Learning Goals for Computer Organization II

Main theme	Prerequisites	Approaching Learning Goals	Reaches Learning Goals	Deepens Learning Goals
Digital logic	Knows the concepts Boolean variable and truth value (Discrete mathematics)	Can explain computer operation at electrical component (gates, circuits, flip-flops) level. Can build a truth table to describe circuit function. Can describe basic functionality of flip-flops. Can list basic gates and explain the basic idea of how circuits fgunction and describe their use in data storage.	Can implement a given logical function with combinational circuits and optimize it with Karnaugh maps Can explain the operation of clocked flip-flops and the register implementation with flip-flops	Can explain the Quine-McKluskey or Lugue methods to simplify combinational circuits Can explain the combinatorial circuit implementation of addition and ROM-memory circuits Can consider timing and voltage problems in implementing circuits
Bus operation	Can explain the need and operation of bus hierarchy (CO-I)	Can describe the meaning and function of one or multiple buses in component communication. Can list and describe the central concepts relating to buses (e.g., speed, width, signaling, timing, synchronization, arbitration) and their meaning in overall system, using PCI bus as an example.	Can list the differences of arbitration methods and describe their signal level implementation. Can explain at the signal and bus event level operation for at least one bus type.	Can explain the signal level operation for the latest bus types.

Main theme	Prerequisites	Approaching Learning Goals	Reaches Learning Goals	Deepens Learning Goals
Memory	Can explain the huge speed differences of various memory devices, as well the significance of them (CO-I). Can explain why cache memory is needed (CO-I). Know the basic ideas of cache and virtual memory operations (CO-I) .	Can explain memory reference actions when both cache and virtual memory are used. Can list and describe cache organization methods, their types and policies governing them. Can explain address translation in (multi-level) virtual memory, possibly using inverted page table. Can list the requirements and basic features for TLB operation.	Can explain the need for and implementation of cache memory hierarchy with separate instruction and data caches. Can explain the details of set-associative cache operation. Can explain with details how TLB operates, including its replacement policies. Can explain the similarities and differences of TLB and cache. Can estimate the memory reference time considering both virtual memory and cache effects.	Can explain differences between Rambus DRAM and normal DRAM. Can explain TLB and cache memory details for some certain CPUs. Can explain various replacement policies for virtual memory back-up store.
Arithmetic	Can explain addition and multiplication algorithms with paper and pencil (high school). Can explain the IEEE floating point presentation (CO-I).	Can explain integer basic arithmetic circuit level implementation. Can explain Booth's algorithm. Can explain IEEE floating point presentation for very large and very small numbers. Can list and describe floating point rounding methods.	Can explain the implementation of 2's complement addition and subtraction. Can apply Booth's algorithm for multiplication. Can explain the basics of IEEE floating point arithmetic implementation.	Can explain precisely integer division implementation Can explain detailed implementation for IEEE arithmetic implementation.
Instruction sets	Can explain instruction execution cycle and the differences between machine and symbolic assembly language (CO-I). Can explain different data reference methods and where data can locate (CO-I).	Can describe and compare instruction sets based on their fundamental features Can explain machine instruction components, meaning of registers, and ways to store multi-byte data.	Can classify instruction based on their features and processors based on their instruction sets. Can explain the structure and data reference methods for real instruction sets (e.g., Intel Pentium and ARM11).	Can give examples of Load-Store processors. Can explain registers, data types and data reference methods for Intel Pentium and Arm 11 processors.

Main theme	Prerequisites	Approaching Learning Goals	Reaches Learning Goals	Deepens Learning Goals
Processor structure and operation	Can explain the basic idea of von Neumann processor architecture (CO-I) Can explain instruction fetch execute cycle operation (CO-I)	Can explain processor general structure at register and control unit level. Can explain the basic idea of pipelining and superscalar processor and give examples of their dependency problems. Can compare RISC and CISC architectures.	Can compute the speed advantage obtained with pipelining Can give solutions to problems caused by structural, control and data pipeline dependencies. Can explain how RISC and CISC architectures can be combined	Can explain special registers and interrupt mechanisms in Intel Pentium and ARM11. Can explain basic ideas of combining multiple processor architectures in Intel Pentium II and Transmeta Crusoe processors. Can explain exact operation in executing multiple instructions simultaneously in IBM PowerPC and Intel IA-64 processors
Control		Can explain machine language execution at micro-programmed control level. Can explain how control signals for instruction fetch-execute cycle are produced with a state automata. Can explain the basic idea of micro-programmed control. Can explain the function of control memory in micro-programmed control.	Can explain hard-wired and micro-programmed control. Can explain how clock cycle length is determined. Can explain the differences, advantages and disadvantages of horizontal and vertical micro-programming. Can explain the advantages and disadvantages of various ways to select next micro-program instruction.	Can explain, how modern processor combines micro-programmed and direct control. Can explain precise implementation of some current processors.