**DD Block 1: Binary Math & Boolean algebra**

* DD Textbook Chapter 1
* [Gray Code](https://www.geeksforgeeks.org/what-is-gray-code/)
* [BCD or Binary Coded Decimal](https://www.geeksforgeeks.org/bcd-or-binary-coded-decimal/)
* [Excess-3 Code](https://www.geeksforgeeks.org/excess-3-code/)
* [Implicants in K-Map](https://www.geeksforgeeks.org/digital-logic-implicants-k-map)
* [Quine-McCluskey Method (Tabular Method)](https://www.geeksforgeeks.org/quine-mccluskey-method/)
* [Variable Entrant Map (VEM)](https://www.geeksforgeeks.org/digital-logic-variable-entrant-map-vem)
* [Consensus theorem](https://www.geeksforgeeks.org/digital-logic-consensus-theorem)
* Interview Q 1-3
* Write a parameterized binary-to-Gray converter & Excess-3 Module

**DD Block 2: Combinational building blocks**

* DD Textbook Chapter 2
* [Combinational Circuit Introduction](https://www.geeksforgeeks.org/what-is-combinational-circuit/)
* [Half-Adder and Half-Subtractor using NAND NOR Gates](https://www.geeksforgeeks.org/half-adder-half-subtractor-using-nand-nor-gates)
* [Full-Adder](https://www.geeksforgeeks.org/full-adder-digital-electronics)
* [Full Subtractor](https://www.geeksforgeeks.org/digital-logic-full-subtractor)
* [BCD Adder](https://www.geeksforgeeks.org/digital-electronics-bcd-adder)
* [Code Converters – Binary to/from Gray Code](https://www.geeksforgeeks.org/digital-logic-code-converters-binary-gray-code)
* [Multiplexers](https://www.geeksforgeeks.org/multiplexers-digital-electronics)
* [Demultiplexer](https://www.geeksforgeeks.org/what-is-demultiplexerdemux/)
* [Encoders and Decoders](https://www.geeksforgeeks.org/digital-logic-encoders-decoders)
* [Encoder](https://www.geeksforgeeks.org/digital-logic-encoder)
* [Binary Decoder](https://www.geeksforgeeks.org/digital-logic-binary-decoder)
* [Combinational circuits using Decoder](https://www.geeksforgeeks.org/combinational-circuits-using-decoder)
* [Magnitude Comparator](https://www.geeksforgeeks.org/magnitude-comparator)
* [Static Hazards](https://www.geeksforgeeks.org/digital-logic-static-hazards)
* Design a 4-way priority encoder + carry-look-ahead adder

**DD Block 3: Sequential elements**

* DD Textbook Chapter 2
* [Difference Between Combinational and Sequential Circuits](https://www.geeksforgeeks.org/difference-between-combinational-and-sequential-circuit/)
* [Latches](https://www.geeksforgeeks.org/digital-logic-latches)
* [One bit memory cell](https://www.geeksforgeeks.org/computer-organisation-one-bit-memory-cell)
* [Flip-Flops(Types and Conversions)](https://www.geeksforgeeks.org/flip-flop-types-and-their-conversion)
* [SR Flip-Flops](https://www.geeksforgeeks.org/sr-flip-flop/)
* [JK Flip-Flops](https://www.geeksforgeeks.org/what-is-jk-flip-flop/)
* [D Flip-Flops](https://www.geeksforgeeks.org/d-flip-flop/)
* [T Flip-Flops](https://www.geeksforgeeks.org/t-flip-flop/)
* [Master Slave JK Flip Flop](https://www.geeksforgeeks.org/master-slave-flip-flop)
* [Synchronous Sequential Circuits](https://www.geeksforgeeks.org/synchronous-sequential-circuits)
* [Asynchronous Sequential Circuits](https://www.geeksforgeeks.org/digital-logic-asynchronous-sequential-circuits)
* [Synchronous vs Asynchronous Sequential Circuits](https://www.geeksforgeeks.org/difference-between-synchronous-and-asynchronous-sequential-circuits/)
* [State Reduction and State Assignment](https://www.geeksforgeeks.org/state-reduction-and-state-assignment/)
* Interview Q 5-7
* Implement Code edge-triggered D-FF with async reset; RTL vs gate sim comparison

**DD Block 4: Counters & clock-dividers**

* [Registers](https://www.geeksforgeeks.org/what-is-register-digital-electronics/)
* [Shift Registers](https://www.geeksforgeeks.org/digital-logic-shift-registers)
* [Serial In Serial Out (SISO) Shift Register](https://www.geeksforgeeks.org/siso-shift-register/)
* [Serial In Parallel Out (SIPO) Shift Register](https://www.geeksforgeeks.org/sipo-shift-register/)
* [Parallel In Serial Out (PISO) Shift Register](https://www.geeksforgeeks.org/piso-shift-register/)
* [Parallel In Parallel Out (PIPO) Shift Register](https://www.geeksforgeeks.org/pipo-shift-register/)
* [Universal Shift Register](https://www.geeksforgeeks.org/digital-logic-universal-shift-register)
* [Counters](https://www.geeksforgeeks.org/counters-in-digital-logic)
* [Ripple Counter](https://www.geeksforgeeks.org/digital-logic-ripple-counter)
* [Ring Counter](https://www.geeksforgeeks.org/ring-counter-in-digital-logic/)
* [Johnson Counter](https://www.geeksforgeeks.org/n-bit-johnson-counter-in-digital-logic/)
* [Design of Counters for Specific Sequences](https://www.geeksforgeeks.org/design-counter-given-sequence/)
* [Amortized analysis for increment in counter](https://www.geeksforgeeks.org/amortized-analysis-increment-counter)
* [Registers Vs Counters](https://www.geeksforgeeks.org/registers-vs-counters/)
* <https://www.mikrocontroller.net/attachment/177198/Clock_Dividers_Made_Easy.pdf>
* Implement ÷3.5 clock using even-odd toggle network

**DD Block 5: Static Timing Analysis & constraints**

* As a fresher, you might not be expected to have worked on STA directly but nevertheless, you should know the concept of what STA is, why it’s the standard way of verifying the timing viability of the design, and what you would if you were to encounter a relatively common case of STA failure. Questions on this topic can range from fairly simple to very complex but it all depends on the level of your experience. -Metastability: This topic forms the core of the issue that STA is trying to solve. It is of outmost importance for one to have a clear understanding of this topic to ace both STA and CDC.
* Derive setup/hold equations for two-FF pipeline, verify with STA

**DD Block 6: Finite-State Machines**

* <https://www.sunburst-design.com/papers/CummingsSNUG2019SV_FSM1.pdf?utm_source=chatgpt.com>
* Code 3-stage handshake FSM in one-hot & gray-encoded styles
* <https://www.sunburst-design.com/papers/CummingsSNUG1998SJ_FSM.pdf>

**DD Block 7: Verilog language deep-dive**

* <https://www.sunburst-design.com/papers/CummingsSNUG2000SJ_NBA.pdf?utm_source=chatgpt.com>
* <https://www.sunburst-design.com/papers/CummingsSNUG2006Boston_SystemVerilog_Events.pdf>
* *SystemVerilog for Design* ch 3-5
* Rewrite a buggy traffic-light controller and explain race fixups

**DD Block 8: RTL pitfalls & synthesis mismatch**

* <https://www.sunburst-design.com/papers/CummingsSNUG1999SJ_SynthMismatch.pdf?utm_source=chatgpt.com>
* Create intentional latch-inference bug, synthesize, show mismatch

**DD Block 9: Clock-Domain Crossing (CDC)**

* <https://www.sunburst-design.com/papers/CummingsSNUG2008Boston_CDC.pdf?utm_source=chatgpt.com>
* <https://thedatabus.in/cdc_complete_guide>
* Build Gray-code async FIFO, run self-checking SVTB

**DD Block 10: FIFO depth calc & flow-control**

* <https://www.sunburst-design.com/papers/CummingsSNUG2002SJ_FIFO1.pdf?utm_source=chatgpt.com>
* Parameterize depth from clk ratio; drive random traffic

**DD Block 11: Reset, gearboxes, power & DFT**

* <https://www.sunburst-design.com/papers/CummingsSNUG2003Boston_Resets.pdf>
* <https://docs.amd.com/v/u/en-US/wp272>
* *VLSI Test Principles* ch 2-4
* Code 64-to-40 bit gearbox with power-safe hand-off
* Low Power VLSI tricks

**DD Block 12: Assertions & event regions (optional but high ROI)**

* <https://sunburst-design.com/papers/CummingsSNUG2009SJ_SVA_Bind.pdf?utm_source=chatgpt.com>
* Add SVA to FIFO for overflow/underflow and CDC timing

**DD Block 13: Special Topics:**

* <https://zipcpu.com/blog/2017/08/14/strategies-for-pipelining.html>
* Skid Buffers
* Arbiters
* Protocols: I2C, SPI, AXI,

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Block** | **Days** | **Topics build-up** | **Core reading / viewing** | **Suggested activities** |
| **A. ISA & performance metrics** | 2 | ISA design, CPI, Amdahl | *CA: A Quantitative Approach* (CAQA) §1.1-1.6; GfG COA “Intro” list ([GeeksforGeeks](https://www.geeksforgeeks.org/computer-organization-and-architecture-tutorials/" \o "Computer Organization and Architecture Tutorial | GeeksforGeeks)) | Summarize trade-offs of RISC vs CISC |
| **B. Five-stage pipeline & hazards** | 3 | datapath, structural/data/control hazards | CAQA App C (§C.1-C.3) ([Chipress](https://chipress.online/2024/03/30/a-must-read-book-list-for-rtl-design-interviews/" \o "A Must Read Book List for RTL Design Interviews – Chipress)) | Modify 5-stage RV32I Verilog core; insert bypass nets |
| **C. Branch prediction & superscalar basics** | 2 | static vs dynamic BP, Tomasulo intro | CAQA ch 3 (§3.1-3.3) ([Chipress](https://chipress.online/2024/03/30/a-must-read-book-list-for-rtl-design-interviews/" \o "A Must Read Book List for RTL Design Interviews – Chipress)) | Implement 2-bit predictor simulator; measure mis-pred impact |
| **D. Memory hierarchy** | 3 | cache taxonomy, write policies | CAQA App B (review) ([Chipress](https://chipress.online/2024/03/30/a-must-read-book-list-for-rtl-design-interviews/" \o "A Must Read Book List for RTL Design Interviews – Chipress)) | RTL model direct-mapped vs 2-way cache; simulate hit rate |
| **E. Virtual memory & TLBs** | 1 | page tables, address translation | GfG COA “Memory Organisation” list items 7-12 ([GeeksforGeeks](https://www.geeksforgeeks.org/computer-organization-and-architecture-tutorials/" \o "Computer Organization and Architecture Tutorial | GeeksforGeeks)) | Walk through x86-64 four-level PT in Python |
| **F. Out-of-order execution & ILP** | 3 | ROB, register renaming, reservation stations | CAQA ch 3 (§3.4-3.6) ([Chipress](https://chipress.online/2024/03/30/a-must-read-book-list-for-rtl-design-interviews/" \o "A Must Read Book List for RTL Design Interviews – Chipress)); Onur Mutlu lec 10-14 (CMU 18-447) ([YouTube](https://www.youtube.com/user/cmu18447?utm_source=chatgpt.com)) | Build small OoO issue queue simulator |
| **G. Coherence & consistency** | 3 | MSI/MESI/MOESI, memory models | FinalRoundAI tip “Owned state in MOESI” ([Final Round AI](https://www.finalroundai.com/blog/computer-architecture-interview-questions?utm_source=chatgpt.com)) | Draw MESI finite-state diagram; explain “Owned” workloads |
| **H. Multicore & interconnects** | 2 | mesh vs ring, NUMA, directory | Onur Mutlu interconnect lecture ([YouTube](https://www.youtube.com/watch?v=fTHwFe0yjTg&utm_source=chatgpt.com)) | Compare ring vs mesh latency in log book |
| **I. GPUs, TPUs & accelerators (overview)** | 2 | SIMT, systolic arrays | ETH “Digital Design & CA” Mutlu lecture 28 (optional) ([YouTube](https://www.youtube.com/onurmutlulectures?utm_source=chatgpt.com)) | Sketch dataflow of a systolic GEMM kernel |
| **J. Review & mock interviews** | 4 | create cheat-sheets, timed white-board drills | Chipress “Crack the HW Interview” suggestion ([Chipress](https://chipress.online/2024/03/30/a-must-read-book-list-for-rtl-design-interviews/" \o "A Must Read Book List for RTL Design Interviews – Chipress)) | Dry-run RTL + µarch questions with peer |

|  |  |  |
| --- | --- | --- |
| **#** | **Resource (title / edition)** | **Type & location** |
| 1 | **Digital Design & Computer Architecture** – Harris & Harris, 4 th ed. | Full-book PDF (389 pp.) ([R-5](https://www.r-5.org/files/books/computers/hw-layers/hardware/digital-desigh/David_Harris_Sarah_Harris-Digital_Design_and_Computer_Architecture-EN.pdf?utm_source=chatgpt.com)) |
| 2 | **“Finite-State-Machine (FSM) Design & Synthesis Using SystemVerilog”** (Cummings 2019) | Sunburst-Design PDF ([sunburst-design.com](https://www.sunburst-design.com/papers/CummingsSNUG2019SV_FSM1.pdf?utm_source=chatgpt.com)) |
| 3 | **“Non-blocking Assignments in Verilog – Coding Styles That Work in Simulation & Synthesis”** (Cummings 2000) | Sunburst-Design PDF ([sunburst-design.com](https://www.sunburst-design.com/papers/CummingsSNUG2000SJ_NBA.pdf?utm_source=chatgpt.com)) |
| 4 | **“RTL Coding Styles That Yield Simulation/Synthesis Mismatches”** (Cummings 1999) | Sunburst-Design PDF ([sunburst-design.com](https://www.sunburst-design.com/papers/CummingsSNUG1999SJ_SynthMismatch.pdf?utm_source=chatgpt.com)) |
| 5 | **“Clock-Domain-Crossing (CDC) Design & Verification Techniques”** (Cummings 2008) | Sunburst-Design PDF ([sunburst-design.com](https://www.sunburst-design.com/papers/CummingsSNUG2008Boston_CDC.pdf?utm_source=chatgpt.com)) |
| 6 | **“Simulation & Synthesis Techniques for Asynchronous FIFO Design”** (Cummings 2002) | Sunburst-Design PDF ([sunburst-design.com](https://www.sunburst-design.com/papers/CummingsSNUG2002SJ_FIFO1.pdf?utm_source=chatgpt.com)) |
| 7 | **“SystemVerilog Event Regions, Race Avoidance & Guidelines”** (Cummings 2006) | Sunburst-Design PDF ([sunburst-design.com](https://www.sunburst-design.com/papers/CummingsSNUG2006Boston_SystemVerilog_Events.pdf?utm_source=chatgpt.com)) |
| 8 | **“SystemVerilog Assertions: Design Tricks & SVA Bind Files”** (Cummings 2009) | Sunburst-Design PDF ([sunburst-design.com](https://sunburst-design.com/papers/CummingsSNUG2009SJ_SVA_Bind.pdf?utm_source=chatgpt.com)) |
| 9 | **SystemVerilog for Design — 2 nd Edition** (Sutherland et al.) | Full-book PDF ([Embedic](https://www.embedic.com/uploads/files/20201009/SystemVerilog%20for%20Design%20Second%20Edition%20A%20Guide%20to%20Using%20SystemVerilog%20for%20Hardware%20Design%20and%20Modeling%20by%20Stuart%20Sutherland%2C%20Simon%20Davidmann%2C%20Peter%20Flake%2C%20P.%20Moorby%20%28z-lib.org%29.pdf?srsltid=AfmBOoomO5mk5RL62XKWnPxUrCkh2t3mNKFXF4DnVNUP4PPP0IfTkWJ9&utm_source=chatgpt.com" \o "[PDF] SystemVerilog For Design - EmbedIc)) |
| 10 | **VLSI Test Principles & Architectures** (Wang / Wen / Wu) | Full-book PDF ([LNIV](https://lniv.fe.uni-lj.si/courses/pev/TestPrinciples.pdf?utm_source=chatgpt.com)) |
| 11 | **Computer Architecture: A Quantitative Approach** – 6 th ed. | Full-book PDF (Paterson & Hennessy) ([acs.pub.ro](https://acs.pub.ro/~cpop/SMPA/Computer%20Architecture%2C%20Sixth%20Edition_%20A%20Quantitative%20Approach%20%28%20PDFDrive%20%29.pdf?utm_source=chatgpt.com)) |
| 12 | **Digital-Electronics Interview Questions** (GeeksforGeeks) | Web article list ([GeeksforGeeks](https://www.geeksforgeeks.org/digital-electronics-interview-questions/?utm_source=chatgpt.com" \o "Digital Electronics Interview Questions for 2024 [Updated])) |
| 13 | **Computer Organization & Architecture Tutorials** (GeeksforGeeks – master index) | Web tutorial hub ([GeeksforGeeks](https://www.geeksforgeeks.org/computer-organization-and-architecture-tutorials/?utm_source=chatgpt.com" \o "Computer Organization and Architecture Tutorial | GeeksforGeeks)) |
| 14 | **Preparing for an RTL Design Interview – Exhaustive Topic List** (theDataBus) | Web article ([theDataBus](https://thedatabus.in/interview_list/?utm_source=chatgpt.com)) |
| 15 | **Static-Timing-Analysis & other STA notes** (within theDataBus list, see section headers) | Same page as #14 ([theDataBus](https://thedatabus.in/interview_list/?utm_source=chatgpt.com)) |
| 16 | **“A Must-Read Book List for RTL Design Interviews”** (Chipress) | Web article ([Chipress](https://chipress.online/2024/03/30/a-must-read-book-list-for-rtl-design-interviews/?utm_source=chatgpt.com" \o "A Must Read Book List for RTL Design Interviews - Chipress)) |
| 17 | **Computer-Architecture Interview Questions You Should Know** (FinalRoundAI) | Web article ([Final Round AI](https://www.finalroundai.com/blog/computer-architecture-interview-questions?utm_source=chatgpt.com)) |
| 18 | **Onur Mutlu Computer-Architecture Lectures – CMU 18-447 channel** | YouTube playlist hub (all lectures, incl. 10-14) ([YouTube](https://www.youtube.com/user/cmu18447?utm_source=chatgpt.com)) |
| 19 | **Lecture 22: Interconnects (Fall 2024)** – Onur Mutlu | Specific YouTube video for the “Interconnects” deep-dive ([YouTube](https://www.youtube.com/watch?v=fTHwFe0yjTg&utm_source=chatgpt.com)) |
| 20 | **Digital Design & Computer Architecture – ETH Zürich (Spring 2020) playlist** | YouTube playlist (35 videos, accelerators incl. VLIW – Lecture 28) ([YouTube](https://www.youtube.com/playlist?list=PL5Q2soXY2Zi_FRrloMa2fUYWPGiZUBQo2&utm_source=chatgpt.com)) |