AutoChip — Report of 5 Examples

Course/Lab: Lab 2-AutoChip

Student: **Naveen Kumar Senthil Kumar**

NetID: **ns6503**

Date: September 18, 2025

1. Binary to BCD Converter

Folder: /content/binary_to_bcd

Prompt source file: ./binary_to_bcd/config.json ('prompt' field)

Testbench file: ./binary_to_bcd/binary_to_bcd_tb.v

Module name (expected): top_module

Prompt (summary)

Generate Verilog-2001 module top_module converting 5-bit binary_input (0–31) to 8-bit BCD (tens in [7:4], ones in [3:0]) using division/modulo; output only the module.

Run configuration

Model: ChatGPT (gpt-4o-mini) via AutoChip loop

Num candidates per iteration: 5

Results

• Iteration 0: all 5 candidates: Testbench ran successfully; Mismatches: 0 / Samples: 32.

• Costs per candidate (approx): $0.0000903, $0.0001047, $0.0001167, $0.0001041, $0.0001101.

• Ranks for all responses: 1.0 (perfect).

Key insight

Well-specified prompt with exact header/ports + simple arithmetic led to consistent, correct completions across candidates.

2. State Sequence Detector (FSM)

Folder: /content/sequence_detector

Prompt source file: ./sequence_detector/config.json ('prompt' field)

Testbench file: ./sequence_detector/sequence_detector_tb.v

Module name (expected): top_module

Prompt (summary)

Synthesize Verilog-2001 top_module with 8 states S0–S7 and exact transition rules; active-low async reset to S0; combinational output sequence_found=1 only when state==S7 AND data==3'b101.

Run configuration

Model: ChatGPT (gpt-4o-mini) via AutoChip loop

Num candidates per iteration: 5

Results

• Iteration 0: all 5 candidates simulated with exactly 1 error.

• Common failure: 'Error: Cycle 8, Expected: 1, Got: 0'; Mismatches: 1 / Samples: 12 (rank ≈ 0.9167).

• Costs per candidate (approx): ~$0.000323–$0.000328.

Key insight

Likely edge/condition mismatch: output asserted one cycle late/early or S7 gating wrong. Tighten prompt to clarify Moore/Mealy timing and require explicit registered/comb structure for sequence_found.

3. Die Roller (LFSR-based RNG)

Folder: /content/dice_roller

Prompt source file: ./dice_roller/config.json ('prompt' field)

Testbench file: ./dice_roller/dice_roller_tb.v

Module name (expected): top_module

Prompt (summary)

Verilog-2001 top_module with LFSR[7:0], roll rising-edge detection, taps x^8+x^7+x^6+x^5+1, modulo mapping for 4/6/8/20-sided dice; single always @(posedge clk or negedge rst_n), nonblocking ops.

Run configuration

Model: ChatGPT (gpt-4o-mini) via AutoChip loop

Num candidates per iteration: 5

Results

• Iteration 0: all 5 candidates: Testbench ran successfully; Mismatches: 0 / Samples: 4000.

• Costs per candidate (approx): ~$0.000349–$0.000369.

• Ranks for all responses: 1.0 (perfect).

Key insight

Precise behavioral spec (roll edge, modulo ranges 1..N, feedback taps) ensured deterministic compliance across candidates.

4. Shift Register

Folder: /content/shift_register

Prompt source file: ./shift_register/config.json ('prompt' field)

Testbench file: ./shift_register/shift_register_tb.v

Module name (expected): top_module

Prompt (summary)

Generate a simple 8-bit shift register with active-low reset and shift_enable; output-only module, no testbench generation.

Run configuration

Model: ChatGPT (gpt-4o-mini) via AutoChip loop

Num candidates per iteration: 5

Results

• Iteration 0: all 5 candidates: Testbench ran successfully; Mismatches: 0 / Samples: 7.

• Costs per candidate (approx): ~$0.0001225–$0.0001412.

• Ranks for all responses: 1.0 (perfect).

Key insight

Short, unambiguous prompt; simple sequential logic with reset/shift made it robust across completions.

5. Token Bucket Rate Limiter

Folder: /content/tokenbucket

Prompt source file: ./tokenbucket/config.json ('prompt' field)

Testbench file: ./tokenbucket/tokenbucket_tb.v

Module name (expected): top_module

Prompt (summary)

Verilog-2001 top_module with parameters (DEN,RATE_NUM,BURST_MAX,TOKEN_COST), 32-bit tokens, MAX_TOKENS=BURST_MAX*DEN, POST-ADD semantics, ready_o derived from tokens_sat, registered 1-cycle grant pulse, no SV features.

Run configuration

Model: ChatGPT (gpt-4o-mini) via AutoChip loop

Num candidates per iteration: 5

Results

• Run output truncated in the provided export; setup, prompt, and file staging are visible.

• Prior logs indicate adherence to POST-ADD readiness and saturation; final pass/fail not visible.

Key insight

The strict POST-ADD wording is critical; ensuring ready_o is based on tokens after addition/saturation avoids off-by-one grant/ready errors.

Overall Observations

• Strong, explicit prompts that fix module headers, reset behavior, and exact arithmetic/timing conditions lead to 100% pass rates across multiple candidates.
• For FSMs, specify Moore vs Mealy output timing and whether detection is combinational or registered, to prevent one-cycle misalignments.
• Including compile/sim constraints (no SV, nonblocking in sequential, no initial blocks) reduces variance and warnings.

**ChatGPT was used to summarize the insights found during execution of Auto Chip scripts*