# Computer Performance

Overview

# Performance Measurement หน่วยวัดประสิทธิภาพ

Instruction Per Sec
MIPS --> Million Instruction Per Second --> CPU Main Focus

Operation Per Sec
FLOPS --> Floating Point Operation Per Second --> Super Computer Main Focus

Cycle/Clock Per Instruction (CPI)
1 คำสั่งใช้สัญญาณนาฬิกาเท่าไหร่
ยิ่งสัญญาณ นาฬิกาน้อยยิ่งดี --> ไม่ใช้แล้ว เพราะ 1 cycle ทำได้มากกว่า 1 คำสั่ง

Instruction Per Cycle
1 Clock Cycle ทำงานได้กี่คำสั่ง --> ปัจจุบัน0

# Performance Effects

1. Clock Speed --> ความเร็วประมวลผลของ CPU หน่วนเป็น MHz, GHz จำนวนครั้งที่คอมดำเนินการได่ในแต่ละวิ | เอา Clock Speed มาคูณกัย Instruction in Program ได้
2. Instruction Execution Time --> โปรแกรมใช้เวลา Execute นานแค่ไหน / เอาผลรวมของ Clock Cycle ของทุกคำสั่งของโปรแกรมมารวม จะได้จำนวน Clock Cycle ที่โปรแกรมใช้งาน
3. Instructions in Program --> โปรแกรมสั้นยาวแค่ไหน คำสั่งเยอะแค่ไหน ออกแบบ algorithm มาดีหรือเปล่า

Benchmark/Standards --> ใช้โปรแกรมวัดความเร็ว CPU / โปรแกรมต่างๆ

# Clock Speed

Clock Speed --> การเปลี่ยนแปลงของค่า 1 กับ 0 อย่างคงที่ในช่วงเวลานึง (วินาที) หน่วยเป็น Hz
ยิ่งเพิ่ม Clock Speed ยิ่งกินไฟ (กินไฟตอนกินสถานะ)
ปกติจะกินไฟแบบ n^2
เมื่อไฟเพิ่มก็น้อนขึ้นต้องใช้ heat sink + พัดลม หรือ ชุดน้ำ อนาคตอาจเป็นแอร์

CPU-RAM Speed Difference
เอาเวลาไปรอ RAM หมดแก้ไงดี, แก้ได้ด้วย cache memory

# Pipeline

ทำให้ทำงานหลายคำสั่งพร้อมกันได้
เป็นการทำงานล่วงหน้า (Pre-Execute)

แบ่งการทำงานเป็นย่อยๆ เช่นการ execute 1 คำสั่งจะทำตามนี้ fetch --> decode --> execute
คำสั่งที่ 1 fetch อยู่ กำลังจะ decode จะ fetch คำสั่ง 2 ต่อเลย
คำสั่งที่ 1 execute คำสั่งที่สอง decode ให้ไป fetch คำสั่งที่ 2 มาเรื่อยๆ

เหมาะกับการทำงานคำสั่งที่ต่อๆ กันไปเท่านั้น (a แล้ว b แล้ว c แล้ว d...) จะเกิดประสิทธิภาพสูงสุด

# Pipeline Problem

ปัญหาที่อาจเกิดขึ้นได้

Control Hazard

- โยนลำดับก่อนหน้าทิ้ง แล้วไปเริ่มลำดับใหม่ ทำให้ไม่ได้ประโยชน์เรื่องประสิทธิภาพ
- เช่นทำคำสั่ง 4 ไปโหลดคำสั่ง 5, 6 ไว้ด้วย แต่ทำ 4 อยู่แล้วโดดไปทำคำสั่ง 10 ต่อ, 5, 6 ก็ต้องโยนทิ้ง

Data Hazard

- จังหวะการทำงานไม่สอดคล้องกัน
- สมมุติคำสั่งที่ 3 ต้องการประมวลผลตัวเลข แต่ตัวเลขนั้นต้องมาจากคำสั่งที่ 2
    - คำสั่งที่ 2 ยังไม่ execute แต่ 3 ต้องการข้อมูลแล้ว

Structural Hazard

- หลายๆ คำสั่งทำงานพร้อมกันแล้วใช้วงจรใน CPU ขัดแย้งกัน
- ex. คำสั่ง 2, 3 ต้องการวงจรบวกอันเดียวกัน

# Pipeline Problem Solutions

Branch Prediction

- เดาว่าคำสั่งต่อไปคืออะไร, มีทั้งเดาถูกและผิด

Delayed Execution

- ชะลอคำสั่งให้คำสั่งก่อนหน้า execute ให้เรียบร้อยแล้วค่อยทำคำสั่งต่อมา

# Parallel Processing

เพิ่มจำนวนหน่วยทำงานให้ทำงานพร้อมกัน ง่ายสุดคือการเพิ่ม CPU (ปัจจุบันเรียกว่า core)

ถ้าเพิ่มจำนวน core ก็ต้องเพิ่มวงจร ใช้พื้นที่มากขึ้น แก้ได้ด้วยการแชร์วงจรร่วมกัน ให้เกิดหน่วยทำงานร่วมกันในวงจรเดียวกัน พื้นที่เดียวกันนั้นเรียกว่า Thread

Thread --> ชุด register ที่เพิ่มเข้ามาใน CPU (1 ชุด register = 1 Thread)
ใช้ ALU และ Control Unit พร้อมกันโดยอาศัยจังหวะไม่ตรงกัน

- ถ้า Thread ใช้วงจรเดียวกัน จะแก้โดยการจัดคิวโดยใช้ Control Unit
- 1 Core ไม่เกิน 2 Thread

Multi Thread

- Virtual CPU --> Executing Status, Registers
- Shared CPU Components --> ALU, Control Unit

Multi-Processors --> 1 เครื่องมี CPU หลายตัว

Multi-Computer --> มีคอมพิวเตอร์หลายตัวในการประมวลผล (Multiple CPU-Memory)