain the product throughout its operational life?

These are the questions that connect architecture to assurance.

When answered early, they help manufacturers build medical devices that not only meet today's requirements, but continue to operate safely, securely and reliably throughout the many years they remain in clinical use.

## About QNX

QNX, a division of BlackBerry Limited, enhances the human experience and amplifies technology-driven industries, providing a trusted foundation for software-defined businesses to thrive. The business leads the way in delivering safe and secure operating systems, hypervisors, middleware, solutions, and development tools, along with support and services delivered by trusted embedded software experts. QNX® technology has been deployed in the world's most critical embedded systems, including more than 275 million vehicles on the road today. QNX® software is trusted across industries including automotive, medical devices, industrial controls, robotics, commercial vehicles, rail, and aerospace and defense. Founded in 1980, QNX is headquartered in Ottawa, Canada.

**Learn more at [qnx.com](https://qnx.com) →**

©2026 BlackBerry Limited. Trademarks, including but not limited to BLACKBERRY and EMBLEM Design, QNX and the QNX logo design are the trademarks or registered trademarks of BlackBerry Limited, and the exclusive rights to such trademarks are expressly reserved. All other trademarks are the property of their respective owners. BlackBerry is not responsible for any third-party products or services.

## About NXP

NXP Semiconductors is the trusted partner for innovative solutions in the automotive, industrial & IoT, mobile, and communications infrastructure markets. NXP's "Brighter Together" approach combines leading-edge technology with pioneering people to develop system solutions that make the connected world better, safer, and more secure.

**Find out more at [www.nxp.com](https://www.nxp.com). →**

NXP and the NXP logo are trademarks of NXP B.V. All other product or service names are the property of their respective owners. All rights reserved. © 2026 NXP B.V.

