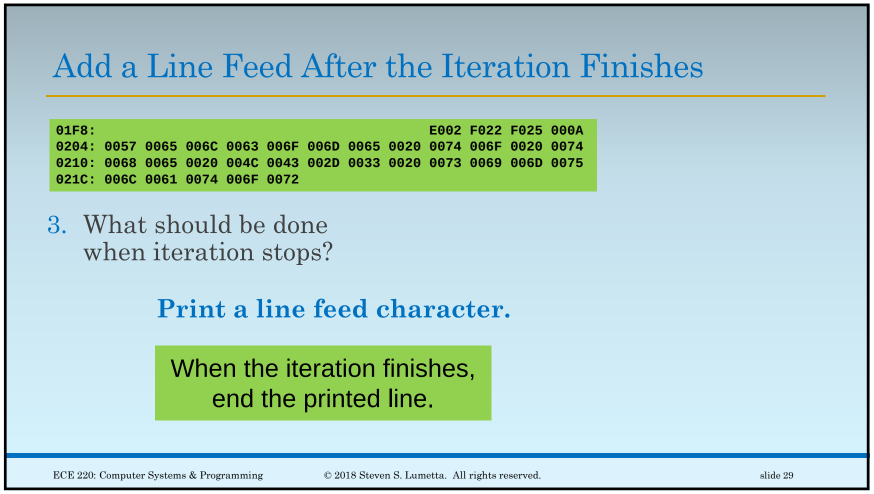


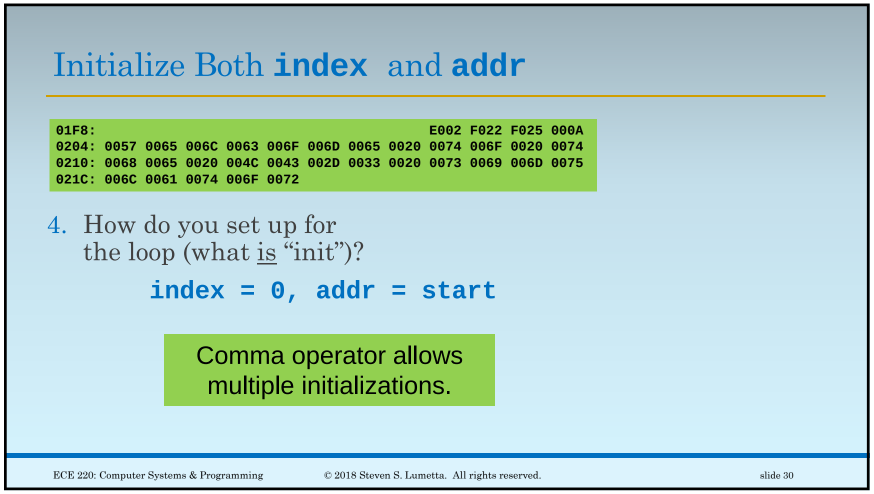
Add a Line Feed After the Iteration Finishes

```
01F8:                                              E002 F022 F025 000A
0204: 0057 0065 006C 0063 006F 006D 0065 0020 0074 006F 0020 0074
0210: 0068 0065 0020 004C 0043 002D 0033 0020 0073 0069 006D 0075
021C: 006C 0061 0074 006F 0072
```

3. What should be done when iteration stops?

Print a line feed character.

When the iteration finishes, end the printed line.



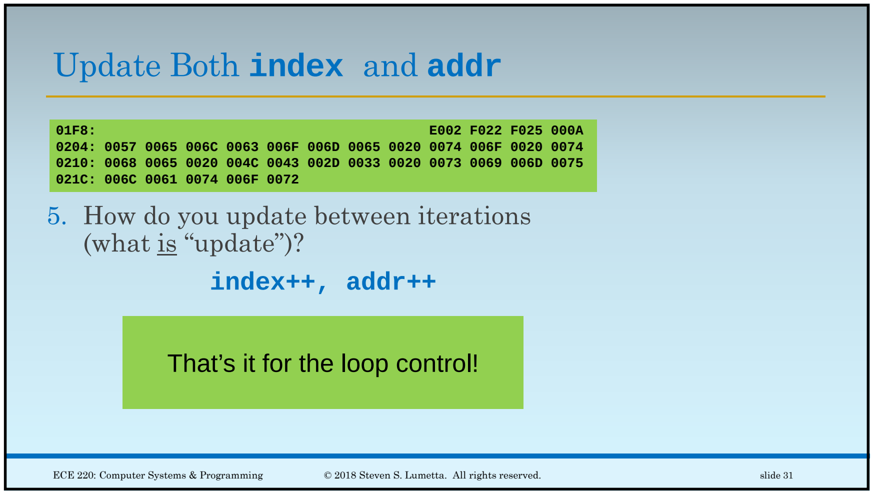
Initialize Both **index** and **addr**

```
01F8:                                              E002 F022 F025 000A
0204: 0057 0065 006C 0063 006F 006D 0065 0020 0074 006F 0020 0074
0210: 0068 0065 0020 004C 0043 002D 0033 0020 0073 0069 006D 0075
021C: 006C 0061 0074 006F 0072
```

4. How do you set up for the loop (what is "init")?

```
index = 0, addr = start
```

Comma operator allows multiple initializations.



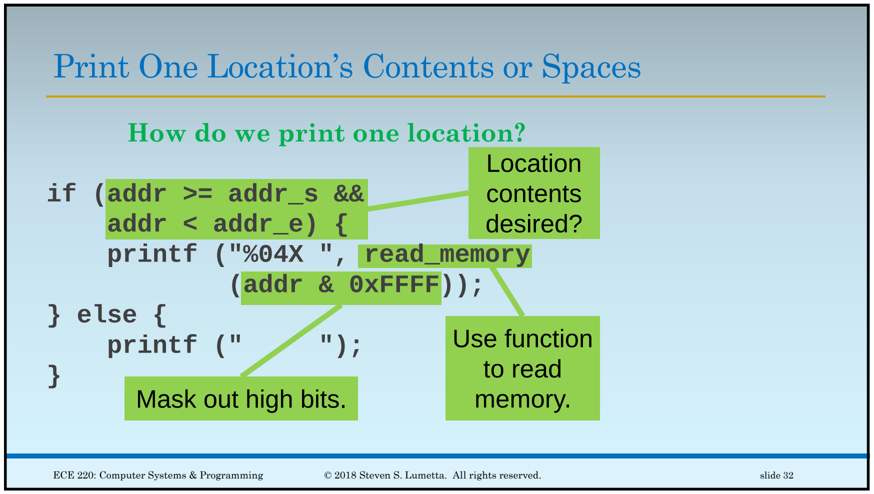
Update Both **index** and **addr**

```
01F8:                                              E002 F022 F025 000A
0204: 0057 0065 006C 0063 006F 006D 0065 0020 0074 006F 0020 0074
0210: 0068 0065 0020 004C 0043 002D 0033 0020 0073 0069 006D 0075
021C: 006C 0061 0074 006F 0072
```

5. How do you update between iterations (what is "update")?

```
index++, addr++
```

That's it for the loop control!



Print One Location's Contents or Spaces

How do we print one location?

```
if (addr >= addr_s &&
    addr < addr_e) {
    printf ("%04X ", read_memory
             (addr & 0xFFFF));
} else {
    printf ("     ");
}
```

Location contents desired?

Use function to read memory.

Mask out high bits.