Okay, class, I've turned in your exams for grading. Now—

Miss Lenhart?

I used a #3 pencil instead of a #2. Will that mess anything up?

Aieee! Blam!

You what?

Oh god!

Oh god! I've never seen so much blood!
EECS 370 Discussion

Topics Today:

- ARM Addressing
  - Endianness, Loading, and Storing Data
- Data Layout
  - Struct Packing
- Control Flow
  - Branches and PSR (Program Status Register)
- Conditional Assembly
  - Predicated Assembly Instructions
- C to ARM Translation
EECS 370 Discussion

ARM Addressing
EECS 370 Discussion

ARM Addressing

Big-Endian means Big End First

- Most Significant Byte (MSB) at first address in memory

In EECS370 we use Big-Endian

Applies to Bytes only!
Example:

sub r1, r1, #0x2
strh r1, [r2, #0x0]
ldrsb r2, [r2, #0x1]
ldrh r1, [r0, #0x1002]
EECS 370 Discussion

Data Layout

Golden Rule

Start address of a variable is aligned based on the variable’s type

- char – byte aligned
- short – halfword aligned
- int – word aligned
- pointer – word aligned
- double – two-word aligned

Structs

- start aligned based on alignment of largest member
- end padded to make overall size a multiple of largest member
Example:

```c
char a;
char* b;
short c;
double* d;
struct {
    int e;
    char f[10];
} g;
```
EECS 370 Discussion

Control Flow

ARM Registers:
16 total registers

R0 – R12: General Purpose
R13: Stack Pointer
R14: Link Register
R15: Program Counter ← Dangerous!
EECS 370 Discussion

Control Flow

Program Status Register (PSR):
Flags set by various assembly instructions

N – result is negative
Z – result is zero
C – result had a carry out of bit 31
V – result had an overflow
Control Flow

Branch Condition Codes based on PSR

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>eq</td>
<td>EQual</td>
<td>Z == 1</td>
</tr>
<tr>
<td>ne</td>
<td>Not Equal</td>
<td>Z == 0</td>
</tr>
<tr>
<td>ge</td>
<td>Greater than/Equal</td>
<td>N == V</td>
</tr>
<tr>
<td>lt</td>
<td>Less Than</td>
<td>N != V</td>
</tr>
<tr>
<td>gt</td>
<td>Greater Than</td>
<td>Z == 0 &amp;&amp; N == V</td>
</tr>
<tr>
<td>le</td>
<td>Less than/Equal</td>
<td>Z == 1</td>
</tr>
<tr>
<td>cs</td>
<td>unsigned higher/Same</td>
<td>C == 1</td>
</tr>
<tr>
<td>cc</td>
<td>unsigned lower</td>
<td>C == 0</td>
</tr>
<tr>
<td>mi</td>
<td>negative</td>
<td>N == 1</td>
</tr>
<tr>
<td>pl</td>
<td>positive/Zero</td>
<td>N == 0</td>
</tr>
<tr>
<td>vs</td>
<td>oVerflow Set</td>
<td>V == 1</td>
</tr>
<tr>
<td>vc</td>
<td>no oVerflow Clear</td>
<td>V == 0</td>
</tr>
<tr>
<td>hi</td>
<td>unsigned H</td>
<td>gher</td>
</tr>
<tr>
<td>ls</td>
<td>unsigned Lower/Same</td>
<td>C == 0</td>
</tr>
<tr>
<td>al</td>
<td>any/Always</td>
<td></td>
</tr>
</tbody>
</table>
Conditional Assembly

Condition Codes can also be applied to other instructions

Example: Translate the following code

```plaintext
while (r0 != 0) {
    if (r0 > 0) {
        r0 = r0 - 1;
    } else {
        r0 = r0 + 1;
    }
}
```
Do some examples.

How to move 32-bit values into registers.