category	Interface	test id	Test	test name	test status	description or objective of test	stimulus geneation procedure	test passes when	test fails when	checking procedure	assertion description
Registers	AXI4-LITE	test_001	axi_reset_test	axi_rst_test	Fail	Verify that reset is working perfect	set the bit RD_RGS_DFT to 1 and RD_RGS to 1 and enable RAL_CHECKER by setting to 1 in config class	The registers have their default values	The registers didnt restored to their default value	Readback the registers after the reset	
		test_002	core_version_test	core_ver_test	Pass	Verify that the major and minor version numbers of the core are correctly reported.	Read the Core Version register (0x00). For this purpose just set core version test bit to 1 in config class	The values match the expected major and minor revisions.	The values do not match the expected core version.	Extract the Major Revision (bits [31:16]) and Minor Revision (bits [15:0]). Compare the values with the expected version Major = 1, Minor = 0.	
		test_003	core_configuration_test	core_cfg_test	Fail	Validate that the configuration of the core (data width, number of channels, and master/slave mode) is accurately reported.	Read specific configuration values of the Core Configuration register (0x04). For this purpose set the bit CORE_CFG_TEST bit 1 in config class	Readback matches programmed configuration.	Readback values differ from programmed settings. Register Core Configuration (0x04) dont have its default value as per described in specs it should be 30201 but got 10201	Compare the readback values with the programmed values.	
		test_004	core_enable_test	core_en_test	Pass	Verify core operations control, including enabling/disabling the core.	Write 1 to bit [0] to enable the core. Write 0 to disable the core. Control Register 0x08	Writing one to bit 0 enable the core and 0 disable the core	The core fails to enable/disable	Verify active/inactive state based on bit [0].	
		test_005	i2s_timing_control_reg_test	sanity_test	Pass	Ensure the SCLK frequency matches the configured divider in the I2S Timing Control register.	Write a divider value to the I2S Timing Control register (0x20). Write 1 to register 0x20	sclk frequency matches the expected value based on the divider. if divider is 1 then it should have frequency half of the MCLK. and period double of the MCLK	sclk frequency does not match the expected value.	Measure the generated sclk frequency using a waveform viewer and ensure it matches the formula: MCLK/SCLK = Divider_value * 2. Compare the measured frequency with the expected value. Use time period of MCLK integer part only to calculate SCK period. If DIvider value is 1 then SCK shloud have period doiuble of the MCLk for FS 192kHz MCLK is Fs*128 and its period is 40ns then SCK should have period of 80ns	
		test_006	read_only_reg_field_test	ral_test	Pass	Ensure read only register fields reject writes and retain default values.	Set Data Pattern in config class to ALL_ONES Set WR_RD_REG bit to 1 to write value to register and read back	Read only register fields maintain default values. Writes are ignored	Read Only register fields accept writes or change from default values.	Verify that read only register fields retain default values and ignore writes. by reading the registers	
		test_007	read_write_reg_field_test	ral_test	Pass	Ensure Read write register fields writes the specic values	Set Data Pattern in config class to ALL_ONES Set WR_RD_REG bit to 1 to write value to register and read back	Read only register fields maintain default values. Writes are ignored for RO fields and for RW fields 1 is written	Read Only register fields accept writes or change from default values. R/W register fields dont get updated with 1's	Verify that read only register fields retain default values and ignore writes. and R/W register fields are updated by reading the registers	
		test_008	default_read_test	ral_test	Fail	After writing or configuring registers apply reset and check if we values are changed to default	Set the bit RD_REGS_DFT in config class after that run ral test	Default values are according to mentioned in specs	Default values are not according to mentioned in specs Configuration Register value mismatched	Verify that registers have their default values by reading the registers	
		test_009	Coverage Driven	rand_reg_test	Pass	Ensure core handle invalid and valid addresses correctly by sending random transactions	Send random read write transaction on interface with random addresses	Read and write on valid addresses with response ok and on invalid address reponse slave error	Read and write on invalid addresses with response ok	Verify core read and write on valid addresses only with response ok otherwise slave error	
		test_010	To test Handshake	axis_tvalid_test	Pass	Verify that the transmitter does not output data when tvalid is de-asserted.	Sent transaction with tvalid 0	No data is transmitted when tvalid is de-asserted.	Data is transmitted when tvalid is low.	Monitor the sdata_0_output output and ensure no data is transmitted when tvalid is low.	

category	Interface	test id	Test	test name	test status	description or objective of test	stimulus geneation procedure	test passes when	test fails when	checking procedure	assertion description
Audio	AXI-Stream	test_011	Test Valid bit tdata[28]	axis_valid_bit_low_test	Pass	Validate that the Validity Bit (TDATA[28]) determines if audio data is processed or ignored.	Send frames with TDATA[28] = 0 By Default Validity Register value is 0	Data validity is determined correctly based on TDATA[28].	Invalid data is processed, or valid data is ignored.	Verify that audio data is processed only when TDATA [28] = 1. Check that data is ignored when TDATA[28] = 0.	
		test_012		axis_valid_bit_hi_test	Pass	Validate that the Validity Bit (TDATA[28]) determines if audio data is processed or ignored.	Send frames with TDATA[28] = 1 for invalid data. By Default Validity Register value is 0	Data validity is determined correctly based on TDATA[28].	Invalid data is processed, or valid data is ignored.	Verify that audio data is processed only when TDATA [28] = 1. Check that data is ignored when TDATA[28] = 0.	
		test_013	For Validity Reg test	validity_reg_test	Pass	Verify control of the validity bit for incoming audio samples.	Write 1 to the register to force all data valid. Write 0 to let the stream decide validity. validity register (0x0c) and TDATA[28] = 1 on AXI-Stream	Sample validity matches the register setting.	Invalid samples are transmitted when forced validity is enabled.	Check if all samples are valid when the register is 1. Verify sample validity corresponds to the stream when the register is 0.	
		test_014	Data Transferred correctl for Left and Right Audio	sanity_test	Pass	Verify the correct handling of multi-channel audio data.	Send audio data for 2 channels with appropriate TID and TDATA[3:0] values. TID = 0 for left and TID = 1 for right Channel	Multi-channel data is transmitted and interleaved as expected. Left audio data is when LRCLK is low and Right audio data when LRCLK is HIGH	Channels are misaligned or interleaving is incorrect.	Ensure Left and Right channels are alternated correctly for all configurations.	
		test_015	Data Transferrend only when core is enable	sanity_test	Pass	Verify core operations control, enabling/disabling the core and justification modes.	Write 1 to bit [0] to enable the core. Write 0 to disable the core. control register 0x08	The core operates and halts correctly.	The core fails to enable/disable	Check if core operations and justification modes behave as expected. Verify active/inactive state based on bit [0]. Check data alignment for left/right justification.	
		test_016	Justification Modes Test	left_just_test	Pass	Verify core operations control, including enabling the core and left Justification justification mode.	Write 1 to bit [0] to enable the core. Configure left justification by writing 01 to bits [2:1]. Control Register 0x08	The core operates and halts correctly, and left justification modes work as expected. MSB are shifted left and LRCLk is 32 bit now data is in upper 24 bits and lower 8 bits are padded with zeros	The core fails to enable/disable or justification is misaligned.	Check if core operations and justification modes behave as expected. Verify active/inactive state based on bit [0]. Check data alignment for left/right justification.	
		test_017		right_just_test	Pass	Verify core operations control, including enabling the core and Right Justification justification mode.	Write 1 to bit [0] to enable the core. Configure left justification by writing 11 to bits [2:1]. Control Register 0x08	The core operates and halts correctly, and right justification modes work as expected. MSB are shifted right and LRCLk is 32 bit now data in lower 24 bits and upper 8 bits are padded with zeros	The core fails to enable/disable or justification is misaligned.	Check if core operations and justification modes behave as expected. Verify active/inactive state based on bit [0]. Check data alignment for left/right justification.	
		test_018		intrpt_ctrl_test	Pass	Validate that enabling and disabling interrupts in the Interrupt Control register works as expected.	Enable interrupts in the Interrupt Control register (0x10). after that Read the Interrupt Status register (0x14). Send the transactions on AXI-Stream interface in parallel	Interrupts are asserted only when enabled. No interrupts are triggered when disabled.	Interrupts occur when disabled.	Observe the irq signal for assertion when interrupts are enabled. Confirm no assertion when interrupts are disabled.	
		test_019		intrpt_stat_test	Pass	Verify that the Interrupt Status register reflects the correct status of triggered events. irq is 0 when write to 0x14 status register	First configure the interrupt control register by writing the R/W fields to enable interrupts Make INTRP_STAT_TEST of config 1 to use slave sequence which will initial write sequence status register if got any innterupt. In paralle send transactions on AXI-Stream Interface	Irq output is zero after writing to status register	Irq is not zero after writing to status register	Observe the irq signal for assertion when interrupts occurred it should be high and when write to status register occurs it should become zero.	

category	Interface	test id	Test	test name	test status	description or objective of test	stimulus geneation procedure	test passes when	test fails when	checking procedure	assertion description
		test_020	Coverage Driven at differet SCLK freq val	sck_freq4_test	Pass	Verify that the core operates fine with different sck divider value at i2s_timing reg 0x20	First configure the i2s timing reg by writing 4 to set the Time period of SCLK = AUD_MCLK*DIV*2 DIV = 4 send the axi-stream transactions	Core operates correctly with diffrent SCK frequency	Core dont operate correctly and Channels are misaligned or interleaving is incorrect.	Ensure Left and Right channels are alternated correctly for different frequencies.	
		test_021		sck_freq2_test	Pass	Verify that the core operates fine with different sck divider value at i2s_timing reg 0x20	First configure the i2s timing reg by writing 2 to set the Time period of SCLK = AUD_MCLK*DIV*2 DIV = 2 send the axi-stream transactions	Core operates correctly with diffrent SCK frequency	Core dont operate correctly and Channels are misaligned or interleaving is incorrect.	Ensure Left and Right channels are alternated correctly for different frequencies.	
		test_022		sck_freq3_test	Pass	Verify that the core operates fine with different sck divider value at i2s_timing reg 0x20	First configure the i2s timing reg by writing 3 to set the Time period of SCLK = AUD_MCLK*DIV*2 DIV = 3 send the axi-stream transactions	Core operates correctly with diffrent SCK frequency	Core dont operate correctly and Channels are misaligned or interleaving is incorrect.	Ensure Left and Right channels are alternated correctly for different frequencies.	
Assertion	I2S Interface	test_023	SCLK and LRCLK Output Assertion	N/A	Pass	Verify that LRCLK toggles perfectly after scpicific cycles of sclk	N/A	Assertion Passes	Assertion Failed	Ensure no assertion failed	Used concurrent assertion which checks if justification is enabled LRCLK should be toggled after 32 cycles of sclk on negedge of sclk otherwise after 24(Audio data width) cycles