Data storage issues

Byte order

What order are bytes stored in memory?

Alignment

Assignment of data elements to memory addresses

Byte order

**It is allowed on all hands, that the primitive way of breaking eggs,
before we eat them, was upon the larger end; but his present majesty's grandfather,
while he was a boy, going to eat an egg, and breaking it according to the ancient practice,
happened to cut one of his fingers. Whereupon the emperor his father published an edict,
commanding all his subjects, upon great penalties, to break the smaller end of their eggs.**

. . .

**It is computed that eleven thousand persons have at several times suffered death,
rather than submit to break their eggs at the smaller end.
Many hundred large volumes have been published upon this controversy:
but the books of the Big-endians have been long forbidden.**

-- Jonathan Swift, Gulliver's Travels

What other computer-related name comes from Swift?

Endianness

Storing data in memory

- 32-bit word: 4 bytes.

Each byte has an individual address.

- Smallest addressable unit

How many hexadecimal digits in each byte?

- Hex digit = 4 bits, so a byte can hold 2 digits.

How do we store the value 01234567_{hex}?

One way: store the digits in order,

- starting with the lowest address.

This is called big-endian, because the biggest-value byte is stored first.

However, this means that the MOST-significant byte has the LOWEST address.

Another way:

- Store the LEAST-significant byte in the lowest address.

This is called little-endian.

Endianness controversy has taken on religious intensity!

- MAC vs. PC, vi vs. emacs, Pepsi vs. Coke
- big-endian: MIPS (selectable), IBM classic, Motorola, Sun
- little-endian: DEC, IBM PC

big-endian

Address	Contents
1000	01
1001	23
1002	45
1003	67
1004	
1005	
1006	
1007	

little-endian

Address	Contents
1000	67
1001	45
1002	23
1003	01
1004	
1005	
1006	
1007	

Endianness

Why is endianness significant?

If you transfer data between machines (removable media or network),

you need to know which order it is stored in.

Things to note:

The order refers to BYTES, not bits. The bits are not simply reversed from lsb to msb.

Ordering only refers to data stored in MEMORY.

Data in registers stored from msb to lsb, left to right, regardless of endianness.

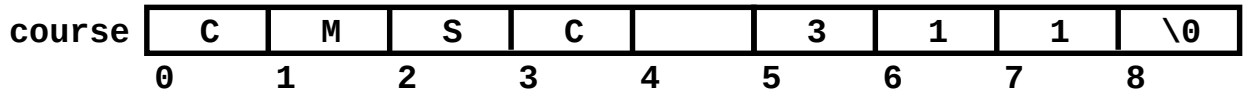
32-bit register: b31....b0

Endianness has nothing to do with the order of C-style strings.

What is a C-style string?

Array of char, ended by '\0'

Each char is 1 byte




```
char course[]= "CMSC 311";
```

Arrays are always stored from element 0 to element (n-1)

Note that C++ style strings are objects, may have more complicated structure

How can we test for endianness on a particular machine?

Need to look at individual bytes within a word.

We will consider this when we look at C language features for manipulating bits and bytes.