100-Day SystemVerilog Schedule (Without UVM)

This 100-day plan comprehensively covers all SystemVerilog (SV) concepts excluding UVM.
It spans from Verilog recap to advanced SystemVerilog features including assertions, coverage, OOP, and synthesis.

Days 1–7 – Foundations & Verilog Recap
1. Digital logic & Verilog recap — modules, ports, gates.
2. Module instantiation, hierarchical references, wire/reg.
3. Continuous assignments, blocking vs non-blocking.
4. always blocks, sensitivity lists, always_comb/ff/latch.
5. Operators, concatenation, replication.
6. Delay & event control, posedge/negedge.
7. initial blocks, simulation control tasks.

Days 8–14 – Data Types & Nets
8. Net types: wire, tri, wand, wor.
9. Variable types: logic, bit, reg, signed/unsigned.
10. Integer types: byte, int, longint, time.
11. Real types: real, shortreal, realtime.
12. enum, struct, union.
13. typedef, localparam, parameter.
14. Strings and string methods.

Days 15–21 – Arrays
15. Packed vs unpacked arrays.
16. Static arrays.
17. Dynamic arrays.
18. Queues.
19. Associative arrays.
20. Arrays of structs/classes.
21. foreach and array iteration.

Days 22–28 – Procedural & Tasks
22. if, case, for, while, do.
23. task/function basics.
24. automatic vs static, recursion.
25. ref/const ref semantics.
26. System tasks: $display, $fopen.
27. fork-join constructs.
28. Delays in tasks and simulation scheduling.

Days 29–35 – Modules & Interfaces
29. Module ports.
30. interface basics.
31. Virtual interfaces.
32. modport details.
33. Interface tasks/functions/clocks.
34. Interface arrays.
35. bind statement.

Days 36–42 – Clocking & Timing
36. Clocking blocks.

37. Clocking vs posedge sync.
38. Sampling/driving via clocking.
39. Metastability checks.
40. disable statement, scopes.
41. NBA scheduling.
42. generate & elaboration timing.

Days 43–49 – Concurrency & IPC
43. Events.
44. Mailboxes.
45. Semaphores.
46. process control.
47. Class-based TB (no UVM).
48. IPC patterns.
49. Scheduler/delta cycle details.

Days 50–56 – Classes & OOP
50. Class basics.
51. Constructors/destructors.
52. Inheritance.
53. Polymorphism.
54. static/local members.
55. protected & packages.
56. Handle semantics.

Days 57–63 – Randomization
57. rand/randc usage.
58. Constraints basics.
59. Dependent constraints.
60. pre/post_randomize hooks.
61. Covergroup-driven randomization.
62. foreach randomization.
63. Randomization debug & seeds.

Days 64–70 – Assertions (SVA)
64. property, sequence, assert.
65. Sequence concatenation/repetition.
66. assert/cover/assume.
67. Immediate vs concurrent assertions.
68. Past-time operators.
69. Formal-friendly assertions.
70. bind for assertions.

Days 71–77 – Functional Coverage
71. covergroup basics.
72. coverpoint bins.
73. Cross coverage.
74. Sampling & per-instance.
75. Conditional coverage.
76. Coverage closure.
77. Coverage-driven flow.

Days 78–84 – Verification (Non-UVM)
78. Testbench architecture.
79. Transactors/TLM.
80. Scoreboard/reference model.

81. Transaction recording.
82. Regression management.
83. Packages & modularity.
84. Debug strategies.

Days 85–91 – Advanced SV
85. DPI basics.
86. import/export DPI.
87. program block.
88. Pragmas/directives.
89. unique/priority case.
90. $fatal/$warning/$error.
91. Protected classes & interfaces.

Days 92–98 – Synthesis & Style
92. Synthesizable subset.
93. Coding styles/FSM/CDC.
94. generate constructs.
95. Memory models.
96. Pragmas/timing.
97. Waveform dumps.
98. Linting/guidelines.

Days 99–100 – Mini Projects
99. DUT + TB with randomization, assertions, coverage.
100. Regression setup, coverage report, documentation.

Tips:
- Keep daily notes & version control your code.
- Practice small focused exercises.
- Use assertions & coverage from day 1 for verification mindset.