# Module-3
# CSEN 3104
# Lecture 25
# 16/09/2019

Dr. Debranjan Sarkar

# Superscalar Architecture

# What is superscalar processor?

- CPI (cycles per instruction) vs. Processor clock speed (Show diagram)
- Conventional CISC processors have CPI more than 1 (say 1 to 20)
- It is tried to lower the CPI using innovative hardware approaches
- With the use of efficient pipeline, the average CPI of RISC instructions is 1 – 2
- Superscalar processor is a subclass of RISC processors
- Superscalar design, like pipelining, is an instance of instruction-level parallelism
- Superscalar Processors allow multiple instructions to be issued simultaneously during each cycle (unlike scalar RISC which issues only 1 instructions per cycle)
- Thus the effective CPI of a superscalar processor should be lower than that of a scalar RISC processor
- The clock rate of superscalar processors matches that of scalar RISC processors

# A problem on Performance of CPU

- A 50 MHz processor was used to execute a program with the following instruction mix and clock cycle counts:

| Instruction type | Instruction Count | Clock Cycle Count |
|---|---|---|
| Integer Arithmetic | 50,000 | 2 |
| Data Transfer | 70,000 | 3 |
| Floating Point Arithmetic | 25,000 | 1 |
| Branch | 4,000 | 2 |

- Calculate the (i) execution time of the program, (ii) effective CPI and (iii) the MIPS rating of the processor

# Superscalar processor

- Scalar processors execute one instruction per cycle.
- Only one instruction is issued per cycle and only one completion of instruction is expected from the pipeline per cycle
- In superscalar processor, multiple instructions are issued per cycle and multiple results are generated per cycle
- Superscalar processors exploit more instruction-level parallelism (ILP) in user programs
- Only independent instructions can be executed in parallel without causing a wait state

# Superscalar processor

- The amount of ILP varies widely depending on the type of code being executed
- It has been observed that the average value of ILP is around 2 for code without loop unrolling
- For these codes, there is not much of benefit by building a superscalar processor that can issue more than 3 instructions per cycle
- The instruction-issue degree in a superscalar processor has, therefore, been limited to 2 to 5 in practice

# Pipelining in Superscalar Processor

- Instruction issue rate is the number of instructions issued per cycle, also called the degree of a superscalar processor
- The scalar base processor may be considered as a superscalar processor of degree 1
- For an m-issue superscalar machine, m instructions are issued per cycle and the ILP should be m in order to fully utilize the pipeline
- Instruction decoding, and execution resources are increased, to form effectively m pipelines operating concurrently
- At some pipeline stages, the functional units may be shared by multiple pipelines
- Show figure of a dual-pipeline superscalar processor
- Here the processor can issue two instructions per cycle, if there is no resource conflict and no data dependence problem

# Pipelining in Superscalar Processor

- There are essentially two pipelines, both having 4 processing stages viz. fetch, decode, execute and store
- Each pipeline has its own fetch unit, decode unit, and store unit
- The two instruction streams, flowing through the two pipelines, are retrieved from a single source stream (I-cache)
- For simplicity, we may assume that each pipeline stage requires one cycle, except the execute stage which may require a variable number of cycles
- Using multiple functional units one can avoid structural hazards
- This concept is used very effectively in constructing superscalar processors
- Four functional units viz. multiplier, adder, logic unit and load unit are available for use in the execute stage
- These functional units are shared by two pipelines on a dynamic basis
- The multiplier itself has 3 pipeline stages, the adder has 2 stages and the others have only 1 stage each

# Pipelining in Superscalar Processor

- The two store units (S1 and S2) can be dynamically used by the two pipelines, depending on availability at a particular cycle
- There is a lookahead window with its own fetch and decoding logic
- This window is used for instruction lookahead in case out-of-order instruction issue is desired to achieve better pipeline throughput
- It requires complex logic to schedule multiple pipelines simultaneously, especially as the instructions are retrieved from the same source
- The aim is to avoid pipeline stalling and minimize pipeline idle time

# Superscalar performance

- Relative performance of a superscalar processor with that of a base scalar machine
- Let there be N independent instructions through the k-stage pipeline
- We estimate the ideal execution time in both the cases
- Time required by the base scalar machine is

  $T_{bs} = 1.k + (N - 1).1$ (cycles) $= k + N – 1$ cycles
- Let us consider an m-issue superscalar machine
- Time required by the superscalar machine is

  $T_{ss} = k + (N - m)/m$ (cycles)
- As the first m instructions are executed in k cycles through the m pipelines simultaneously and for the remaining (N-m) instructions, the rate at which the instructions are executed is m per cycle
- So, the ideal speedup of the superscalar machine over the base scalar machine is

  $T_{ss} / T_{bs} = m(N + k – 1)/ (N + m(k - 1))$
- As $N \rightarrow \infty$, the speedup limit tends to m

Thank you