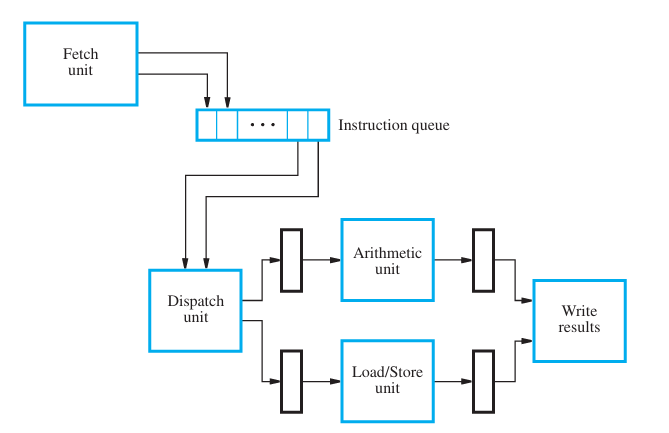
SUPERSCALAR PROCESSOR

-PRAVEEN KUMAR T V

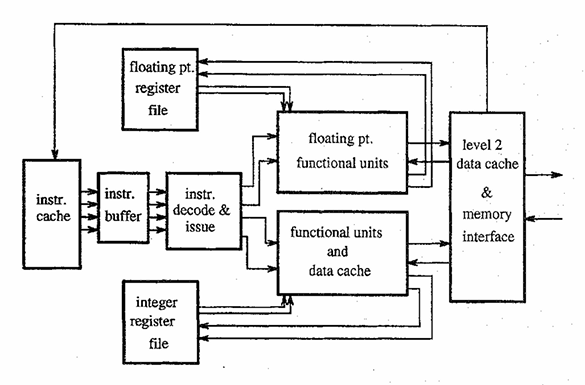
Superscalar is a process of executing multiple instructions concurrently and superscalar processor are capable of achieving an instruction throughput of more than one instruction per clock cycle. This is achieved by feeding the different pipelines through a number of execution units within the processor. The design techniques of superscalar processes normally comprise of out-of-order execution, instruction decoding and execution. The fetch unit fetches more than one instruction per clock cycle and places them in an instruction queue. The dispatch unit dispatches the instruction from the head of the queue and load them to the corresponding execution unit. The problem arises when the output of one instruction depends on the other instruction. It leads to out of order execution where the second instruction is completes its execution first before the first instruction. This can be solved by stalling the second instruction till the write operation of first instruction. Another approach is to maintain the results of executed instruction in a temporary register which may be used by some other instruction for its purpose. This process is called register renaming. Instructions can be executed in out of order, but are stored in the program order. Reorder buffer is used to reorder the instructions in program order. When an instruction completes in execution, it is present in the head of the buffer and retires from it after releasing the temporary registers. Hence, the instructions may complete out of order, but they are retired in program order.

Let’s see the examples of superscalar processors. MIPS R10000, ALPHA 21164, AMD K5 etc…

**MIPS R10000** fetches four instructions at a time from the instruction cache. The instructions are predecoded before they were put into the cache. The predecode generates four additional bits per instruction which helps to determine the instruction type immediately after the fetch from the cache. After being fetched, branch instructions were predicted. Following the instruction fetch, registers are renamed and they are dispatched to the appropriate instruction queue. The destination register is assigned an unused physical register from the free list and the map is updated to reflect the new logical to physical mapping. There are five functional units: an address adder, two integer ALU’s, a floating point multiplier/divider/square rooter and a floating point adder. The R10000 uses a reorder buffer mechanism for maintaining a precise state at the time of an exception condition. When an instruction is committed it frees up the old physical copy of the logical register it modifies.

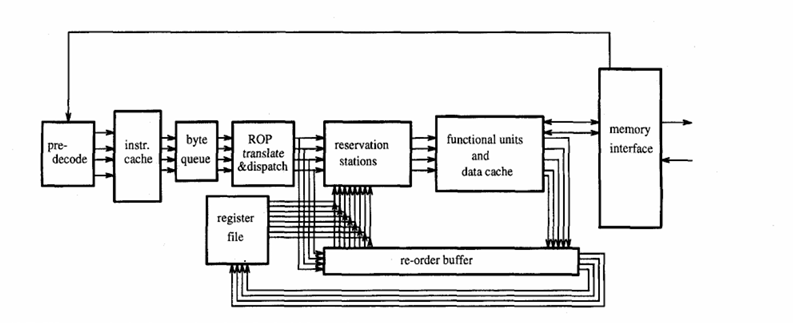
**ALPHA 21164** is an example of a simple superscalar processor that forgoes the advantages of dynamic instruction scheduling in favor of a high clock rate. Instructions are fetched from an 8 Kbytes instruction cache four at a time. These instructions are placed in one of two instruction buffers, each capable of holding four instructions. Instructions are issued from the buffers in program order, and a buffer must be completely emptied before the next buffer can be used. This restricts the instruction issue rate somewhat, but it greatly simplifies the necessary control logic.

Branches are predicted using a table that is associated with the instruction cache. There is a branch history entry for each instruction in the cache; each entry is a two bit counter. There can be only one predicted, and yet unresolved, branch in the machine at a time; instruction issue is stalled if another branch is encountered before a previously predicted branch has been resolved. There are four functional units: two integer ALU's, a floating point adder, and a floating point multiplier. The integer ALU's are not identical-only one can perform shifts and integer multiplies, the other is the only one that can evaluate branches.



**AMD K5** implements an instruction set that was not originally designed for fast, pipelined implementation. It is an example of a complex instruction set the Intel x86. The Intel x86 instruction set uses variable length instructions. This means, among other things, one instruction must be decoded (and its length established) before the beginning of the next can be found. Consequently, instructions are sequentially predecoded as they are placed into the instruction cache. There are five predecode bits per instruction byte. These bits indicate information such as whether the byte is the beginning or end of an instruction and identifies bytes holding opcodes and operands.

Instructions are then fetched from the instruction cache at a rate of up to 16 bytes per cycle. These instruction bytes are placed into a 16-element byte queue where they wait for dispatch. Instructions read operand data (if available) and are dispatched to functional unit reservation stations, up to 4 ROP's per cycle (corresponding to up to 4 x 86 instructions). Data can be in the register file, or can be held in the reorder buffer. When available this data is read and sent with the instruction to functional unit reservation stations. If operand data has not yet been computed, the instruction will wait for it in the reservation station.



There are six functional units: two integer ALU's, one floating point unit, two load/store units, and a branch unit. One of the integer ALU's capable of shifts, and the other can perform integer divides.

Here are some real world applications of superscalar processor:

General-purpose computing: Superscalar processors are commonly used in personal computers, workstations, and servers for tasks such as word processing, web browsing, and data analysis.

High-performance computing: Superscalar processors are used in supercomputers and other high-performance systems for tasks such as scientific simulations, weather forecasting, and financial modeling.

Gaming: Superscalar processors are used in gaming consoles and high-end gaming PCs to provide smooth and fast performance for games.

Embedded systems: Superscalar processors are used in embedded systems such as cars, cellphones, and drones for various tasks such as controlling sensors, interpreting data, etc.

AI/ML : Superscalar processors are increasingly used in AI/ML applications to perform complex computations and matrix operations to support the deep learning models, along with specialized hardware like TPUs/GPUs.

**Recent research about superscalar processor** is the collaborative design of compiler and microarchitecture of superscalar processor.

Compilers generate machine code optimized for the target microarchitecture. Optimization techniques include instruction scheduling, loop unrolling, register allocation, and software pipelining, among others.

**Intel Corporation**: Intel has been a leader in processor design for decades, and they continue to conduct extensive research on superscalar architectures.

**IBM Research**: IBM conducts research on superscalar processor architectures for their Power Systems servers and other computing systems. Their research aims to enhance performance, scalability, and reliability of superscalar designs for enterprise workloads.

**Google**: Google conducts research on superscalar processor architectures for their data centers and cloud computing infrastructure. They optimize superscalar designs for their workloads, which include web services, machine learning, and data analytics.

**NVIDIA Corporation**: NVIDIA is known for its GPU architectures, which often incorporate superscalar execution units. They conduct research on optimizing superscalar designs for parallel computing, AI, and graphics workloads.

Therefore, the concept of superscalar plays a vital role in processors, allowing to perform multitasking which leads to rapid response. Superscalar processors are the most advanced form of processors implemented in every domain of technology and making the process more feasible.