Unit 17

Improving Performance

Caching and Pipelining

Improving Performance

- We want to improve the performance of our computation
- Question: What are we referring to when we say "performance"?
  - _________________
  - _________________
  - _________________
- We will primarily consider ___________ in this discussion

How Do We Measure Speed

- **Fundamental Measurement**: _________
  - Absolute time from __________ to ___________
  - To compare two alternative systems (HW + SW) and their performance, start a timer when you begin a task and stop it when the task ends
  - Do this for both systems and compare the resulting times
- We call this the __________ of the system and it works great from the perspective of the _______________ task
  - If system A completes the task in 2 seconds and system B requires 3 seconds, then system A is clearly superior
- But when we dig deeper and realize that the single, overall task is likely made of _________ small tasks, we can consider more than just latency

Performance Depends on View Point?!

- What's faster to get from point A to point B?
  - A 747 Jumbo Airliner
  - An F-22 supersonic, fighter jet
- If only _______________ to get from point A to point B, then the ___________
  - This is known as _______________ [units of seconds]
  - Time from the start of an operation until it completes
- If _______________ to get from point A to point B, the _____ looks much better
  - This is known as _______________ [jobs/second]
- The ***overall*** execution time (latency) may best be improved by _______________ throughput and not the latency of individual tasks

Improving Latency and Throughput

CACHING AND PIPELINING

Hardware Techniques

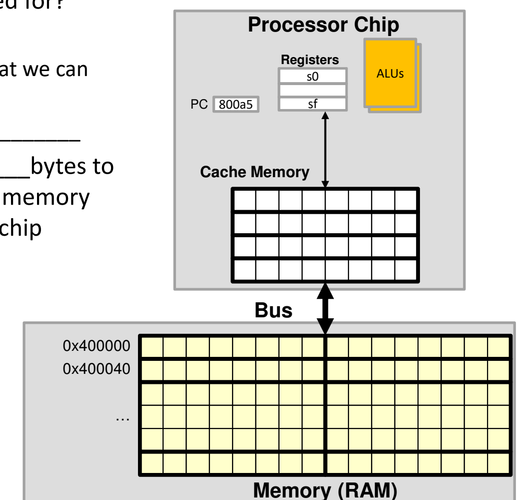
- We can add hardware or reorganize our hardware to improve throughput and latency of individual tasks in an effort to reduce the total latency (time) to finish the overall task
- We will look at two examples:
  - Caching: Improves _______________
  - Pipelining: Improves _______________

Caching

- **Cache (def.)** – "*to store away in hiding or for future use*"
- Primary idea
  - The _____________ you access or use something you expend the ________ amount of time to get it
  - However, store it someplace (i.e. in a cache) you can get it more _____________ the next time you need it
  - The next time you need something check if it is in the cache first
  - If it is in the cache, you can get it quickly; else go get it expending the full amount of time (but then _________ it in the cache)
- Examples:
  - ____________________
  - ____________________
  - ____________________

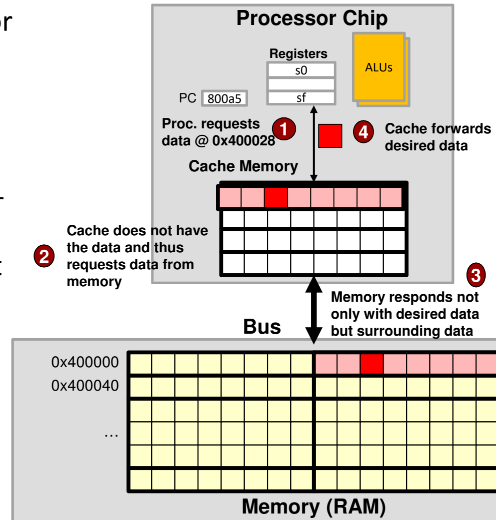
Cache Overview

- Remember what register are used for?
  - Quick access to copies of data
  - Only a _______ (32 or 64) so that we can access really quickly
  - Controlled by the _________________
- Cache memory is a small-ish, (____bytes to a few _____bytes) "_________" memory usually built onto the processor chip
- Will hold ____________ of the latest data & instructions accessed by the processor
- Managed by the ____
  - ___________ to the software



Cache Operation (1)

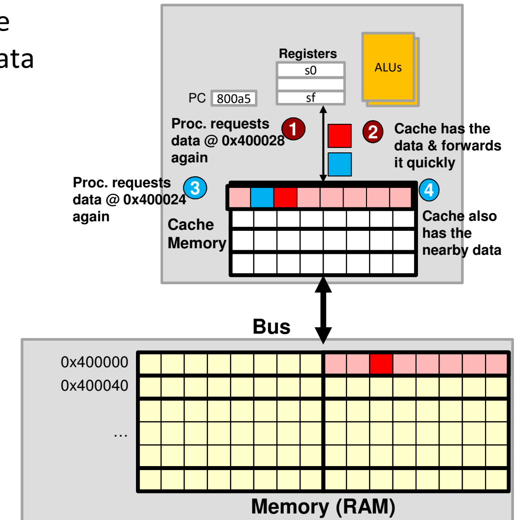
- When processor wants data or instructions it always _________ in the cache first
- If it is there, ______ access
- If not, get it from __________
- Memory will also supply ______________ data since it is likely to be needed soon
  - Why?
  - Things like ______ & ______ (instructions) are commonly accessed sequentially



Cache Operation (2)

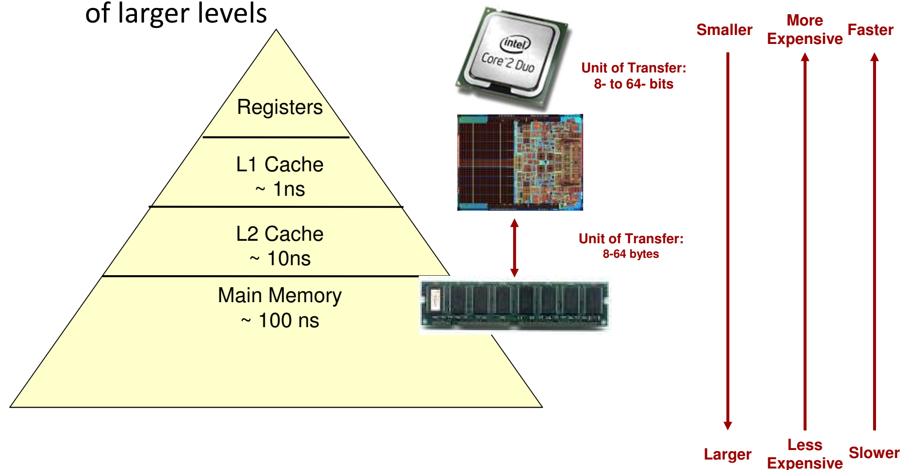
- When processor asks for the data again or for the next data value in the array (or instruction of the code) the cache will likely have it
- Questions?

Main point: Caching reduces the latency of memory accesses which improves overall program performance.



Memory Hierarchy & Caching

- Use several levels of faster and faster memory to hide _______ of larger levels

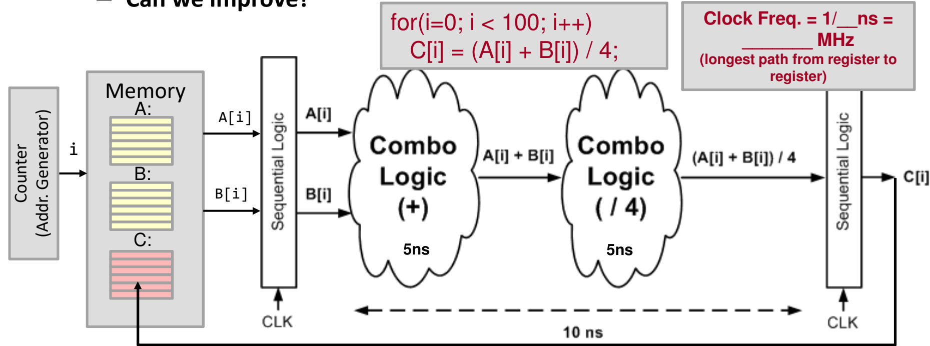


Pipelining

- We'll now look at a hardware technique called **pipelining** to improve _______________
- The key idea is to __________ the processing of multiple "items" (either data or instructions)

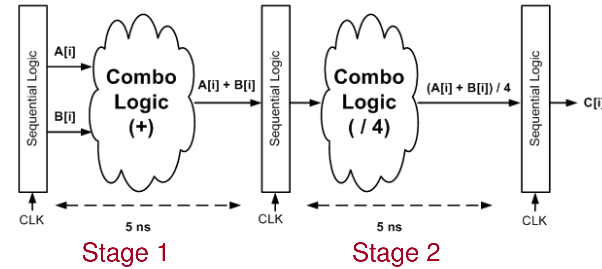
Example

- Suppose you are asked to build dedicated hardware to perform some operation on all 100 elements of some arrays
- Suppose the operation (A[i]+B[i])/4 takes 10 ns to perform
- How long would it take to process the entire arrays: _______ ns
  - **Can we improve?**



Pipelining Example

- Pipelining refers to insertion of registers to split combinational logic into smaller stages that can be overlapped in time (i.e. create an assembly line)



	Stage 1	Stage 2
Clock Cycle 0	A[0] + B[0]	
Clock Cycle 1	A[1] + B[1]	(A[0] + B[0]) / 4
Clock Cycle 2	A[2] + B[2]	(A[1] + B[1]) / 4

for(i=0; i < 100; i++)
C[i] = (A[i] + B[i]) / 4;

Time for 0th elements to complete: __________

Time between each of the remaining 99 element completing: ________

Total: _____________

Define:

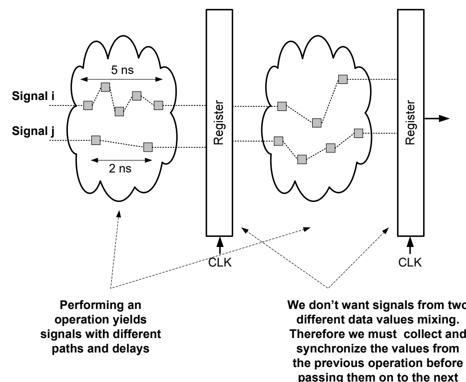
$$speedup = \frac{Original\ time}{Improved\ Time}$$

$$speedup = \frac{1000ns}{____ns} \approx ___$$

Clock freq: = _________

Need for Registers

- Provides separation between combinational functions
  - Without registers, fast signals could "catch-up" to data values in the next operation stage



Pipelining Example

- By adding more pipelined stages we can improve throughput
- Have we affected the latency of processing individual elements? ____________
- Questions/Issues?
  - ____________ stage delays
  - ___________ of registers (Not free to split stages)
    - This limits how much we can split our logic

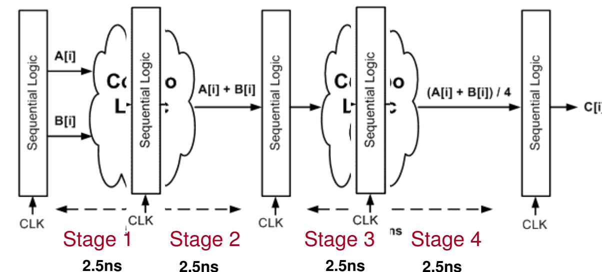
for(i=0; i < 100; i++)
C[i] = (A[i] + B[i]) / 4;

Time for 0th elements to complete: __________

Time between each of the remaining 99 element completing: ________

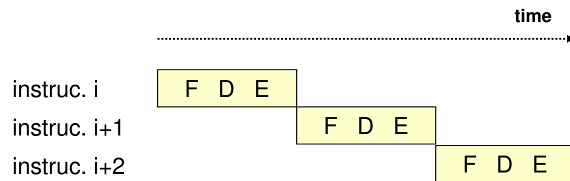
Total: _____________

$$speedup = \frac{1000ns}{257.5ns} \approx 4x$$



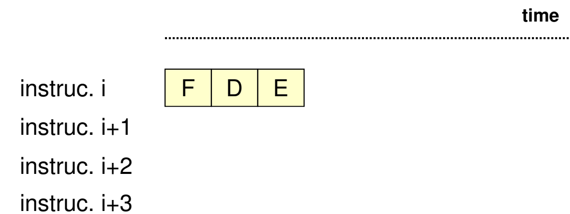
Non-Pipelined Processors

- Currently we know our processors execute software 1 instruction at a time
- 3 steps/stages of work for each instruction are:
  - ___________
  - ___________
  - ___________



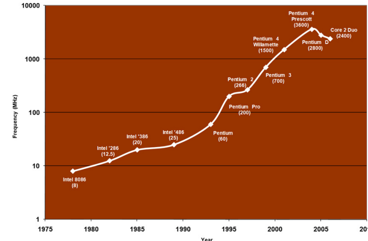
Pipelined Processors

- By breaking our processor hardware for instruction execution into stages we can overlap these stages of work
- Latency for a single instruction is the _____________
- Overall throughput, and thus total latency, are greatly improved



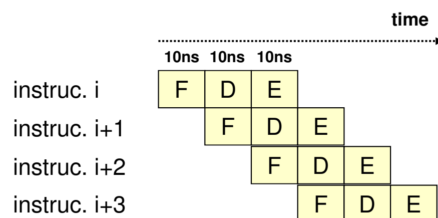
More and More Stages

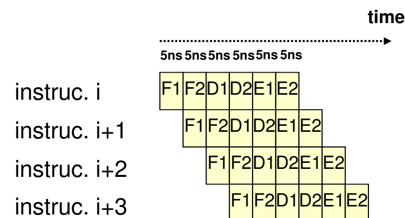
- We can break the basic stages of work into substages to get better performance
- **In doing so our clock period goes ______; frequency goes _____**
- All kinds of interesting issues come up though when we overlap instructions and are discussed in future CENG courses



Clock freq. = 1/10ns = 100MHz

Clock freq. = 1/__ns = ___MHz





Summary

- By investing extra hardware we can improve the overall latency of computation
- Measures of performance:
  - Latency is start to finish time
  - Throughput is tasks completed per unit time (measure of parallelism)
- Caching reduces latency by holding data we will use in the future in quickly accessible memory
- Pipelining improves throughput by overlapping processing of multiple items (i.e. an assembly line)