SPI Controller

AXI4-Lite SPI Master/Slave Controller — Runtime STD / 3-Wire / Dual / Quad

IP Core · spi

RUNX-PB-SPI-001 · 2026-08-11 **Product Brief**

Introduction

The RunX SPI Controller (core ID spi_top) puts a full SPI master and a full SPI slave behind two independent AXI4-Lite register banks in one core, so the same bitstream can drive external SPI devices, answer an external master, or both at once on separate pins. All four SPI modes — standard 4-wire, 3-wire half duplex, Dual and Quad — are selected from the register map at run time when the core is built with its default RUNTIME mode, with no rebuild between them. Build-time generics trim either the role (master-only, slave-only or both) or the transfer mode down to exactly what an application needs, from a 3441-LUT fully flexible build to a 729-LUT master-only, STD-only one.

Features

- Runtime-selectable transfer mode — STD (4-wire), 3-wire, Dual and Quad — switched from the register map with no rebuild, on a core built with the default G_MODE=RUNTIME.
- Two independent AXI4-Lite endpoints in one core: a master and a slave, each with its own register bank, wired to nothing inside the top level but each other's package.
- Build-time G_ROLE selects master-only, slave-only or both; build-time G_MODE can fix a single transfer mode at elaboration. Together they cut the core from 3441 LUTs (both roles, runtime mode) to 729 LUTs (master-only, STD-only).
- All four CPOL/CPHA combinations (SPI modes 0–3), plus MSB-first or LSB-first bit order, both selectable at run time.
- Transfer size programmable in single bits, 1 to 123, over a flat 128-bit payload spread across four TXD/RXD registers.
- Programmable turnaround position for the shared-line modes (3-wire, Dual, Quad), plus master-side pre- and post-transfer delay.
- One level-triggered done interrupt per side, driven from the same bit software polls.
- Read-only VERSION and BUILD_DATE registers on both sides, for driver bring-up checks.

IP Facts	
Core Specifics	
Product Name	SPI Controller
IP Identifier	RUNX-PL-SPI-v1
IP Version	v1.0.0
Bus Interface	AMBA AXI4-Lite (two independent 32-bit slave interfaces)
Release Status	Engineering
Verification Status	436 regression configs, ~49,450 protocol-checker assertions, PASS
Validation Status	Simulation only — third-party hardware validation not started
Provided with Core	
Delivery Format	Portable VHDL-93 RTL source (proprietary, NDA)
Vendor Support	AMD/Xilinx (built & timing-closed) · Intel/Altera (synthesized) · Lattice · Microchip (expected)
Constraints	XDC (Vivado reference design)
Simulation	SystemVerilog regression testbench + passive protocol checker
Supported SW driver	Bare-metal C driver, both sides (Apache-2.0)
Document	
Document Number	RUNX-PB-SPI-001
Document Revision	Rev A
Date	2026-08-11

Synthesis & Implementation

Vendor	LUT	FF	DSP	BRAM	Fmax
Xilinx	2 182	584	0	0	180 MHz
Altera	2 460	551	0	—	222 MHz
Lattice	[—]	[—]	[—]	[—]	[—]
Microchip	[—]	[—]	[—]	[—]	[—]

Support

info@run-x.com · www.run-x.com

- Complete bare-metal C driver for both sides (Apache-2.0), with a blocking path and a non-blocking poll/interrupt path.
- Vendor-agnostic VHDL-93 RTL — no vendor primitives, no block RAM, no DSP, on any target.

Contents

01

SECTION

Introduction

1.1 Overview

The SPI Controller (core ID spi_top) lets a processor or bus master drive an SPI link through memory-mapped registers. Two cores are delivered inside one top level: a master that generates chip select and the serial clock, and a slave that is clocked by an external master. Each has its own AXI4-Lite register bank, and neither is wired to the other inside the top level. A transfer is set up by writing the clock rate, mode and framing into CFG, the payload into the TXD registers, and the enable/start bits into CTRL; the result appears in the RXD registers once the done indication changes state.

1.2 Key Features

The features listed on the cover are the complete feature set of this release. Two are worth calling out for how far they change the core's footprint: the transfer mode is runtime-selectable by default (STD, 3-wire, Dual, Quad, chosen from the register map with no rebuild), and the master/slave role is a build-time choice that can remove up to 79% of the LUTs a fully flexible build would cost — see §3.1.

1.3 Target Applications

1. Sensor and converter links — ADCs, DACs, IMUs and temperature sensors whose framing differs from device to device.
2. Serial flash and configuration memory, where Dual and Quad widths matter for throughput.
3. Board-to-board and chip-to-chip control channels between an FPGA and a second FPGA or an MCU, in either direction, since both a master and a slave are provided.
4. Bring-up and test fixtures that have to speak several SPI dialects from one bitstream.

1.4 Supported Device Families

Vendor	Device family	Status
AMD/Xilinx	Zynq UltraScale+ (xck26, KV260)	Built and timing-closed
AMD/Xilinx	Artix-7, Kintex-7, Virtex-7, Zynq-7000	Expected — not yet built
Intel/Altera	Cyclone, Arria, Stratix, Agilex	Synthesized — see §3.1
Lattice	ECP5, CertusPro	Expected — not yet built
Microchip	PolarFire, IGLOO2	Expected — not yet built

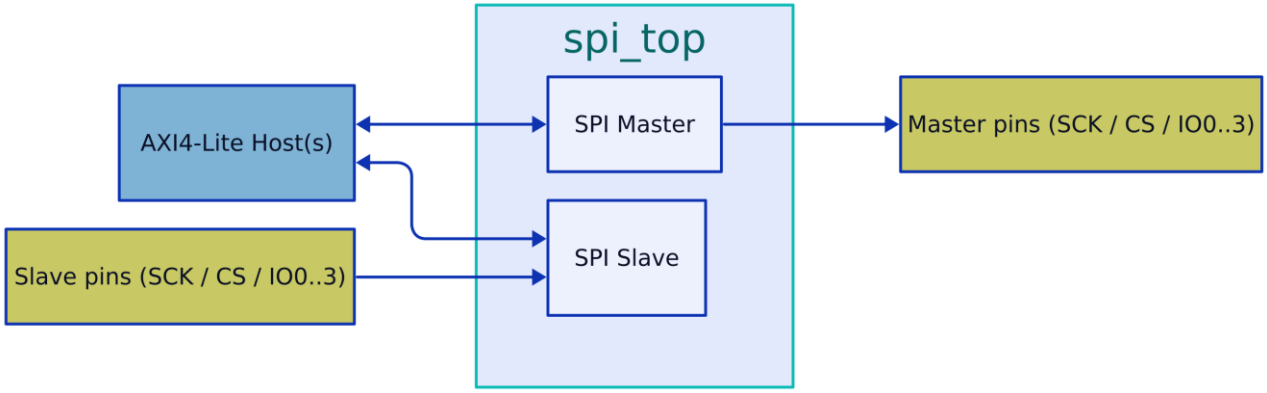
02

SECTION

Functional Overview

2.1 Block Diagram (simplified)

spi_top instantiates an independent master and slave, each behind its own AXI4-Lite register bank, with no connection between them inside the core — on a board they are brought out on separate pins.



spi_top: an independent AXI4-Lite SPI master and SPI slave in one core, sharing no internal connection.

2.2 Port List (names/widths only)

Port	Dir	Width
clk_i / reset_i	in	1 / 1
spi_master_axi_*	in/out	AXI4-Lite, master register bank
m_sck_o / m_cs_n	out	1 / 1
m_io0_mosi_io .. m_io3_io	inout	1 each
m_spi_master_done_interrupt_o	out	1
spi_slave_axi_*	in/out	AXI4-Lite, slave register bank
s_sck_i / s_cs_n	in	1 / 1
s_io0_mosi_io .. s_io3_io	inout	1 each
s_spi_slave_done_interrupt_o	out	1

03

SECTION

Performance Summary

3.1 Resource Utilization (headline)

Figures below are Vivado 2025.2 out-of-context synthesis of spi_top alone on part xck26-sfvc784-2LV-c — the right basis for comparing build-time configurations, since the reference design's own implementation run also carries the processor system wrapper around it. G_ROLE and G_MODE compose; a single-role, single-mode build is the leanest the core gets.

Configuration	CLB LUTs	vs. fully flexible
BOTH roles, RUNTIME mode (fully flexible)	3441	—
MASTER only, STD only (leanest)	729	−79%
SLAVE only, STD only	915	−73%
BOTH roles, QUAD fixed	3254	−5%

Intel/Altera (Agilex 3, A3CY135BM16AE6S), Quartus Prime, RUNTIME mode, slave-only: ~2500 LUT/ALUT, 551 registers, 0 DSP. No Lattice or Microchip figures exist yet.

3.2 Maximum Frequency (Fmax, headline)

AMD/Xilinx (Zynq UltraScale+, xck26): the reference design's routed clk_pl_0 closes at 10.312 ns (about 97.0 MHz), +4.890 ns WNS, +0.038 ns WHS. This is a tested operating point, not a swept maximum — no dedicated Fmax push has been run, and the hold margin is met but tight.

Intel/Altera (Agilex 3): 222–320 MHz across the RUNTIME/STD/3WIRE/DUAL/QUAD × MASTER/SLAVE/BOTH matrix, Quartus Prime, both default and area-optimized strategies. The serial link itself imposes no frequency pressure at either vendor: at 500 kHz with a 100 MHz system clock, both the master's shift engine and the slave's oversampling logic have 200 system-clock cycles per bit.

04

SECTION

Licensing & Ordering

4.1 Licensing Options

RTL source is proprietary and delivered under NDA. The bare-metal C driver (spi_common / spi_master / spi_slave) is licensed separately and more permissively, under Apache-2.0, so it can be redistributed with an end product without carrying the RTL's NDA terms.

4.2 Deliverables

5. Synthesizable VHDL-93 RTL, five modules under spi_top, plus a Verilog wrapper (IP_spi.v).
6. Bare-metal C driver for both sides (spi_common/master/slave .c/.h, Apache-2.0), blocking and non-blocking.
7. Reference pseudocode (spi_master_pseudocode.c / spi_slave_pseudocode.c) documenting the register sequence the driver implements.
8. SystemVerilog regression testbench plus a passive protocol checker.
9. Constraints (spi.xdc) from the reference design.
10. Documentation: product brief, user guide.

4.3 Contact & Support

info@run-x.com · www.run-x.com. Support covers RTL integration questions, driver questions, and errata reports; see the User Guide for known limitations before filing one.

05

SECTION

Revision History

5.1 Document Revision History

The following table lists the revision history of this document. Each entry names the sections that were added, changed, or removed, and the IP Version column records the core release that revision of the document describes.

Date	Doc Rev.	IP Version	Description of Change
2026-08-11	[D]	v1.0.0	• [Section N.n — summarise what was added, changed, or removed.] • [One bullet per edited section, named, so two revisions can be diffed at a glance.]
2026-08-11	[C]	v1.0.0	• [Added Section N.n — <title>.] • [Updated Table N-n to cover <what changed>.] • [Removed <what was dropped> from Section N.n.]
2026-08-11	[B]	v1.0.0	• [Editorial revision — no technical content changed.]
2026-08-11	[A]	v1.0.0	Initial RunX release.

5.2 IP Version History

The following table lists the release history of the IP core itself. Document revisions and core releases do not track in lockstep — one core release may be described by several document revisions, and an editorial document revision may describe no new core release at all.

Release Date	IP Version	Release Status	Change Summary
2026-08-11	v1.0.0	[Production]	• [Functional change — new feature, parameter, or port.] • [Bug fix — reference the errata item it closes.]
2026-08-11	v1.0.0	[Pre-Release]	• [Interface change — record any backward incompatibility here.] • [Performance or resource change, with the numbers it supersedes.]
2026-08-11	[v1.0.0]	[Engineering]	Initial release of the core.