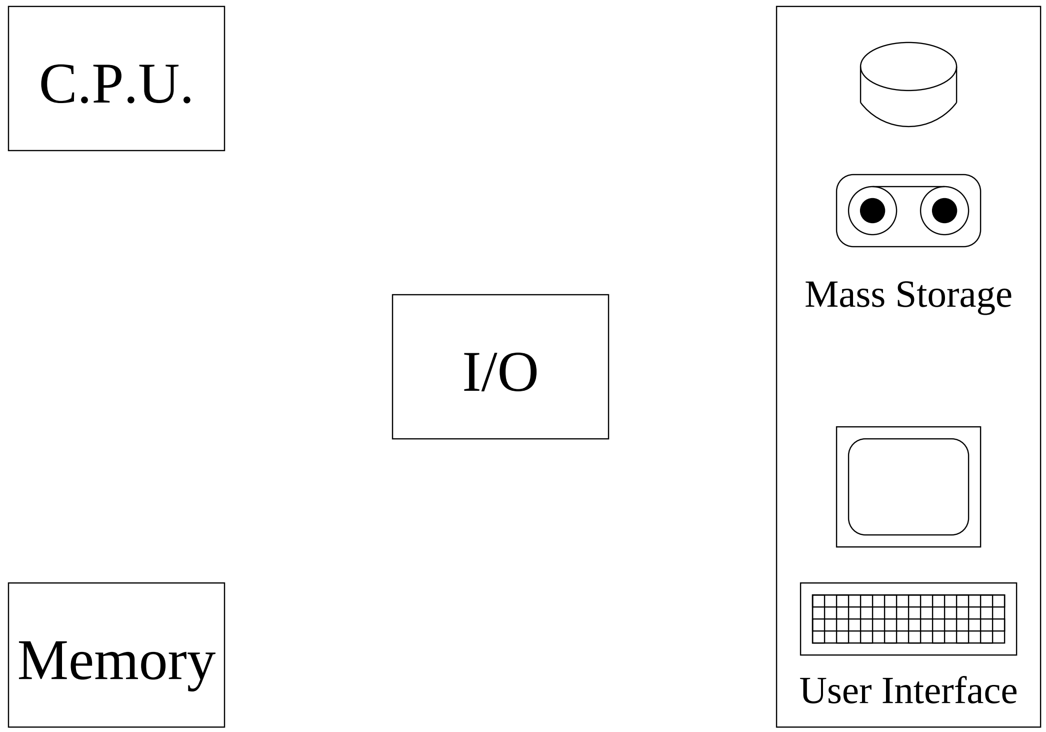
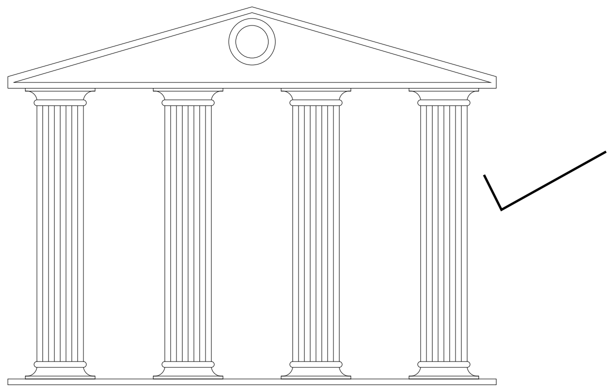
Review of Serial Computer Architecture

Computer components:



Review of Serial Computer Architecture

In this course we are concerned with systems architecture:



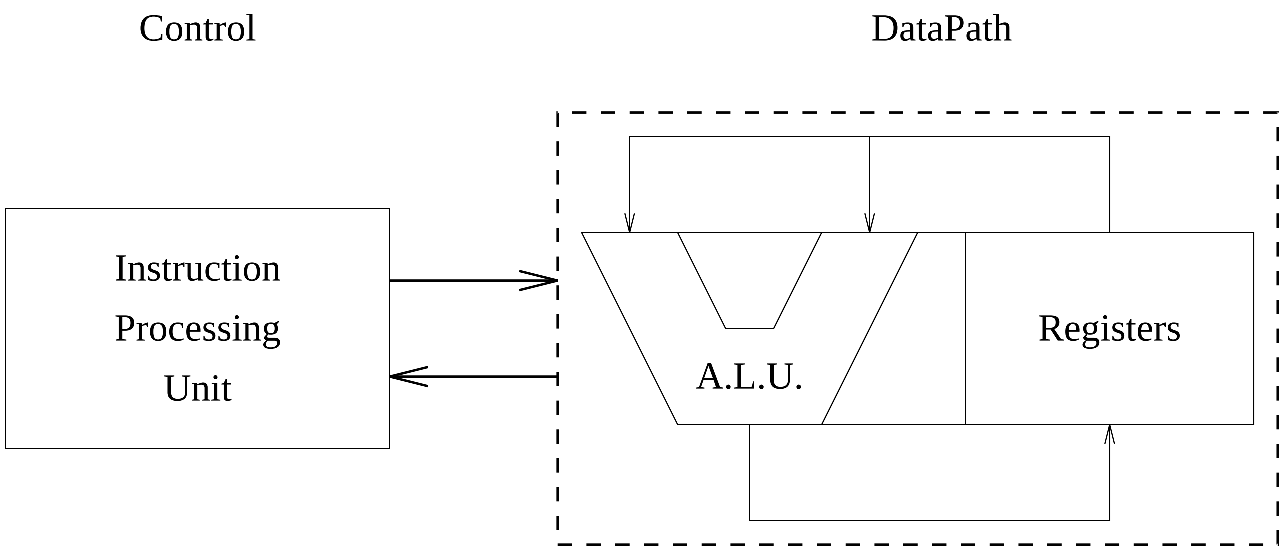
- The arrangement of computer components
- The mechanisms for connecting these components together
- How the problems change as we have multiple copies of certain components

We are less concerned with the components themselves:

- We shall not be discussing the design of a better multiplier.

Review of Serial Computer Architecture

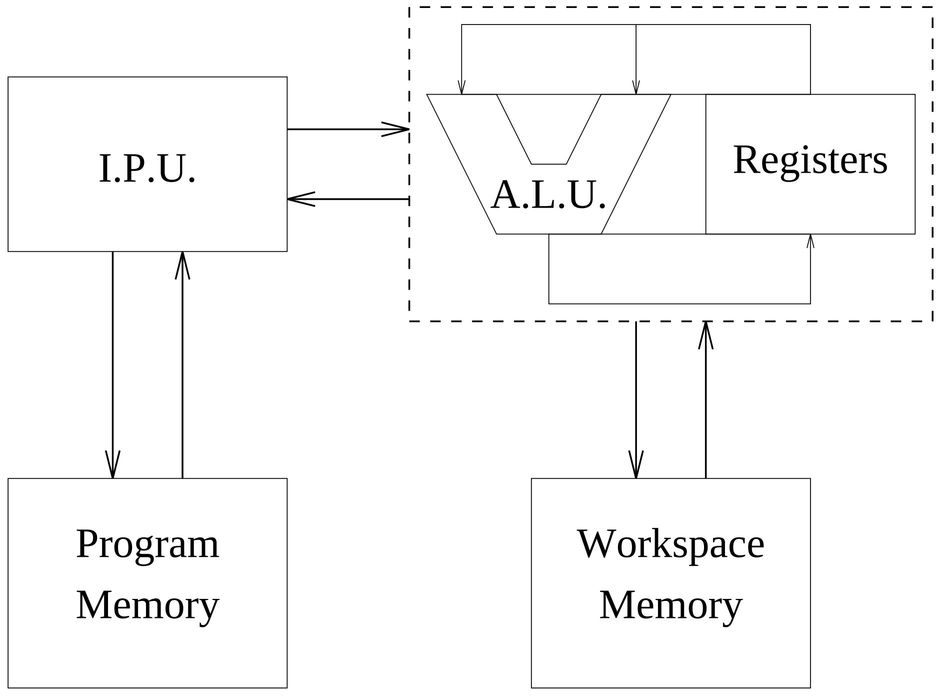
- Central Processing Unit (C.P.U.)
  - Arithmetic Logic Unit (A.L.U.)
  - Instruction Processing Unit (I.P.U.)
  - Registers



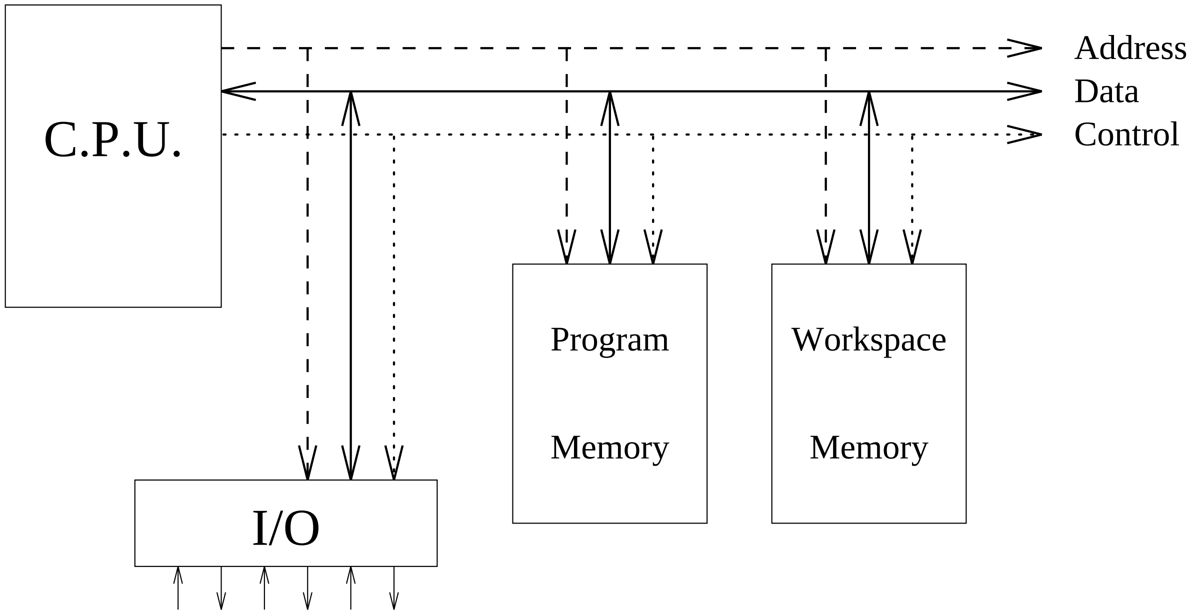
Review of Serial Computer Architecture

- Support for Central Processing Unit

  Both the datapath and the control system require external storage.



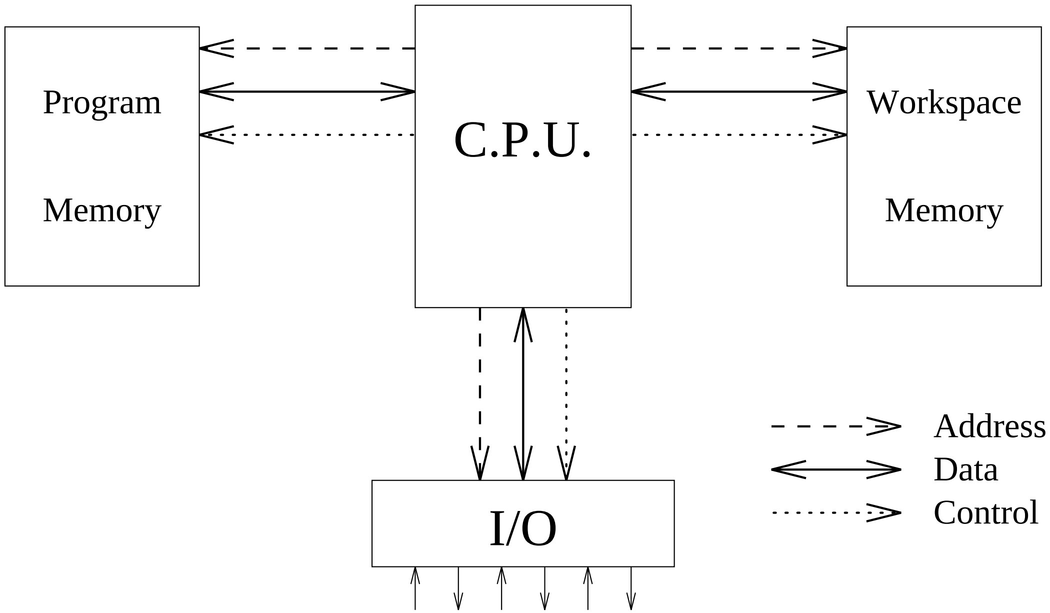
Princeton Architecture for Serial Computers



- The Princeton Architecture uses a common set of buses for all components.

Review of Serial Computer Architecture

Harvard Architecture for Serial Computers



- The Harvard Architecture uses different sets of buses for different components.[1]

[1] Frequently the I/O will share the same set of buses as the Workspace Memory

Princeton Benefits

- Much lower C.P.U. pin count.
- Simpler programming model

  The common bus architecture makes the external components appear to the C.P.U. as a single memory space.[2]
- DMA (Direct memory access by I/O units) can be used for loading of programs and/or data.

Harvard Benefits

- Simpler internal architecture.

  Princeton C.P.U. will frequently contain several internal buses which are multiplexed onto a common external bus.
- Possibility for increased *concurrency*.

[2] All locations can be read or written. Some locations return different values when read than those supplied during writing - this is a problem for the programmer not the C.P.U.

Sequential Nature of the Sequential Computer

Taking a simplified view of instruction execution:

- Fetch Instruction
- Decode Instruction
- Fetch Data
- Execute Instruction
- Store Result

Repeat once for each instruction.

Each operation depends on the completion of the previous operation.

- The process appears inherently sequential.

Concurrency in Serial Computers

Instruction Pre-Fetch in a Princeton sequential computer:

Fetch Instruction		
Decode Instruction		
Fetch Data		
Execute Instruction	Fetch Instruction	
Store Result	Decode Instruction	
	Fetch Data	
	Execute Instruction	Fetch Instruction
	Store Result	Decode Instruction
		Fetch Data
		Execute Instruction
		Store Result

- A common optimization amongst more powerful microprocessors.[3]
- The parallelism is limited by the use of global address and data buses.

[3] Note that it may be necessary to discard the pre-fetched instruction following a successful branch.

Concurrency in Serial Computers

Simple Pipeline Execution in a Harvard sequential computer:

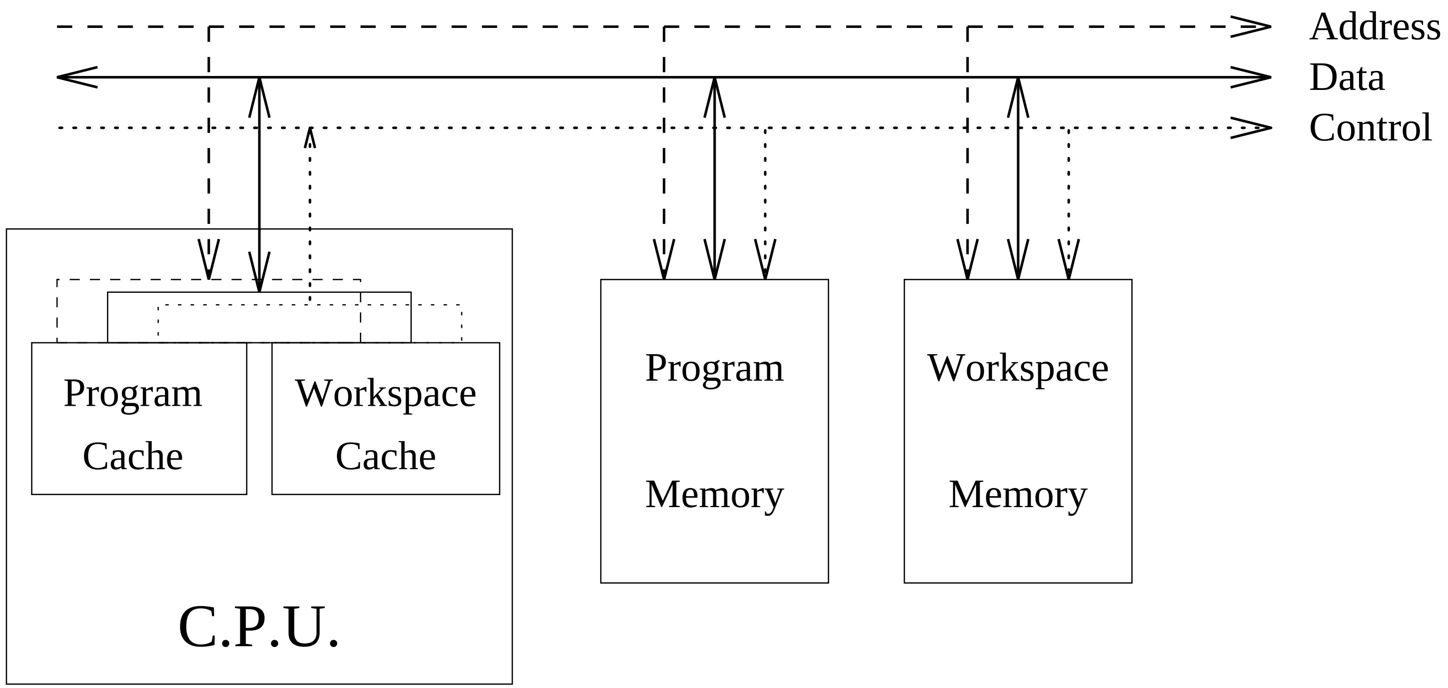
Fetch Instruction			
Decode Instruction	Fetch Instruction		
Fetch Data	Decode Instruction		
Execute Instruction	Fetch Data		
Store Result	Execute Instruction	Fetch Instruction	
	Store Result	Decode Instruction	Fetch Instruction
		Fetch Data	Decode Instruction
		Execute Instruction	Fetch Data
		Store Result	Execute Instruction
			Store Result

- The parallelism is limited by the availability of the workspace address and data buses.[4]

[4]Note that data dependencies must be spotted by the compiler to prevent the fetching of old data.

Concurrency in Serial Computers

Certain modern Princeton machines have a Pseudo-Harvard architecture:



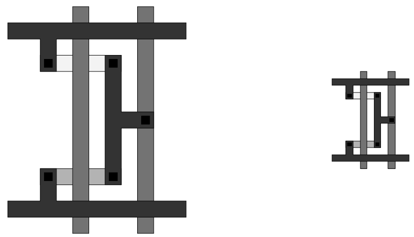
- The separate caches appear to allow access to workspace and program memory in the same cycle.

Performance Limits of Serial Computers

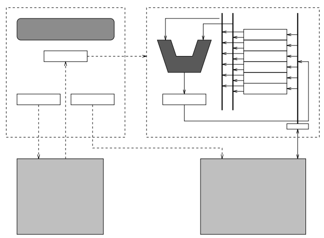
Computer power has been increasing steadily.[5]

What factors enable this increase in power?

- Technology



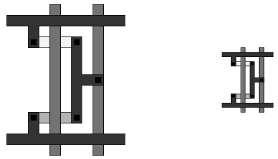
- Architecture



[5] Approx – factor of 10 increase in arithmetic speed every 5 years

Performance Limits of Serial Computers

Technology



- Transistor speed

  Direct effect on computer power.

- Transistors size

  Direct effect by reducing wire length and reduction in slow off-chip wiring.

Limits

- The rate of increase is likely to level-off due to the hard limits of molecular size and light speed.

Performance Limits of Serial Computers

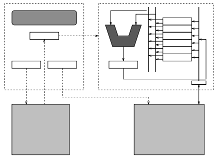
Technology Limits

Just how close are these fundamental limits?

- We have already reached a time when designers must consider the propagation delays of signals being limited by the speed of light - signals take about 6ns to travel 1 meter along 50 ohm co-axial cable.
- As we shrink our computers to take account of this, we require freon cooling to dissipate the heat generated.
- Meanwhile we find device physicists talking about the impact of atom spacing on their ability to make smaller (and faster) transistors.

We must look to architecture to realize our goals for ever increasing computer power.

Architectural Enhancements



Variety of architectural enhancements:

- Wider datapaths (8,16,32,64bits) the simplest form of increased concurrency.
- RISC machines
- Memory Caches
- Hardware floating point units
- Increased use of simple concurrency

Each architectural enhancement requires extra design effort to overcome a specific bottleneck. The benefit in each case is limited, as the system bottleneck moves from one unit to another.

Performance Limits of Serial Computers

The Problem

- Each enhancement has limited effect.
- Many enhancements complicate the system.

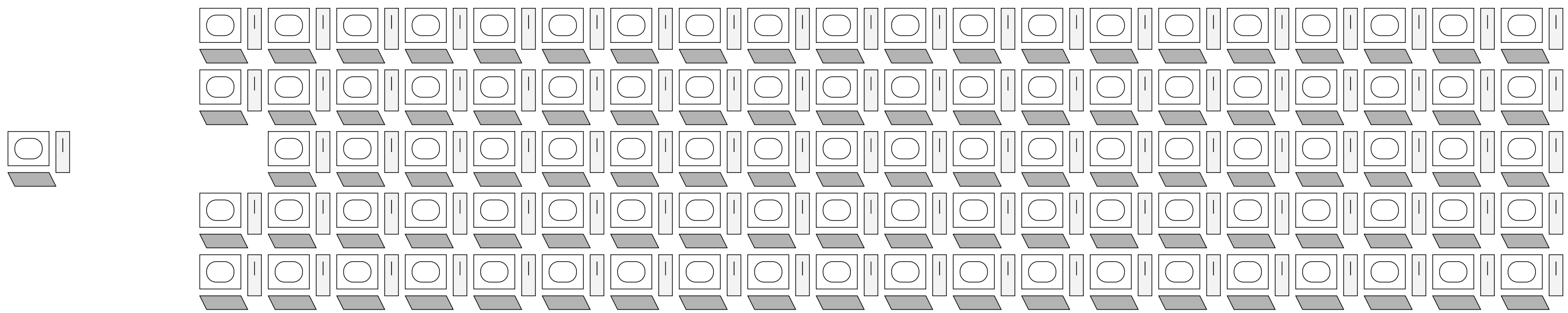
The Alternative

- An enhancement that can increase your computing power 100 fold without any extra design effort.

?

Replication!

BUY ANOTHER 99 COMPUTERS



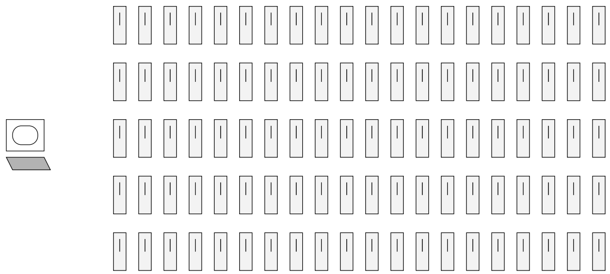
This is the concept of replication

- your computing power is limited only by your budget
- the same units can be used in large and small installations
- no extra design effort for more powerful systems
- linear increase in power with budget

Selective Replication

Replication as described is easy, but more selective replication may yield better results.

- With 100 screens and 100 keyboards we need 100 programmers. With selective replication we need only have one screen and keyboard and replicate the rest.



Two new problems are produced:

- How do we connect the computers to work together with a single screen and keyboard?
- How does one programmer write a program to control all of the computers?

These are the problems of

Parallel Computers