1. **Instruction set architecture (ISA) –** Another term for CPU.
2. **Organization –** Characteristics including such things as the number and types of registers, methods of addressing memory, and basic design and layout of the instruction set.
3. **Very Long Instruction Word (VLIW) –** from Transmeta.
4. **Explicitly Parallel Instruction Computers (EPIC) –** from Intel.
5. **Data Dependency –** Data used in an instruction that depends on the result from a previous instruction.
6. **Control Dependency –** Comes into effect when branches and loops are put into play.
7. **Code-Morphing Layer –** Permanently resident in memory and processes every instruction prior to execution.
8. **Clock –** Provides a master control as to when each step in the instruction cycle takes place.
9. **Fetch Unit –** Portion of the CPU consists of an instruction fetch unit and an instruction decode unit. Instructions are fetched from memory by the fetch unit, based on the current address stored in an instruction pointer (IP) register.
10. **Execution Unit –** Contains the arithmetic/logic unit and the portion of the control unit that identifies and controls the steps that comprise the execution part for each different instruction.
11. **Pipelining –** As each instruction completes a step, the following instruction moves into the stage just vacated.
12. **Instruction Reordering –** Reorders instructions as they are executed to keep the pipelines full and to minimize situations where a delay is necessary.
13. **Scalar Processor –** Single execution unit pipeline, ignoring holes in the pipeline resulting from different instruction types and branch conditions, the CPU can average instruction execution approximately equal to the clock speed of the machine.
14. **Superscalar Processing –** Multiple execution units it is possible to process instructions in parallel, with an average rate of more than one instruction per clock cycle.
15. **Hazard –** Out-of-order instruction execution can cause problems because a later instruction may depend on the results from an earlier instruction.
16. **Data Dependency –** Situation in which the later instruction is supposed to use the results from the earlier instruction in its calculation.
17. **Speculative Execution –** Separate bank of registers is used to hold results from these instructions until previous instruction are complete.
18. **Branch History Table -** Small amount of dedicated memory built into the CPU that maintains a record of pervious choices for each of several branch instructions that have been used in the program being executed to aid in prediction.
19. **Wide Path Memory Access –** Instead of reading 1 byte at a time, the system can retrieve 2, 4, 8 or even 16 bytes simultaneously.
20. **Memory Interleaving –** Dividing memory into parts, so that it is possible to access more than one location at a time.
21. **n-way Interleaving –** 2 or 4 or some other value is substituted for n.
22. **Logical Storage Elements –** Elements that can independently accept a memory request.
23. **Cache Memory –** Small amount of high-speed memory, between the CPU and main storage. Invisible to the programmer and cannot be directly addressed in the usual way by the CPU.
24. **Cache Line –** Organized into blocks, each block provides a small amount of storage perhaps between 8 and 64 bytes.
25. **Tag –** Identifies the location in main memory that corresponds to the data being held in that block.
26. **Cache Controller –** Checks the tags to determine if the memory location of the request is presently stored within the cache.
27. **Write Through –** Writes data back to the main memory immediately upon change in the cache.
28. **Write Back –** Faster, since writes to memory are made only when a cache line is actually replaced.
29. **Locality of Reference –** At any given time, most memory references will be confined to one or a few small regions of memory.
30. **Stall Time –** The time it takes to move data to the cache.
31. **Disk Cache –** When a disk read or write request is made, the system checks the dish cache first.
32. **Multiprocessor systems or Tightly Coupled systems –** Computers that have multiple CPU’s.
33. **Multicore Processors –** When multiple CPU processors are supplied within a single integrated circuit, they are more commonly called this.
34. **Threads –** Each of the CPUs has access to the same memory and I/O, any CPU can theoretically execute any thread or program currently in memory.
35. **Master-slave Multiprocessing –** One CPU, the *master,* manages the system, and controls all resources and scheduling.
36. **Symmetrical Multiprocessing –** Each CPU has identical access to the OS, and to all system resources including memory.
37. **Simultaneous Thread Multiprocessing –** CPUs that implement a simplified, limited for of multiprocessing using parallel execution units within a single CPU to process two or more threads simultaneously.