The Real-Time Edge Intelligence Solutions Handbook

A Practical Framework for Building Commercial Edge Systems based on Physics and Mathematics

Foreword by Joseph Yiu
Distinguished Engineer, Arm

Sanjeev Sarpal

REAL-TIME EDGE INTELLIGENCE SERIES

ISBN: 978-94-6526-559-9 (softcover)

First Edition — Version 1.0.1 (February 2026)

Published by:
Advanced Solutions Nederland BV (ASN)
www.advsolned.com

Printed by: Boekengilde BV, Enschede, the Netherlands

Foreword

Embedded computing is a fascinating area in the electronics industry. There is a huge diversity of embedded systems, all with different performance levels, interfaces and form factors.

Many of these intelligent embedded systems are well hidden within products and often go unnoticed, yet they play a key role in improving performance and ease of use. For example, an audio system inside a modern car provides sophisticated noise suppression capabilities and might also include voice command control features that allow the driver to access various features without taking their eyes off the road. Similarly, the intelligence in the latest wearable devices delivers much more information to the user. Nowadays, instead of just tracking exercise information, like the traditional fitness trackers, some of the latest smart watches provide a range of medical-related features. These could even alert the user that they are ill before they are actually aware of the fact.

The intelligence in these systems can be achieved using AI/ML techniques, DSP techniques, or combinations of both. Thanks to new technologies like Arm Helium technology in the Cortex-M processors, and in the Ethos-U Neural Processing Units (NPU), it is now possible to deliver an impressive amount of AI/ML and DSP processing power in tiny microcontrollers and low-power system-on-chips (SoCs). However, there are still a number of challenges that product designers need to overcome, particularly in the software design domain — and this is where Sanjeev's experience and this book's focus are especially valuable.

Unlike most books that either focus on just the DSP algorithms or the ML software frameworks, this book takes a system-level view, e.g. it shows the interactions between DSP and ML, which is an essential part of many 'Edge Intelligence' applications. It also illustrates how to implement Edge Intelligence using several practical examples. These examples cover a knowledge gap that is not covered by many other books. Additionally, this book also covers a range of product design topics, making it a great resource for new product developers.

When Sanjeev asked me to help review the content for this book, I immediately said, "yes" as I was sure that I would learn a lot from it. Sanjeev has worked with Arm technologies for many years and is not only an expert on DSP algorithms (he created the ASN Filter Designer software), but is also very experienced in applying DSP and AI/ML techniques in real-world applications. If you have time, check out the articles that he

has published (you can find the links on the ASN website or the various LinkedIn posts and articles). Many of these articles provide practical insights into deploying DSP and AI/ML technologies in real products — insights that go well beyond what is typically found in many textbooks and application notes.

I hope you will enjoy this book as much as I have.

Joseph Yiu
Distinguished Engineer, Arm

Acknowledgements

This book would not have been possible without the support, feedback, and insight of many individuals and organisations across industry and the wider embedded systems community.

I would like to express my sincere thanks to colleagues and reviewers from Arm, and the broader embedded and signal processing community for their technical feedback, encouragement, and constructive critique throughout the development of this work. Over the years, I have had the privilege of working with and learning from many exceptionally talented engineers and researchers — far too many to name individually — whose ideas and practical experience have shaped both this book and my wider work in this exciting field.

In particular, I would like to thank Dr. Jayakumar Singaram for his insights into AI and AIoT, and Joseph Yiu and Dan Boschen for their detailed feedback, technical depth, and diligent review of the chapters, helping to ensure the technical accuracy and quality of the material presented.

I am especially grateful to the Arm CMSIS team and the wider Arm developer community for their guidance and support over the years. While it is impossible to acknowledge everyone individually, I would certainly like to thank Christophe Favergeon and Fabien Klein for their practical insights and continued support, and Reinhard Keil for his encouragement.

Special thanks are due to the users of ASN Filter Designer for their continued feedback, real–world use cases, and practical insights, that directly shaped many of the examples and design decisions presented in this book.

Finally, my thanks to family, colleagues, and friends for their unwavering support throughout the long process of writing, revision and refinement.

— Dr. Sanjeev Sarpal

Contents

Acronyms and Definitions

ACC	Adaptive Cruise Control (Automotive)
ADC	Analog-to-Digital Converter
AES	Advanced Encryption Standard (typically, AES-256)
AFE	Analog front-End
AI	Artificial Intelligence
AIoT	Artificial Intelligence of Things
AoP	Antennas-on-package
Arm	Arm is a processor IP licensing company, derived from the name *Acorn RISC Machine*.
ASIL	Automotive Safety Integrity Level
ASNFD	ASN Filter Designer
AWS	Amazon Web Services
BLE	Bluetooth Low Energy
BIBO	Bounded-Input Bounded-Output
BLW	Baseline Wander
BOM	Bill of Materials
BPF	Bandpass Filter
BPM	Beats-per-minute (biomedical)
BSF	Bandstop Filter
BSP	Board Support Package
CAN	Controller Area Network

The Evolution of Intelligent Products

As we enter what is widely believed to be the fifth industrial revolution, it is prudent to reflect on the fact that intelligent products have actually existed for centuries. Even before advanced computing platforms ever existed, many products were already designed to follow rules, enforce timing, and respond to inputs in a predictable manner. For example, early mechanical systems such as musical organs driven by punch cards implemented fixed logic using mechanical hardware — these were followed by more advanced electromechanical systems, such as the famous *Colossus* code-breaking machine, that extended these principles into early forms of programmable computation.

As technology evolved, many of these functions were realised using analog electronics that provided reliable and efficient operation across a wide range of applications. However, in many domains, these *analog solutions* were later replaced or augmented by digital solutions — not because the underlying problems had changed, but because digital implementations enabled programmability and repeatable, production-consistent behaviour across large numbers of systems.

While the concept of embedding intelligence into products has remained consistent, the technologies used to realise it have evolved. Each major technological transition — from mechanical systems, to analog electronics, and later to digital implementations — expanded what could be sensed, controlled, and decided in real-time. As a result, intelligent products became progressively more capable, while preserving the core requirements of predictable and reliable behaviour.

One well-known example of this progression is factory automation. Industrial systems were built around deterministic control architectures implemented using analog electronics, relays, and later PLCs (programmable logic controllers). These systems enabled machines to operate reliably and predictably over long service lifetimes, and they remain in widespread use today. Similar design principles also appeared across many other domains, including test and measurement and early embedded products.

As digital computation became practical within embedded hardware, these systems gradually evolved into what we now recognise as embedded systems. With the emergence of early microcontrollers and Digital Signal Processors (DSPs), it became possible for the first time to perform numerical computation directly in software. This development did not replace established control architectures, but complemented them. Digital processing extended what could be measured, analysed, and interpreted in real-time, while preserving the predictable behaviour required by real-world systems.

As sensing became cheaper and connectivity more widespread, embedded platforms were increasingly networked, giving rise to what became known as the Internet of Things (IoT). Design priorities expanded from individual machines to fleets of connected devices, enabling remote monitoring, diagnostics, and system-level visibility. More recently, advances in Machine Learning (ML) and Artificial Intelligence (AI) have further influenced system design. In cloud environments, this has often involved large models trained on extensive datasets, but at the edge, the situation is fundamentally different.

Edge deployments operate under strict constraints, i.e. known latency, low power consumption, deterministic behaviour and long-term reliability. They rarely involve large models or large datasets. Instead, they rely on compact, task-specific models, or no learned models at all. In such environments, the effectiveness of any learning component is fundamentally bounded by the quality and physical correctness of the input data presented to it. Even when the underlying physical process remains unchanged, variations in noise or even sensor artefacts can cause the measured data to deviate from the training datasets. In such cases, no ML model can compensate for these deficiencies, as they must be addressed at the signals and systems level.

This book contends that *signals and systems* engineering remains the decisive factor in real-time edge intelligence. In practice, this is realised through DSP, which provides a systematic, scientifically grounded framework for interpreting sensor data. It enables noise reduction, timing control, spectral isolation, and the extraction of physically meaningful features in a manner that is predictable, analysable, and verifiable. These properties are not optional, as they remain essential in industrial, medical, automotive, and other compliance-driven domains shaped by IEC and ISO standards.

We further contend that intelligence at the edge is not synonymous with Artificial Intelligence. In many deployed systems, carefully engineered signal-chains combined with deterministic decision logic already constitute a form of intelligence. What is often described as *feature engineering* is, in practice, the explicit encoding of domain knowledge into a system—an approach that delivers robustness, transparency and dependable behaviour under real operating conditions. Learning-based methods can augment this process, but they do not replace it.

From this perspective emerges the concept of *Real-Time Edge Intelligence (RTEI).* RTEI unifies signals and systems reasoning, intelligent data conditioning, and the selective use of AI into a coherent engineering workflow. It recognises the strengths of ML models, while placing them within a disciplined framework that prioritises interpretability and predictable real-time behaviour. More importantly, it also recognises

that AI is optional, as many edge systems achieve excellent performance through carefully engineered DSP algorithms combined with well-chosen decision thresholds.

Figure 1 provides a high-level view of the evolution of intelligent products over the past five decades. The diagram shows how early embedded systems based on PLCs, dedicated hardware, and early microcontroller (MCU) and DSP platforms gradually evolved into connected and AI-enabled architectures, culminating into today's Real-Time Edge Intelligence systems.

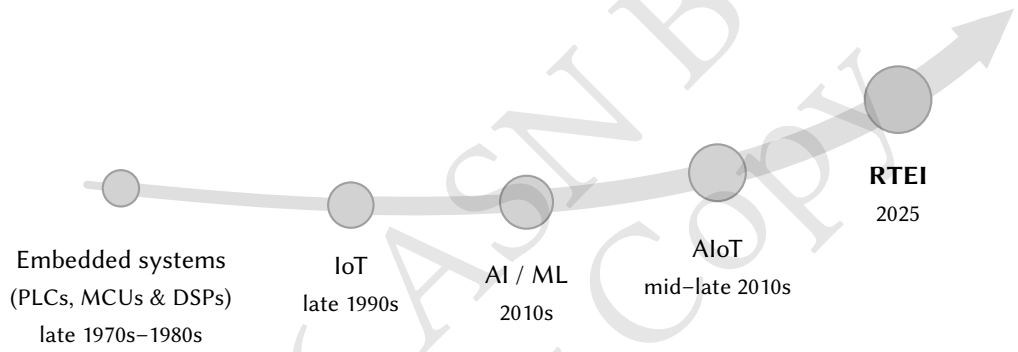

Figure 1: The journey towards Real-Time Edge Intelligence (RTEI): the evolution from early DSP-capable embedded systems, through IoT and AIoT, to a disciplined, standards-aware edge engineering methodology.

Intended Audience and Prerequisites

This book is written for software and system architects, algorithm developers, researchers, and product designers involved in the design and deployment of intelligent products operating under real-world constraints. It addresses the needs of those who are responsible for making architectural, algorithmic, and system-level decisions, and who require practical, engineering-led guidance for implementation on Arm-based microcontrollers and SoCs, rather than abstract or purely data-driven approaches.

The material is especially relevant to professionals working in industrial, medical, automotive, instrumentation, and other compliance-driven domains, where long service lifetimes, IEC/ISO standards compliance, and verifiable behaviour are as important as performance. While Machine Learning and Artificial Intelligence techniques are discussed, the book does not assume a cloud-centric or data-science-driven workflow, and it is not intended as an introduction to large-scale AI model development.

The Journey Ahead

The book is structured as a three-part journey, guiding the reader from foundational concepts, through signals and systems, and finally to real-world commercial deployment workflows and use cases.

- **Part I (Context and Foundations):** establishes the conceptual and engineering framework for Real-Time Edge Intelligence. It clarifies what is meant by intelligence at the edge, and distinguishes between IoT, AIoT, Edge AI, and RTEI. This part also places modern AI techniques in context by examining both their strengths and weaknesses in embedded and edge settings, including their purely data-driven nature, limited interpretability, susceptibility to hallucination, and lack of causal, physics-based reasoning when operating outside trained conditions. It introduces key system-level concepts such as real-time operation, determinism, mission-critical systems, and the constraints imposed by embedded and edge deployments. In addition, it provides historical context by reviewing the evolution of DSP technologies, embedded platforms, and design toolchains over the past five decades, explaining how current practices and capabilities have emerged.
- **Part II (Signals and Systems):** provides system designers with a practical toolkit of state-of-the-art signals and systems techniques, such as transform-based analysis, IIR/FIR digital filter design methods and feature extraction and tracking methods for designing real-time edge applications. Rather than an academic treatment, it presents a detailed overview of essential concepts and best design practices, supported by worked examples drawn from successful deployments. This part assumes that the reader has an undergraduate-level background in signals and systems.
- **Part III (Implementing Commercial RTEI Systems):** provides a framework for translating the discussed methods into commercially deployable products and systems through a series of detailed real-world use cases. It addresses practical implementation on real hardware, including numerical precision effects, data-rate variability, toolchains and best practice design workflows. This part also covers the New Product Development (NPD) process, showing how customer and regulatory requirements are translated into robust, deployable Real-Time Edge Intelligence solutions.

Throughout the book, the emphasis remains focused on systems that work in practice, i.e. systems that can be deployed, certified, maintained, and trusted in the field.

Chapter 1

Introduction to AI, AIoT, DSP and RTEI

Artificial Intelligence or AI has been glorified as the future of automation, often portrayed as the ultimate solution for efficiency, decision-making, and innovation across industries. It is marketed as a transformative technology for everything from healthcare and finance to autonomous systems and industrial processes.

In practice, this narrative does not reflect present reality, as AI in its current form remains too limited to be relied upon for mission-critical applications that require deterministic behaviour, such as those stipulated by ISO/IEC standards. Although AI performs impressively in controlled settings, it often struggles when exposed to the complexity, variability and unpredictability of real-world environments — particularly when those conditions fall outside the assumptions used to train the model.

This degradation in performance occurs because AI lacks common sense reasoning and struggles with real-world subtlety, i.e. *it does not understand the real world in the same way that humans do.* Trained largely on synthetic or otherwise limited datasets, it has no intrinsic grasp of the physical or situational subtleties present in operational environments. When real-world conditions differ from those seen during training, AI systems often misinterpret context, leading to unreliable or misleading outcomes — an unacceptable limitation for any mission-critical application.

1.0.1 Mission-Critical Applications: Definition and Examples

An application is classed as *mission-critical* when results must be delivered predictably, repeatably, and within guaranteed timing specifications as stipulated by the application and stakeholder requirements. More importantly, the term extends well beyond traditional safety-critical domains, as in many operational settings, unreliable or de-

layed behaviour leads directly to financial loss, disrupted processes, or regulatory non-compliance.

Mission-critical requirements appear across a wide range of industrial, commercial, and safety-critical systems. Examples include logistics operations that rely on accurate track-and-trace data, smart-grid infrastructures that must detect voltage dips and swells within milliseconds (e.g. IEC 61000-4-30), and high-integrity processes where incorrect decisions or delayed responses can halt production, increase costs, or compromise product quality.

Mission-critical systems therefore depend on processing chains that behave in a controlled and repeatable manner under all expected operating conditions. Digital Signal Processing (DSP) supports this by providing rules-based signal processing algorithms whose behaviour is deterministic, making it a natural foundation for time-sensitive sensing and control applications.

All of these examples highlight a key point: mission-critical performance is not achieved by adopting a purely DSP-centric or a purely AI-centric design. A design may be predominantly *DSP-centric*, using established signal-processing blocks as the foundation, with trained AI models to improve adaptivity, provided that hardware resources can still guarantee bounded execution times. Conversely, solutions that rely more heavily on trained AI models often require DSP components to meet timing, stability, or compliance requirements.

In practice, what ultimately matters is that the complete system — regardless of how much DSP or AI it contains — can deliver predictable, repeatable results within its required timing constraints and regulatory requirements.

Examples of Mission-Critical Behaviour

- **Automotive systems:** advanced driver assistance, sensor fusion, and powertrain control (in the case of electric vehicles) require consistent, bounded-latency decisions. Errors can compromise ASIL-rated safety functions. A more detailed breakdown of these requirements will be given in the next section.
- **Avionics systems:** flight-control and autopilot functions, as well as the TCAS anti-collision avoidance system, rely on deterministic, rules-based logic to maintain stability, separation, reliability, and safety margins.
- **Predictive maintenance:** vibration and acoustic asset-monitoring applications require consistent interpretation. Early signs of failure must not be missed, while false alarms can lead to unnecessary downtime and costly interventions.

- **Power-quality monitoring:** smart-grid systems rely on reproducible, time-aligned measurements to maintain stability and support billing and operational decisions.
- **Industrial automation and process control:** factories rely on synchronised sensing and actuation to maintain throughput and quality. For example, a jam at a single station — such as the labelling unit on a bottling line — can quickly create a backlog that stops the entire production line. Prompt detection and reporting are therefore essential to minimise any downtime and financial loss.
- **Medical devices:** devices such as insulin pumps, cardiac monitors, ventilators, and assisted–living equipment require fault–tolerant operation. Missed events, incorrect dosing, or delayed response can directly endanger patient safety.
- **Logistics automation:** modern BLE/UWB track-and-trace tags combine traditional barcodes with real-time location (RTLS) capabilities. Any mis-identification or tracking error can propagate across a supply chain, leading to costly delays and operational disruption.

Why Determinism Matters

Mission-critical systems are defined not only by the importance of their outcomes but also by the operational guarantees they must uphold. In practice, this requires:

- Predictability and bounded latency.
- Robustness under real-world variability.
- Traceability and explainability.
- Reproducibility for auditing and standards compliance.

These requirements form the basis for the next section, which examines the specific limitations of trained AI models when applied to mission-critical scenarios.

1.0.2 Operational Implications of AI's Limitations

The lack of common sense about how the physical world works and limited training data are fundamental limitations of AI systems. This can lead to costly failures due to false predictions when operating conditions move outside the scope of the training data — making them less suitable for dynamic environments where reliability is paramount.

Another limitation, particularly for large-scale models such as LLMs, is that AI models require powerful computing resources, making them inefficient for real-time, low-power edge applications. That being said, advancements in Nvidia's latest chipsets, such as the Jetson Orin series, are certainly helping bridge this gap by providing high-performance, power-efficient AI processing directly on edge devices.

While these new chipsets allow AI models to run locally and reduce reliance on cloud computing, AI in general still faces challenges such as excessive power consumption compared to deterministic DSP algorithm-based solutions, reliance on limited datasets, and a lack of explainability. These factors make AI unsuitable for safety-critical functionality subject to strict regulatory compliance. In such cases, only algorithms with strict, verifiable deterministic behaviour are acceptable.

This is clearly emphasized in the automotive domain through the ISO 26262 ASIL (Automotive Safety Integrity Level) framework, as shown below in Table 1.1. As seen, ISO 26262 classifies vehicle functions according to the level of safety integrity they require, with AI generally restricted to the lower-risk, assistive functions.

ASIL	Safety Level	Typical Functions	AI Suitable?
QM	None	Entertainment, Navigation, Comfort Systems	Yes
A	Low	Seatbelt reminder, Cabin Occupancy Detection, Lane-Departure Warning, Driver-fatigue warning	Yes
B	Moderate	Blind-Spot Monitoring, Forward obstacle warning	Yes (assistive only)
C	High	Airbag sensing, Cruise Control (throttle-only), traction/torque support	Limited
D	Highest	Braking, steering, torque control, emergency braking, Adaptive Cruise Control (radar/LiDAR-based)	No

Table 1.1: ISO 26262 ASIL levels and AI's suitability.

Analysing the various ASIL categories, it can be seen that at the QM, ASIL-A and ASIL-B levels, functions are primarily advisory or perception-driven, so any malfunction is generally manageable by the driver — making these the natural domain for AI and ML techniques. ASIL-C introduces functions that influence vehicle dynamics through throttle or engine-torque control, where stronger behavioural guarantees are required.

12.1 Use Case 1: Smart Factory Operator Safety Validation

In modern smart-factory environments, many machines require operators to comply with strict safety conditions before operation is permitted — typical examples include CNC milling machines and lathes. In all cases, operators are required to wear appropriate personal protective equipment (PPE), such as hard hats and safety goggles. As such, this use case considers an automated safety check that validates operator readiness at the point of operation and provides immediate feedback.

Figure 12.1: Smart-factory scenario illustrating real-time operator safety validation.

Figure 12.1 illustrates the scenario. An operator approaches a machine and is required to stand within a designated zone on the floor. The system monitors this zone and determines whether a human operator is present. This includes rejecting any false positives, such as someone walking past. Once a valid operator presence has been established, the system checks compliance with the required PPE rules. The system then provides immediate feedback indicating whether machine operation is permitted — providing an automated safety check between operators and dangerous machinery.

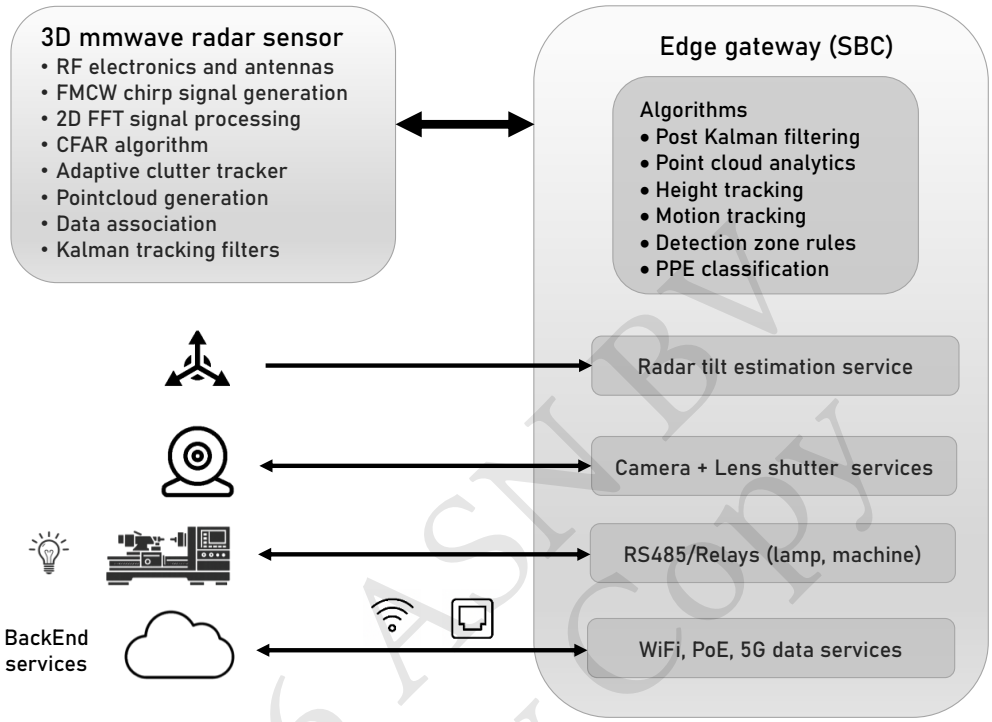

Figure 12.2: Proposed system block diagram for smart-factory operator safety validation using camera-based PPE assessment and 3D mmwave radar-based presence and position estimation.

The use of an SBC platform running a full operating system provides native support for network connectivity, e.g. Ethernet, Power-over-Ethernet (PoE), Wi-Fi, enabling secure communication with enterprise systems and cloud-based services for logging, monitoring, and device management. Optional cellular connectivity via an external 5G dongle may also be used in scenarios where fixed network infrastructure is not available. In addition, the presence of a full OS allows the system to host heterogeneous software components, including high-level application logic in languages such as Python, alongside performance-critical modules implemented in C or C++.

In this architecture, the edge gateway is responsible for hosting all system-level services, including sensor interfaces, data acquisition, control logic, and user interaction. This includes management of the camera subsystem and its associated lens shutter mechanism, which is exposed through a dedicated service layer and controlled by the fusion logic. The gateway therefore acts as the central integration point, coordinating sensing, decision-making, and machine control in real-time.

Get the full handbook

This preview edition contains selected excerpts from the Real-Time Edge Intelligence Solutions Handbook (RTEI).

The full handbook includes complete DSP derivations, implementation workflows, and detailed use-case studies across industrial, biomedical, and smart-grid domains.

Website: https://www.advsolned.com/rteihandbook/
ISBN: 978-94-6526-559-9 (softcover)

Includes complete workflows implemented using ASN Filter Designer and Arm CMSIS-DSP frameworks.