Generated Specification Document

INTRODUCTION

-- Overview and Scope --

ac97_top is a compact AC’97 controller core that bridges a system bus/DMA side to a standard external AC’97 codec over the AC-link. It implements full AC-link framing (256-bit frames with a 16-bit tag and 20-bit data/control slots), serial TX/RX shifting, per-slot enable/latch timing, and codec register access (CRA) via slots 1/2. The core targets SoCs that need a lightweight, register-programmable front end for playback and capture, expecting high-rate sample movement to be serviced by an external DMA engine. Two asynchronous clock domains are used: a system/bus clock domain for control, registers, FIFOs, interrupts, and DMA requests, and an AC-link bit clock domain for precise frame/slot timing; crossings are minimized and qualified by stable, latched conditions. Playback (TX) supports slots 3, 4, 6, 7, 8, and 9; capture (RX) latches the slot 0 tag plus slots 1–4 and 6; unused TX slots (5, 10–12) are driven zero. Small per-channel FIFOs (depth 4 words) buffer audio with selectable 16/18/20-bit sample modes; 16-bit samples are packed/unpacked as halves, while 18/20-bit samples are zero-extended on RX and left-aligned for TX. A simple Wishbone slave interface provides single-cycle register access for configuration/status, TX write apertures, RX read apertures, and CRA commands/data. Interrupts and DMA requests are threshold-driven and maskable, including underflow/overflow attempt indications to help software maintain service. Reset handling includes a timed deassertion for the codec reset pin, automatic suspend on AC-link bit-clock loss, and a resume sequence that forces SYNC per AC’97. Scope: standard AC’97 framing and codec register access without sample-rate conversion, mixing/effects, deep buffering, or codec-specific extensions. System assumptions: an external AC’97 codec supplies clocks and analog I/O, software configures modes and thresholds, and DMA promptly services the small FIFOs to avoid over/underruns.

-- Key Features --

- Complete AC'97 link controller: generates SYNC, per-slot timing, detects bit-clock loss, and performs a timed resume sequence.
- Dual clock domains (system clk and AC-link bit clock) with pipelined strobes for per-frame alignment and safe crossings.
- Full serial framing: 256-bit TX/RX frames (16-bit Slot 0 tag + twelve 20-bit slots); reserved TX slots (5, 10, 11, 12) forced to zero.
- Programmable per-channel sample widths (16/18/20-bit) for playback (o3–o9) and capture (i3, i4, i6) via configuration registers.
- Integrated TX/RX FIFOs (4-word depth): TX left-aligns and zero-extends to 20 bits; RX unpacks and repacks to 32-bit words.
- Slot 0 tag builder advertises active slots and gates FIFO transactions with single-shot enables to prevent retriggers.
- Codec Register Access (CRA) engine: formats Slot 1/2 for read/write, asserts transaction-valid, captures read data, and provides done strobes.
- Wishbone slave interface: compact memory map for control/config/CRAC and per-channel apertures; single-cycle acknowledge.
- Masked interrupt system: sticky status aggregation (clear-on-read) plus per-channel causes

(threshold/level, underflow-attempt, overflow-attempt).
- Configurable DMA request generation: threshold-based requests held until acknowledge to service FIFOs without underrun/overrun.
- Timed reset sequencer: clean, delayed deassertion of the AC'97 reset pin; supports forced resets from CSR.
- Suspend/resume control via CSR and automatic suspend on AC-link inactivity; precise valid-window gating and per-slot latch-enable alignment.

-- Standards Compliance (AC’97) --

Implements AC’97 AC-link per spec: 256-bit frames with a 16-bit Slot 0 tag followed by twelve 20-bit slots; SDATA_OUT is shifted MSB-first per slot and reserved/unused slots (5, 10–12) are driven to zero. SYNC is asserted high for the first 16 bit clocks of every frame and for a controlled resume sequence; BITCLK loss is detected and the link enters suspend with the frame counter held until resume, consistent with AC’97 handshaking. Per-slot latch enables align to 20-bit boundaries after the tag, and SDATA_IN is captured on the falling edge and shifted on the rising edge to meet AC’97 timing. Slot 0 tag fields comply: bit[15] indicates a valid frame (OR of advertised slots), bits[14:13] advertise Codec Register Access (read/write), and bits[12:6] advertise active PCM/data slots (3, 4, 6–9); other bits are zero. Codec Register Access follows the two-frame protocol: Slot 1 carries R/W and 7-bit address (RW in bit[19], addr in bits[18:12]); Slot 2 carries 16-bit data (bits[19:4]); writes complete in the issuing frame and reads return data in the subsequent frame. Audio data formatting matches AC’97: 16/18/20-bit samples are left-justified and zero-padded to the 20-bit slot width on transmit; receive captures full 20-bit slots and repackages to system words with correct zero-extension. Reset sequencing provides a timed, glitch-free ac97_rst_ assertion/deassertion suitable for codec initialization. Overall frame timing, bit ordering, slot tagging, control access, and suspend/resume behavior are consistent with AC’97 requirements.

-- Clocking and Reset Domains --

- Clock domains
- System clock (clk): Wishbone/bus interface (ac97_wb_if), register file (ac97_rf), codec register access (ac97_cra), slot-0 tag builder (ac97_prc), interrupt logic (ac97_int), DMA gating (ac97_dma_if), audio FIFOs (ac97_out_fifo, ac97_in_fifo), and reset sequencer (ac97_rst). ac97_soc also includes clk-domain logic for bit-clock monitoring and resume timing.
- AC-link bit clock (wclk): Serial TX shifter (ac97_sout), RX shifter (ac97_sin), and ac97_soc frame/timing (256-count frame counter, sync_beat, valid window, per-slot out_le[n]). Naming note: some child modules expose a single "clk" port that is driven by wclk at top-level.

- Cross-domain synchronization
- Bit-clock presence: wclk sampled into clk via two-flop synchronizer and XOR edge detector within ac97_soc; clk-domain timeout asserts suspended.
- Suspend feedback: suspended is generated in clk and consumed in wclk to hold the frame counter at 0xFF; synchronize suspended into wclk (two-flop) before use.
- SYNC generation: sync = sync_beat (wclk) OR sync_resume (clk-originated resume sequence). Synchronize sync_resume into wclk before OR’ing.
- Frame qualifiers: valid/in_valid originate in wclk; use synchronized two-cycle pipelines in clk (valid_s, in_valid_s) to qualify per-frame bus/FIFO actions.
- Data movement assumptions: TX data (slot-0/1/2 and FIFO outputs) must be registered in clk and stable well before so_ld; RX slot words captured in wclk are sampled/written in clk only within in_valid windows using the synchronized qualifiers.

- Reset strategy and polarity
- Active-low asynchronous resets (rst/rst_n) in clk domain: ac97_rf, ac97_cra, ac97_int, ac97_rst, and

DMA-related logic; ac97_soc uses active-low reset in its clk-domain paths.
- No explicit reset: ac97_sout, ac97_sin, ac97_prc (and fifo_ctrl), ac97_wb_if; rely on protocol framing, valid windows, and bus quiescence after power-up for deterministic alignment.
- FIFO reset: ac97_out_fifo/ac97_in_fifo clear synchronously when channel en deasserts; system software must toggle en low-high to reset.
- Codec reset output: ac97_rst drives ac97_rst_ (active-low) and deasserts high after ~200 clk cycles; ensure codec timing compliance and that wclk is present or suspend logic holds framing until stable.
- Polarity alignment: ac97_int uses rst_n (active-low); keep reset polarity consistent across clk-domain modules.

-- Functional Block Summary --

- Top-level (ac97_top): Compact AC'97 controller bridging a system bus/DMA side to an external AC’97 codec; orchestrates AC-link framing, serial TX/RX, slot/tag control, codec register access, FIFOs, interrupts, DMA, and RESET# timing.
- Clocking: Two domains—system clock for bus/control/DMA and AC-link bit clock for timing/serial; includes safe CDC for suspended status and bit-clock presence.
- Timing & suspend (ac97_soc): Generates 256-tick frame counter, frame-load pulse, SYNC (normal and forced resume), per-slot latch enables, and late-valid window; monitors bit-clock activity to enter/exit suspended state.
- TX serial (ac97_sout): Builds a 256-bit MSB-first frame and drives SDATA_OUT; populates Slot0 tag, CRAC Slots1/2, PCM slots (3,4,7,8,9) per enable; forces zeros on Slots5 and 10–12; aligns with slot enables.
- RX serial (ac97_sin): Samples SDATA_IN with two-edge staging, deserializes 20-bit words, and latches Slot0 (16-bit), Slots1–4 and 6 (20-bit) at their enables; provides captured data to input FIFOs/CRAC.
- Slot tag & gating (ac97_prc + ac97_fifo_ctrl): Forms Slot0 tag (activity, CRAC presence/write, active PCM slots) and issues one-shot per-slot enables once per valid window when channel enabled, ready, and FIFO capacity conditions are met, preventing intra-frame retriggers.
- Codec Register Access (ac97_cra): Formats Slot1/2 for register writes/reads (RW, 7-bit address, 16-bit data); asserts transaction-valid; captures read data on the subsequent frame; produces crac_wr_done/crac_rd_done strobes.
- Playback FIFO (ac97_out_fifo): 4-word buffer delivering 20-bit-aligned samples; modes—16-bit (two per 32-bit, left-aligned to 20), 18-bit (one per 32-bit, left-shift to 20), 20-bit (pass-through); coarse status; clears on en=0.
- Capture FIFO (ac97_in_fifo): 4-word buffer producing 32-bit words; modes—16-bit (packs two samples), 18/20-bit (zero-extends single sample); coarse status; clears on en=0.
- DMA interface (ac97_dma_if): Per-stream request generation with enable gating and configurable threshold modes based on FIFO status; qualifies requests and holds until dma_ack to avoid chattering.
- Interrupts (ac97_int + ac97_rf): Per-channel causes—level/threshold, underflow on empty read, overflow on full write; register file latches into INTS (read-to-clear) and masks via INTM; outputs a single aggregated interrupt.
- Wishbone slave (ac97_wb_if): Single-cycle-ack, word-aligned bus; decodes register region, playback write apertures (o3/o4/o6/o7/o8/o9), capture read apertures (i3/i4/i6); generates one-cycle strobes for accesses.
- Register file (ac97_rf): Control/config/status—CSR (suspended readback; pulses for forced reset/resume), OCC0/OCC1 (playback channel modes), ICC (capture modes), CRAC command/data ports, INTM, INTS; aggregates events for interrupt.
- Reset sequencer (ac97_rst): Drives AC’97 RESET# with timed hold and release (~200 system clocks); supports forced resets via CSR.
- Channel/slot map: Playback channels o3/o4/o6/o7/o8/o9; capture channels i3/i4/i6. Slot0 tag; Slots1/2 CRAC; Slots3/4/7/8/9 playback; Slot6 capture; Slot5, Slots10–12 are unused on TX (zeros).
- End-to-end operation: At frame start, timing asserts load; tag advertises active slots; CRAC and TX

FIFOs fill outgoing slots; RX latches incoming slots to FIFOs; per-slot gating aligns transactions; DMA and interrupts reflect FIFO state and events; bus programs configuration, initiates CRAC, and moves samples.

-- Assumptions and Limitations --

- Two asynchronous clock domains (clk=AC-link bit clock, wclk=system monitor); crossings are largely unsynchronized (e.g., suspended signal), so integration must add CDC synchronizers/constraints. Glitches/runt pulses can occur around suspend/resume.
- AC'97 framing/timing: Continuous 256-bit frames (16-bit tag + 12×20-bit slots). so_ld must assert exactly once per frame aligned to SYNC/tag; SDATA_IN sampled on falling edge, shifted on rising. Many one-shots require valid to drop between frames.
- Slot coverage limitation: Only slots 0–4 and 6–9 are supported; slots 5, 10, 11, 12 are driven as zero on TX and ignored on RX. Codecs depending on other slots are unsupported.
- Codec register access (CRAC): Single outstanding transaction; do not assert crac_we while crac_valid is active. Writes complete in the initiating frame; reads complete in the next data frame; no queueing/retry.
- Reset behavior: ac97_rst_ deasserts after a fixed delay (~200 clk cycles) from async reset/rst_force; not parameterized. Several modules lack explicit reset; internal registers may power up unknown until first valid/frame activity.
- Resume/SYNC: sync_resume duration depends on external ps_ce period; integration must provide a stable periodic ps_ce or resume length is undefined.
- Wishbone interface: Single-beat transfers via edge-detected re/we; single-cycle acknowledge; no wait-states, stalls, or error responses. Address decode limited to wb_addr_i[31:29]==0; out-of-map accesses are not acknowledged—system-level decode must prevent bus hangs. wb_if lacks reset and may need system reset discipline.
- FIFO limitations: Shallow 4×32-bit memories; no internal protection against overflow/underflow—external logic must gate accesses. Mode-dependent packing (16-bit: two samples/word; 18/20-bit: one/word). TX empty is meaningful only in 16-bit mode; RX empty meaningful only in 16-bit; status is coarse. Known RX FIFO quirks: 20-bit zero-extend width typo; mode==3 missing default may cause latch-like behavior.
- Interrupts: ac97_int outputs are cycle-level pulses; ac97_rf INTS is sticky and cleared on read globally (all bits). Masking via INTM is coarse; threshold int_set[0] always ORs full_empty.
- DMA requests: Require cfg[6] and cfg[0] enabled; two-cycle qualification filters glitches; requests held until dma_ack; not reasserted until condition re-qualifies; no arbitration for multiple channels.
- Data width/mode: Per-channel cfg[3:2] must match software packing. TX left-aligns and zero-pads to 20 bits; RX right-justifies and zero-extends to 32 bits. Mismatched mode/packing corrupts audio.
- Valid-window dependence: Enables/pulses (FIFO control, CRAC edges, tag latching) rely on valid returning low between frames; holding valid high can suppress retriggers; initial Xs may persist until framing stabilizes.
- AC’97 electrical/timing: Codec must meet AC-link timing (SDATA_IN stable on falling edge, continuous wclk). Suspend declared if clk edges stop for ~33 wclk cycles; behavior during suspend/resume must follow AC’97.
- Feature coverage: Limited subset of AC’97 control/status and PCM channels implemented (TX: 3,4,6–9; RX: 3,4,6). No variable slot maps beyond those wired; no error reporting on bus/link beyond interrupts.

IO PORTS

-- Clocks and Resets --

Two independent clocks: (1) AC-link bit clock drives serial TX/RX (SDATA_OUT/IN), frame counter/sync, and codec register access timing; used by ac97_sout, ac97_sin, ac97_soc, ac97_cra. (2) System/bus clock drives Wishbone, register file, DMA, interrupts, reset sequencer, and sample FIFOs; used by ac97_wb_if, ac97_rf, ac97_out_fifo, ac97_in_fifo, ac97_dma_if, ac97_int, ac97_rst. ac97_soc spans both domains (SYNC generation, suspend/resume control).
Clock-domain crossings: Frame/slot strobes (valid, in_valid, out_le[n]) originate in bit-clock domain and are re-timed with two-stage pipelines in the system domain to form one-shot enables per frame. suspended is detected in the system domain and must be synchronized before holding the bit-domain frame counter. CRAC control (crac_we/data) comes from system writes and must be synchronized/handshaked into ac97_cra's bit-clock timing. SYNC is the OR of sync_beat (bit-domain) and sync_resume (system-domain); implement without combinational paths across domains.
Reset architecture: Active-low resets are used consistently. Asynchronous active-low resets apply to ac97_rf, ac97_cra, ac97_int, ac97_dma_if, ac97_soc (bit-clock logic), and ac97_rst. No explicit reset on ac97_sout, ac97_sin, ac97_wb_if, and parts of fifo_ctrl/prc; rely on protocol alignment and initial idle frames; a dummy Wishbone cycle after reset is recommended to settle ack logic. FIFOs implement a synchronous soft reset: asserting en=0 clears pointers/data in the system domain.
Codec reset (ac97_rst_): Generated in the system domain; held low after global reset or ac97_rst_force, then released high after ~200 system clock cycles (prescaler ≈50 clocks, 4 pulses).
Resume: resume_req from CSR triggers a sync_resume sequence to raise SYNC and exit suspend.
Constraints: Preserve valid/in_valid pipelines to prevent re-triggering within a frame; add proper synchronizers for all CDC paths; ensure reset sequencing deasserts system reset before bus activity.

-- AC-Link Interface Signals (SYNC, SDATA_OUT, SDATA_IN) --

AC-Link Interface Signals

SYNC (Frame Sync)
- Direction/source: Driven by ac97_soc in clk (AC-Link BIT_CLK) domain; monitored/managed from wclk domain.
- Polarity/timing: Asserted high for the first 16 BIT_CLK cycles of each 256-cycle frame.
- Generation: An 8-bit counter (cnt) in clk asserts ld at cnt==0 (frame start) and defines sync_beat when cnt is 0x00..0x0F. Final SYNC = sync_beat OR sync_resume.
- Suspend/resume: BIT_CLK loss is detected in wclk; when suspended, cnt is held at 0xFF (no sync_beat). A CSR-driven resume_req creates sync_resume (forced SYNC high) for a short burst (~5 ps_ce pulses from a wclk prescaler), then normal framing resumes.
- CDC note: suspended (wclk→clk) and resume control cross domains; treat as CDC paths.

SDATA_OUT (Codec Receive: host transmit)
- Direction/source: Driven by ac97_sout in clk domain.
- Load/shift: At ld (frame start), a chained register bank loads Slot 0 (16-bit tag) and Slots 1..12 (20-bit each; unused forced 0). Thereafter, bits shift MSB-first on each BIT_CLK rising edge.
- Bit/slot order: Outputs slt0[15] first, then remaining tag bits, followed by slots 1..12 MSB-first. Total 256 bits/frame. After the last bit, zeros continue until next ld.
- Slot content: Slot 0 tag from ac97_prc; Slots 1–2 (control/CRA) from ac97_cra; Slots 3,4,6,7,8,9 from playback FIFOs; Slots 5,10,11,12 are zero.
- Reset: No explicit reset; correctness relies on ld alignment from ac97_soc.

SDATA_IN (Codec Transmit: host receive)
- Direction/sampling: Sampled from codec into sdata_in_r on BIT_CLK falling edge for margin; shifted

into 20-bit sr on BIT_CLK rising edge (sr <= {sr[18:0], sdata_in_r}).
- Capture strobes: ac97_soc asserts out_le[n] pulses on BIT_CLK rising edge at fixed counts (e.g., 0x11, 0x25, 0x39, 0x4D, 0x61, 0x89 hex) to latch complete slot words.
- Latching: Slot 0 tag captures sr[15:0]; Slots 1–4 and 6 capture full 20-bit sr. Slot 5 unused (out_le[5] maps to Slot 6).
- Alignment: out_le pulses are timed so sr holds a full slot at latch; downstream logic uses the latched words.

Common/frame timing
- Clocks: clk = AC-Link BIT_CLK; wclk = system/control domain.
- Frame: 256 BIT_CLK cycles (16-bit tag + 12×20-bit slots). ac97_soc provides ld (start-of-frame), SYNC window (first 16 cycles), and valid windowing (e.g., cnt > 0x39) for downstream use.

-- System Bus Interface (Wishbone) --

32-bit Wishbone Classic slave, single-beat transfers only, synchronous to system clk. No stall or retry; fixed one-cycle acknowledge per accepted transfer.

Interface signals and behavior
- Inputs: wb_cyc_i, wb_stb_i, wb_we_i, wb_addr_i[31:0], wb_data_i[31:0]. Byte lane selects (wb_sel_i) are not used; transfers are full 32-bit words. All signals sampled on clk rising edge.
- Outputs: wb_ack_o (asserts for exactly one clk per accepted transfer), wb_err_o (always 0), wb_data_o[31:0] (registered; valid when wb_ack_o=1).
- Handshake: A transfer request is detected on the rising edge of (wb_cyc_i & wb_stb_i). Read/write strobes are derived as one-cycle pulses from this edge; wb_ack_o pulses once per request. Holding wb_cyc_i/wb_stb_i asserted without deasserting will not generate additional acks; masters must deassert and reassert for subsequent beats.

Address decode and mapping
- Region select: The slave only responds (ack/data) when wb_addr_i[31:29] == 3'b000. Requests outside this region do not generate wb_ack_o; system-level address decode must prevent masters from hanging on unmapped regions.
- Word-aligned addressing: Lower 2 address bits are ignored. Word offset is wb_addr_i[6:2].
- Memory map within the region:
- 0x00..0x1C (word offsets 0..7): Register file (ac97_rf)
• Read: rf_re pulse; wb_data_o <= rf_din.
• Write: rf_we pulse; dout <= wb_data_i.
• Defined registers at offsets 0..6: CSR (RO; write generates control pulses), OCC0, OCC1, ICC, CRAC (codec access), INTM (mask), INTS (status; read-to-clear). Offset 7 reserved; read returns rf_din default; write has no defined side effects.
- 0x20..0x34 (word offsets 8..13): Playback channel write-only portals
• On write: assert oX_we (one clk) for channels o3,o4,o6,o7,o8,o9 at offsets 8,9,10,11,12,13; dout <= wb_data_i. FIFO fullness is not checked here; software/DMA must avoid overflow.
- 0x38..0x40 (word offsets 14..16): Capture channel read-only apertures
• On read: assert iX_re (one clk) for channels i3,i4,i6 at offsets 14,15,16; wb_data_o <= iX_din. Software must avoid reading when FIFOs are empty.
- Reads to other offsets within the region return rf_din mux default; writes within the region to undefined offsets are ignored but still acked (no wb_err_o).

Timing
- Single-cycle acknowledge per transfer; no stall signaling. Write data (wb_data_i) is captured into a local dout register on the same cycle the write strobe is generated. Read data is registered and presented on wb_data_o coincident with wb_ack_o.

Reset and clocking
- All bus-side logic is synchronous to system clk. The Wishbone slave itself has no explicit reset; external modules (e.g., ac97_rf) may have their own resets (ac97_rf uses async active-low). Ensure the bus master and this interface share the same clock domain.

Protocol limitations
- No burst/CTI/BTE support; each request is a single beat.
- No error or retry: wb_err_o=0 permanently; wb_rty_o not used.
- Byte enables not supported; transfers must be 32-bit word-aligned.

-- DMA Interface --

Purpose: Per-channel DMA request generation for AC97 audio FIFOs to avoid TX underrun and RX overrun by asserting requests when programmable occupancy thresholds are crossed.

Channels: TX/playback o3, o4, o6, o7, o8, o9 (ac97_out_fifo); RX/capture i3, i4, i6 (ac97_in_fifo).

Configuration (per stream, from register file):
- Gate bits: cfg[6] (global/stream gate) and cfg[0] (enable) must both be 1 for requests to assert.
- Threshold select cfg[5:4]:
- 00: request when full_empty is asserted OR status < 3.
- 01: request when full_empty is asserted OR status[1] == 0.
- 10: request when full_empty is asserted OR status == 0.
- 11: request only when full_empty is asserted.
- Sample width/mode cfg[3:2] affects FIFO packing semantics; it does not directly change DMA logic.
- Register mapping: TX o3..o9 use oc0_cfg..oc5_cfg; RX i3/i4/i6 use ic0_cfg..ic2_cfg.

FIFO status inputs:
- status[1:0]: coarse, modulo-4 word-level occupancy.
- full_empty boundary flag: TX uses FIFO empty; RX uses FIFO full.

Request generation and handshake:
- Two-cycle qualification: the selected condition must be true for two consecutive clk cycles before asserting a request.
- dma_req is level-true when qualified and remains asserted until dma_ack is observed.
- dma_ack clears the request; if the condition still holds, it must re-qualify for two cycles to raise a new request.
- Synchronous to system clk; asynchronous reset forces dma_req low immediately.

Behavioral notes and caveats:
- Prevents chattering around thresholds via two-cycle filter; supports back-pressure (holds request until ack).
- One outstanding request per channel until acknowledged.
- ac97_out_fifo empty is reliable primarily in 16-bit mode; for 18/20-bit TX streams rely on status[1:0] thresholds more than empty.
- ac97_in_fifo full detection is reliable; its empty flag is specialized to 16-bit packing and otherwise hardwired.
- status is coarse and not sample-accurate in mixed-width modes; choose threshold modes accordingly.

System integration:

- DMA requests can be used alongside the interrupt generator (ac97_int), which uses similar cfg gating; INTM masks in the register file control interrupt causes.
- Separate from Wishbone/PIO paths (ac97_wb_if); PIO accesses must still respect FIFO full/empty.
- Channel/slot association: o3/o4 → playback slots 3/4; o6/o7/o8/o9 → slots 6/7/8/9; i3/i4/i6 → capture slots 3/4/6.
- Link suspend/resume does not automatically gate DMA; software should coordinate DMA enable with AC-link state.

-- Interrupt Output --

- Signal: int_o is a single, level-true interrupt output generated and registered in the clk domain. It asserts when any enabled interrupt source is set and remains asserted until cleared.
- Formation: int_o = OR-reduction of (INTM & INTS). INTS is a 29-bit sticky status register; INTM is a 29-bit mask register that gates only the output (events are still recorded in INTS even if masked).
- Clear behavior: Reading INTS performs a read-to-clear of all its bits, deasserting int_o if no new events are pending. int_o will reassert if new sources set after the read.
- Reset: Asynchronous reset clears INTM and INTS to 0, deasserting int_o.
- Sources recorded in INTS (bit layout): [0]=CRAC read done; [1]=CRAC write done; [2:4]=oc0; [5:7]=oc1; [8:10]=oc2; [11:13]=oc3; [14:16]=oc4; [17:19]=oc5; [20:22]=ic0; [23:25]=ic1; [26:28]=ic2.
- Per-channel causes (each channel contributes 3 bits via ac97_int):
- int_set[0]: level/threshold interrupt (enabled by cfg[0]; threshold via cfg[5:4], qualified by full/empty).
- int_set[1]: underflow attempt (read while empty), one-cycle pulse latched sticky.
- int_set[2]: overflow attempt (write while full), one-cycle pulse latched sticky.
- CRAC events: crac_wr_done and crac_rd_done are single-cycle pulses that set INTS[1] and INTS[0], respectively.
- Timing/clocking: All interrupt aggregation, masking, and output registration occur in the clk domain. int_o reflects the registered OR-reduction on the next clk edge after sources/mask changes.
- Software usage: Enable desired sources via INTM; service interrupts by reading INTS to identify and clear causes. DMA requests are independent of int_o and do not clear INTS.

-- Channel Data Apertures --

Per-channel, 32-bit Wishbone apertures provide host access to AC’97 playback (TX) and capture (RX) FIFOs. Addresses are word-aligned and accepted when wb_addr_i[31:29]==0; the effective aperture index is wb_addr_i[6:2]. TX (write-only) slots: o3@0x20, o4@0x24, o6@0x28, o7@0x2C, o8@0x30, o9@0x34. RX (read-only) slots: i3@0x38, i4@0x3C, i6@0x40.

Access semantics: Each accepted transfer generates a single-cycle wb_ack_o; wb_err_o=0. TX writes: 32-bit wb_data_i is captured into the selected out FIFO and oX_we pulses for one clk. RX reads: wb_data_o returns a 32-bit word and iX_re pulses for one clk to advance the in FIFO.

Data formatting modes per channel are selected by cfg[3:2]: TX oc0_cfg..oc5_cfg map to slots o3..o9; RX ic0_cfg..ic2_cfg map to slots i3,i4,i6. Mode values: 00/0=16-bit (two 16-bit samples packed per 32-bit word); 01/1=18-bit (one sample per 32-bit word); 10/2=20-bit (one sample per 32-bit word). TX samples are left-aligned/zero-padded to 20 bits on the AC-link; RX samples are zero-extended to 32 bits.

Flow and timing: TX writes enqueue data; RX reads dequeue data. Actual AC-link transfers occur once per frame when gated by ac97_prc/fifo_ctrl under valid-slot timing and channel enable. Hardware does not block over/underflow at the bus apertures; software/DMAs must observe FIFO full/empty/thresholds to avoid we-while-full and re-while-empty.

Notes: AC-link TX uses slots 3,4,6,7,8,9; RX captures slots 3,4,6. All aperture transactions and FIFOs operate in the clk domain; serialization/deserialization to the AC-link occurs in wclk with frame-boundary CDC.

-- Signal Timing and Polarity --

- Clock domains and edges
- Two asynchronous domains: AC-link bit clock ("clk" in ac97_sin/sout/soc) and system/control clock ("wclk"). All bit-clocked signals are aligned to clk edges; all control/Wishbone/DMA/INT/CRAC/FIFO control signals are aligned to wclk.
- Unless stated, signals are synchronous to their domain's rising edge.

- Reset polarity/behavior
- rst inputs are active-low (assert low). ac97_rst_ output is active-low and deasserts high after ~200 wclk cycles (prescaled counter).
- FIFO en acts as a synchronous clear: en=0 forces pointers/outputs to 0; en=1 enables normal operation.
- One-shot control pulses (e.g., ac97_rst_force, resume_req) are active-high, one wclk cycle.

- AC-link frame, SYNC, and slot timing (bit clock domain)
- Frame length: 256 bit clocks (16-bit Slot0 tag + 12×20-bit slots).
- SYNC is active-high for the first 16 bit clocks of each frame (sync_beat). During resume, SYNC is additionally forced high (sync_resume); overall SYNC = sync_beat OR sync_resume.
- so_ld (frame load) is an active-high, one-clk pulse at frame start (cnt rolls to 0x00).
- valid is active-high late in the frame (cnt > 0x39); used to qualify one-shot per-frame operations; pipelined copies exist for downstream gating.
- out_le[n] per-slot latch-enables are active-high, one-clk pulses aligned to 20-bit slot boundaries.

- Serial data timing and polarity
- SDATA_OUT (ac97_sout):
- so_ld is a one-clk, active-high load strobe; a new frame is captured on its posedge.
- Shifting is MSB-first within each slot; the first bit of a new frame appears immediately after the load edge (output tap slt0_r[15]).
- Unused slots (5, 10, 11, 12) are driven with zeros each frame.
- SDATA_IN (ac97_sin):
- Input is staged on the falling edge of clk and shifted on the rising edge.
- Slot captures occur on the rising edge when out_le[n] is high.
- Slot 0 captures 16 bits; slots 1, 2, 3, 4, and 6 capture full 20-bit words; slot 5 is intentionally skipped.

- Suspend/resume signaling (cross-domain)
- Bit-clock presence is monitored in wclk: if no clk edges for ~33 wclk cycles, suspended asserts (level-high).
- When suspended and resume_req (one-cycle, active-high) are true, sync_resume drives SYNC high for a short programmed sequence (bit-clock domain) then deasserts.
- While suspended, the bit-clock frame counter is held at 0xFF, suppressing normal SYNC/so_ld behavior.

- Codec Register Access (CRAC) timing/polarity
- crac_we is an active-high, one-wclk-cycle strobe to launch a transaction; crac_out[31]=1 selects read, 0 selects write.
- crac_valid is level-high while driving a command frame (slots 1/2 meaningful).
- Write: crac_wr is level-high during the valid window; crac_wr_done pulses high for one wclk at valid

falling edge.
- Read: command drives during a valid window; readback captured the following frame; crac_rd_done pulses high for one wclk at the next valid rising edge following the data frame; in_slt2[19:4] is latched to crac_din then.
- Slot field mapping: out_slt1[19]=crac_out[31] (1=read), out_slt1[18:12]=address, out_slt2[19:4]=write data; remaining lower bits zero.

- Slot 0 tag word polarity
- Active-high indicators: [14]=CRAC present, [13]=CRAC write present, [12..6]=channels 3,4,6,7,8,9 present; [15] is the OR of [14..6]. Unused slots forced low. Tag and CRAC state latch when valid=1 and hold for the frame.

- FIFO control timing
- ac97_fifo_ctrl emits en_out as a single, active-high, one-wclk pulse per valid window when channel enable, ready, and snapshotted FIFO status qualify. Status/full/empty are snapshotted while valid=0 and held through the slot to prevent retriggers.

- DMA request timing/polarity
- dma_req is level-true; asserts only when enabled by cfg[6]&cfg;[0] and the threshold condition holds for two consecutive wclk cycles; held high until dma_ack clears it. Reset forces dma_req low. If dma_ack is high, dma_req does not assert.

- Interrupt timing/polarity
- ac97_int produces int_set[2:0] registered each wclk:
- int_set[0]: level indicator (active-high) gated by cfg[0] and selected threshold; includes full/empty boundary.
- int_set[1]: active-high, one-cycle pulse on read-while-empty.
- int_set[2]: active-high, one-cycle pulse on write-while-full.
- ac97_rf aggregates into sticky INTS (masked by INTM). int_o is level-true when any masked INTS bit is set; cleared when INTS is read.

- Wishbone handshake timing
- Single-beat transfers: re/we are active-high, one-wclk-cycle pulses generated by edge-detect of wb_cyc_i & wb_stb_i.
- wb_ack_o pulses high for exactly one wclk per accepted transfer.
- i3_re/i4_re/i6_re pulse high for one wclk on reads of their addresses; o3_we..o9_we pulse high for one wclk on writes to their addresses.

- Miscellaneous pulse semantics
- One-shot strobes (so_ld, out_le[n], crac_wr_done, crac_rd_done, iX_re, oX_we, en_out) are active-high, one cycle in their respective clock domains.
- ac97_sout has no explicit reset; serial contents are undefined until the first frame load; zeros propagate due to zero-fed tail.

ARCHITECTURE

-- Top-Level Block Diagram --

Depict two clock domains with explicit CDC boundaries: (1) System clock domain (sysclk) hosting wb_if, rf, cra, prc/fifo_ctrl, per-channel FIFOs, int, dma_if, rst; (2) AC-Link bit clock domain (wclk) hosting soc (bit-clock side), sout, sin. Show external interfaces: Wishbone bus (adr/dat/we/re/ack) into ac97_wb_if; AC-Link pins BITCLK(wclk), SYNC(sync_o), SDATA_OUT, SDATA_IN; DMA per-channel dma_req out and dma_ack in; single interrupt int_o; AC97 reset pin ac97_rst_. Place ac97_soc centrally with two-part representation: bit-clock side generates ld (frame load), out_le[n] (per-slot latch enables), sync_o (SYNC), and valid window; system/control side monitors BITCLK presence, asserts suspended, handles resume_req. Arrows: ac97_soc(valid) -> ac97_prc and ac97_cra (sysclk domain); ac97_soc(ld) -> ac97_sout (wclk domain); ac97_soc(out_le[n]) -> ac97_sin (wclk domain); ac97_soc(sync_o) -> external SYNC. In TX path: ac97_prc builds Slot0 tag and emits one-shot FIFO ops; per-channel ac97_out_fifo (o3,o4,o6,o7,o8,o9) receive Wishbone writes from ac97_wb_if, expose status to ac97_int/dma_if, and provide 20-bit dout to ac97_sout input slots; ac97_cra formats Slot1/2 (CRAC address/data) and drives parallel out_slt1/out_slt2 toward ac97_sout (CDC captured at ld). ac97_sout (wclk) parallel-to-serial shifts SDATA_OUT MSB-first, loading Slot0/1/2 and data slots at each ld; slots 5/10/11/12 forced zero. In RX path: ac97_sin (wclk) deserializes SDATA_IN, latching slots on out_le[n]; exposes slt0 to ac97_prc/rf status, slt2 to ac97_cra for CRAC readback (CDC into sysclk), and slt3/4/6 to per-channel ac97_in_fifo (i3/i4/i6) via CDC boundary; in_fifos pack to 32-bit words for Wishbone reads via ac97_wb_if, and export status to ac97_int/dma_if. Bus/CSR/interrupts: ac97_wb_if decodes RF (0..7), TX write ports (o3..o9), RX read ports (i3/i4/i6), returns rf_dout or FIFO data; ac97_rf holds CSR (suspended, ac97_rst_force, resume_req), OCC*/ICC per-channel mode/thresholds, CRAC command/data (crac_out/we, crac_din), INTM/INTS aggregation; int_o driven by masked INTS; ac97_int per channel consumes FIFO status and one-shot ops to raise causes (threshold/level, underflow-attempt, overflow-attempt) to RF. DMA: ac97_dma_if generates per-channel dma_req based on mode/thresholds and FIFO status; external dma_ack clears requests. Reset: ac97_rst times deassertion of ac97_rst_ and accepts rst_force from RF. CDC annotations: (a) valid/in_valid pipelined in sysclk for clean one-shots; (b) ld-driven capture of parallel slot inputs into ac97_sout; (c) out_le-driven slot latching in ac97_sin with subsequent transfer of slt2 and capture samples into sysclk domain; (d) suspended crosses to wclk side to hold frame counter; resume_req flows from RF to ac97_soc. Label data/control flows: TX path Wishbone -> out_fifos -> prc one-shots -> sout slots -> SDATA_OUT; RX path SDATA_IN -> sin slots -> in_fifos -> Wishbone; Control path RF CRAC -> cra -> sout Slot1/2 -> sin slt2 -> cra readback -> RF; Service path fifo status -> int/dma_if -> RF/req/ack.

-- Clock Domain Partitioning and CDC Strategy --

Clock domains
- Two asynchronous clock domains are used:
- System clock (clk): Wishbone/control/DMA, register file and interrupts, FIFO management, CRAC formatting/readback, tag generation, suspend monitor, reset sequencer.
- AC-link bit clock (wclk): AC’97 serial framing/timing, SDATA_OUT parallel-load and serial shift, SDATA_IN sampling/deserialization, per-slot strobes and frame counter.

Domain placement by function
- wclk domain:
- ac97_sout: loads parallel slot words at frame-load (so_ld) and shifts MSB-first each wclk.
- ac97_sin: samples SDATA_IN on both wclk edges, assembles slot words, asserts per-slot strobes.
- ac97_soc (bit portion): frame counter; out_le[] strobes; sync_beat generation; valid/in_valid windows.
- clk domain:
- ac97_wb_if: Wishbone slave and address decode.
- ac97_rf: register file (CSR/OCC/ICC/CRAC/INTM/INTS) and masked interrupt aggregation.
- ac97_prc (+ ac97_fifo_ctrl): Slot0 tag builder; per-channel one-shot enables for FIFO service.
- ac97_cra: CRAC formatter for slots 1/2; readback capture and done strobes.

- ac97_out_fifo / ac97_in_fifo: synchronous sample FIFOs; 20-bit AC-link to 32-bit system reformatting.
- ac97_dma_if (+ ac97_dma_req): threshold-qualified, held-until-ack DMA request generation.
- ac97_int: interrupt cause (level/underflow/overflow attempts).
- ac97_rst: timed deassertion of active-low codec reset.
- ac97_soc (system portion): bit-clock presence monitor; suspend/resume control.

CDC strategy and crossings
- Framing strobes wclk→clk:
- valid and in_valid are created in wclk and synchronized into clk via two-stage pipelines (valid→valid_s1→valid_s; in_valid→in_valid_s1→in_valid_s).
- The synchronized strobes qualify CRAC operations, tag latching, and FIFO service so system actions align to stable frame boundaries.
- Slot data wclk→clk:
- ac97_sin captures slot words in wclk; transfer into system FIFOs only under in_valid_s and channel enables to avoid mid-slot sampling.
- CRAC readback (e.g., in_slt2[19:4]) is captured on a clk-domain strobe (crac_rd_done) derived from valid_s, never from raw wclk-domain valid.
- System drive clk→wclk:
- Slot0 tag (ac97_prc), CRAC slots 1/2 (ac97_cra), and TX FIFO outputs are produced in clk and must be stable across the wclk frame-load event (so_ld).
- Ensure setup/hold at so_ld by either double-registering these buses in wclk or only updating them while valid_s is low; avoid combinational CDC paths to ac97_sout inputs.
- Bit clock presence and suspend/resume:
- wclk is sampled into clk with a two-flop synchronizer; an XOR edge detector creates bit_clk_e to feed a timeout counter that asserts suspended when edges cease.
- suspended (generated in clk) that affects wclk logic (e.g., frame counter hold) must be synchronized into wclk with a two-flop synchronizer.
- resume_req (clk) produces sync_resume; before ORing with sync_beat (wclk) to drive SYNC, sync_resume must be synchronized into wclk.
- One-shot enables and snapshots (clk domain):
- ac97_fifo_ctrl snapshots FIFO full/empty while valid_s is low; generates a single pulse per valid_s window when conditions are met. This prevents retriggers and avoids sampling fast-changing status across domains.

CDC safety expectations
- Use two-flop synchronizers for all level crossings (e.g., suspended clk→wclk, sync_resume clk→wclk, wclk sampling into clk).
- Convert edges to pulses in the receiving domain; qualify all actions with valid_s/in_valid_s.
- Hold parallel slot contents constant throughout the load/capture aperture around so_ld.
- Do not sample live FIFO status across domains; rely on clk-domain snapshots and per-frame pulses.

Reset strategy
- Asynchronous active-low resets are used in ac97_soc (frame counter preset) and ac97_rst; release resets per-domain to avoid metastability on deassertion.
- Most system blocks reset synchronously. For blocks without explicit reset (e.g., ac97_wb_if), consider adding a top-level reset or power-on init to avoid indeterminate handshakes.

STA and constraints
- Declare clk and wclk asynchronous. Add false paths between domains except for explicitly synchronized nets.
- Constrain two-flop synchronizers; apply appropriate multicycle/false-path settings to the wclk→clk edge detector path (bit_clk_e).

- Constrain the clk→wclk paths feeding ac97_sout so parallel data meets so_ld timing (or treat as CDC if double-registered in wclk).

Known crossings to review/harden
- suspended clk→wclk: add two-flop sync before gating the wclk-domain frame counter.
- sync_resume clk→wclk: synchronize before OR with sync_beat to form SYNC.
- valid/in_valid wclk→clk: maintain two-flop pipelines and STA constraints.
- CRAC read capture: drive crac_rd_done from clk-domain valid_s edges to guarantee stable data.

Guidelines for future signals
- For any new control/status that crosses domains, synchronize levels with two flops, derive per-frame pulses in the receiving domain, and align operations to valid_s/in_valid_s windows.

-- Frame/Timing Generation (SYNC, ld, valid, out_le) --

Generates all AC-link frame/timing strobes in the bit-clock (clk) domain using an 8-bit modulo-256 counter (cnt). All pulses are synchronous to clk and one cycle wide unless stated.
- Counter (cnt):
- Free-runs when not suspended, incrementing each clk and wrapping 0xFF->0x00 to define a 256-tick frame.
- Asynchronous preset to 0xFF on reset, and forced to 0xFF while suspended (no frame activity while suspended).
- Frame-load (ld):
- Asserted for one clk at cnt==0x00, exactly once per frame; used to load TX slots. Suppressed while suspended (cnt held at 0xFF).
- SYNC (sync_o):
- sync_o = sync_beat OR sync_resume.
- sync_beat: high for the first 16 bit clocks of every frame (cnt in 0x00..0x0F).
- sync_resume: high during the resume burst (originates from wclk domain) and forces SYNC high independent of cnt/suspend.
- valid (late-frame qualifier):
- Asserted when cnt > 0x39 (i.e., cnt in 0x3A..0xFF), providing a stable late-frame window for Slot0 tag generation and CRA operations.
- Slot latch-enables (out_le[5:0]):
- Single-cycle pulses placed at the end of specific slot windows (MSB-first alignment) after the 16-bit tag:
- out_le[0] at cnt==0x11,
- out_le[1] at cnt==0x25,
- out_le[2] at cnt==0x39,
- out_le[3] at cnt==0x4D,
- out_le[4] at cnt==0x61,
- out_le[5] at cnt==0x89.
- Pulses are spaced by 0x14 (20-bit) steps after the tag, with a larger gap to 0x89 to skip an unused slot. No out_le pulses occur while suspended.
- Guarantees:
- Exactly one ld and at most one out_le[k] per frame; sync_beat always covers cnt 0x00..0x0F when running; valid continuously covers cnt 0x3A..0xFF.
- On reset/suspend release, the first ld occurs at the first cnt==0x00 boundary and timing proceeds per the schedule above.

-- Serial Transmitter (SDATA_OUT Shifter) --

Parallel-to-serial AC’97 SDATA_OUT shifter that formats a 256-bit frame from per-slot parallel words. Operates in the AC-link bit-clock domain (clk). A one-cycle load strobe (so_ld), generated by the frame/timing block (ac97_soc) at the frame boundary (cnt==0), must occur exactly once every 256 bit clocks. On so_ld, the shifter captures: Slot 0 tag slt0[15:0] from ac97_prc; control slots slt1[19:0] and slt2[19:0] from ac97_cra (considered meaningful when crac_valid is asserted; codec interpretation is governed by the tag); data slots slt3, slt4, slt6, slt7, slt8, slt9 (20 bits each) typically sourced by per-channel TX FIFOs, with samples left-aligned and zero-padded to 20 bits according to channel mode. Reserved slots slt5, slt10, slt11, slt12 are forced to 20’h0 every frame. After load, bits are emitted MSB-first through a chained shift: sdata_out taps slt0_r[15]; slt0_r shifts left pulling its incoming MSB from slt1_r[19]; each subsequent slot pulls from the next slot’s MSB; the final slot shifts in zero. Bit order per frame is slt0[15:0] followed by slt1..slt12, each [19:0], for a total of 16 + (12×20) = 256 bits; after slot 12 is exhausted, zeros continue until the next so_ld. AC’97 alignment: sync_beat is high for the first 16 clocks (Slot 0) while sdata_out presents slt0[15:0]. Tag semantics: slt0[15] is the OR of [14:6] (valid frame), [14] indicates CRAC present (Slot 1), [13] indicates CRAC write (Slot 2), and [12:6] map to active data slots (3,4,6,7,8,9); reserved slots are advertised inactive and transmitted as zero. No explicit reset; prior to the first load contents are undefined, and if frames stop the output trends to zero due to the zero-fed tail. Integration/timing: data sources must present stable sltN values at the so_ld edge; MSB-first ordering and zeroed reserved slots ensure AC’97-compliant framing; all relevant paths (ac97_sout, ac97_soc, ac97_cra, ac97_prc) operate in the bit-clock domain at load/shift, and any cross-domain controls must be synchronized before use.

-- Serial Receiver (SDATA_IN Deserializer) --

Serial Receiver (SDATA_IN Deserializer)

Purpose:
- Convert the AC'97 SDATA_IN serial stream into parallel per-slot words and present them at precise bit boundaries for downstream logic.

Clocking and sampling (AC'97 bit-clock domain):
- Two-edge sampling for timing margin: SDATA_IN is sampled on the falling edge into a staging flip-flop; on the next rising edge a 20-bit shift register shifts left and ingests the staged bit at its LSB.
- Provides a half-cycle of setup prior to shifting, matching AC'97 timing (codec drives around rising edge; controller samples around falling edge).

Shift register and bit order:
- Shift operation: sr <= {sr[18:0], staged_bit}; newest bit enters sr[0], older bits move toward sr[19].
- With MSB-first streams, after 20 shifts sr[19:0] contains the word MSB..LSB in correct order.

Slot capture and mapping:
- Single-cycle latch-enable pulses (out_le[x]) mark the end of each slot; on the rising edge when out_le[x] is asserted, sr is copied into the slot register.
- Captures:
- Slot 0 (tag): slt0 <= sr[15:0] (16-bit)
- Slots 1–4: slt1..slt4 <= sr (20-bit each)
- Slot 6: slt6 <= sr (20-bit)
- Slot 5 on SDATA_IN is intentionally not captured; out_le[5] is used to trigger slt6 capture (no slt5 register).

Timing alignment requirements:
- out_le pulses must coincide with the last bit time of each target field so sr holds a complete, correctly ordered word at latch time.

- The deserializer performs no self-alignment; it relies entirely on the external AC'97 frame-timing block for SYNC, frame counting, and out_le scheduling.
- Typical out_le schedules (from frame-timing) occur at defined bit counts within the 256-bit frame (e.g., 0x11, 0x25, 0x39, 0x4D, 0x61, 0x89) and are spaced primarily by 20 bits.

Downstream usage:
- slt0[15:0] carries the received frame tag.
- slt2[19:4] is consumed by Control Register Access logic for inbound CRA read data returned on SDATA_IN Slot 2.
- slt3/slt4/slt6 feed RX channel paths and their input FIFOs per AC'97 conventions.
- in_valid windows generated by frame-timing (e.g., after cnt > 0x4D, > 0x61, > 0x89) qualify downstream FIFO transactions; the deserializer itself captures solely on out_le.

Latency and update points:
- Each incoming bit experiences ~half-cycle latency (negedge sample to next posedge shift).
- Slot registers update synchronously on the rising edge when the corresponding out_le pulse occurs.

Reset and power-up:
- No internal reset; registers power up unknown. Correct operation is achieved once external framing (SYNC, frame counter, out_le pulses) is aligned and runs for at least one frame.
- System logic should ignore early values until framing is active and aligned.

Domain and integration notes:
- All logic operates in the AC'97 bit-clock domain; parallel outputs must be consumed in the same domain or transferred via proper CDC to other domains.
- CDC for movement to system/DMA clocks is not handled inside this deserializer.
- Ensure STA/constraints treat these outputs as bit-clock domain signals.

Assumptions:
- AC'97-compliant MSB-first serial data on SDATA_IN.
- Accurate, slot-end-aligned out_le strobes from the frame-timing generator.
- Downstream consumers adhere to bit-clock domain timing or apply CDC for cross-domain transfers.

-- Slot■0 Tag and FIFO Control --

Purpose: Generate the AC’97 Slot■0 tag (16■bit) that declares which slots in the next frame are valid, and produce one■shot per■frame enables that trigger exactly one safe FIFO transaction for each advertised slot.

Tag composition (out_slt0[15:0]):
- [15] = OR of [14:6] (Valid Frame)
- [14] = crac_valid_r (Slot 1: CRAC active)
- [13] = crac_wr_r (Slot 2: CRAC write phase; 0 = read)
- [12] = o3_re_l (Slot 3 present)
- [11] = o4_re_l (Slot 4 present)
- [10] = 0 (Slot 5 unused)
- [09] = o6_re_l (Slot 6 present)
- [08] = o7_re_l (Slot 7 present)
- [07] = o8_re_l (Slot 8 present)
- [06] = o9_re_l (Slot 9 present)
- [5:0] = 0 (Slots 10–15 unused)
- If none of [14:6] are set, [15]=0 marking the frame invalid per AC’97.

CRAC latching:
- crac_valid/crac_wr are sampled into crac_valid_r/crac_wr_r only while valid=1 to hold a stable indication across the upcoming frame. Module has no explicit reset; these become defined after the first valid window.

Timing/integration:
- ac97_soc provides the per■frame valid window and load strobes (ld/so_ld). During valid, ac97_prc samples CRAC and qualifies per■slot enables; out_slt0 is then loaded by ac97_sout at next frame start (so_ld) and shifted MSB■first on SDATA_OUT. Reserved/unused slots are forced to zero. Use the aligned valid from the top■level pipelines.

FIFO control (per slot 3,4,6,7,8,9):
- Generate a single■cycle enable pulse (oX_re_l) once per frame when all are true: valid=1, channel enabled (ch_en), interface ready (crdy), pre■slot snapshot indicates safe (!full_empty_r), and either srs=0 or (srs=1 and req=1).
- Snapshot full_empty into full_empty_r only while valid=0 to avoid in■slot races; choose full_empty polarity appropriate to direction (e.g., TX: empty=bad; RX: full=bad).
- Use edge detection on the qualified condition so sustained readiness yields one pulse per frame, not retriggers.
- oX_re_l both asserts the corresponding Slot■0 presence bit and triggers exactly one FIFO transaction for that slot.

End■to■end behavior:
- Any slot advertised in Slot■0 will transfer exactly one sample in that frame when FIFO and readiness conditions permit; control slots 1/2 reflect CRAC state to correctly advertise and time control accesses.

-- Codec Register Access Path --

Purpose: Implements AC’97 Codec Register Access (CRA) end-to-end, from host register programming to AC-link command transport, readback capture, and completion/interrupt signaling.
Programming Model (host/Wishbone RF):
- CRAC (RF addr index 4) launches a transaction on write and provides readback on read.
- Write command word crac_out[31:0]: [31]=1 read / 0 write; [22:16]=7-bit codec register address; [15:0]=write data. ac97_rf asserts crac_we for one clk on CRAC write and latches fields.
- CRAC readback format: {RW flag, 8’h00, addr[6:0], crac_din[15:0]}.
- Interrupts: INTS (RF addr 6) bit[0]=crac_rd_done, bit[1]=crac_wr_done; masked by INTM (RF addr 5) and OR’d to int_o. Reading INTS clears all status bits.
- Wishbone: ac97_wb_if provides single-cycle ack; RF region decode places CRAC at adr=4, INTM at 5, INTS at 6, CSR at 0.
AC-link formatting and tagging:
- Slot 0 tag: Tag[14]=crac_valid (Slot1 present), Tag[13]=crac_wr_r (Slot2 present for writes), Tag[15]=OR of [14:6].
- Slot 1 (command): out_slt1[19]=R/W flag; [18:12]=addr[6:0]; [11:0]=0.
- Slot 2 (data): out_slt2[19:4]=write/read data[15:0]; [3:0]=0.
- ac97_sout serializes SYNC, Slot 0, 1, 2 MSB-first each frame; ac97_sin deserializes and latches slots on out_le strobes.
Timing and sequencing (clk domain):
- valid window asserted late in frame (cnt > 0x39). ac97_cra uses valid_pe/ne edges.
- Write: crac_wr asserted for the command frame’s valid; crac_valid high that frame; crac_wr_done pulses at valid_ne.
- Read: crac_rd asserted for the command frame’s valid; crac_valid high for that frame. Read data

returns next frame in Slot 2; crac_rd_done pulses at the next valid_pe and crac_din[15:0] captures in_slt2[19:4].
Suspend/resume:
- Link valid depends on ac97_soc; when suspended (bit-clock loss) the frame counter is held. Software resumes via CSR (RF addr 0, resume_req pulse); issue CRA only when valid is active.
Constraints and rules:
- No queuing: do not write CRAC while crac_valid is active; only one outstanding transaction.
- Slots 1/2 are meaningful only when advertised by Slot 0 tag (crac_valid and crac_wr_r).
- Data alignment: Slot 2 uses bits [19:4]; lower 4 bits are zero for both TX and RX.
Software flows:
- Write: program CRAC (RW=0, addr, data) -> wait for INTS[1] or poll -> optional CRAC read for confirmation.
- Read: program CRAC (RW=1, addr) -> wait for INTS[0] -> read CRAC to obtain crac_din.
Clocking and CDC: CRA, RF, PRC, SIN/SOUT operate in clk domain; behavior assumes stable valid provided by ac97_soc.

-- Playback FIFOs (Output) --

Purpose and scope: Per-channel output FIFOs buffer 32-bit system writes and feed 20-bit AC’97 playback slots (3, 4, 6, 7, 8, 9) at frame rate without underruns.

Structure and modes: 4×32-bit circular buffer per channel. Mode from ocX_cfg[3:2]: 00=16-bit (two 16-bit samples per word, left-shift by 4 on read), 01=18-bit (one sample per word, shift by 2), 10=20-bit (one sample per word, pass-through). Effective capacity: 8 samples (16-bit) or 4 samples (18/20-bit).

Read/write behavior: Write (we) stores din at mem[wp[1:0]] and advances wp by one word; reads (re) present a single 20-bit MSB-aligned sample on dout. Read pointer advances half-word per frame in 16-bit mode (toggle lower bit) or one word per frame in 18/20-bit. At most one read per frame is issued by ac97_prc/fifo_ctrl when the valid window is active, the channel is enabled, data is available, and timing permits; otherwise no read and the slot is untagged for that frame.

Output to link: On scheduled read, Slot 0 tag bit for that channel is set and the 20-bit dout is loaded into the respective playback slot; SDATA_OUT shifts MSB-first. Slots 3, 4, 6, 7, 8, 9 carry playback; other TX slots (5, 10–12) are forced to zero.

Bus interface (system writes): Wishbone write apertures per channel: 0x20 (o3), 0x24 (o4), 0x28 (o6), 0x2C (o7), 0x30 (o8), 0x34 (o9). A bus write generates a one-cycle oX_we with wb_data_i as din; single-beat, 1-cycle ack; data is latched centrally and presented to the FIFO in the same cycle.

Status and flags: full indicates 4-word full; empty is meaningful only in 16-bit mode (unreliable in 18/20-bit). status[1:0] exposes modulo-4 word-level occupancy (coarse, not sample-accurate). The FIFO does not internally block writes when full or reads when empty; overflow/underflow attempts are detected for interrupt purposes, and top-level logic must gate we/re.

Interrupts and DMA: Per-channel configuration via ocX_cfg. Level/threshold interrupt enabled by cfg[0], threshold via cfg[5:4] and a boundary flag; single-cycle causes for overflow-attempt and underflow-attempt. DMA request asserted when FIFO crosses programmable thresholds (cfg[5:4]), gated by cfg[6] and cfg[0], and held until dma_ack.

Enable/reset behavior: en acts as synchronous enable/reset; when deasserted, pointers and dout clear, dropping buffered data; normal operation resumes when asserted.

Throughput and framing: Consumption is one sample per frame per enabled slot. In 16-bit mode, one

32-bit word supplies two frames; in 18/20-bit, one word supplies one frame. fifo_ctrl snapshots FIFO boundary status at the start of the valid window to issue one clean read per frame and set Slot 0 tagging only when a transfer occurs, preventing mid-frame chattering.

Limitations and guidance: empty flag should not be used in 18/20-bit modes; rely on scheduler gating and status thresholds. status[1:0] is coarse; plan DMA thresholds carefully given small depth. No internal flow control—system/DMA must prevent overflow/underflow. Data alignment: 16-bit shift by 4, 18-bit by 2, 20-bit none. Disabling a channel immediately clears buffered content.

-- Capture FIFOs (Input) --

Three capture FIFOs (i3, i4, i6) buffer AC'97 SDATA_IN from slots 3, 4, and 6 for system/DMA consumption.
- Buffer and modes: Each FIFO is a 4-word × 32-bit synchronous buffer (ac97_in_fifo). Mode selects sample width/packing from ic0_cfg/ic1_cfg/ic2_cfg[3:2]: 0=16-bit (pack two samples per 32-bit word using din[19:4]), 1=18-bit (zero-extend din[19:2] to 32 bits), 2=20-bit (zero-extend din[19:0]).
- Pointer behavior: 16-bit mode uses a half-sample toggle to pack pairs (advance per incoming sample; commit one 32-bit word per two samples). 18/20-bit modes write one 32-bit word per sample (advance one word per sample). Reads advance the read pointer by one word per access.
- Ingress path: ac97_sin deserializes SDATA_IN and latches slot words on out_le[x] strobes; slots 3/4/6 feed i3/i4/i6. Per-channel one-shot write enables are gated per frame (valid/in_valid, channel enable, !full, codec/controller readiness), ensuring at most one FIFO write per advertised slot per frame and preventing overruns. in_valid windows open late in the frame; top-level pipelines in_valid for stable qualification.
- Egress path: Wishbone reads at 0x38 (i3), 0x3C (i4), 0x40 (i6) return the next 32-bit word and assert a one-cycle iX_re pulse to advance the read pointer. A DMA interface can request service based on programmable thresholds; requests are level-held until dma_ack.
- Status/flags: Full detection uses ring-buffer logic (e.g., (wp[2:1]==rp[1:0]) && (wp[3]!=rp[2])). Empty reporting is meaningful in 16-bit packing; in 18/20-bit modes use full plus status for flow control. status provides a coarse 2-bit word-level occupancy metric for thresholding/requests.
- Interrupts: Per-channel causes include configurable level/threshold (cfg[5:4], gated by cfg[0]), underflow attempt (read while empty), and overflow attempt (write while full). Register file aggregates i3/i4/i6 interrupt bits into INTS and masks via INTM.
- DMA: Requests use cfg[6] and cfg[0] enables plus cfg[5:4] threshold selection; conditions are qualified for one cycle and requests held until acknowledged.
- Reset/enable: Deasserting en synchronously clears pointers and idles FIFOs. FIFOs do not internally block operations when full/empty; external gating (fifo_ctrl, bus/DMA logic) must prevent overflow/underflow.
- Implementation notes: ac97_in_fifo uses 4×32 memory with wp[3:0] (half-sample toggle in mode 0) and rp[2:0] word index. Zero-extension in 20-bit mode is to 32 bits; verify RTL handling of mode==3 (unused/default). CDC/timing in ac97_sin uses negedge/posedge staging; slot strobes align to end-of-slot boundaries to ensure complete words on capture.

-- Register File and Interrupt Aggregation --

Register File and Interrupt Aggregation
- Bus/Addressing: ac97_wb_if exposes a Wishbone slave register file (ac97_rf) at word offsets wb_addr[5:2] = 0..7. All accesses are single-cycle with immediate acknowledge (no wait states). Word offsets 14..16 are FIFO data windows and not part of the register file.

Register map (offsets 0..6)
- 0: CSR (Control/Status)
- Read: bit[1] = suspended (codec/link suspend indicator); bit[0]=0; all other bits 0.

- Write: bit[0] pulses ac97_rst_force (timed reset sequencer); bit[1] pulses resume_req (link resume sequence). Writes do not update any persistent readback field.
- 1: OCC0 (Output Channel Config 0)
- 32-bit RW. Bytes: oc0_cfg = [7:0], oc1_cfg = [15:8], oc2_cfg = [23:16], oc3_cfg = [31:24].
- 2: OCC1 (Output Channel Config 1)
- Lower 24 bits RW, readback zero-extended. Bytes: oc4_cfg = [7:0], oc5_cfg = [15:8].
- 3: ICC (Input Channel Config)
- Lower 24 bits RW, readback zero-extended. Bytes: ic0_cfg = [7:0], ic1_cfg = [15:8], ic2_cfg = [23:16].
- 4: CRAC (Codec Register Access Control)
- Write: packs command as crac_out <= {rf_din[31], 8'h00, rf_din[22:16], rf_din[15:0]}; crac_we qualifies the write.
- Read: returns live readback and last command header as rf_dout = {crac_r[7], 8'h00, crac_r[6:0], crac_din}.
- Interface: ac97_cra consumes crac_out/crac_we and produces crac_rd_done, crac_wr_done pulses and crac_din[15:0].
- 5: INTM (Interrupt Mask)
- 29-bit RW mask (bits [28:0]). A 1 enables the corresponding INTS bit to contribute to the global interrupt. Reset default 0.
- 6: INTS (Interrupt Status)
- 29-bit sticky status (bits [28:0]). Read-to-clear: any read clears all bits. Reset default 0.

Configuration byte semantics (oc*/ic*)
- cfg[5:4]: Select threshold mode used by both interrupt level detection (int_set[0]) and DMA request logic (e.g., boundary-only, empty-only, below-half, below-almost-full; interpretation depends on TX/RX direction).
- cfg[0]: Enables/gates the level/threshold interrupt cause (int_set[0]).
- Other bits (e.g., cfg[6]) may gate DMA and do not directly affect ac97_int.

Interrupt sources and aggregation
- INTS bit assignments [28:0]:
- [0] crac_rd_done (CRA read completes; data latched)
- [1] crac_wr_done (CRA write frame completed)
- [2:4] oc0_int_set[2:0]
- [5:7] oc1_int_set[2:0]
- [8:10] oc2_int_set[2:0]
- [11:13] oc3_int_set[2:0]
- [14:16] oc4_int_set[2:0]
- [17:19] oc5_int_set[2:0]
- [20:22] ic0_int_set[2:0]
- [23:25] ic1_int_set[2:0]
- [26:28] ic2_int_set[2:0]
- Per-channel int_set encoding (from ac97_int):
- int_set[0]: Level/threshold condition; asserted when enabled by cfg[0] and the selected threshold via cfg[5:4] is met, with the channel boundary flag (full_empty) included in the condition.
- int_set[1]: Underflow attempt (read enable while FIFO empty).
- int_set[2]: Overflow attempt (write enable while FIFO full).
- Sticky behavior: ac97_rf ORs incoming int_set causes into ints_r (sticky). Set bits remain asserted until INTS is read (which clears all bits).

Global interrupt output
- Masking and assertion: int_all = intm_r & ints_r (bitwise AND). The global interrupt output is a registered single-wire level that asserts when any bit in int_all is 1 and deasserts after INTS is read or

masked bits are cleared.
- Level reassertion: Because int_set[0] is level-driven and INTS is sticky/read-to-clear, a persistent threshold condition will immediately reassert the corresponding status bit after a clear unless masked or the FIFO level condition is resolved.

Reset behavior
- Asynchronous active-low reset clears OCCx, ICC, INTM, INTS, and the CRAC command header; global interrupt deasserts. The CRAC write-data latch is undefined until the first CRAC write. CSR read under reset only reflects suspended (other CSR fields read as 0).

Software usage
- Program OCCx/ICC for per-channel modes and thresholds.
- Enable desired interrupt sources in INTM.
- Use the global interrupt or polling on INTS; reading INTS both reports causes and clears all sticky bits (no per-bit clear).
- For codec register access, write CRAC to launch, wait for crac_wr_done or crac_rd_done (via INTS), then read CRAC to obtain crac_din for reads.
- Use CSR writes to initiate AC’97 reset or resume sequences; CSR reads to monitor suspended.
- If selective suppression is required, adjust INTM via read-modify-write because INTS clears all bits on any read.

-- DMA Request Generation --

Per-channel, level-style DMA requests are generated in the system clk domain to move audio sample words between system memory and on-chip FIFOs. Each stream (TX playback o3..o9, RX capture i3/i4/i6) has an independent requester with a single dma_req output and dma_ack input. Requests do not depend on AC’97 frame timing.

Enables and gating: A channel can request DMA only if cfg[6] (channel enable) = 1 and cfg[0] (DMA enable) = 1. Changes to cfg take effect immediately; disabling either bit deasserts/blocks dma_req; re-enabling requires re-qualification before dma_req reasserts.

Inputs used for decision: full_empty is the direction-specific boundary flag (TX: empty; RX: full) and status[1:0] is a 2-bit coarse FIFO level/space indicator (0..3). In non-16-bit sample modes, empty may be unreliable; robust behavior relies on full and status.

Threshold selection (cfg[5:4]):
- 00: request if (full_empty == 1) OR (status < 3)
- 01: request if (full_empty == 1) OR (status[1] == 0) // status in {0,1}
- 10: request if (full_empty == 1) OR (status == 0)
- 11: request only if (full_empty == 1)
In modes 00–10, full_empty acts as an immediate trigger; in mode 11 it is the sole trigger.

Qualification and handshake: A request condition must be true for two consecutive clk cycles (glitch filter) while dma_ack is low to assert dma_req. Once asserted, dma_req remains high until dma_ack is observed high, which clears the request. While dma_ack is high, no new request is captured. After dma_ack deasserts, if the condition persists, a new request will reassert after passing the two-cycle qualification again.

Reset and timing: All requester state is synchronous to clk; async active-low reset immediately clears dma_req. Requests reflect FIFO service needs at the system rate and are independent from interrupt generation.

Behavioral guarantees: At most one outstanding request per channel; no chatter due to two-cycle qualification and hold-until-ack behavior. Sample format bits cfg[3:2] select FIFO data width but do not affect DMA request logic.

-- Reset Sequencer --

Reset Sequencer for AC’97 codec pin reset (ac97_rst_, active-low).

Purpose
- Provide a deterministic, glitch-free hardware reset pulse to the external AC’97 codec after power-up or a software-forced reset.

Interfaces
- clk: Sequencer clock input.
- rst: Asynchronous, active-low global reset input.
- rst_force: Synchronous force-reset request (1-cycle pulse from CSR write).
- ac97_rst_: Active-low reset output to the codec.

Behavior
- Assertion: ac97_rst_ drives low immediately when rst==0 (async) or while rst_force==1 (sync). Internal counters are cleared during assertion.
- Release sequencing (only when rst==1 and rst_force==0):
- Prescaler: 6-bit counter counts 0..49 and emits a 1-cycle enable pulse (ps_ce) every 50 clk cycles.
- Timeout: 3-bit counter increments once per ps_ce while its value <4; on reaching 4, a terminal flag (to) asserts.
- Deassertion: On the clock edge after to asserts, ac97_rst_ drives high and remains high until the next rst or rst_force.
- Hold-low duration: (50 prescaler cycles × 4 increments) + 1 final cycle ≈ 201 clk cycles from the end of reset/force to deassertion.
- After release: Prescaler continues free-running; timeout counter saturates at 4 (no further state changes).

Software-forced reset
- Writing CSR address 0 with bit0=1 generates a one-clock rst_force pulse that immediately reasserts ac97_rst_ low and restarts the full ~200-cycle hold.

Independence from link suspend/resume
- Resume requests (CSR bit1) control AC-link SYNC in ac97_soc and do not affect ac97_rst_. Reset sequencing and resume are independent.

Corner cases and guarantees
- If clk halts, the prescaler does not advance and ac97_rst_ stays asserted low (fail-safe).
- ac97_rst_ remains high once released until a new rst or rst_force event.
- Sequencing is fully synchronous to clk except for the async rst input.

Configuration
- Duration is defined by two constants: prescaler terminal count=49 (50-cycle period) and timeout terminal count=4; these can be tuned in RTL if a different reset width is required.

Integration notes
- ac97_rst_ controls only the external codec reset pin and is independent of internal block resets or enable-based initialization in other AC’97 logic.

OPERATION

-- Power■Up and Reset Sequence --

Power-up sequence and resets:
- Assert global rst low with clocks running. While rst=0: ac97_rf, ac97_cra, ac97_soc, ac97_int, and ac97_dma_if clear to 0; ac97_rst drives ac97_rst_ low to hold the codec in reset; interrupts and DMA requests are disabled; FIFOs remain cleared when en=0.
- Deassert rst high. ac97_rst keeps ac97_rst_ low for a fixed delay (~200 clk cycles) and then releases ac97_rst_ high. After release, ac97_rf contents are 0 (INTM=0, INTS=0, configuration registers cleared).
- AC-link activity check: ac97_soc monitors the bit clock. If no edges are seen, suspended=1, the frame counter is held at 0xFF, and SYNC is suppressed. When edges resume, suspended clears and 256-count framing restarts.
- Resume the link: Software writes CSR[1]=1 (resume_req pulse). ac97_soc asserts a short sync_resume and then returns to normal sync_beat. Poll CSR suspended status to confirm recovery.
- Stabilization before traffic: Wait at least one full frame after resume before enabling channels or starting codec register access (CRAC). Modules without explicit reset (ac97_sout, ac97_sin, ac97_prc, ac97_fifo_ctrl, and channel FIFOs) settle on ld/valid/out_le; SDATA_IN/OUT are undefined until valid framing signals appear. Keep FIFO en=0 during configuration; en=0 synchronously clears FIFO pointers and data.
- Configuration and enable: Program OCC0/OCC1/ICC for per-channel modes and enables. Enable interrupts (INTM) and configure DMA thresholds. Gate FIFO re/we and CRAC drive with the per-slot valid window to prevent underflow/overflow, then enable streams and CRAC.
- Forced reset (runtime): Writing CSR[0]=1 generates ac97_rst_force, reasserts ac97_rst_ low, and repeats the same fixed release delay. After release, perform resume and reconfigure before resuming traffic.
- Bit-clock loss recovery: If the bit clock stops, ac97_soc asserts suspended, freezes the frame counter, and suppresses SYNC. Restore the clock and issue resume_req to reestablish framing.
- Integration cautions: The Wishbone bus interface has no explicit reset; avoid accesses until global reset is complete and ac97_rst_ has been released.

-- AC’97 Frame Lifecycle and Slot Timing --

Summary
- Each AC’97 frame is 256 bit clocks long per SDATA stream: a 16-bit Slot 0 (tag) followed by twelve 20-bit slots (1..12). Bits transmit MSB-first, and all timing derives from the AC-link bit clock.

Frame lifecycle (clk domain)
- Frame counter cnt[7:0] free-runs 0x00..0xFF when link is active.
- ld (frame-load) asserts for one clk at cnt==0x00, marking the start of every frame and the time TX data is (re)loaded.
- SYNC is asserted high for the first 16 bit clocks of each frame (cnt 0x00..0x0F). The output sync_o = sync_beat | sync_resume, where sync_resume is a short burst used only during resume.
- Total bit allocation per frame (reference indexing from ld): [0..15] Slot 0, [16..35] Slot 1, [36..55] Slot 2, [56..75] Slot 3, [76..95] Slot 4, [96..115] Slot 5, [116..135] Slot 6, [136..155] Slot 7, [156..175] Slot 8, [176..195] Slot 9, [196..215] Slot 10, [216..235] Slot 11, [236..255] Slot 12.

Suspend/resume behavior
- A wclk-domain clock monitor asserts suspended if the bit clock stops. While suspended, cnt is held at 0xFF, suppressing normal ld and SYNC framing.
- A host-driven resume_req triggers a brief sync_resume burst to re-enter normal framing; regular SYNC then reverts to the standard 16-bit high at the top of frame.
- Note: Because cnt is forced to 0xFF during suspend, a simple threshold like (cnt > 0x39) would read true; production logic must additionally qualify actions with ld/sync or not-suspended.

Slot end strobes and windows (clk domain)
- Per-slot latch-enable pulses out_le[n] assert at these cnt values (aligning to the end of each field):
- out_le[0] at 0x11 (17): end of Slot 0 (16-bit tag)
- out_le[1] at 0x25 (37): end of Slot 1 (20-bit)
- out_le[2] at 0x39 (57): end of Slot 2 (20-bit)
- out_le[3] at 0x4D (77): end of Slot 3 (20-bit)
- out_le[4] at 0x61 (97): end of Slot 4 (20-bit)
- out_le[5] at 0x89 (137): end of Slot 6 (20-bit) — Slot 5 is intentionally skipped on RX
- valid (frame/data window) is high when cnt > 0x39 (i.e., late in the frame) and is used to qualify PCM/data operations.
- in_valid windows (RX data-accept enables) open after these boundaries:
- in_valid[0]: cnt > 0x4D (after Slot 3 boundary)
- in_valid[1]: cnt > 0x61 (after Slot 4 boundary)
- in_valid[2]: cnt > 0x89 (after Slot 6 boundary)

Transmit serialization (SDATA_OUT)
- At each ld (so_ld), the TX shifter loads Slot 0 (16-bit tag) and Slots 1..12 (20-bit each). Slots 5, 10, 11, and 12 are forced to zero every frame.
- Bits shift MSB-first across slots with a chained shifter; the serial output starts at Slot0[15] immediately after load and proceeds through Slot 12 without gaps.
- so_ld must assert exactly once per 256 bit clocks at the frame boundary; missing or extra loads will corrupt slot alignment.

Receive deserialization (SDATA_IN)
- SDATA_IN is sampled on the falling edge, then shifted into a 20-bit shift register on the rising edge to maximize timing margin.
- On rising edges when an out_le pulse asserts, the current shift register contents are latched into slot registers:
- out_le[0] -> slt0[15:0] (tag)
- out_le[1]..[4] -> slt1..slt4 (20-bit)
- out_le[5] -> slt6 (20-bit). Slot 5 is intentionally not captured
- Correct operation requires out_le pulses to coincide with each slot’s end-of-field so the full field is captured coherently.

Slot 0 tag and control slots (context for timing)
- Slot 0 tag formation (during valid window):
- tag[15] = OR(tag[14:6]) indicating a valid frame
- tag[14] = Slot 1 (CRAC command) present; tag[13] = Slot 2 (CRAC write data) present
- tag[12:6] map to enabled TX audio/data slots (3,4,6,7,8,9); tag[10] and [5:0] are 0
- CRAC timing relative to frames:
- Writes (R/W=0 in Slot 1): command and data (Slot 2) are asserted in the current valid window; completion aligns with valid falling edge in the same frame
- Reads (R/W=1 in Slot 1): command is asserted in the current valid window; returned read data

appears in incoming Slot 2 of the following frame; completion strobe occurs at the start of the frame after that data frame
- The tag must advertise the presence of Slots 1/2 in the same frame they are driven (crac_valid ensures coherence).

Design notes
- RX slot capture and TX slot drive share the same 256-bit cadence; any change to counter placement or sample/shift edges must preserve the specified out_le positions and the one-shot ld at cnt==0x00.
- FIFO/DMA control should issue at most one transaction per frame per active slot, qualified by valid/in_valid and not-suspended, to avoid mid-frame retriggering.

-- Tag Generation and Slot Advertising --

- Purpose: Build and transmit the AC'97 Slot 0 tag each frame to advertise which downstream TX slots contain valid data and whether a Control Register Access (CRAC) is present. The codec uses this tag to accept Slots 1–2 (control) and the enabled PCM/data slots.
- Bit mapping (16-bit Slot 0 tag):
- [15] Valid Frame = OR of bits [14:6]. If none of [14:6] are set, the frame is not marked valid.
- [14] CRAC present (Slot 1). Driven by latched crac_valid.
- [13] CRAC write data present (Slot 2). 1 for writes (crac_wr=1), 0 for reads (no Slot 2 on TX).
- [12] Slot 3 present (o3_re_l).
- [11] Slot 4 present (o4_re_l).
- [10] 0 (Slot 5 never advertised).
- [09] Slot 6 present (o6_re_l).
- [08] Slot 7 present (o7_re_l).
- [07] Slot 8 present (o8_re_l).
- [06] Slot 9 present (o9_re_l).
- [05:00] 0 (Slots 10–15 never advertised).
- Per-slot advertising sources:
- Bits [12:6] derive from single-cycle enable pulses: o3_re_l, o4_re_l, o6_re_l, o7_re_l, o8_re_l, o9_re_l.
- Pulses are generated by ac97_fifo_ctrl per channel, producing at most one enable per frame when all of the following hold at the slot's valid window: channel enabled, link/codec ready, FIFO status (snapshotted before window) permits transfer, and any per-channel request gating is met. This guarantees the tag advertises only slots that will actually be serviced in that frame.
- CRAC interaction (ac97_cra):
- When a control transaction is active, crac_valid=1 asserts tag[14] (Slot 1). For writes, crac_wr=1 asserts tag[13] (Slot 2 present); for reads, tag[13]=0.
- ac97_cra supplies Slot 1 (R/W+address) and Slot 2 (write data) contents when crac_valid is high.
- Latching and timing/stability:
- crac_valid and crac_wr are sampled and latched only when the frame valid window is asserted, stabilizing tag[14:13] across the frame and preventing mid-frame glitches. These latched values are used on the next frame load.
- Per-slot pulses are edge-detected and constrained to one per frame by ac97_fifo_ctrl; retriggering within a frame is prevented by the pre-window snapshot and gating logic.
- Prior to the first qualified valid window after reset, tag latches may be undefined; thereafter the tag is stable frame-to-frame.
- Frame load and serial placement (ac97_soc/ac97_sout):
- ac97_soc asserts so_ld at the frame start (cnt==0) and drives SYNC high for the first 16 bit clocks (Slot 0).
- On so_ld, ac97_sout loads the 16-bit tag and shifts it MSB-first; SDATA_OUT presents tag[15] immediately after load, aligning with AC'97 timing.
- Unimplemented TX slots (5, 10–15) are forced to zero on load; the tag never advertises them.

- Receive-side note:
- Slot 0 on SDATA_IN is captured as a 16-bit tag for observation by higher layers; it does not influence transmit-side advertising.
- Advertised slots summary: 1 (CRAC), 2 (CRAC writes only), 3, 4, 6, 7, 8, 9. Slots 5 and 10–15 are never advertised.

-- Transmit Data Flow --

- Source and configuration
- Host programs per-channel TX config in ac97_rf (oc0_cfg..oc5_cfg); cfg[3:2] selects sample width (16/18/20), other bits gate DMA/interrupts and channel enable.
- System writes to ac97_wb_if channel apertures at 0x20..0x34, asserting o3_we..o9_we to push 32-bit words into per-channel ac97_out_fifo instances.
- ac97_dma_if monitors FIFO level against cfg[5:4]-selected thresholds, raises dma_req when crossed, and holds until dma_ack.

- Frame timing and tagging (AC-link domain)
- ac97_soc runs the 256-tick frame on the AC-link bit clock: ld at cnt==0, SYNC during first 16 ticks; data-phase is qualified when cnt>0x39 (valid window).
- ac97_prc builds Slot 0 tag: [12:6]=active TX channels (o3..o9), [14]=CRAC present, [13]=CRAC write; [15] is the OR of [14:6] so the frame is valid only if CRAC or any TX channel is active.

- Read scheduling from FIFOs
- fifo_ctrl issues exactly one read-enable (re) pulse per active channel per frame when: valid=1, channel enabled, controller/link ready, and the latched FIFO status indicates not-empty.
- FIFO status is snapshotted outside the valid window to avoid mid-slot flag changes; one-shot gating prevents multiple reads in the same frame.

- Sample extraction and formatting (ac97_out_fifo)
- Each re produces one 20-bit-aligned sample:
- 16-bit mode (cfg[3:2]=00): outputs {sample16, 4'b0}; two samples per 32-bit word; rp advances by half-word per re.
- 18-bit mode (01): outputs {word[17:0], 2'b0}; rp advances by one word per re.
- 20-bit mode (10): outputs word[19:0]; rp advances by one word per re.
- FIFO does not block invalid ops; external gating must avoid re on empty/full. Empty is reliable primarily in 16-bit mode; ac97_int flags underflow (re on empty) and overflow (we on full).

- CRAC transmit (control register access)
- On crac_we, ac97_cra formats Slot 1/2 and asserts crac_valid during the command frame.
- Writes complete at valid falling edge; reads return next frame via in_slt2 and pulse crac_rd_done.

- Frame assembly and serialization
- On each ld, ac97_sout loads slot registers: Slot 0 tag; Slots 1–2 from ac97_cra when crac_valid; Slots 3,4,6,7,8,9 from corresponding ac97_out_fifo outputs; Slots 5,10–12 forced to 0.
- ac97_sout shifts MSB-first one bit per clk to drive SDATA_OUT: slt0[15:0], then slt1..slt12[19:0] contiguous.

- Flow control, interrupts, and DMA
- ac97_int evaluates per-channel causes each clk: level/threshold (cfg gating), underflow attempt, overflow attempt; ac97_rf aggregates into sticky INTS, masks via INTM, asserts int_o on any enabled cause; reading INTS clears.
- ac97_dma_if generates qualified one-cycle dma_req on threshold crossings and holds until dma_ack

to sustain FIFO fill.

- Clocking, suspend/resume, and reset
- ac97_soc/ac97_sout operate in the AC-link bit-clock domain; bus/CFG/FIFOs are system-clocked; the top integrates small pipelines to align cross-domain strobes.
- If the bit clock stops, cnt holds at 0xFF; ld and normal SYNC cease; resume issues a forced SYNC to restart timing.
- ac97_rst_ is held low after reset and released high after a fixed delay (~200 clk cycles) to meet codec reset timing before transmit proceeds.

-- Receive Data Flow --

Receive path converts AC'97 SDATA_IN into per-slot parallel words, qualifies them safely in-frame, and writes formatted samples into per-channel input FIFOs (i3/i4/i6) for software/DMA readout.

Clocking and CDC
- Bit clock domain: deserialization and slot capture; system clock domain: qualification, FIFO write, bus/DMA/interrupts. Slot data is captured in the AC-link domain and only transferred after in_valid windows open; the top level pipelines these windows to generate one-shot enables per frame in the system domain.

Deserialization and slot capture
- SDATA_IN sampled on bit-clock falling edge; a 20-bit shift register advances on rising edge.
- Per-slot latch strobes (out_le) capture completed words into slot registers at window ends: slot0 tag 16 bits, slots1/2/3/4/6 as 20 bits; slot5 is intentionally skipped. RX audio channels use slots 3, 4, and 6; control/readback uses slots 1 and 2.

Frame timing and qualification
- A 256-count frame counter aligns strobes to slot boundaries and opens in_valid windows once captured data is stable: slot3 after cnt > 0x4D, slot4 after cnt > 0x61, slot6 after cnt > 0x89. These are pipelined to system clock to form single-shot write enables per active frame.

FIFO write discipline (system domain)
- Each channel issues at most one FIFO write per frame when: its in_valid window is active, the channel is enabled by configuration, the FIFO is not full (status snapshotted before the window), and any additional ready gating is met. This one-shot gating prevents overruns and double-writes.

Sample formatting (ac97_in_fifo)
- Each channel uses a 4-word synchronous FIFO to repack slot samples into 32-bit words by mode: 16-bit mode packs two consecutive samples (din[19:4]) per word using an internal hold register; 18-bit mode zero-extends din[19:2] per word; 20-bit mode zero-extends din[19:0] per word. Write pointer advances by half-word in 16-bit mode or by word in wider modes; read pointer advances by words. Full/empty and coarse status flags support backpressure, DMA, and interrupts. Mode==3 is reserved; verify 20-bit zero-extension width in RTL.

Readout and DMA
- Wishbone reads at 0x38 (i3), 0x3C (i4), 0x40 (i6) return data and assert a one-cycle read enable to advance the FIFO.
- DMA interface raises RX requests when per-channel thresholds are met; requests are one-clock qualified and held until ack.

Interrupts
- Per-channel interrupts assert on configured level/threshold crossings and always include full/empty

boundary flags; pulses are generated on read-when-empty and write-when-full attempts. An aggregator forms INTS and masks with INTM to drive int_o.

Control register readback (CRAC)
- Read commands are issued, then read data is captured from incoming slot2 (bits 19:4) in the subsequent frame; a done pulse indicates availability and data is exposed to software.

Suspend/resume
- If the bit clock stops, framing and valid windows halt (no new FIFO writes). On resume, a SYNC sequence restarts framing and RX flow continues with the next valid frames.

-- Codec Register Access (CRAC) Read/Write --

Provides AC’97 Codec Register Access over AC-link slots 1 (command/address) and 2 (data) to read or write codec registers. Host issues a transaction by writing RF index 4 with a 32-bit word: bit[31]=1 for read (0 for write), bits[22:16]=7-bit register address, bits[15:0]=write data (ignored for reads). This write asserts crac_we for one cycle and latches crac_out={rw, 8’h00, addr[6:0], wr_data[15:0]}. During the active command frame (crac_valid=1), slot 1 drives [19]=rw, [18:12]=addr, [11:0]=0; for writes, slot 2 drives [19:4]=data, [3:0]=0. Slot 0 tags advertise the transaction: bit[14]=1 when slot 1 (CRAC) is present; bit[13]=1 only for writes (slot 2 data present); bit[15] indicates any activity (OR of [14:6]). Write completes at the end of the command frame (crac_wr_done pulse; INTS[1] set). Read spans two frames: command in the current frame (slot 1); codec returns data in incoming slot 2 of the following frame; crac_din captures in_slt2[19:4] and crac_rd_done pulses at the start of the frame after the data (INTS[0] set). Reading RF index 4 returns {rw, 8’h00, addr[6:0], crac_din}. Do not issue a new crac_we while crac_valid is high; requests are not queued. In suspend (no bit clock), valid edges may not occur; software must serialize operations, and for reads should wait for crac_rd_done or poll INTS before reading back data; INTS is cleared on read and INTM masks completion interrupts into int_o.

-- Suspend Detection and Resume Handshake --

Suspend Detection and Resume Handshake

Clock domains and inputs
- clk: AC-link bit clock.
- wclk: system/control clock used for monitoring and software interaction.
- ps_ce: prescale/enable tick defining coarse timing for resume pulse length.
- CSR interface (rf addr 0):
- Read: bit[1] = suspended status.
- Write pulses: bit[1]=1 -> resume_req; bit[0]=1 -> ac97_rst_force (independent of suspend/resume).

Suspend detection (wclk domain)
- Edge monitor samples clk into wclk and XORs successive samples to detect bit-clock transitions.
- A timeout counter increments each wclk cycle without a detected clk edge; when it reaches 0x21 (33 wclk cycles), suspended is asserted.
- Suspended deasserts automatically when clk edges resume (counter clears on edge).

Behavior while suspended (clk domain)
- Frame counter (cnt) is held at 0xFF, freezing frame timing.
- Frame start (ld) and sync_beat (first 16 clk cycles high) are suppressed.
- Per-slot strobes and load pulses (e.g., out_le[*], so_ld) do not fire; TX/RX shifters stop updating/capturing.
- valid remains true because cnt == 0xFF; downstream logic must qualify transfers with ld/sync or !suspended to avoid unintended activity.

Resume handshake
- Software requests resume by writing CSR bit[1]=1, generating a one-cycle resume_req (clk/rf domain), which is synchronized into wclk.
- When suspended && resume_req in wclk, sync_resume is asserted to force SYNC high regardless of cnt.
- A small counter (res_cnt) increments on each ps_ce tick while sync_resume is asserted; when res_cnt == 5, resume_done is generated and sync_resume is cleared.
- SYNC output (clk domain): sync_o = sync_beat | sync_resume. During suspend, sync_beat is inactive, so SYNC is driven solely by sync_resume.
- Absolute SYNC-high duration is 5 ps_ce ticks (e.g., with ps_ce = 1 per 50 clk cycles, total ≈ 250 clk cycles).

Exit from suspend
- Upon detection of clk edges, suspended clears, cnt resumes free-running from 0x00, sync_beat returns, and per-slot timing restarts.
- Recommended software sequence: write CSR bit[1]=1 to request resume; poll CSR bit[1] until it clears (indicating bit-clock present); then restart channel/DMA activity.

CDC requirements
- Use proper synchronizers for:
- clk→wclk sampling for edge detection.
- clk→wclk transfer of resume_req.
- wclk→clk consumption of suspended and sync_resume (affects cnt hold and SYNC composition).

Constants/parameters
- Suspend timeout: 33 wclk cycles (0x21).
- cnt hold value during suspend: 0xFF.
- Resume SYNC-high length: 5 ps_ce ticks.

Reset independence
- Forced reset (ac97_rst_force) and ac97_rst_ timing are independent of suspend/resume.

-- FIFO Transaction Gating --

Purpose
- Provide capacity-safe, one-shot read/write enables to AC'97 FIFOs so exactly one transaction occurs per advertised timeslot/frame, preventing underflow/overflow.

Location and Interfaces
- Implemented in ac97_prc via ac97_fifo_ctrl; coordinated with ac97_soc valid windows and per-slot strobes.
- Inputs: valid (from ac97_soc), ch_en, crdy, full/empty/status (via full_empty snapshot), optional service request select srs and per-channel req.
- Outputs: single-cycle en_out pulses per channel (e.g., o3..o9_re) to FIFOs and Slot 0 tag builder.

Core Gating Behavior
- en_out asserts for one clock within a valid window only if all are true:
- valid window is open,
- channel enabled (ch_en),
- codec/controller ready (crdy),
- FIFO capacity is available based on a stable snapshot (!full_empty_r),

- request qualification passes (srs=0 or (srs=1 and req=1)).
- Capacity snapshot full_empty_r is captured only while valid=0; in-window decisions use this stable view to ignore mid-slot flag changes.
- One-shot edge: en_out derives from a level latch with edge detection, so at most one pulse per valid window even if conditions remain asserted.

TX (Playback) Path
- Per-channel pulses (o3..o9_re) simultaneously:
- drive ac97_out_fifo read enable for exactly one sample per frame when safe, and
- inform the Slot 0 tag builder to advertise active output slots [12:6].
- Mode handling in ac97_out_fifo:
- 16-bit: re advances half-words; empty flag is meaningful.
- 18/20-bit: re advances whole words; use status/full for gating (empty specialized flag not reliable).
- Gating must ensure the FIFO can supply data (not empty/equivalent) before asserting re; the FIFO does not self-block on empty.

RX (Capture) Path
- Write enable to ac97_in_fifo asserted once per captured input slot while not full, aligned to ac97_soc in_valid windows and out_le strobes.
- Mode handling in ac97_in_fifo:
- 16-bit: packs two samples per 32-bit word; continue issuing we per incoming sample if not full.
- 18/20-bit: one we per sample.
- Gating must ensure !full before asserting we; the FIFO does not self-block on full.

Timing and Alignment
- ac97_soc asserts valid late in the frame and provides out_le strobes; gating pulses are produced only within the valid window.
- Top-level pipelines (valid -> valid_s1 -> valid_s; in_valid -> in_valid_s1 -> in_valid_s) provide stable qualification for gating and tag capture.

Slot 0 Tag Consistency
- ac97_prc collects per-channel enables (o3_re_l..o9_re_l) to form Slot 0 bits [12:6].
- crac_valid/crac_wr drive bits [14:13]; bit [15] is the OR of [14:6].
- Only advertise a slot when a gating pulse will actually occur to maintain link consistency.

Protection and Observability
- Interrupts (ac97_int):
- int_set[1] = empty & re (underflow attempt),
- int_set[2] = full & we (overflow attempt),
- int_set[0] = level/threshold per cfg, combining full_empty and status.
- DMA (ac97_dma_if/ac97_dma_req):
- Requests gated by cfg and FIFO thresholds; level-true until dma_ack; complements gating to prompt timely service.

Reset, Enable, and Suspend
- When a FIFO’s en=0, pointers/data clear; gating suppresses re/we; capacity snapshot updates on the next valid-low period.
- Ensure valid is low at reset so fifo_ctrl latches initialize benignly.
- In suspend (cnt held at 0xFF), normal sync_beat is suppressed; gating relies on the pipelined valid to avoid spurious pulses.

Mode-Specific Notes

- 16-bit mode: use empty for TX gating; RX packs two samples per word.
- 18/20-bit modes: prefer status/full for gating; empty specialized flag is not reliable; TX/RX advance by whole words.

Assumptions and Cautions
- FIFOs do not internally block re/we on empty/full; correctness relies on gating.
- Use the snapshot-based capacity check (full_empty_r) for in-window decisions.

Net Effect
- Single, mode-aware, capacity-safe FIFO transactions per advertised timeslot, aligned to AC'97 valid windows and per-slot timing, with interrupts/DMA aiding timely service and fault visibility.

-- DMA Request/Acknowledge Flow --

Scope and timing: dma_req/dma_ack are generated per channel in ac97_dma_if, synchronous to clk and independent of AC-link frame timing.
Enable gating: Requests are possible only when cfg[6]=1 (stream/global enable) and cfg[0]=1 (DMA enable for the channel).
Inputs: status[1:0] is a coarse modulo-4 FIFO level; full_empty is a boundary flag mapped by direction (TX uses empty, RX uses full).
Threshold modes (cfg[5:4]):
- 00: request when full_empty=1 OR status!=3 (below max level)
- 01: request when full_empty=1 OR status in {0,1} (lower half)
- 10: request when full_empty=1 OR status==0 (empty-only)
- 11: request only when full_empty=1 (boundary-only)
Qualification: The selected request condition must be true for two consecutive clk cycles with dma_ack=0 before asserting dma_req; dma_ack masks new captures.
Handshake behavior: dma_req is level-style—once qualified it asserts and stays high until dma_ack=1. dma_ack immediately clears an active dma_req and prevents assertion while high.
Re-request: After dma_ack returns low, dma_req can reassert only after the condition re-qualifies for two cycles.
Reset: Asynchronous active-low reset forces dma_req=0 and clears qualification; requests resume only after cfg enables and thresholds re-qualify.
Direction intent: TX requests refill to avoid underrun (empty boundary); RX requests drain to avoid overrun (full boundary).
Concurrency: Each enabled channel can assert dma_req independently; system DMA must arbitrate and return per-channel dma_ack.
FIFO note: Coarse 2-bit status is complemented by the boundary flag to guarantee service at critical limits, including 18/20-bit modes where some empty/full flags are specialized.

-- Interrupt Generation, Masking, and Clearing --

ac97_top implements a centralized, masked, sticky interrupt mechanism. All event sources drive one-cycle pulses that are latched into a 29-bit sticky status register INTS[28:0] by ac97_rf. Sources are: Codec Register Access done pulses (crac_rd_done -> INTS[0], crac_wr_done -> INTS[1]) and per-channel ac97_int int_set[2:0] groups. Outbound channels oc0..oc5 map to INTS[2:19] (three bits per channel, contiguous per channel), and inbound channels ic0..ic2 map to INTS[20:28]. For each channel, int_set[0] is a level/threshold cause gated by cfg[0] and selected by cfg[5:4]: 00 = full_empty OR (status < 2), 01 = full_empty OR (status[1] == 0), 10 = full_empty OR (status == 0), 11 = full_empty only. int_set[1] reports an underflow attempt (empty & re) and int_set[2] reports an overflow attempt (full & we); these pulses do not depend on cfg and do not block FIFO operation.
Masking is provided by a 29-bit mask register INTM. Masking is applied only to the global interrupt

output: int_all = INTM & INTS (bitwise), and int_o asserts when any bit of int_all is 1. Masked causes still set INTS and must be cleared by software. int_o is registered in the clk domain, adding one cycle of latency, and remains asserted as a level until the corresponding INTS bits are cleared or masked. Clearing is global and read-to-clear: reading INTS at rf adr==6 returns the current INTS value and simultaneously clears all bits in the same acknowledged cycle; there is no per-bit clear. INTM is accessed at adr==5 (read/write). Asynchronous reset clears INTS and INTM to 0 and deasserts int_o; after reset, interrupts are disabled (INTM==0) until software programs the mask.
Timing and access: ac97_wb_if provides single-beat Wishbone accesses; int_set pulses are cycle-wide and reliably latched into INTS; CRAC done pulses are captured similarly. Software should read INTS only when ready to consume and clear all causes, as every read clears the entire register.

-- Error Conditions and Recovery --

- Bit-clock loss and suspend:
- Detection: AC-link bit clock stops toggling; wclk-side timeout counter reaches 0x21 (~33 wclk cycles) and CSR.suspended reads 1.
- Effects: Frame counter holds at 0xFF (SYNC/tag suppressed); link timing freezes.
- Recovery: Host writes CSR[1]=1 (resume_req) while suspended; sync_resume pulses for ~5 ps_ce periods then clears; normal frames resume.

- Reset sequencing:
- Condition: Power-on rst_n=0 or CSR[0]=1 (ac97_rst_force).
- Effects: ac97_rst_ asserted low; prescaler counts to 4 (~200 clk cycles); internal state clears.
- Recovery: After ac97_rst_ releases high, deassert then reassert per-channel en to reset FIFO pointers/outputs; reprogram modes/thresholds as needed.

- FIFO boundary violations (overflow/underflow attempts):
- Condition: Write while full or read while empty; FIFOs do not block invalid we/re.
- Detection: int_set[2]=1 on full & we; int_set[1]=1 on empty & re; INTS sticky bits capture events; optional int_set[0] level via cfg[5:4] thresholds.
- Effects: Data loss/corruption; pointer misalignment.
- Recovery: Gate we/re externally (fifo_ctrl, DMA, software); read INTS to clear; deassert channel en to synchronously clear pointers; retune DMA thresholds/masks.

- DMA threshold requests:
- Condition: FIFO occupancy crosses cfg[5:4] thresholds with cfg[6]/cfg[0] enabled; two-cycle qualification.
- Detection/Effects: dma_req asserted and held until dma_ack; may chatter if thresholds tight.
- Recovery: Assert dma_ack; adjust cfg gates/thresholds; reset immediately clears dma_req.

- Mode/format misconfiguration:
- RX FIFO: mode==3 undefined; 20-bit zero-extend path width typo (31-bit); empty flag meaningful only in 16-bit mode.
- TX FIFO: empty flag meaningful only in 16-bit mode; rely on full/status for 18/20-bit modes.
- Effects: Stale/invalid data; misleading status.
- Recovery: Program oc*/ic*[3:2] correctly (avoid RX mode==3); if corruption suspected, deassert en to reset pointers and reconfigure; use proper status signals per mode.

- CRA sequencing/protocol misuse:
- Condition: Launching new crac_we while crac_valid is active; misinterpreting read timing.
- Effects: Overlapping commands undefined; read data valid only at crac_rd_done (aligned to next frame); no codec read timeout.
- Recovery: Issue one transaction at a time; wait for crac_rd_done before consuming crac_din;

implement software timeouts; writes complete at crac_wr_done; abort via reset if necessary.

- Serial alignment/framing errors:
- Condition: so_ld/out_le misaligned to frame boundaries; shifters lack explicit reset and power up unknown.
- Effects: Wrong slot captures; X/invalid data until first proper load.
- Recovery: Rely on ac97_soc ld/sync_beat for realignment; ensure so_ld occurs once per 256 bit clocks; keep channels disabled until alignment established; valid data after next frame load.

- Wishbone bus access issues:
- Condition: Accesses outside mapped region; wb_err_o tied low; only acknowledges valid map.
- Effects: No ack on unmapped accesses; masters can hang without system decode/timeouts.
- Recovery: Provide system-level address decoding; masters implement timeouts; within mapped region, single-cycle ack is guaranteed.

- Interrupt handling and recovery:
- Detection: INTS captures CRAC done and per-channel int_set bits; INTM masks sources; INT asserts when any masked bit is set.
- Recovery: Read INTS to clear sticky status; adjust INTM to mask/unmask sources; correct underlying causes (flow control, configuration).

- CDC considerations (suspended crossing domains):
- Condition: suspended generated in wclk domain and used in clk domain without explicit resynchronization.
- Effects: Potential metastability/glitches in pathological async relationships; clk-domain cnt held at 0xFF.
- Recovery/Mitigation: Add synchronizers if modifying design; rely on global reset/stable clocking; current usage is tolerant.

REGISTERS

-- Memory Map Overview --

Wishbone slave, 32-bit word-aligned. Internal word index = wb_addr_i[6:2]; byte address = word_offset*4. Region acknowledges only when wb_addr_i[31:29]==0; each valid access produces a single-cycle ack. No bus errors.

Word offsets:
0x00 (0) CSR (R/W side-effects): Read returns {30'h0, suspended, 1'b0}. Write: bit[0] pulses ac97_rst_force; bit[1] pulses resume_req. No persistent write image beyond pulses.
0x01 (1) OCC0 (R/W): Output channel config. oc0_cfg=bits[7:0], oc1_cfg=bits[15:8], oc2_cfg=bits[23:16], oc3_cfg=bits[31:24].
0x02 (2) OCC1 (R/W): Lower 24 bits used; readback zero-extended. oc4_cfg=bits[7:0], oc5_cfg=bits[15:8].
0x03 (3) ICC (R/W): Lower 24 bits used; readback zero-extended. ic0_cfg=bits[7:0], ic1_cfg=bits[15:8], ic2_cfg=bits[23:16].
0x04 (4) CRAC (Codec Register Access): Write packs {R/W, addr, data} into crac_out and pulses crac_we to launch; Read returns {crac_r[7], 8'h0, crac_r[6:0], crac_din}. Do not issue a new write while

crac_valid is active.
0x05 (5) INTM (R/W): Interrupt mask in bits[28:0]; readback zero-extended.
0x06 (6) INTS (RO, sticky, read-to-clear): Read returns {3'h0, ints_r}; reading clears all status bits. Sources include CRAC done (rd/wr) and per-channel causes for oc0..oc5, ic0..ic2.
0x07 (7) Reserved/unused (reads may default to rf_din if implemented; writes have no effect).
0x08..0x0D (8..13) Playback write apertures (WO): o3,o4,o6,o7,o8,o9 map to AC'97 slots 3,4,6,7,8,9. Each write enqueues one 32-bit word to the corresponding playback FIFO and asserts oX_we for one cycle. Reads are undefined.
0x0E..0x10 (14..16) Capture read apertures (RO): i3,i4,i6 map to AC'97 slots 3,4,6. Each read dequeues one 32-bit word from the corresponding capture FIFO, drives wb_data_o with iX_din, and asserts iX_re for one cycle. Writes are ignored.

Data semantics at channel apertures: 32-bit transfers. Sample width/mode is selected via cfg[3:2] in oc*/ic*: 16/18/20-bit packing to/from AC-link slots (playback packs/aligs to 20-bit; capture pops two 16-bit samples per word or zero-extends 18/20-bit samples).

Interrupts: INTS is read-to-clear; masked by INTM. Interrupt output asserts if any masked status bit is set.

Handshake: rf_we/rf_re asserted for one cycle on adr<8; oX_we and iX_re are one-cycle enables aligned to the acknowledged write/read.

-- Control/Status Register (CSR) --

Control/Status Register (CSR)
- Location/width: 32-bit register at word offset 0x00 in the register-file address map via Wishbone (ac97_wb_if, wb_addr_i[5:2]==0). Single-cycle ACK; no bus errors.

Access semantics
- Read: returns status only; no write history is reflected.
• [31:2] = 0
• [1] = suspended (1 = AC-link suspended, sourced from link-monitor domain)
• [0] = 0
- Write: side effects only; no bits are stored.
• Write bit[0]=1: generate a one RF-clock pulse ac97_rst_force to start the reset sequencer (holds ac97_rst_ low, then releases high after a fixed delay ≈200 RF clocks).
• Write bit[1]=1: generate a one RF-clock pulse resume_req to initiate AC'97 resume (causes a short SYNC burst; duration controlled internally).
• Writes of 0 have no effect; all other bits are ignored.

Timing/clock domains
- Reads/writes occur in the register-file/Wishbone clock domain.
- suspended is produced in the AC-link monitor domain and forwarded for readback.
- ac97_rst_force and resume_req are single-cycle pulses in the RF/Wishbone domain; downstream logic handles any required clock-domain crossings and sequencing.

Software usage
- Codec reset: write 0x00000001 to CSR; ac97_rst_ will be asserted low then released after the sequenced delay.
- Link resume: when suspended=1, write 0x00000002 to CSR; the controller emits the resume SYNC burst; suspended clears when bit-clock activity resumes.
- Status polling: read CSR and check bit[1] (suspended). Previously written values are not readable.
- Issue one action per write; avoid reissuing resume while a resume is in progress. Multiple identical

writes are benign but unnecessary.

Miscellaneous
- Reserved bits [31:2] read as 0; writes to them have no effect.
- CSR accesses do not directly affect interrupts; suspended is not an interrupt source. Use INTM/INTS for interrupt handling.

-- Output Channel Configuration (OCC0/OCC1) --

OCC0/OCC1 define per-channel output configuration bytes for AC'97 playback slots 3, 4, 6, 7, 8, and 9. Register file mapping: RF[1]=OCC0 contains oc0_cfg[7:0], oc1_cfg[15:8], oc2_cfg[23:16], oc3_cfg[31:24]; RF[2]=OCC1 contains oc4_cfg[7:0], oc5_cfg[15:8] (upper 16 bits read as 0). Wishbone addresses: OCC0 at adr=1, OCC1 at adr=2. Channel-to-slot mapping: oc0_cfg→slot 3 (o3), oc1_cfg→slot 4 (o4), oc2_cfg→slot 6 (o6), oc3_cfg→slot 7 (o7), oc4_cfg→slot 8 (o8), oc5_cfg→slot 9 (o9). Each ocX_cfg[7:0] controls data format and DMA/interrupt behavior: bits[3:2] Mode select the TX FIFO sample width/format and 20-bit link alignment (00=16-bit, two samples per 32-bit word; 01=18-bit, one per word, left-shifted by 2 to 20b; 10=20-bit, one per word, passthrough to 20b; 11 reserved). Bits[5:4] Threshold select the FIFO occupancy threshold used by the level/threshold interrupt and DMA request (typical: 00/01=below-half, 10=empty-only, 11=boundary-only using full/empty flag). Bit[6] DMA/stream enable must be 1 to allow DMA requests; when 0, DMA requests are suppressed. Bit[0] Level/threshold enable gates the level/threshold interrupt source and participates in DMA request gating. Other bits are reserved. Reset: OCC0 and OCC1 reset to 0; playback channels start disabled for DMA/level interrupts with Mode=00 (16-bit). Readback: OCC0 returns all 32 bits; OCC1 reads zero in upper 16 bits (only oc4_cfg/oc5_cfg are used). Effects: Mode directly drives o3_mode..o9_mode, the ac97_out_fifo read stepping (16-bit mode reads half-words; 18/20-bit read full words), and 20-bit slot formatting on SDATA_OUT. Threshold and enable bits configure when level-style interrupts assert (ac97_int) and when ac97_dma_if issues DMA requests per channel; DMA requests require ocX_cfg[6]=1 and ocX_cfg[0]=1, use the selected threshold, are qualified for one cycle, and held until dma_ack. Slot 0 tag generation advertises output slots only when fifo_ctrl grants a transfer; some integrations use a bit in ocX_cfg as a channel enable (not explicitly defined here). Software should mask upper bits on OCC1 read/modify/write.

-- Input Channel Configuration (ICC) --

ICC (Input Channel Configuration) is a 24-bit register at RF address index 3. It contains three 8-bit configuration bytes for the three input capture channels:
- [7:0] IC0 → channel i3 (captures SDATA_IN slot 3)
- [15:8] IC1 → channel i4 (captures SDATA_IN slot 4)
- [23:16] IC2 → channel i6 (captures SDATA_IN slot 6)
Reset: 0x000000. Writes take effect immediately.

Per-channel ICx[7:0] fields:
- [6] EN: Channel enable. 1 enables capture and FIFO logic; 0 synchronously holds/resets the input FIFO (pointers/data cleared) and blocks samples. Required together with SRV_EN for DMA requests.
- [5:4] THR[1:0]: Shared DMA and level-interrupt threshold select based on FIFO status:
- 00: request/interrupt when (full_empty == 1) OR (status < 3)
- 01: request/interrupt when (full_empty == 1) OR (status[1] == 0)
- 10: request/interrupt when (full_empty == 1) OR (status == 0)
- 11: request/interrupt only when (full_empty == 1)
Notes: full_empty is the FIFO boundary flag; status[1:0] is a coarse 2-bit fill level.
- [3:2] MODE[1:0]: Sample width/packing in ac97_in_fifo:
- 00: 16-bit samples; packs two samples per 32-bit word.
- 01: 18-bit samples; one sample per 32-bit word, zero-extended.

- 10: 20-bit samples; one sample per 32-bit word, zero-extended.
- 11: reserved.
- [0] SRV_EN: Service/IRQ enable. Gates the level/threshold interrupt cause and is required (with EN) to allow DMA requests.
- [7], [1]: reserved; write 0.

Behavior and integration:
- MODE[1:0] (per IC0/IC1/IC2) drives how 20-bit slot words from ac97_sin are reformatted and how FIFO pointers advance:
- 16-bit mode: packs two samples per 32-bit word; write pointer steps by half-sample; FIFO empty semantics are meaningful.
- 18/20-bit modes: one sample per 32-bit word (zero-extended); write pointer steps by words; empty flag semantics are implementation-specific.
- EN=0 clears the input FIFO each clock; transitioning 1→0 immediately drops accumulated samples; 0→1 restarts accumulation.
- DMA requests per channel assert when EN=1 and SRV_EN=1 and the THR condition qualifies; requests hold until acknowledged by DMA.
- Interrupts: per-channel int_set[2:0]:
- int_set[0]: level/threshold per THR and gated by SRV_EN.
- int_set[1]: read-while-empty.
- int_set[2]: write-while-full.
Reported in INTS: i3 → [20..22], i4 → [23..25], i6 → [26..28]; globally masked by INTM; INTS is read-to-clear.

Notes:
- THR[1:0] is consumed by both ac97_dma_if (DMA requester) and ac97_int (level interrupt).
- Treat THR modes as progressively stricter level qualification.
- Keep reserved bits 0 for forward compatibility.

-- Codec Register Access (CRAC) --

Codec Register Access (CRAC) implements AC'97 control register read/write over the AC-link with bus-side programming, slot formatting, frame-edge timing, and completion/interrupt signaling. Host interface: Wishbone -> ac97_wb_if -> ac97_rf.CRAC (adr=4); a write launches a transaction. CRAC write fields: [31]=1 read/0 write, [22:16]=7-bit codec register address, [15:0]=write data; CRAC readback returns {crac_r[7], 0x00, crac_r[6:0], crac_din} where crac_din is the last captured 16-bit read data. Slot formatting: out_slt1[19]=R/W, [18:12]=address, [11:0]=0; out_slt2[19:4]=write data, [3:0]=0. Slot 0 tags (ac97_prc): bit14=crac_valid (Slot1 present), bit13=crac_valid & crac_wr (Slot2 present on write), bit15=OR of bits [14:6] (Valid Frame). Timing: valid from ac97_soc defines the frame window; write drives the current frame while crac_valid and completes at valid falling edge (crac_wr_done pulse); read drives the command in the current frame, data returns in the next frame's Slot 2 (captured by ac97_sin), completion at the next valid rising edge (crac_rd_done pulse and crac_din load). crac_valid is high only while a transaction is active; no queuing—do not assert crac_we until the corresponding done strobe; reset clears state; avoid use during suspend/resume. Return path: codec SDATA_IN -> ac97_sin -> ac97_cra.in_slt2 -> crac_din -> ac97_rf readback. Interrupts: INTS[0]=read done, INTS[1]=write done; INTM masks these sources; int_o asserts if any enabled INTS bit is set; reading INTS clears all bits. Constraints: address 7 bits, data 16 bits; lower Slot1/2 bits zero-padded; ac97_wb_if acknowledges in one cycle; after a write, the CRAC readback data field reflects the last read, not the write. Typical sequences: Write—program CRAC (R/W=0, addr, data) and wait for INTS[1] or poll; Read—program CRAC (R/W=1, addr), wait for INTS[0], then read CRAC to obtain crac_din.

-- Interrupt Mask (INTM) --

Interrupt Mask (INTM) is a 29-bit read/write mask register that enables individual interrupt sources to drive the global interrupt output. It resides in the register file at address index 5 (Wishbone access). Reset value is 0x00000000 (all interrupts masked).
- Width/readback: bits [28:0] are implemented; reads return {3'b000, intm_r}. Bits [31:29] read as 0.
- Write: rf_din[28:0] are stored; upper write bits are ignored.
- Function: int_o asserts level-true while any bit in (INTM & INTS) is 1. INTM only gates sources; it does not clear status.
- Status interaction: INTS is a sticky, read-to-clear status register. If a level/threshold condition persists, INTS may re-set after a clear when the corresponding mask bit remains enabled.
- Bit assignments:
- [0] CRAC read done
- [1] CRAC write done
- Per-channel groups (each group = {threshold/level, underflow attempt (empty & read), overflow attempt (full & write)}):
- oc0: [2:4], oc1: [5:7], oc2: [8:10], oc3: [11:13], oc4: [14:16], oc5: [17:19]
- ic0: [20:22], ic1: [23:25], ic2: [26:28]
- Access side effects: reads have no side effects; single-cycle capture on writes.
- Software notes: program INTM after configuring channel thresholds; enable CRAC bits for command completion interrupts; clear INTS by reading it when servicing. Default mask=0 disables all interrupts until configured.

-- Interrupt Status (INTS) --

Interrupt Status (INTS) is a 29-bit, read-to-clear, sticky status register in ac97_rf (address index 6) that latches one-cycle interrupt-event pulses from the codec control path and per-channel FIFO/threshold causes.

Function:
- Set: Captures single-cycle pulses in the clk domain and holds them until software reads INTS.
- Clear: Any read of address index 6 atomically clears all 29 bits; read data returns {3'b000, INTS}.
- Reset: Asynchronous reset initializes INTS to 0.
- Masking/Global IRQ: INTS is unmasked; INTM (29-bit mask) gates INTS to form int_all, whose OR-reduction drives int_o. Masking does not modify INTS contents.

Bit assignments (LSB first):
- [0] CRAC read done (ac97_cra read completes; data captured from Slot 2).
- [1] CRAC write done (ac97_cra write command completes).
- [2:4] oc0_int_set[2:0] – output channel 0 (o3).
- [5:7] oc1_int_set[2:0] – output channel 1 (o4).
- [8:10] oc2_int_set[2:0] – output channel 2 (o6).
- [11:13] oc3_int_set[2:0] – output channel 3 (o7).
- [14:16] oc4_int_set[2:0] – output channel 4 (o8).
- [17:19] oc5_int_set[2:0] – output channel 5 (o9).
- [20:22] ic0_int_set[2:0] – input channel 0 (i3).
- [23:25] ic1_int_set[2:0] – input channel 1 (i4).
- [26:28] ic2_int_set[2:0] – input channel 2 (i6).

Per-channel cause encoding (int_set[2:0]):
- bit 0: Level/threshold interrupt (enabled by cfg[0]; threshold selected by cfg[5:4]).
- bit 1: Underflow attempt (read-enable when FIFO empty).
- bit 2: Overflow attempt (write-enable when FIFO full).

Notes:
- Multiple events may be latched before a read; all bits clear together on read.
- All contributing pulses are synchronous to clk; INTS updates synchronously and is asynchronously reset low.

-- Reset Values and Access Types --

Global reset and domains:
- rst_n is asynchronous active-low. Some submodules have no explicit reset and may power up unknown until first valid activity.

ac97_rst (codec reset generator):
- Reset value/polarity: ac97_rst_ held low while rst_n=0 or rst_force=1.
- Release: ac97_rst_ goes high ~200 clk cycles (4 prescaled ticks) after rst_n=1 and rst_force=0.
- Access type: rst_force is a 1-cycle pulse generated by CSR write (see CSR below).

ac97_soc (framing/timing):
- Reset value: 8-bit frame counter asynchronously preset to 0xFF on reset.
- Suspend behavior: while suspended, counter forced to 0xFF every clk.
- Access: internal only.

ac97_wb_if (Wishbone interface):
- Reset: none; edge detectors/ack flops power up unknown. System must ensure clean first access.
- Access map (word offsets):
- 0..7: Register file window (writes assert rf_we; reads assert rf_re).
- 8..13 (0x20..0x34): o3..o9 write-only channel apertures (writes assert oX_we for 1 cycle; data from wb_data_i).
- 14..16 (0x38..0x40): i3/i4/i6 read-only channel apertures (reads assert iX_re for 1 cycle; returns iX_din).
- Handshake: single-cycle wb_ack_o per accepted transfer; wb_err_o=0.

ac97_rf (register file):
- Reset: async active-low clears all storage to 0.
- Registers and access types:
- CSR (adr=0): RO status readback {30'h0, suspended, 1'b0}. Writes do not store; they produce 1-cycle pulses only:
• ac97_rst_force <= rf_din[0]
• resume_req <= rf_din[1]
- OCC0 (adr=1): 32-bit RW, reset 0. oc0_cfg..oc3_cfg derived from bytes.
- OCC1 (adr=2): RW lower 24 bits, reset 0; readback zero-extended {16'h0, occ1_r}. oc4_cfg/oc5_cfg from lower 16 bits.
- ICC (adr=3): RW lower 24 bits, reset 0; readback {8'h0, icc_r}. ic0_cfg..ic2_cfg from bytes.
- CRAC (adr=4): RW with side effects; reset crac_r=0; crac_dout_r undefined until first write. Readback {crac_r[7], 8'h0, crac_r[6:0], crac_din}.
- INTM (adr=5): 29-bit RW interrupt mask, reset 0; readback {3'h0, intm_r}.
- INTS (adr=6): 29-bit sticky RO status, reset 0; readback {3'h0, ints_r}. Read-to-clear: a read at adr==6 with rf_re clears all bits.
- Interrupt output: int_o = OR of masked INTS (intm_r & ints_r); reset low; level-active until INTS is cleared or masked.

ac97_cra (codec register access):

- Reset: active-low clears crac_wr=0, crac_rd=0, crac_din=0, internal edge/state flags to 0.
- Access: launched by 1-cycle crac_we (from CRAC write). Provides 1-cycle crac_wr_done/crac_rd_done pulses; crac_valid qualifies Slot1/2 data.

ac97_int (interrupt cause aggregator):
- Reset: active-low clears int_set[2:0] to 0.
- Access: int_set are cycle-by-cycle causes (not sticky). Captured into INTS by ac97_rf; INTS is read-to-clear.

ac97_dma_if / ac97_dma_req (DMA request):
- Reset: active-low clears dma_req=0 immediately.
- Access: level-style dma_req gated by cfg[6], cfg[0], and thresholds cfg[5:4]; asserts after 2-cycle qualification, remains high until dma_ack.

ac97_sout / ac97_sin (serial shifters):
- Reset: none; internal registers power up unknown. sout tends to zeros after many shifts; sin relies on framing for valid slots.
- Access: internal only.

ac97_prc / ac97_fifo_ctrl (tag builder and FIFO control):
- Reset: none; latches stabilize when valid=0.
- Access: internal only.

ac97_out_fifo (output FIFO):
- Reset: en=0 acts as synchronous reset (wp<=0, rp<=0, dout<=0). No async reset.
- Access: we writes 32-bit words; re outputs 20-bit samples per mode. No internal overflow/underflow protection; external gating required.

ac97_in_fifo (input FIFO):
- Reset: en=0 acts as synchronous reset (wp<=0, rp<=0). No async reset.
- Access: we writes repacked 32-bit words per mode; re reads words. empty flag only meaningful in 16-bit mode.

Summary of access types:
- RO: CSR status (readback only), INTS sticky status (read-to-clear), i3/i4/i6 channel read apertures.
- WO: o3..o9 channel write apertures; CSR writes produce pulses (no storage).
- RW: OCC0, OCC1, ICC, INTM; CRAC is RW with side effects (write launches CRA; read returns live crac_din).

Operational cautions:
- Modules without explicit reset may output X until first valid activity.
- CRAC path: crac_dout_r undefined until first CRAC write; ac97_cra resets crac_din to 0.
- ac97_rst release is delayed; wait ~200 clk cycles after rst_n high and rst_force low before starting link transactions.
- INTS is sticky and masked by INTM; int_o remains asserted until INTS is cleared or masked.
- FIFOs rely on external gating; underflow/overflow attempts are detected by int_set[1]/[2].
- DMA requires cfg[6] and cfg[0] set; uses 2-cycle qualification.
- During suspend, ac97_soc counter remains 0xFF; downstream logic should also consider ld/sync/suspended for validity.

-- Bit Field Definitions --

- AC'97 Slot 0 Tag out_slt0[15:0]
- [15] Valid Frame (derived: OR of [14:6], read-only)
- [14] Slot 1 present (CRAC address phase)
- [13] Slot 2 present (CRAC write data; 1 when issuing a write)
- [12] Slot 3 present (TX/playback channel o3)
- [11] Slot 4 present (TX/playback channel o4)
- [10] 0 (Slot 5 unused)
- [9] Slot 6 present (TX/playback channel o6)
- [8] Slot 7 present (TX/playback channel o7)
- [7] Slot 8 present (TX/playback channel o8)
- [6] Slot 9 present (TX/playback channel o9)
- [5:0] Reserved (write 0)

- AC'97 Control Slots
- out_slt1[19:0]
- [19] R/W (1=read, 0=write)
- [18:12] Codec register address [6:0]
- [11:0] 0
- out_slt2[19:0]
- [19:4] Write data [15:0] (valid when R/W=0)
- [3:0] 0
- in_slt2[19:4]
- [19:4] Read data [15:0] (captured on CRAC reads)

- CRAC command word crac_out[31:0]
- [31] R/W (1=read, 0=write)
- [22:16] Codec register address [6:0]
- [15:0] Write data (used when R/W=0)
- Readback (on CRAC register read)
- [31] Latched R/W
- [30:23] 0
- [22:16] Latched address [6:0]
- [15:0] Captured read data (from in_slt2[19:4])

- Register File Bit Fields
- CSR (adr=0)
- Read:
- [1] suspended (link suspended status)
- [0] 0; [31:2] 0
- Write (one-cycle pulses):
- [0] ac97_rst_force
- [1] resume_req
- OCC0 (adr=1)
- [7:0] oc0_cfg
- [15:8] oc1_cfg
- [23:16] oc2_cfg
- [31:24] oc3_cfg
- OCC1 (adr=2)
- [7:0] oc4_cfg
- [15:8] oc5_cfg
- [31:16] 0 (read as 0)
- ICC (adr=3)

- [7:0] ic0_cfg
- [15:8] ic1_cfg
- [23:16] ic2_cfg
- [31:24] 0 (read as 0)
- CRAC (adr=4)
- Fields per CRAC command word above
- INTM (adr=5)
- [28:0] Interrupt mask bits (1:1 with INTS)
- [31:29] 0
- INTS (adr=6) (sticky, read-to-clear)
- [0] crac_rd_done
- [1] crac_wr_done
- [4:2] oc0 int_set[2:0]
- [7:5] oc1 int_set[2:0]
- [10:8] oc2 int_set[2:0]
- [13:11] oc3 int_set[2:0]
- [16:14] oc4 int_set[2:0]
- [19:17] oc5 int_set[2:0]
- [22:20] ic0 int_set[2:0]
- [25:23] ic1 int_set[2:0]
- [28:26] ic2 int_set[2:0]
- [31:29] 0
- Per-channel int_set[2:0]
- [0] Threshold/level interrupt (gated by cfg[0]; threshold via cfg[5:4] vs status[1:0] with boundary override)
- [1] Underflow attempt (read while empty)
- [2] Overflow attempt (write while full)

- Per-channel configuration byte (ocN_cfg, icN_cfg)
- [6] DMA request enable
- [5:4] Threshold select (for DMA/interrupts; uses status[1:0] with boundary override)
- 00: Boundary OR low-level threshold
- 01: Boundary OR lower-half threshold
- 10: Boundary OR empty-only threshold
- 11: Boundary-only
- [3:2] Sample width/mode
- 00: 16-bit
- 01: 18-bit
- 10: 20-bit
- 11: Reserved
- [0] Channel/IRQ/DMA enable (master enable)
- [7], [1] Reserved (write 0)

- FIFO mode field (shared convention)
- mode[1:0]
- 00: 16-bit (pack 2 samples per 32-bit word)
- 01: 18-bit (1 sample per 32-bit word)
- 10: 20-bit (1 sample per 32-bit word; zero-extend upper bits when packing)
- 11: Reserved

- Interrupt generation inputs (ac97_int)
- Threshold compare: cfg[5:4] vs status[1:0] (with boundary full_empty)

- Level gating: cfg[0]
- Boundary full_empty: TX uses empty boundary; RX uses full boundary

- DMA request generation (ac97_dma_req)
- Enables: cfg[6] AND cfg[0]
- Threshold select: cfg[5:4] vs status[1:0] (with boundary full_empty)

- Reserved/behavioral notes
- All reserved bits read 0; software must write 0
- INTS is sticky, read-to-clear; INTM masks by AND; global interrupt is OR of masked set bits

CORE CONFIGURATION

-- Per■Channel Sample Mode Selection (16/18/20■bit) --

Defines per-channel sample width (16/18/20-bit) selection for AC’97 transmit (playback) and receive (capture) paths, controlling how 32-bit system words are packed to/from 20-bit AC-link slots.

Programming interface
- Valid modes: 00=16-bit, 01=18-bit, 10=20-bit; 11 reserved/unsupported.
- Transmit channels: o3,o4,o6,o7,o8,o9.
- OCC0 (RF adr 1): oc0_cfg..oc3_cfg bytes map to o3,o4,o6,o7.
- OCC1 (RF adr 2): oc4_cfg..oc5_cfg bytes map to o8,o9.
- Mode field: ocX_cfg[3:2] -> o3/o4/o6/o7/o8/o9_mode.
- Receive channels: i3,i4,i6.
- ICC (RF adr 3): ic0_cfg..ic2_cfg bytes.
- Mode field: icX_cfg[3:2] -> i3/i4/i6_mode.

Transmit data path (ac97_out_fifo -> AC-link)
- System writes are 32-bit; AC-link slots are always 20-bit.
- 16-bit mode (00): two 16-bit samples per 32-bit word; each sample left-aligned and zero-padded to 20 bits: {sample[15:0], 4'b0}. Read pointer advances by half-word.
- 18-bit mode (01): one sample per 32-bit word using din[17:0]; left-align to 20 bits: {din[17:0], 2'b0}. Read pointer advances by word.
- 20-bit mode (10): one sample per 32-bit word using din[19:0]; passes through as 20 bits. Read pointer advances by word.
- Flags: empty is meaningful only in 16-bit mode; use full/status for flow control in 18/20-bit modes.

Receive data path (AC-link -> ac97_in_fifo -> system)
- AC-link always delivers 20-bit slot words.
- 16-bit mode (0): take din[19:4] (upper 16 bits); pack two consecutive samples per 32-bit word: {current[15:0], previous[15:0]}. Write pointer advances by half-sample for packing.
- 18-bit mode (1): use din[19:2] (18 MSBs); zero-extend to 32 bits. Write pointer advances by word.
- 20-bit mode (2): use din[19:0]; zero-extend to 32 bits. Write pointer advances by word.
- Flags: empty reflects "no complete packed word" only in 16-bit mode.

Operational guidance

- Do not program mode=11; behavior is undefined.
- Changing a channel’s mode alters FIFO packing and pointer increments. Disable/flush the channel (ensure FIFO en=0) before changing mode to avoid corruption; FIFOs clear when en=0.
- AC-link timing and per-slot enables (ac97_prc/ac97_soc) are unchanged by mode; mode only affects data width/padding/truncation.
- Software should adjust DMA/interrupt thresholds acknowledging that empty semantics differ by mode.

-- Channel Enable and Request Gating --

- Purpose: Ensure each channel issues exactly one, well-timed enable pulse per frame only when the slot window is valid, the channel is enabled, the interface is ready, FIFO capacity/state allows it, and (optionally) a per-channel request is present.
- Scope: Applies to TX channels (slots 3/4/6/7/8/9) using valid windows and RX channels (slots 3/4/6) using in_valid windows.
- Inputs: valid/in_valid (window open), ch_en (channel enable), crdy (codec/controller ready), full_empty_r (latched FIFO can-serve/can-accept), req (per-channel request), srs (per-channel config: request required).
- Optional request gating: If srs=1, req must be asserted to permit the transfer; if srs=0, req is ignored and gating relies only on valid, ch_en, crdy, and full_empty_r.
- FIFO readiness snapshot: full_empty is sampled only while the window is closed (valid/in_valid=0) then held as full_empty_r during the entire slot, preventing mid-slot decision changes.
- Enable pulse generation: On the first cycle of a valid/in_valid window where all conditions pass, en_out asserts for exactly one cycle (edge-detected on the window), guaranteeing one pulse per frame even if inputs remain true.
- TX vs RX requests: For TX, req corresponds to per-channel write strobes from the bus side (e.g., oX_we); for RX, req corresponds to per-channel read strobes (e.g., iX_re). These are honored only when srs=1.
- Channel disable behavior: When ch_en=0, gating is blocked and associated FIFOs are synchronously cleared (pointers/outputs reset), ensuring clean re-enable with no stale data.
- Timing and CDC: valid/in_valid are pipelined into the system clock domain before qualification with ch_en/req/status, avoiding metastability and ensuring deterministic gating.
- Outputs and tagging: The one-cycle en_out pulses drive oX_re_l/iX_re_l strobes used to transact FIFO data and to set Slot 0 tag bits; the frame-valid tag bit reflects the presence of any latched activity.
- Safety and observability: Gating prevents underflow/overflow by requiring full_empty_r. Interrupts flag attempted violations (read-empty/write-full) and separate threshold-based service needs; DMA request logic is independent and complements gating with buffer-level servicing.
- Mode independence: Sample width/mode settings affect data formatting/pointer stepping but do not alter the gating criteria described above.

-- Interrupt Thresholds and Enables --

Each audio I/O channel supports a level/threshold interrupt source plus two attempt-event sources. The level/threshold source int_set[0] is controlled by cfg[5:4] (threshold mode) and cfg[0] (enable). Modes (evaluated using status[1:0] and the direction-appropriate full_empty boundary flag) are: 00 = trigger when full_empty is asserted OR status is below the maximum level (loosest), 01 = trigger when full_empty is asserted OR status indicates below half (status[1] == 0), 10 = trigger when full_empty is asserted OR status is empty-only (status == 0; strictest), 11 = trigger only on full_empty (boundary-only). full_empty is always an immediate override in all modes (e.g., TX: empty; RX: full). cfg[0] enables the level/threshold interrupt for that channel; when cfg[0] = 0, int_set[0] is disabled regardless of status/full_empty. Attempt-event sources int_set[1] (underflow attempt: empty & re) and int_set[2] (overflow attempt: full & we) are not gated by cfg[0]. Delivery of any source to the external interrupt pin is further controlled by the global mask register INTM (per-source bits must be set to

propagate), with sticky latching in INTS. After reset, INTM defaults to 0 (all sources masked); level/threshold interrupts require both cfg[0] = 1 and the corresponding INTM mask bit = 1 to reach int_o. status[1:0] is a coarse occupancy measure; rely on it and full_empty for thresholds across sample-width modes.

-- DMA Threshold Modes and Enables --

DMA requests are generated per channel by ac97_dma_if using a 7-bit configuration field cfg[6:0]. Two enables must be set for any request to occur: cfg[6] = 1 and cfg[0] = 1. When enabled, the request condition is determined by cfg[5:4] (threshold mode) applied to a 2-bit status level and a boundary flag full_empty.

Inputs and semantics:
- status[1:0]: coarse level indicator in the range 0..3 derived from FIFO pointer relations (word-based, not sample-based). Lower values indicate closer to empty (TX) or further from full (RX) depending on direction-specific wiring.
- full_empty: boundary-critical flag indicating an immediate risk (e.g., empty for TX channels, full for RX channels), as wired by each channel.

Threshold modes (cfg[5:4]):
- 00: Request if full_empty is asserted OR status != 3.
- 01: Request if full_empty is asserted OR status[1] == 0 (status in {0,1}).
- 10: Request if full_empty is asserted OR status == 0.
- 11: Request only when full_empty is asserted.

Qualification and handshake:
- The selected condition must be stable for two consecutive clk cycles before dma_req asserts (internal 1-cycle prequalification, assertion on the second cycle).
- Once asserted, dma_req remains high until dma_ack is received; dma_ack clears the request.
- On reset, dma_req is forced low. After re-enabling (cfg[6] and cfg[0] set), the condition must re-qualify before a new request is raised.
- Changing cfg on-the-fly immediately gates/ungates requests; when re-enabled, qualification restarts.

Direction/wiring notes:
- The same mechanism serves TX and RX channels; exact behavior depends on how each channel wires status and full_empty from its FIFO (TX typically reflects space/underrun risk; RX reflects occupancy/overrun risk).

Register context:
- The per-channel cfg bits reside in OCC0/OCC1 for TX (o3..o9) and ICC for RX (i3/i4/i6). Bits [6], [5:4], and [0] control DMA enables/thresholds; bits [3:2] select sample width and are unrelated to DMA thresholds.

Scope note:
- ac97_int uses a similar threshold concept for interrupts but with different comparisons; the mapping above is authoritative for DMA requests.

-- Slot Usage and Reserved Slots Policy --

The interface services only AC’97 Slots 1, 2 (conditional), 3, 4, 6, 7, 8, and 9. Slots 5, 10, 11, and 12 are permanently reserved. Reserved-slot policy: they are never advertised in the Slot 0 tag, are always driven as zero on SDATA_OUT every frame, and are never latched or used on SDATA_IN (Slot 5 is explicitly skipped in the input capture sequence). Slot 0 tag policy: bit[15] (Valid Frame) equals the OR

of presence bits [14:6]; presence bits are asserted only for active slots (1, 2 when writing, 3, 4, 6, 7, 8, 9). Bits for reserved slots are forced low: bit[10] (Slot 5) and bits [5:0] (Slots 10–15) are always 0. SDATA_OUT behavior: the transmitter loads the tag plus data for enabled slots; Slots 5, 10, 11, and 12 are hard-zeroed each frame. Slot 2 is driven only during CRAC write cycles; during reads, only Slot 1 is advertised and driven. SDATA_IN behavior: the receiver captures Slot 0 and Slots 1, 2, 3, 4, and 6; Slot 5 is not captured (the capture strobe for that position maps to Slot 6), and all reserved slots are ignored. CRAC readback uses incoming Slot 2 data; no other reserved slots participate in protocol or data paths. Only the supported slots generate internal activity (e.g., FIFOs/DMA/interrupts); reserved slots are inert by design.

-- Suspend/Resume Control Configuration --

Suspend/Resume Control Configuration
- Clock domains
- System (control) domain and AC’97 bit-clock domain are used. The bit clock is sampled into the system domain for suspend detection.

- Suspend detection (fixed threshold)
- Condition: If no AC’97 bit-clock edge is observed for 33 system clocks, the core enters suspended state.
- Mechanism: A system-domain edge detector drives a timeout counter; timeout at 0x21 (33) sets suspended=1.

- Suspended-state behavior
- Frame/timing: In the bit-clock domain, the frame counter is held at 0xFF; no frame loads and sync_beat is suppressed.
- SYNC output: Remains low unless a resume sequence is active.
- Data-valid caveat: Internal valid (cnt > 0x39) may read true while cnt=0xFF; do not use valid alone for transfers during suspend.

- Control/status registers (CSR at register-file address 0)
- Read: bit[1] = suspended (1 = suspended), bit[0] = 0, upper bits = 0.
- Write pulses (one-cycle each): bit[0]=1 -> ac97_rst_force (forces AC’97 reset low for a timed interval); bit[1]=1 -> resume_req (requests resume; honored only while suspended=1). Writes are not sticky and are not reflected in readback.

- Resume request and SYNC behavior
- Eligibility: resume_req is effective only when suspended=1; otherwise it is ignored. Software may re-issue if necessary.
- SYNC control: sync_o = sync_beat | sync_resume. During resume, sync_resume forces SYNC high regardless of frame timing.
- Duration: sync_resume stays high for 5 ps_ce pulses, then clears automatically. With the default prescaler (ps_ce every 50 system clocks), SYNC-high lasts ~250 system clocks (+up to 1 cycle to clear).
- Exit from suspend: After resume signaling completes, the link clears suspended when AC’97 bit-clock edges resume.

- Reset interaction (ac97_rst)
- Forcing reset: ac97_rst_force asserts ac97_rst_ low and holds it until 4 ps_ce pulses elapse (~200 system clocks by default), then releases high.
- Shared timing source: The same prescaler generates ps_ce used for both reset duration and resume SYNC-high timing; changing the prescaler affects both.
- Post-reset: If the AC’97 bit clock is absent, the core remains or re-enters suspended state.

- Configuration knobs and fixed values
- Suspend detect threshold: fixed at 33 system clocks without a bit-clock edge.
- Resume SYNC-high length: fixed at 5 ps_ce pulses; ps_ce default period is 50 system clocks (adjustable via the reset prescaler), thus resume duration scales with prescaler configuration.
- CSR access: write-only pulse controls on bits [1:0]; status via read bit[1]. No sticky control fields.

- CDC and usage considerations
- CDC: suspended is generated in the system domain and used to gate the bit-clock-domain frame counter without explicit resynchronization; limit cross-domain uses or add synchronizers for new consumers.
- Software guidance: Poll CSR bit[1] for suspend detection; issue resume_req (CSR bit[1]=1) only while suspended; wait until bit[1] clears to confirm exit; if recovery fails, consider ac97_rst_force (CSR bit[0]=1) and verify clock presence.
- Traffic gating: Avoid initiating DMA/interrupt-driven transfers while suspended; qualify transfers with additional conditions (e.g., not-suspended and sync/frame qualifiers).

-- Reset Sequencer Parameters --

- Output: ac97_rst_ (active-low codec reset)
- Clock domain: clk
- Reset inputs: rst (asynchronous, active-low); rst_force (synchronous hold/reset)
- Prescaler: ps_cnt[5:0] counts 0..49; ps_ce pulses 1 clk at 49 (period = 50 clk cycles)
- Timeout counter: cnt[2:0] increments on ps_ce while cnt < 4; saturates at 4
- Release threshold: cnt == 4; ac97_rst_ deasserts high on next clk edge
- Deassertion delay: 4 × 50 = 200 clk cycles (+1 clk edge for registered transition); T_release ≈ 200 / f_clk seconds
- Runtime programmability: none (prescale=50 cycles, ticks=4 fixed)
- CSR linkage: write ac97_rf (adr=0) with rf_din[0]=1 generates 1-cycle rst_force; multi-cycle rst_force holds reset
- Post-release behavior: ps_cnt free-runs; cnt stays at 4 until new rst/rst_force
- Scope: ac97_rst_ only drives external codec; internal modules use independent rst signals

-- Bus Decode Region and Acknowledge Behavior --

Bus decode and acknowledge behavior are implemented in ac97_wb_if as follows:

Decode region qualification:
- The slave responds only when wb_addr_i[31:29] == 3'b000. Any access with [31:29] != 0 is ignored and not acknowledged.
- Addressing is word-aligned; the effective address index is adr = wb_addr_i[5:2]. The decoded map within the qualified region is:
• adr 0..7 (0x00..0x1C): Register file window (ac97_rf). Writes assert rf_we; reads assert rf_re; wb_data_o returns rf_din. Interrupt status is read-to-clear but does not alter acknowledge timing.
• adr 8..13 (0x20..0x34): Write-only playback channel apertures. Writes assert the corresponding o3_we..o9_we and place write data on dout; reads default to rf_din.
• adr 14..16 (0x38..0x40): Read-only capture channel apertures. Reads assert i3_re/i4_re/i6_re and wb_data_o sources i3_din/i4_din/i6_din; writes are ignored.
• Any other address within the qualified region on reads muxes rf_din by default. Writes are only acted upon for the ranges above.

Acknowledge behavior:

- Single-beat acknowledge per accepted transfer. wb_ack_o pulses high for exactly one clock when wb_cyc_i & wb_stb_i are asserted and a new read or write is edge-detected: wb_ack_o = (re | we) & wb_cyc_i & wb_stb_i & ~wb_ack_o.
- Read/write strobes (re/we) are generated as one-clock pulses via internal edge detectors, ensuring one ack per request with no inserted wait states, stalls, or retries.
- wb_err_o is tied low; the core does not generate bus errors.
- Accesses outside the qualified region are not acknowledged. Writes to addresses within the region but outside the defined write apertures are not acted upon and may not be acknowledged; system-level decode should prevent masters from targeting unmapped addresses to avoid hung transactions.
- wb_data_o is registered and returned according to the address map. Write data is captured into dout synchronously in the same cycle the write enable pulses.

Integration note:
- No explicit reset is shown inside ac97_wb_if; re/we edge detectors and ack may power up unknown. Provide a proper reset or benign initial bus conditions in the system.

-- Parameterization and Build Options --

Current status: No synthesizable Verilog parameters or compile-time defines are exposed. All widths, depths, timing constants, address maps, channel mappings, and interrupt widths are fixed in RTL; there are no generate blocks or conditional compilation.

Build-time options: None. The design's AC'97 framing (256 bits/frame; Slot 0 tag 16 bits; Slots 1–12 20 bits), slot utilization (TX: 3,4,6,7,8,9; RX: 0,1,2,3,4,6; unused SDATA_OUT slots 5,10,11,12 tied to 0), FIFO depths (4 words, 32-bit), sample width handling (16/18/20-bit packing rules), Wishbone map (RF 0..7; TX write 8..13; RX read 14..16), interrupt vector width/mapping (29 causes in INTS with 1:1 mask in INTM), CRA packing ([31] RW, [22:16] addr, [15:0] data), timing (frame valid threshold, slot latch points, suspend timeout ≈0x21 wclk cycles, resume pulse length ≈5 ps_ce, ac97_rst deassert after 50-cycle prescaler ×4), clocks/resets (fixed clk and wclk domains; reset polarities where present) are all hard-coded.

Runtime configurables (not build-time): Channel mode/enable/threshold via OCC0/OCC1/ICC (incl. sample width mode [3:2]); interrupt masking via INTM and sticky status via INTS; DMA request enable/threshold policy via ac97_dma_if cfg bits; Codec Register Access via CRAC writes; resume/reset sequencing via CSR writes.

Integration/tooling expectations: No base-address parameter; no bus-level reset signal. CDC crossings are fixed; some modules intentionally have no explicit resets and may power up X—do not optimize away framing-dependent initialization. Simulation may require reset sequencing to ensure valid low windows before operation.

Potential future parameterization targets (if customization is required): FIFO depth and word width (and empty/status logic for 18/20-bit modes); AC'97 frame timing points (valid threshold, per-slot out_le counts) and slot enable/utilization; suspend timeout, resume pulse length, and ac97_rst prescaler/delay; interrupt vector width/mapping and DMA threshold policy; Wishbone base address/window and optional bus reset.

Dependency notes if enabling additional slots or changing channel mapping: Requires coordinated edits in ac97_prc (slot-0 tag), ac97_sout (slot loading), ac97_sin (slot capture), and the bus interface/register map to maintain consistency across TX/RX paths and software-visible registers.

-- Timing and Performance Constraints --

- Clocks and clock domains
- Two asynchronous clock domains: clk (system/bus/DMA) and wclk (AC-link BIT_CLK). Treat them as asynchronous clock groups. Add false paths between domains except through explicit two-flop synchronizers.
- AC-link edge detection in the system domain: only constrain the two-flop sampler path used for edge detect; cut all other clk↔wclk crossings. For robust edge detection, require fclk ≥ 2 × fBIT_CLK (e.g., ≥ 24.576 MHz for a 12.288 MHz BIT_CLK); slower ratios can miss edges and falsely trigger suspend.
- CDC caveat: suspended is generated in clk domain and consumed in wclk domain; add a two-flop synchronizer in the wclk domain or formally cut timing and validate functional robustness.
- Naming: some submodules name the AC-link clock port "clk"; ensure constraints bind the AC-link I/O to wclk and the system logic to clk.

- AC-link I/O and frame timing
- Frame: 256 wclk edges per frame (16-bit Slot 0 tag + 12×20-bit slots). Typical fBIT_CLK=12.288 MHz → frame rate ≈ 48 kHz.
- SYNC: asserted for the first 16 wclk bit times of each frame. so_ld must assert exactly once per 256 wclk edges at the frame boundary.
- SDATA_OUT: MSB-first, change/launch on wclk rising edges. Constrain with set_output_delay relative to wclk to meet codec setup/hold.
- SDATA_IN: codec drives on wclk rising; controller stages on wclk falling and samples on the next rising (negedge-to-posedge half-cycle path). Constrain with set_input_delay accounting for the half-cycle capture and duty-cycle distortion; ensure the shortest half-period minus input delay/skew meets setup/hold.
- Duty-cycle requirement: wclk duty-cycle distortion must not violate the negedge-to-posedge capture path. Budget margins for clock skew, IO delay, and device variability.

- Internal frame/slot generation (wclk domain)
- Free-running 8-bit counter when not suspended; ld at count 0; per-slot latch-enables (out_le[n]) and valid window are deterministic from the count.
- valid window opens late in the frame (cnt > 0x39) and is pipelined (valid → valid_s1 → valid_s); any added pipelines must preserve slot alignment. out_le strobes must terminate on slot boundaries so full 16/20-bit words are captured.

- Suspend/resume behavior
- Suspend timeout: no AC-link edge detected in the system domain for 33 clk cycles triggers suspended. Sensitivity scales with fclk; verify at target fclk to avoid false positives/negatives.
- Resume: SYNC is forced high for approximately 5 ps_ce pulses; ps_ce source/frequency must be chosen to meet AC’97 resume timing requirements.

- Reset timing
- ac97_rst_ (active-low) held low after reset or rst_force and released high ≈ 200 clk cycles later (50-cycle prescaler × 4 + 1). Ensure this exceeds the codec’s minimum reset width at the chosen clk frequency. Ensure bus fabric provides a system reset before first Wishbone access.

- Throughput and latency (FIFOs/service)
- Playback FIFO: 4×32-bit.
- 16-bit mode: 8 samples effective depth → ≈ 167 µs service window at 48 kHz.
- 18/20-bit modes: 4 samples → ≈ 83 µs service window at 48 kHz.
- out_fifo empty flag is only reliable in 16-bit mode; prefer status/full for flow control in wider modes.
- Capture FIFO: 4×32-bit.
- 16-bit mode: packs two halves per word → 8 halves (≈ 167 µs at 48 kHz).

- 18/20-bit modes: 4 samples (≈ 83 µs).
- DMA/CPU must service each enabled channel within these worst-case windows to avoid underflow/overflow.

- Bus/DMA/interrupt timing
- Wishbone: single-cycle acknowledge for each accepted access; no wait-state insertion. Combinational address decode/mux paths must meet clk timing.
- DMA: dma_req is qualified for one extra cycle and held until dma_ack; ack latency does not affect correctness but overall service must meet FIFO latency budgets.
- Interrupts: underflow/overflow attempts generate one-clk pulses; register file latches them as sticky status, easing ISR latency requirements.

- Codec Register Access (CRAC)
- Writes: Slot1/Slot2 driven and complete within the same valid window; crac_wr_done pulses at valid falling edge.
- Reads: command frame in Slot1; read data returns next frame in Slot2; crac_rd_done pulses on the subsequent valid edge. Budget at least one frame of latency between command and data availability.

- STA/constraint guidance
- Define separate clocks for clk and wclk; set_clock_groups -asynchronous between them (or explicit false paths), except designated synchronizer paths.
- Constrain AC-link I/O relative to wclk with set_input_delay/set_output_delay per codec timing; add half-cycle constraints for SDATA_IN (falling-to-rising capture) and account for wclk duty-cycle distortion.
- Constrain only the clk-domain edge sampler used for suspend detection; cut other cross-domain paths, including suspended if not resynchronized.
- Verify timing on FIFO combinational read path (mem $\rightarrow$ dout_tmp $\rightarrow$ registered dout on re) at the target clk frequency.
- Ensure so_ld periodicity (1 per 256 wclk) is preserved; missed/jittered pulses will break frame alignment and violate slot timing.