Superscalar Processors

Contents of Lecture:

- Superscalar Architecture
- Superscalar machine
- Superpipelining
- Superscalar vs. Superpipeline

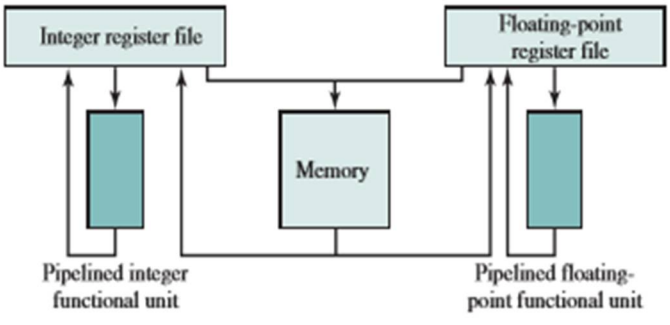
References for This Lecture:

- ✓ William Stallings, Computer Organization and Architecture Designing For Performance, 9th Edition, Chapter 16: *Instruction Level Parallelism and Superscalar Processors*

Superscalar Architecture:

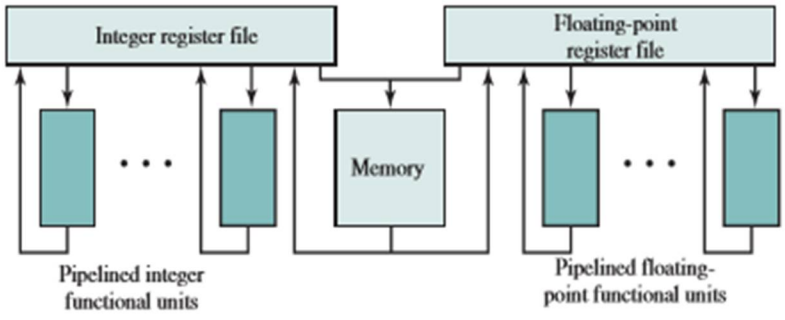
- **Superscalar** is a computer designed to improve the performance of the execution of scalar instructions.
  - ✓ A scalar is a variable that can hold only one atomic value at a time, e.g., an integer or a real (float).
  - ✓ A scalar architecture processes one data item at a time.
  - ✓ Examples of non-scalar variables:
    - Arrays
    - Matrices
    - Records

- In a superscalar architecture (SSA), several scalar instructions can be initiated simultaneously and executed independently.
- Pipelining allows also several instructions to be executed at the same time, but they have to be in different pipeline stages at a given moment.



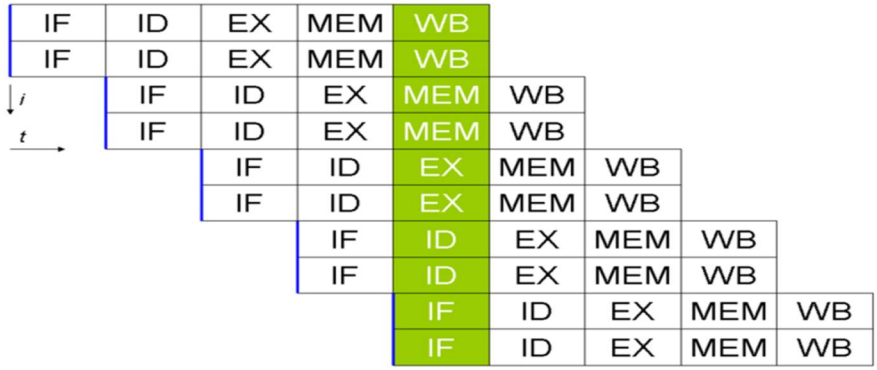
(a) Scalar organization

- SSA includes all features of pipelining but, in addition, there can be several instructions executing simultaneously in the same pipeline stage.



(b) Superscalar organization

❖ SSA introduces therefore a new level of parallelism, called **instruction-level parallelism**.



Superscalar machine:

❖ A Superscalar machine executes multiple independent instructions in parallel.
❖ They are pipelined as well:
  ✓ "Common" instructions (arithmetic, load/store, conditional branch) can be executed independently.
  ✓ Equally applicable to RISC & CISC, but more straightforward in RISC machines.
  ✓ The order of execution is usually assisted by the compiler.

Superpipelining:

❖ Superpipelining is based on dividing the stages of a pipeline into several sub-stages, and thus increasing the number of instructions which are handled by the pipeline at the same time.
❖ For example, by dividing each stage into two sub-stages, a pipeline can perform at twice the speed in the ideal situation.
  ✓ Many pipeline stages may perform tasks that require less than half a clock cycle.
  ✓ No duplication of hardware is needed for these stages.

- For a given architecture and the corresponding instruction set there is an optimal number of pipeline stages/sub-stages.

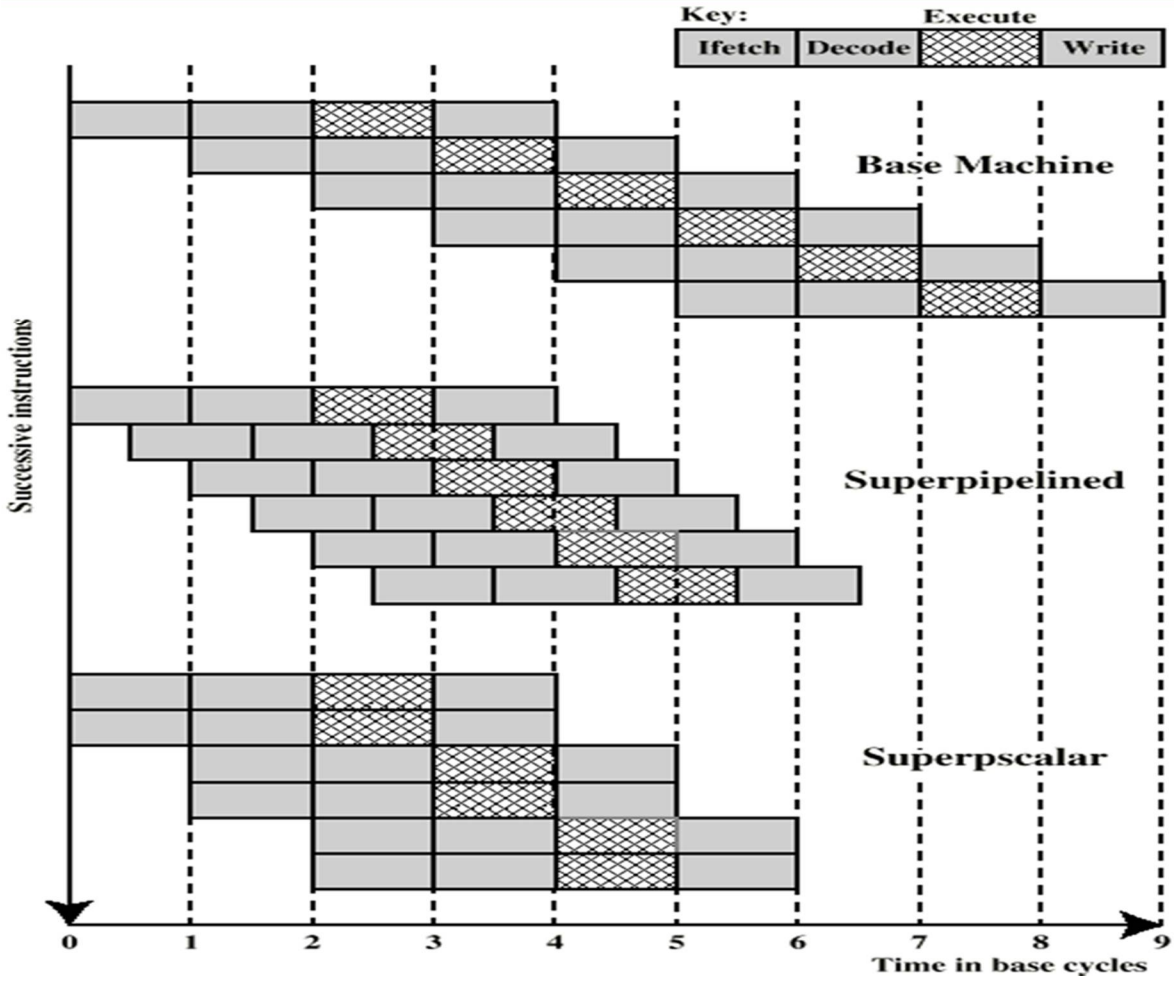
- Increasing the number of stages/sub-stages over this limit reduces the overall performance.
  - ✓ Overhead of data buffering between the stages.
  - ✓ Not all stages can be divided into (equal-length) sub-stages.
  - ✓ The hazards will be more difficult to resolved.
  - ✓ More complex hardware.

Superscalar vs. Superpipeline:

- Base machine: 4-stage pipeline
  - ✓ Instruction fetch
  - ✓ Operation decode
  - ✓ Operation execution
  - ✓ Result write back

- Superpipeline of degree 2
  - ✓ A sub-stage often takes half a clock cycle to finish.

- Superscalar of degree 2
  - ✓ Two instructions are executed concurrently in each pipeline stage.
  - ✓ Duplication of hardware is required by definition.



Limitations of Superscalar:

- Data dependency
- Procedural dependency
- Resource conflicts