Guide2Code - Computer Organization & Architecture (COA) Roadmap

🟢 Phase 1: Beginner Level

📚 Topics to Learn:

1. **Introduction to Computer Architecture** (Von Neumann vs Harvard, Components)
2. **Number Systems & Arithmetic** (Binary, Octal, Hex, Signed & Unsigned Numbers)
3. **Boolean Algebra & Logic Gates** (AND, OR, NOT, XOR, Universal Gates)
4. **Combinational Circuits** (Multiplexers, Demultiplexers, Encoders, Decoders)
5. **Sequential Circuits** (Flip-Flops, Registers, Counters)
6. **Memory Hierarchy** (RAM, ROM, Cache, Virtual Memory)
7. **Instruction Set Architecture (ISA)** (RISC vs CISC, Addressing Modes)
8. **CPU Organization** (ALU, Control Unit, Register Set)
9. **Input-Output Mechanisms** (Interrupts, DMA, Polling)
10. **Basics of Assembly Language** (Registers, Instructions, Stack Operations)

🛠️ Beginner Project Ideas:

- **Binary Converter** – Convert numbers between bases
- **Logic Gate Simulator** – Simulate basic logic circuits
- **Simple ALU Implementation** – Perform basic arithmetic operations
- **Memory Address Calculator** – Calculate physical & logical addresses
- **Basic Assembler Simulator** – Translate simple assembly to machine code

---

🟡 Phase 2: Intermediate Level

📚 Topics to Learn:

1. **Processor Architecture** (Pipeline Processing, Hazards & Solutions)
2. **Cache Memory Organization** (Mapping Techniques, Write Policies)
3. **Microprogramming & Control Unit Design** (Hardwired vs Microprogrammed)

4. **Instruction-Level Parallelism (ILP)** (Superscalar, VLIW, Out-of-Order Execution)
5. **Memory Management Techniques** (Paging, Segmentation, TLB)
6. **I/O System Design** (I/O Controllers, Bus Systems, RAID Storage)
7. **Multiprocessor & Multicore Architectures** (Shared vs Distributed Memory)
8. **Performance Optimization** (Branch Prediction, Loop Unrolling, Prefetching)
9. **Embedded Systems & Real-time Computing** (Microcontrollers, IoT Hardware)
10. **Basic FPGA & Hardware Description Languages** (VHDL, Verilog Basics)

🛠 **Intermediate Project Ideas:**

- **Cache Simulator** – Implement LRU, FIFO, Direct-Mapped Cache
- **Pipeline CPU Simulator** – Visualize instruction execution in stages
- **Assembly Code Interpreter** – Execute basic assembly commands
- **Virtual Memory Manager** – Implement paging and address translation
- **RISC vs CISC Performance Comparison** – Simulate execution times

---

🔴 **Phase 3: Advanced Level**

📚 **Topics to Learn:**

1. **Advanced Pipelining & Superscalar Architectures** (Tomasulo's Algorithm, Register Renaming)
2. **Memory Coherence & Consistency Models** (MESI Protocol, Directory-Based)
3. **GPU & Parallel Computing** (CUDA, OpenCL, SIMD vs MIMD)
4. **Advanced Cache Optimization** (Prefetching, Write Combining, Multi-Level Caches)
5. **Quantum Computing Basics** (Qubits, Quantum Gates, Superposition)
6. **Processor Security & Spectre/Meltdown Attacks** (Side-Channel Attacks)
7. **Domain-Specific Architectures** (TPUs, Edge AI Processors, Neuromorphic Chips)
8. **Reconfigurable Computing** (FPGA Optimization, Hardware-Software Co-Design)
9. **High-Performance Computing (HPC)** (Cluster Computing, Parallel Algorithms)

10. **Advanced Embedded Systems** (RTOS, Bare-Metal Programming, IoT Edge AI)

🛠 **Advanced Project Ideas:**

- **FPGA-Based CPU Emulator** – Implement a simple CPU on FPGA
- **Branch Predictor Analyzer** – Study performance of prediction techniques
- **Parallel Processing Simulator** – Compare CPU vs GPU execution times
- **Quantum Circuit Simulator** – Simulate basic quantum operations
- **Hardware Security Analyzer** – Detect vulnerabilities in cache & memory