RISC ARCHITECTURE

Pronounced risk, acronym for reduced instruction set computer, a type of microprocessor that recognizes a relatively limited number of instructions. Until the mid-1980s, the tendency among computer manufacturers was to build increasingly complex CPUs that had ever-larger sets of instructions. At that time, however, a number of computer manufacturers decided to reverse this trend by building CPUs capable of executing only a very limited set of instructions. One advantage of reduced instruction set computers is that they can execute their instructions very fast because the instructions are so simple. Another, perhaps more important advantage, is that RISC chips require fewer transistors, which makes them cheaper to design and produce. Since the emergence of RISC computers, conventional computers have been referred to as CISCs (complex instruction set computers).

RISC (Reduced Instruction Set Computer) is used in portable devices due to its power efficiency. For Example, Apple iPod and Nintendo DS. RISC is a type of microprocessor architecture that uses highly-optimized set of instructions. RISC does the opposite, reducing the cycles per instruction at the cost of the number of instructions per program Pipelining is one of the unique feature of RISC. It is performed by overlapping the execution of several instructions in a pipeline fashion. It has a high-performance RISC processor take simple instructions and are executed within a clock cycle

RISC ARCHITECTURE CHARACTERISTICS

* Simple Instructions are used in RISC architecture.
* RISC helps and supports few simple data types and synthesize complex data types.
* RISC utilizes simple addressing modes and fixed length instructions for pipelining.
* RISC permits any register to use in any context.
* One Cycle Execution Time
* The amount of work that a computer can perform is reduced by separating “LOAD” and “STORE” instructions.
* RISC contains Large Number of Registers in order to prevent various number of interactions with memory.
* In RISC, Pipelining is easy as the execution of all instructions will be done in a uniform interval of time i.e. one click.
* In RISC, more RAM is required to store assembly level instructions.
* Reduced instructions need a less number of transistors in RISC.
* RISC uses Harvard memory model means it is Harvard Architecture.
* A compiler is used to perform the conversion operation means to convert a high-level language statement into the code of its form.advantage over CISC.

CISC ARCHITECTURE

The CISC stands for complex instruction set computer. It is read only memory (ROM) based processor architecture which are designed with a full set of instruction that were intended to provide needed capabilities in the most efficient way. Basically, the CISC processor architecture is designed to simplify compilers and to improve performance under constraints such as small and slow memories. This processor is easy to program and make efficient use of memory. CISC processor has variable length instructions where the length often varies according to the addressing mode hence instructions require multiple clock cycles to execute. It is easy for micro-coding new instructions which allow designers to make CISC processor upwardly compatible means a new processor could run the same programs as earlier processors because the new processor would contain a superset of the instructions of the earlier processors. In CISC architecture which has software control unit consists of micro programmed control memory. The fig of CISC architecture given in below, the fig shows the CISC architecture with micro programmed control memory and unified cache.

The CISC approach attempts to minimize the number of instructions per program, sacrificing the number of cycles per instruction. Computers based on the CISC architecture are designed to decrease the memory cost. Because, the large programs need more storage, thus increasing the memory cost and large memory becomes more expensive. To solve these problems, the number of instructions per program can be reduced by embedding the number of operations in a single instruction, thereby making the instructions more complex.

* MUL loads two values from the memory into separate registers in CISC.
* CISC uses minimum possible instructions by implementing hardware and executes operations.
* Instruction Set Architecture is a medium to permit communication between the programmer and the hardware. Data execution part, copying of data, deleting or editing is the user commands used in the microprocessor and with this microprocessor the Instruction set architecture is operated.
* The main keywords used in the above Instruction Set Architecture are as below

VLIW ARCHITECTURE

Very-Long Instruction Word (VLIW) architectures are a suitable alternative for exploiting instruction-level parallelism (ILP) in programs, that is, for executing more than one basic (primitive) instruction at a time. These processors contain multiple functional units, fetch from the instruction cache a Very-Long Instruction Word containing several primitive instructions, and dispatch the entire VLIW for parallel execution. These capabilities are exploited by compilers which generate code that has grouped together independent primitive instructions executable in parallel. The processors have relatively simple control logic because they do not perform any dynamic scheduling nor reordering of operations (as is the case in most contemporary superscalar processors).

VLIW has been described as a natural successor to RISC, because it moves complexity from the hardware to the compiler, allowing simpler, faster processors. As stated in Microprocessor Report (2/14/94):

"The objective of VLIW is to eliminate the complicated instruction scheduling and parallel dispatch that occurs in most modern microprocessors. In theory, a VLIW processor should be faster and less expensive than a comparable RISC chip."

The instruction set for a VLIW architecture tends to consist of simple instructions (RISC-like). The compiler must assemble many primitive operations into a single "instruction word" such that the multiple functional units are kept busy, which requires enough instruction-level parallelism (ILP) in a code sequence to fill the available operation slots. Such parallelism is uncovered by the compiler through scheduling code speculatively across basic blocks, performing software pipelining, reducing the number of operations executed, among others.

VLIW has been perceived as suffering from important limitations, such as the need for a powerful compiler, increased code size arising from aggressive scheduling policies, larger memory bandwidth and register-file bandwidth, limitations due to the lock-step operation, binary compatibility across implementations with varying number of functional units and latencies. In recent years, there has been significant progress regarding these issues, due to general advances in semiconductor technology as well as to VLIW-specific activities. For example, our tree-based VLIW architecture provides binary compatibility for VLIW implementations of varying width, and our VLIW compiler contains state-of-the-art parallelizing/optimizing algorithms.