Pipelining & Caches

Scott Beamer (cs61c-ta)

Structural Hazards

Component	Conflicting Stages	Solution

Control Hazards

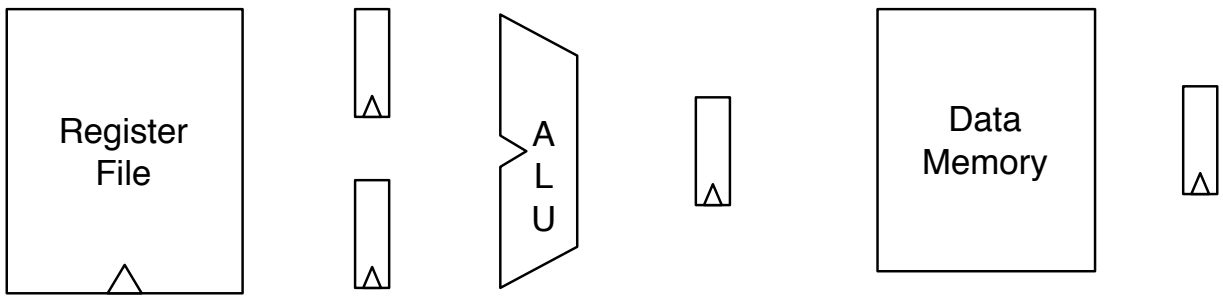
Instruction	Solution

Data Hazards

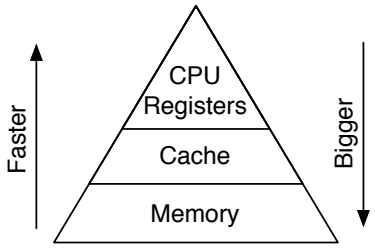
Instruction	Needed	Available	Solution

Pipelined Datapath

- Fill in the basic datapath with the forwarding paths
- It only needs to handle R-type instructions and don't worry about the IF stage



Cache Summary



- Memory is slow relative to the CPU, so caches are used to speed up memory accesses
- *Memory Hierarchy*: A system in which a subset of memory is stored at each level. Smaller levels contain less data, but are faster to access.
- Caches try to maximize usefulness by usually exploiting spatial and temporal locality
  - *Spatial Locality*: memory near where we just accessed is more likely to accessed
  - *Temporal Locality*: memory that was just accessed is more likely to be accessed again

Direct-Mapped Cache Addressing

- Break address into Tag, Index, and Offset fields (TIO)

Tag	Index	Offset

- Cache Size = (number of rows) x (row width)
- Capacity Equation - capacity is 2^N bytes, blocks size is 2^B bytes, and there are 2^R rows
  $2^N = 2^B \times 2^R = 2^{(B+R)}$
- Each row contains data, tag bits, valid bit (for now)

Cache Problems

- Fill in the gaps in the following table

Address Bits	Cache Size	Block Size	Tag Bits	Index Bits	Offset Bits	Bits per row
16	4KB	4B	4	10	2	37
16	16KB	8B	2	11	3	67
32	8KB	8B	19	10	3	84
32	32KB	16B	17	11	4	146
32	64KB	16B	16	12	4	145
32	512KB	32B	13	14	5	270
64	1024KB	64B	44	14	6	557
64	2048KB	128B	43	14	7	1068

Cache Design

- Consider the following scenarios and what effect they would have on the cache

Scenario	Effect
no tag bits	cache is the same size as memory
no index bits	fully associative
no offset bits	each block contains only 1 element