15-213 Recitation 5 – 2/19/01

Outline

- Structured Data: `structs` / `unions`
- Alignment
- Floating Point

Reminders

- Lab 2: Wednesday, 11:59
- EXAM 1: Tuesday, 2/27
  - UC McConomy

Shaheen Gandhi

e-mail:

sgandhi@andrew.cmu.edu

Office Hours:

Wednesday 12:30 – 2:30

Wean 3108

`struct`**s and** `union`**s**

- Organize data
- `struct`s store multiple elements, `union`s store a single element at a time
- Members of a `union` change how you look at data
- `union`s used for mutually exclusive data

Alignment

- Contiguous areas of memory
- Each block is *aligned*
  - Size is a multiple of a base value
  - “Base value” is the largest alignment of data types in structure
- Why?
  - Efficient load/store from memory
  - Virtual Memory paging
- This applies to any variable type

Structure of a `struct`

- Find largest alignment
  - Size of structure must be a multiple of this
- For each element e (top to bottom):
  - Find alignment of e
    - Starting offset must be a multiple of this
  - Pad previous element with empty space until alignment matches
  - Allocate alignment worth of space to e
- Pad last element with empty space until alignment of structure matches
- Note this isn't optimal!

Structure of a `union`

- Find largest alignment
  - Size of structure must be a multiple of this
- Allocate this much space

Examples

```
struct one {
  int i;
  double d;
  char c[2];
}
```

```
union two {
  int i;
  double d;
  char c[2];
}
```

Floating Point

(Better known as "I'm going to kill the person that thought this up")

- IEEE Floating Point
  - Standard notation
  - Tons of features we won't look at
- Floating Point at a bit level:

s	**exp**	**frac**

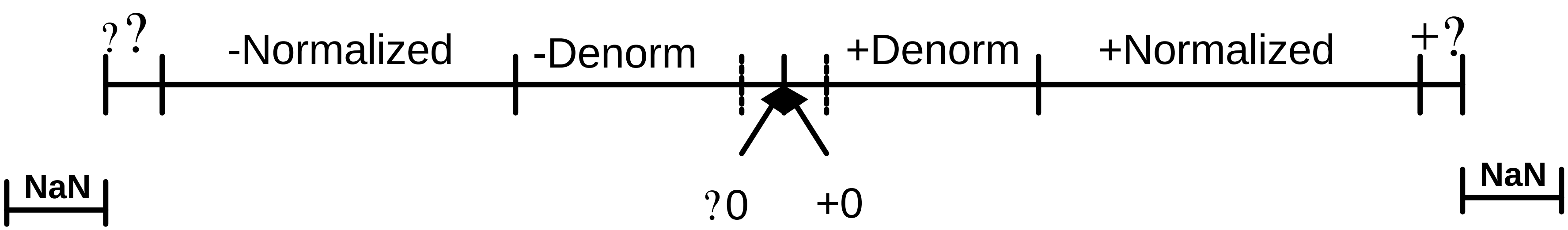
  - `s` – sign bit (S)
  - `exp` – exponent (maps to E, has e bits)
  - `frac` –significand (maps to M, has f bits)
- Numerical Equivalent: $-1^{s}\ M\ \ 2^{E}$
- "Normalized" and "Denormalized" encoding

“Normalized” Encoding

- exp ? `0` and exp ? `111…1`
  - If exp = 111…1, it’s ? or NAN
- `E = exp – B`
  - B is the “Bias”
  - Usually $2^{e-1} – 1$, but can be different
  - `exp`: Unsigned integer value $[1, 2^{e} – 1]$
- `M = 1.{frac}`
  - `{frac}` are the bits of `frac`
  - `frac` is a fractional binary number
- Normalized Numbers have range $[2^{1-B}, 2^{B+1})$
  - And their negatives

“Denormalized” Encoding

- exp = 0
- E = -B+1
- M = 0.{frac}
- Denormalized Numbers have Range $[0, 2^{1-B})$
  - And their negatives



Examples

- 8 bit FP, 1 bit sign, 4 bit exponent, 3 bit significand, Bias of 7

Representation -> Number

0 0101 011	0.34375
0 0000 101	0.009765625
1 1011 110	-28.0