Replacement Policy: Which block to replace from the set?

- Direct mapped: no choice
- Associative: evict *least recently used* (LRU)
  - difficult/costly with increasing associativity
- Alternative: random replacement
  - simple and fast, not too much worse than LRU
- Combination LRU+Random
Multilevel Caches

- primary (level-1) cache attached to CPU
  - small, but fast (usually instruction speed)
- level-2 cache services misses from L1 cache
  - larger, slower, but still faster than main memory
- main memory services L2 cache misses
- some high-end systems have even more levels
- on-chip vs. off-chip placement of cache
- If L2 contains the requested data item, Miss penalty for L1 will be the access time for L2. The Miss Penalty will be higher if data not in L2 either.
Multilevel Cache Example

- assume
  - CPU base CPI = 1, clock rate = 4GHz
  - miss rate = 2%
  - main memory access time = 100ns
- with just primary cache
  - miss penalty = 100ns/0.25ns = 400 cycles
  - effective CPI = 1 + 0.02 * 400 = 9
Multilevel Cache Example

- now add L2 cache
  - access time = 5ns
  - global miss rate to main memory = 0.5%
- primary miss with L2 hit
  - penalty = 5ns/0.25ns = 20 cycles
- primary miss with L2 miss
  - extra penalty = 400 cycles

\[ \text{CPI} = 1 + 0.02 \times 20 + 0.005 \times 400 = 3.4 \]

Processor with L2 cache is \( \frac{9}{3.4} = 2.6 \) times faster!
I/O Devices

- I/O devices can be characterized by
  - behaviour: input, output, storage
  - partner: human or machine
  - data rate: bytes/sec, transfers/sec

- I/O device vs. I/O controller
  - controller: processor that controls device
I/O System Characteristics

- dependability important – especially for storage
- performance
  - latency (response time)
  - throughput (bandwidth)
- desktops & embedded systems
  - mainly interested in response time & device diversity
- servers
  - mainly interested in throughput & expandability
Interconnecting Components

- need interconnection between
  - CPU, memory, I/O controllers
- Bus: shared communication channel
  - parallel set of wires for data and synchronization
- performance limited by physical factors
  - wire length, number of connections
I/O Bus

- Processor
- Cache
- Memory-I/O Interconnect
  - Main memory
  - I/O controller
    - Disk
  - I/O controller
    - Graphics output
  - I/O controller
    - Network
Bus Types

- processor-memory bus
  - Connects processor to main-memory
  - short, high-speed
  - design matches memory organization
- I/O bus: connect different devices together
  - longer, multiple connections
  - interoperability standards
- I/O Bus does not directly connect to Memory. **Bridge** connects memory & I/O bus
Bus Signals

- data lines
  - carry addresses and data
  - Multiplexed (shared) or separate
- control lines: carries info about the data being carried
  - indicate data types, synchronize transactions
- Serial Communication: One bit at a time on one line
- Parallel Comm: Multiple bits at a time, sent on many parallel lines (still one bit on one line at a time)
Bus Synchronization

- The information being sent back and forth on a bus needs to be synchronized so that the component making a request knows when it is receiving the response.
- **Synchronous buses** include a clock in the control lines, and follow a fixed protocol for communicating.
- For example: Memory will receive the address and read on the first clock cycle, then memory is required to respond with the data word in question on the sixth cycle.
- Every device on the bus must run at the same clock rate
- Clock skew is an issue, limits our buses length
- **Asynchronous Buses** are not clocked, instead agree upon some kind of a handshaking protocol: *Send out signal, wait for acknowledgement before sending next*
- Can support any device that follows protocol, doesn't need clocks, Can be lengthened as needed
I/O Management

- I/O access mediated by operating system
- multiple programs share resources
  - need protection and scheduling
- I/O causes asynchronous interrupts
- I/O programming is fiddly
  - OS provides simpler abstractions
I/O Programming

- memory mapped I/O
  - I/O “registers” are mapped into memory
  - address decoder distinguishes between them
  - OS uses memory protection to protect from direct user access

- I/O instructions
  - special instructions to access I/O operations
  - can only be executed in privileged mode
I/O Interaction

- polling: periodically check I/O status register
  - if device ready, perform operation
  - if error, take action
- interrupts: I/O controller interrupts main CPU
  - asynchronous instruction execution
  - interrupt handler must preserve state – costly
  - can give priorities to interrupts
  - at certain times: mask low-priority interrupts
I/O Data Transfer

- Traditional
  - CPU transfers data from/to memory
  - time consuming for high-speed devices
- Direct Memory Access (DMA)
  - OS provides memory address to device
  - I/O controller autonomously transfers data
  - interrupt/polling signal on completion or error
DMA/Cache Interaction

- DMA writing to cached memory block
  - cached copy becomes stale
- write-back cache with dirty block
  - DMA would read stale data
- need to ensure cache coherence
  - flush blocks from cache if they will be used for DMA
  - or use non-cacheable memory locations for I/O