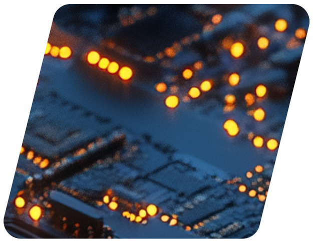
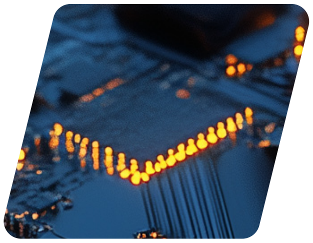
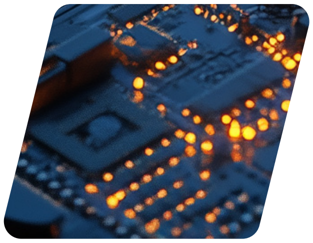
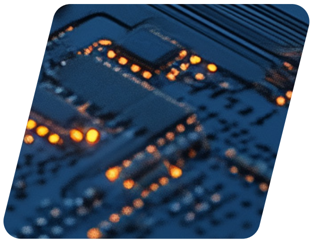
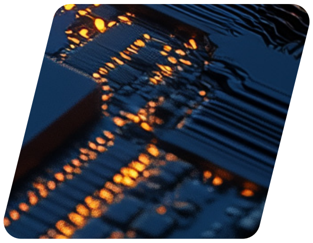



Enabling High-performance Servo Drives With MCU–FPGA Architectures

White Paper

Author:
Karl Wachswender, Senior Principal System Architect, Lattice Semiconductor

DISCLAIMERS

Lattice makes no warranty, representation, or guarantee regarding the accuracy of information contained in this document or the suitability of its products for any particular purpose. All information herein is provided AS IS, with all faults, and all associated risk is the responsibility entirely of the Buyer. The information provided herein is for informational purposes only and may contain technical inaccuracies or omissions, and may be otherwise rendered inaccurate for many reasons, and Lattice assumes no obligation to update or otherwise correct or revise this information. Products sold by Lattice have been subject to limited testing and it is the Buyer's responsibility to independently determine the suitability of any products and to test and verify the same. Lattice products and services are not designed, manufactured, or tested for use in life or safety critical systems, hazardous environments, or any other environments requiring fail-safe performance, including any application in which the failure of the product or service could lead to death, personal injury, severe property damage or environmental harm (collectively, "high-risk uses"). Further, buyer must take prudent steps to protect against product and service failures, including providing appropriate redundancies, fail-safe features, and/or shut-down mechanisms. Lattice expressly disclaims any express or implied warranty of fitness of the products or services for high-risk uses. The information provided in this document is proprietary to Lattice Semiconductor, and Lattice reserves the right to make any changes to the information in this document or to any products at any time without notice.

INCLUSIVE LANGUAGE

This document was created consistent with Lattice Semiconductor's inclusive language policy. In some cases, the language in underlying tools and other items may not yet have been updated. Please refer to Lattice's inclusive language FAQ 6878 for a cross reference of terms. Note in some cases such as register names and state names it has been necessary to continue to utilize older terminology for compatibility.

ABSTRACT

Modern servo drives face increasing demands for higher bandwidth, tighter real-time control, functional safety, and support for advanced power technologies such as silicon carbide (SiC) and gallium nitride (GaN). Traditional microcontroller architectures struggle to meet these requirements as control loop frequencies rise and timing margins shrink. This white paper explains how an MCU-FPGA architecture, a partitioned system design that combines a microcontroller unit (MCU) with a field-programmable gate array (FPGA), addresses these challenges by separating software-defined control-plane functions from deterministic, hardware-based real-time execution. Through practical use-cases, architectural analysis, and emerging power-stage considerations, this paper demonstrates why low power industrial FPGAs are becoming a foundational element of next-generation high-performance servo drives.

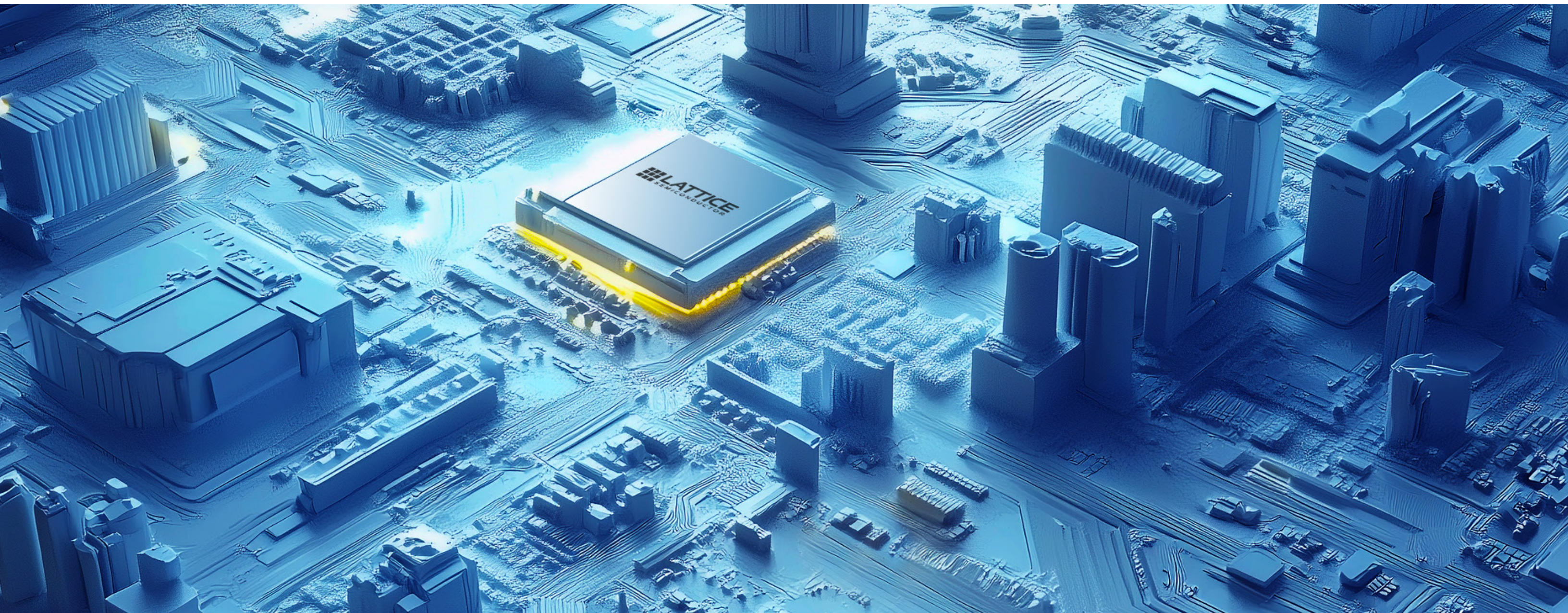


TABLE OF CONTENTS

Introduction

High-performance servo drives are core building blocks in applications such as robotics, CNC machines, and semiconductor manufacturing equipment. As these systems evolve, designers must deliver higher precision, faster dynamic response, and robust functional safety while supporting increasingly complex connectivity and diagnostics. Meeting these goals with MCU-only designs becomes difficult as software load, interrupt latency, and timing jitter increase.

A complementary MCU–FPGA architecture offers a scalable solution. By assigning time-critical, deterministic functions to hardware while preserving software flexibility at the system level, designers can achieve higher performance without sacrificing reliability or extensibility.

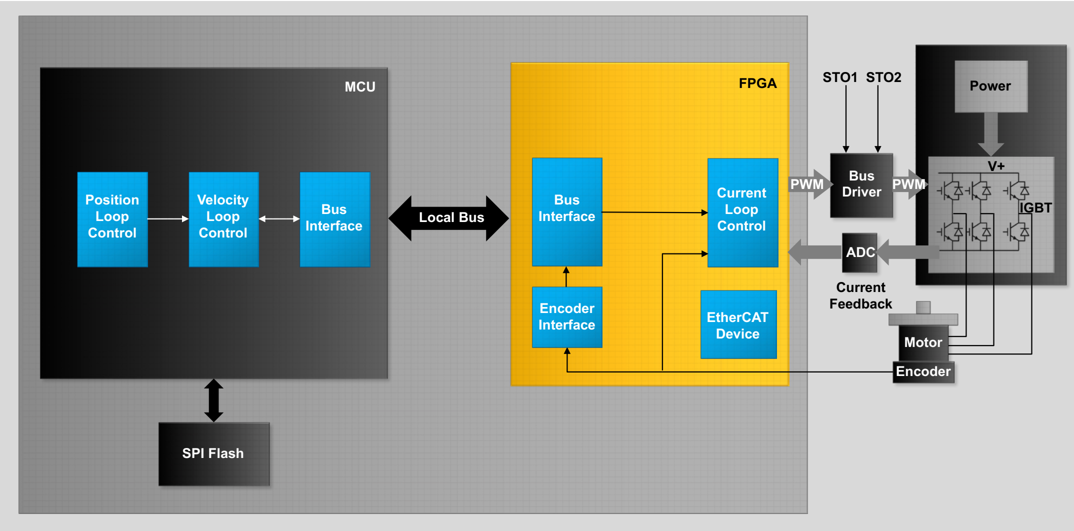
Use Cases

1. High-Bandwidth Motor Control in Advanced Servo Drives

In high-performance servo applications, control loop frequencies continue to increase to improve accuracy and dynamic response. In these systems, the FPGA implements ultra-fast and deterministic motor-control functions, including pulse width modulation (PWM) generation, field-oriented control (FOC) calculations, and high-resolution encoder processing. These operations benefit from parallel execution and nanosecond-level timing accuracy.

The MCU operates alongside the FPGA to execute higher-level motion algorithms, trajectory planning, parameter management, and diagnostics. This separation allows performance to scale without overloading the MCU or compromising real-time behavior, while maintaining software flexibility for tuning and feature expansion. See Figure 1.

Figure 1: Servo Driver Block Diagram



2. Deterministic Industrial Ethernet and Fieldbus Offload

Distributed motion systems rely on real-time industrial Ethernet protocols such as EtherCAT®, PROFINET IRT, and EtherNet/IP with CIP Motion. These protocols impose strict synchronization and latency requirements that are difficult to guarantee using MCU-only solutions.

In this use case, the FPGA offloads time-critical protocol functions such as frame parsing, cycle-accurate data exchange, distributed clock synchronization, and low-latency I/O handling. Hardware implementation ensures deterministic response and minimal jitter, even under heavy network load, while the MCU focuses on higher-level control logic and system integration.

3. Functional Safety and State Management Separation

Industrial servo drives often operate under functional safety requirements such as safe torque off (STO), safe speed monitoring, and controlled stop. In an MCU–FPGA architecture, safety-related monitoring and protection functions can be cleanly partitioned between hardware and software domains.

The FPGA implements fast, deterministic safety mechanisms such as signal monitoring, redundancy checks, and immediate shutdown logic. The MCU manages diagnostics, fault logging, and coordination with higher-level safety controllers. This separation simplifies safety analysis and supports certification by establishing clear fault-containment boundaries.

4. Modular and Scalable Servo Drive Platforms

OEMs increasingly design servo drives as modular platforms that scale across power classes, motor types, and communication standards. FPGAs enable this scalability by allowing hardware functionality to be adapted through logic reconfiguration without redesigning the control software.

A single platform can support multiple encoder interfaces, feedback standards, or Ethernet variants by modifying FPGA logic, while preserving a common MCU firmware base. This approach reduces time to market and enables efficient product differentiation.

Compute Partitioning in Servo Drive Architectures

Modern servo drives benefit from a compute-partitioned architecture that separates control-plane functions from real-time data-plane execution. Assigning responsibilities according to the intrinsic strengths of MCUs and FPGAs enables tighter real-time behavior, improved robustness, and long-term scalability.

MCU Responsibilities: Control Plane

The MCU serves as the primary control-plane processor, handling software-defined functionality that benefits from programmability and rich ecosystems. Typical responsibilities include:

- Motion state management and operational mode handling
- System diagnostics, logging, and fault reporting
- Commissioning, configuration, and firmware management
- Industrial Ethernet stack integration and application-layer processing
- Supervisory control loops and coordination with higher-level controllers
- Functional safety coordination, depending on system requirements

This role provides adaptability and extensibility while remaining isolated from time-critical execution paths.

FPGA Responsibilities: Real-Time Data Plane

The FPGA implements the real-time data plane, where deterministic timing, low latency, and parallel execution are essential. Hardware-based execution ensures consistent behavior independent of software load or interrupt latency. Typical responsibilities include:

- Deterministic processing pipelines for sampling, signal conditioning, and control computations
- High-speed capture and alignment of encoder, resolver, and ADC signals
- Fast protection and interlock logic operating independently of MCU scheduling
- Parallel coprocessor functions such as PWM generation, dead-time insertion, timestamping, and trigger generation

By executing these functions in hardware, the FPGA enables precise timing control and rapid response to fault conditions.

Why Add an FPGA When an MCU Exists

Deterministic Inner-Loop Control

Ultra-fast current-loop control demands precise timing for sampling alignment, FOC computation, PWM updates, and protection. FPGA parallelism and determinism make these functions more reliable at high bandwidths.

Stable Real-Time Behavior Under Software Load

As features such as connectivity, logging, and predictive maintenance expand, MCU interrupt load and jitter increase. FPGA-based pipelines maintain consistent latency regardless of MCU workload.

Always-On Safety and Protection

FPGAs enforce protection boundaries such as overcurrent detection, illegal PWM prevention, and fast shutdown without relying on software deadlines.

Platform Flexibility

FPGAs enable feature scaling and interface variation by modifying logic configuration, enabling reuse across product families without redesigning the MCU architecture.

FPGA Enablement for SiC- and GaN-Based Servo Drives

Servo drives are rapidly transitioning from silicon-based power devices to wide-bandgap technologies such as SiC and GaN to achieve higher switching frequencies, improved efficiency, and greater power density. These devices enable faster dynamic response and reduced passive component size, but they also introduce sensitivity to timing errors and delayed fault response.

At high PWM frequencies, even small variations in timing, dead-time control, or protection latency can result in excessive switching stress, EMI, or device failure. MCU-based architectures struggle to guarantee the sub-microsecond determinism required in this operating regime.

FPGAs address these challenges by implementing PWM generation, dead-time insertion, sampling alignment, and protection logic directly in hardware with nanosecond-level precision. Fast protection mechanisms such as cycle-by-cycle current limiting and immediate fault shutdown significantly reduce stress on power devices and improve system robustness. See Table 1.

Table 1: FPGA vs MCU in SiC and GaN Servo Drivers

Requirement	MCU only	FPGA
PWM jitter	Variable	Deterministic
Dead-time control	Limited	Per-edge precision
Short-circuit protection	µs latency	<200 ns
High PWM (>30–50 kHz)	Hard	Natural
ADC-PWM alignment	Approximate	Exact
Multi-axis sync	Software-timed	Hardware-timed

Conclusion

As servo drives evolve toward higher bandwidth, tighter safety requirements, advanced networking, and wide-bandgap power stages, traditional MCU-only architectures face fundamental limitations. A partitioned MCU–FPGA architecture provides a scalable and robust foundation by combining software flexibility with deterministic, hardware-based real-time control.

READY TO LEARN MORE?

To learn more about Lattice low power FPGA-based solutions for industrial, automotive, communications, computing, and consumer applications, visit **www.latticesemi.com** or contact us at **www.latticesemi.com/contact** or **www.latticesemi.com/buy**.

TECHNICAL SUPPORT ASSISTANCE

Submit a technical support case through **www.latticesemi.com/techsupport**.
For frequently asked questions, please refer to the Lattice Answer Database at **www.latticesemi.com/Support/AnswerDatabase.**