Today's topics

Machine Architecture
More Low-level Programming

Upcoming
Language Translation (G.I. Chapter 9)

Reading
Great Ideas, Chapters 8

Programming Loops

- Now use new instructions to do the equivalent of `while`
- We noted that syntax for `if` and `while` were same
  - Assembler code surprisingly similar for these two
  - Major addition is the update
  - Also need jump back to beginning of loop
- Demonstrate with code equivalent to:

```java
{  
    limit = 0;
    sum = 0;
    x = a.getInt();
    while (limit < x)
    {
        sum = (sum + x);
        x = a.getInt();
    }
    b.setInt(sum);
}
```

sum.as

```
0    copy    ax,  #C0
1    copy    limit, ax
2    copy    ax,  #C0
3    copy    sum, ax
4    in      ax
5    copy    x, ax
6    #L0     copy    ax, limit
7    cmp     ax, x
8    jnb     #L1
9    copy    ax, sum
10   add     ax, x
11   copy    sum, ax
12   in      ax
13   copy    x, ax
14   jmp     #L0
15   #L1     copy    ax, sum
16   out     ax
```

Another looping example

- Calculate N! (N factorial) but do it with a loop this time
- Code is equivalent to the following Java:

```java
{  
    n = a.getInt();
    i = 1;
    fact = 1;
    while (i < n+1)
    {
        fact = (fact * i);
        i = (i + 1);
    }
    b.setInt(fact);
}
```
fact.as

1  in   ax
2  copy n, ax
3  copy ax, #C1
4  copy i, ax
5  copy fact, ax
6  #L0 copy ax, n
7  add ax, #C1
8  copy E0, ax
9  copy ax, i
10 cmp ax, E0
11 jnb #L1
12 copy ax, fact
13 mult ax, i
14 copy fact, ax
15 copy ax, i
16 add ax, #c1
17 copy i, ax
18 jmp #L0
19 #L1 copy ax, fact
20 out ax
21 halt

Notes:
#L0=6
#L1=19

Assembler Programming Notes

- Note that previous program added the `mul` instruction
  - Most hardware has standard arithmetic support
  - Historically not the case
- The best way to follow such a program is by tracing
  - See trace for fact.as program on web page
- Writing assembler programs from scratch
  - Not that hard
  - Can get quite used to working at this level
  - Was done for efficiency reasons
    - Could do better than automatic translation (e.g., compiler)
  - However, remember 15 lines of code a day
    - This figure is language independent!
- Compilers have gotten better than the average programmer

Handling List or Arrays

- Need extra hardware to do this well
  - Have registers that point to the list/array
  - Increment these registers to step through list/array
- Can be done with our limited hardware
  - Involves having the program modify itself
  - Not hard to write
  - Errors in such self-modifying code very hard to find!
- Additional Features Desired (minimal upgrade)
  - Need for more registers
  - Handling function/method calls
    - Need to “remember” where you came from
    - Jump to statement after that when done

Modern Hardware

- Memory Size
  - PC’s often have gigabyte of memory now
  - What does this do to the size of the instruction?
- Lots of Registers
  - It is not unusual to have 32 accumulators
  - What does this do to the size of the instruction?
- Memory Hierarchy
  1. Registers
  2. Cache Memory
  3. Main Memory
  4. Disk (virtual memory)
  5. Offline storage (tapes, CDROMs, DVDs, etc.)