Arithmetic Instructions

- General form
- and must be registers; may be a register or a signed 13-bit number

```
add %o1, %o2, %g3
sub %i1, 2, %g3
```

- Some SP ARCs have no multiply and divide instructions
  see Appendix E of the SP ARC Architecture Manual, §4.10 in Paul

- Standard run-time library provides multiply and divide routines
  .mul .rem .div
  .umul .urem .udiv

Data Movement

- Load a constant into a register

```
e.g., direct addressing
sethi %hi(a), %g1
ld [%g1], %g2
```

Synthetic Instructions

- Synthetic instructions or pseudo-instructions are implemented by the assembler
  by one or more „real“ instructions implemented by the assembler

```
move register to register
mov src, dst
or %g0, src, dst
```

- Clearing registers and memory; note the use of %g0 to stand for 0

```
clear register, memory
add %g0, %g0, reg
```

Bitwise Logical Instructions

- General form

```
~ = not
& = and
|= = or
^ = xor
```

- Corresponding C bitwise operators:
  is a register or a signed 13-bit number

```
~reg = not reg
reg & mask = and reg, mask
reg | mask = or reg, mask
reg ^ mask = xor reg, mask
```

Machine representation:

```
 1111 1111 0000 0000 0000 0000 0000 0000
~reg = 0111 1111 1111 1111 1111 1111 1111 1111

= each register and memory; note the use of 0x90 to stand for 0

ld 0x90, %a
and %a, 0x90
```

Data Movement

- Load a constant into a register

```
if reg = 0, omit if 0x90 = 0, omit ox
else
```

```
ld 0x90, %a
and %a, 0x90
```
Bitwise Logical Instructions, cont'd

- Complement
  - 2's complement: neg
  - 1's complement: not
  - xnor: reg, %g0, reg

- Synthetic instructions
  - btst
  - andcc: reg, bits, %g0
  - bset
  - or: reg, bits, reg
  - bclr
  - andn: reg, bits, reg
  - btog
  - xor: reg, bits, reg

For example:
- btst 0x8, %g1

Shift Instructions

- General form
- Instruction format
- Shift instructions do not modify the condition codes.

<table>
<thead>
<tr>
<th>sll</th>
<th>src 0</th>
<th>src 1</th>
<th>src 2</th>
<th>dst</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>00000000</td>
<td>00000000 0..31</td>
<td>31 29 24 18 13 12 4</td>
<td></td>
</tr>
<tr>
<td>srl</td>
<td>00000001</td>
<td>00000000 0..31</td>
<td>31 29 24 18 13 12 4</td>
<td></td>
</tr>
<tr>
<td>sra</td>
<td>00000001</td>
<td>00000000 0..31</td>
<td>31 29 24 18 13 12 4</td>
<td></td>
</tr>
</tbody>
</table>

Floating Point Instructions

- Floating point instructions are performed using the floating point unit (FPU).
- 32 floating point registers: %f0—%f31
- Floating point load and store instructions:
  - ld [address], freg
  - ldd [address], freg
  - st freg, [address]
  - std freg, [address]
- Doubles use even-odd register pair.

Other instructions: set L, set G, delete floating point registers.

Floating Point Instructions, cont'd

- Floating point instructions are performed using the floating point unit.

**Floating Point Instructions**

1. Hodges
2. Hennes
3. Linus
4. Christian

**Shift Instructions**

- General form
- Instruction format
- Shift instructions do not modify the condition codes.

<table>
<thead>
<tr>
<th>Instruction Format</th>
<th>Source 0</th>
<th>Source 1</th>
<th>Source 2</th>
<th>Destination</th>
</tr>
</thead>
<tbody>
<tr>
<td>sll</td>
<td>00000000</td>
<td>00000000 0..31</td>
<td>31 29 24 18 13 12 4</td>
<td></td>
</tr>
<tr>
<td>srl</td>
<td>00000001</td>
<td>00000000 0..31</td>
<td>31 29 24 18 13 12 4</td>
<td></td>
</tr>
<tr>
<td>sra</td>
<td>00000001</td>
<td>00000000 0..31</td>
<td>31 29 24 18 13 12 4</td>
<td></td>
</tr>
</tbody>
</table>

Floating Point Instructions, cont'd

- Comparison and branching

**Floating Point Instructions, cont'd**

1. Hodges
2. Hennes
3. Linus
4. Christian

**Shift Instructions**

- General form
- Instruction format
- Shift instructions do not modify the condition codes.

<table>
<thead>
<tr>
<th>Instruction Format</th>
<th>Source 0</th>
<th>Source 1</th>
<th>Source 2</th>
<th>Destination</th>
</tr>
</thead>
<tbody>
<tr>
<td>sll</td>
<td>00000000</td>
<td>00000000 0..31</td>
<td>31 29 24 18 13 12 4</td>
<td></td>
</tr>
<tr>
<td>srl</td>
<td>00000001</td>
<td>00000000 0..31</td>
<td>31 29 24 18 13 12 4</td>
<td></td>
</tr>
<tr>
<td>sra</td>
<td>00000001</td>
<td>00000000 0..31</td>
<td>31 29 24 18 13 12 4</td>
<td></td>
</tr>
</tbody>
</table>