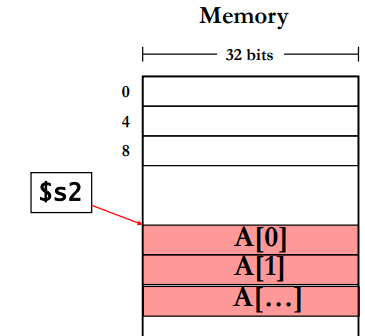
Computer Systems Lecture 5

Getting at the Data

Goal: g = h + A[0] where h is in register $s1, A[0] is the first element of array A and is pointed to by $s2

MIPS: lw $t1,0($s2)

Add $t2, $s1, $t1

Putting lw <register>,constant(<register>) tells the processor to get the data pointed to by the value in the specified register with an offset specified by the constant.  
Data-Transfer Instructions

Load Word: lw r1,n(r2) => r1 = memory[n+r2]

Store Word: sw r1,n(r2) => memory[2+r2]=r1

Load Byte: lb r1,n(r2) => r17-0 = memory[n+r2], r131-8 = sign extension

Store Byte: sb r1,n(r2) => memory[n+r2]=r17-0 no sign extension

Memory Addressing

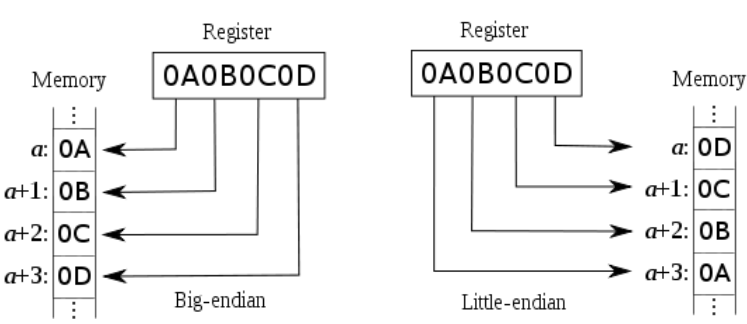
Memory is byte addressable, but it is organised so that a whole word can be accessed directly.

Where can a word be stored?

* Option 1: anywhere (unaligned)
* Option 2: at an address that is a multiple of the word size (aligned)
* Both options are used, MIPS requires alignment, x86 doesn’t

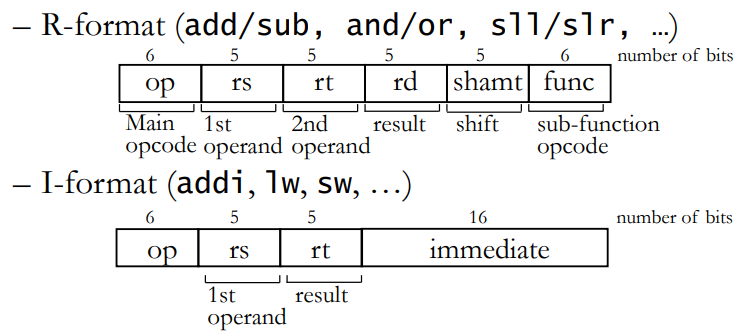
Memory Addressing: Endianness

Given a memory address, Endianness tells us where to find the starting byte of a word.



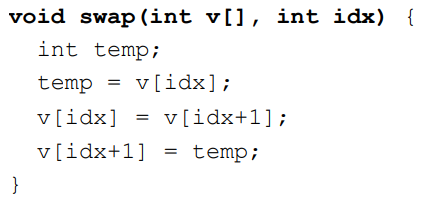
Instruction Formats

Instruction representation is composed of bit-fields, similar instructions have the same format. MIPS instruction formats are:

R formats are for working with registers

L formats are for working with memory

A Simple Function to Swap Array Elements



In MIPS this is written:

