INTRODUCTION TO MEMORY SYSTEM (II)

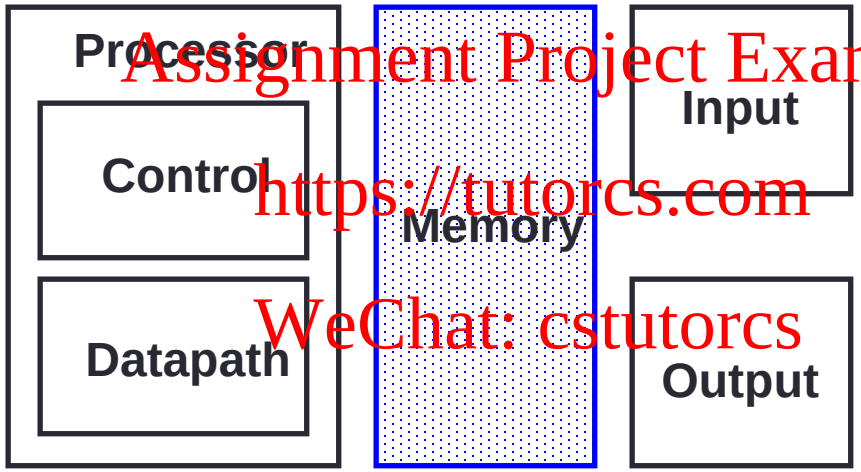
Lecturer: Hui Annie Guo
h.guo@unsw.edu.au
K17-501F

Lecture overview

- **Topics**
  - **Memory hierarchy**
- **Suggested reading**
  - **H&P Chapter 5.2**

The big picture:

- **Five classic components of a computer**



Memory hierarchy

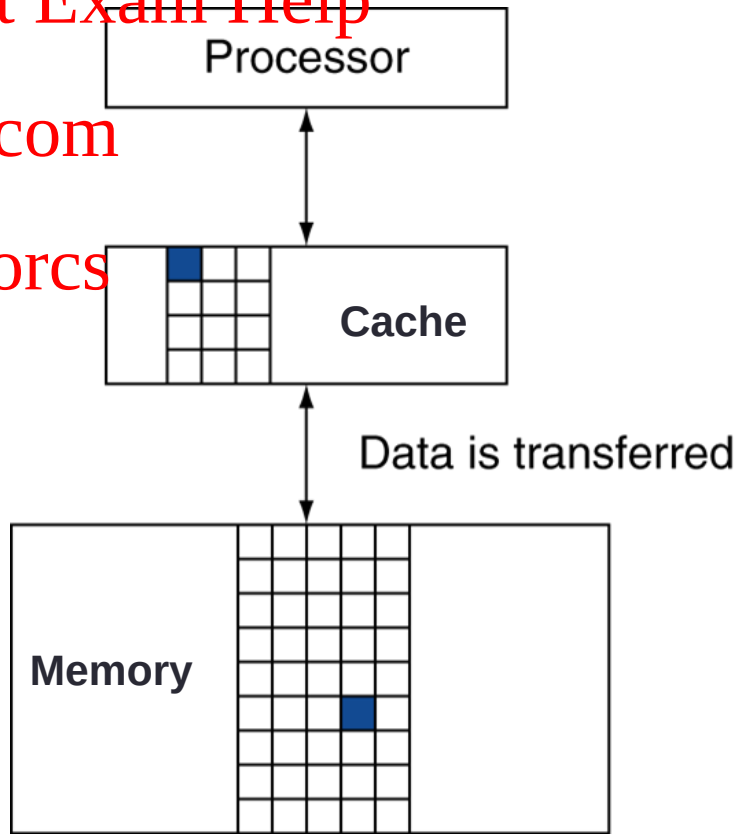
- **Memory hierarchy provides an illusion that a large, fast, cheap memory is available**
  - **How?**
  - **Why?**

How?

- Store everything on disk
- Copy recently required data from disk to smaller DRAM memory
  - main memory
- Copy more recently accessed (and nearby) items from DRAM to even smaller SRAM memory
  - Cache
- See example in the next slide

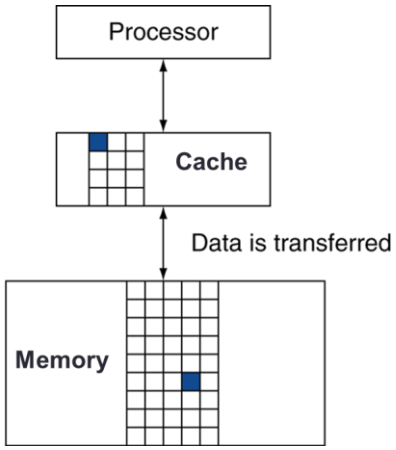
Memory hierarchy (cont.)

- **A typical example – two-level memory hierarchy**
  - **Processor accesses cache for data**
  - **Cache is fast but small**
  - **Memory is larger but slow**



Memory hierarchy (cont.)

- **Cache hit**
  - **If accessed data is present in cache**
- **Cache miss**
  - **If accessed data is not in cache**
  - **Solution: copy the data block from memory to the cache**
    - **The accessed data is then available in cache**

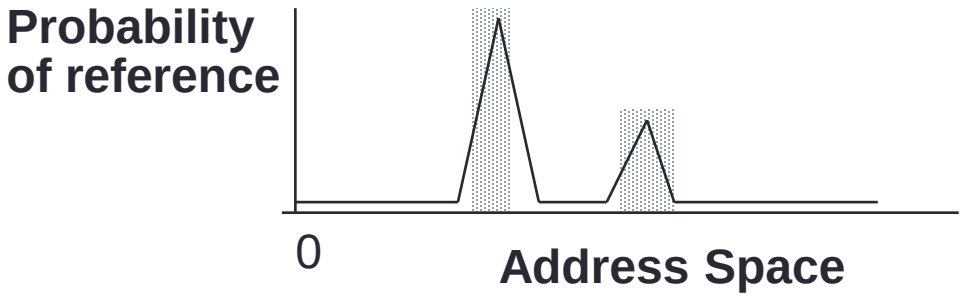


Memory hierarchy (cont.)

- **Cache hit rate**
  - **hits/accesses**
- **Hit time**
  - **Time taken to access cache**
- **Cache miss rate**
  - **misses/accesses**
    = 1 – hit rate
- **Miss penalty**
  - **Time taken to copy data block from memory to cache**

Why?

- **Principle of locality**
  - **Programs tend to access relatively small portions of the address space over a small period of time.**

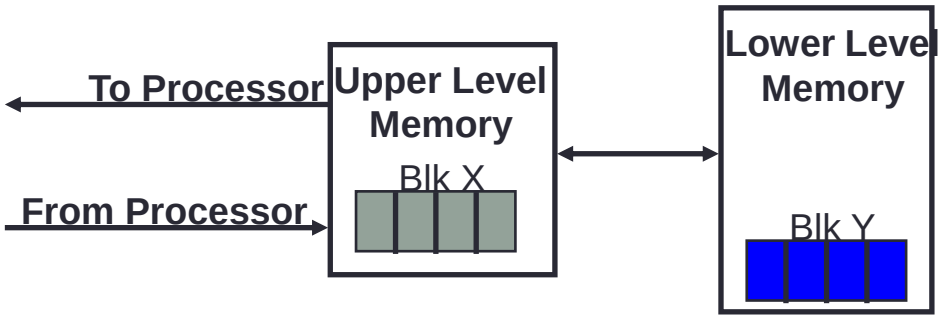


Principle of locality

- **Two types of localities:**
  - **Temporal Locality (locality in time)**
    - **If an item is referenced, it will tend to be referenced again soon.**
  - **Spatial Locality (locality in space)**
    - **If an item is referenced, items whose addresses close by tend to be referenced soon thereafter.**

Principle of locality (cont.)

- **Apply principle of locality on memory hierarchy, we can achieve "a large, fast, cheap memory"**
  - **Temporal locality**
    - **Keep most recently accessed data items closer to the processor**
  - **Spatial locality**
    - **Move multiple neighbourhood data items together (a data block) to cache**



Average access time

- **Average memory access time (AMAT)**
  - **AMAT = Hit time + Miss rate × Miss penalty**

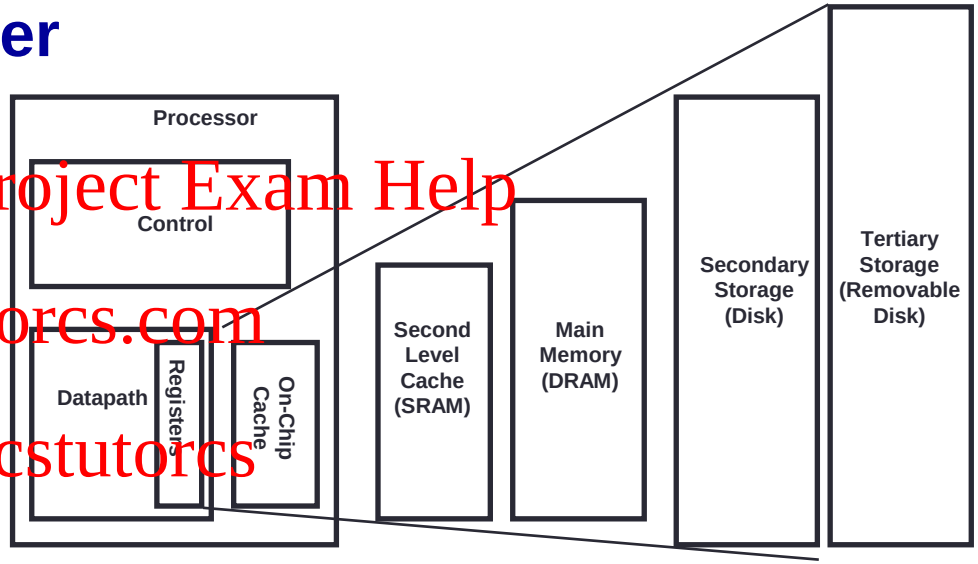
- **Example**

  - **CPU with 0.2 ns clock, hit time = 1 cycle, miss penalty = 20 cycles, I-cache miss rate = 5%**

  - **What is AMAT of instruction memory?**
    - **AMAT = (1 + 0.05 × 20)*0.2 = 0.4ns**
    - **2 cycles per instruction**

Overview of memory system hierarchy

- registers ↔ memory
  - by compiler/programmer
- cache ↔ memory
  - by the hardware
- memory ↔ disks
  - by hardware and operating system (virtual memory)
  - by programmer



We want AMAT as small as possible!

About Tutorial Solutions

About Group Project