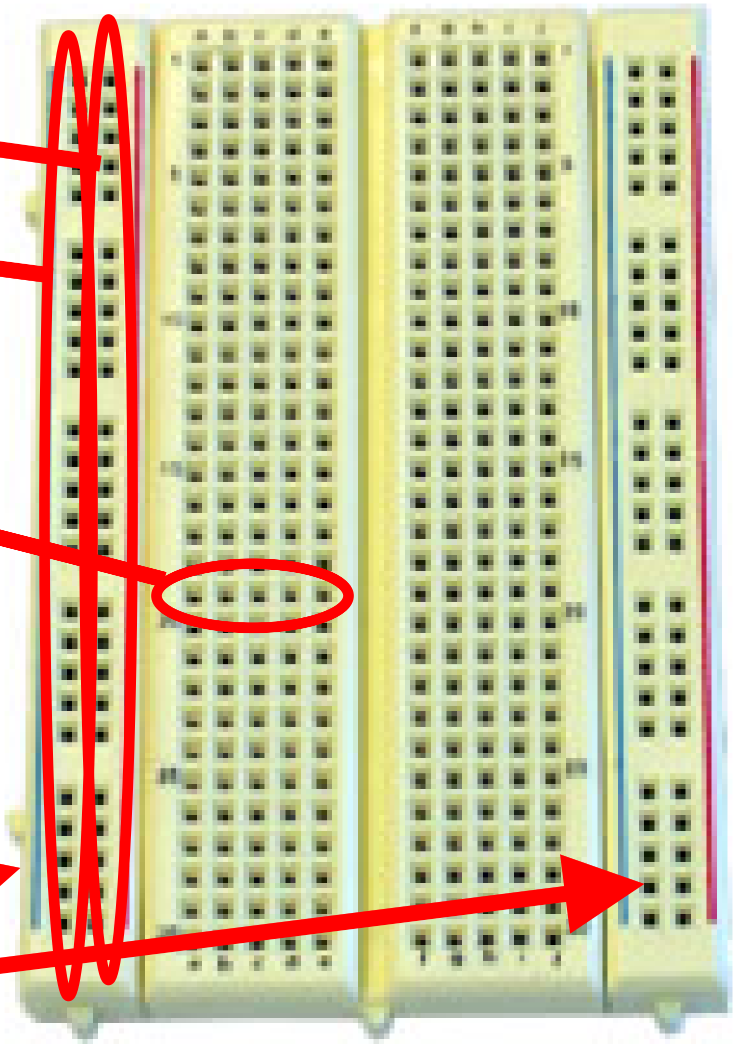
Solderless Breadboards

Power bus (red)

Ground bus (blue)

Component bus

Note that the two sides are not connected



mbus.net

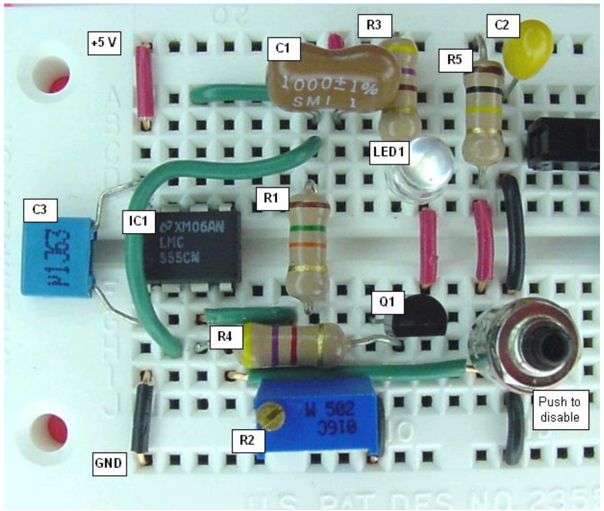
Wiring Standards

When possible, use wire colors for different types of signals:

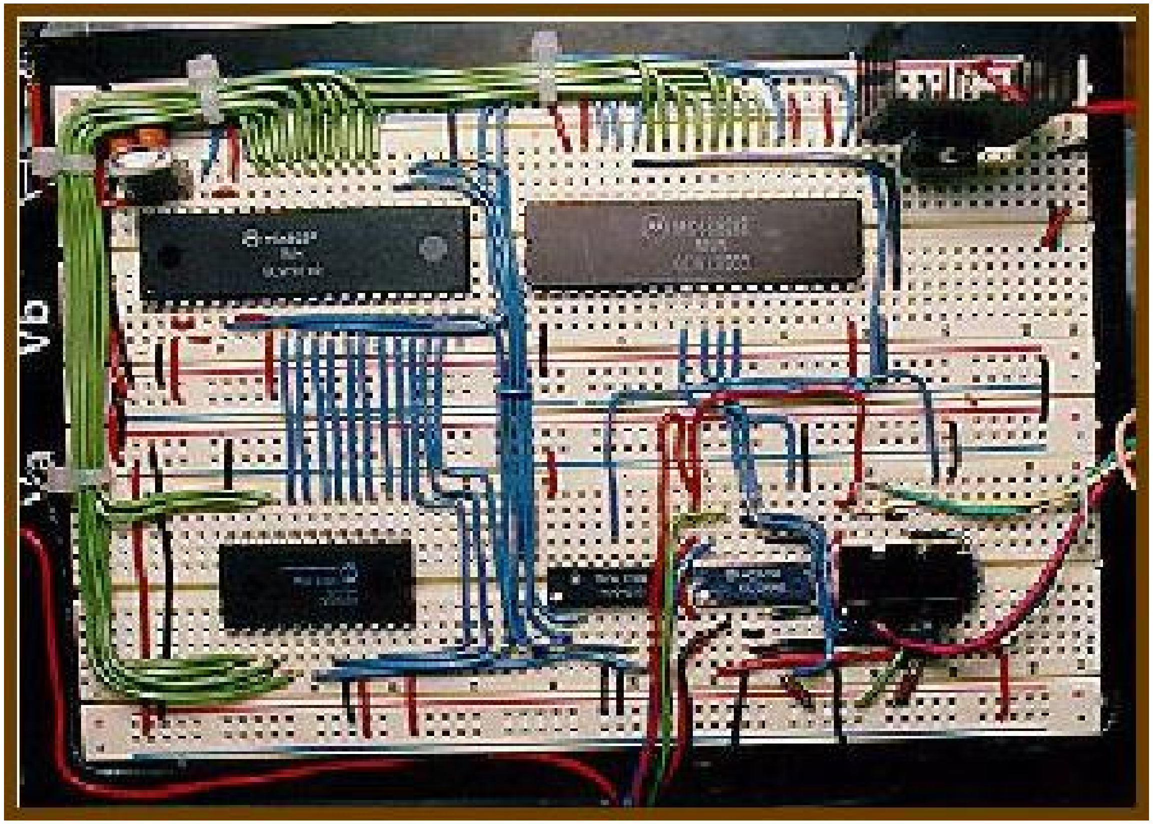
- Black: ground
- Red: power
- Other: various signals

Clean Wiring

A clean breadboard will make debugging easier – and it makes circuits more robust



www.linefollowing.com



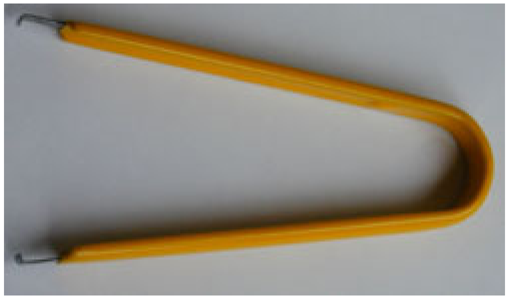
tangentsoft.net

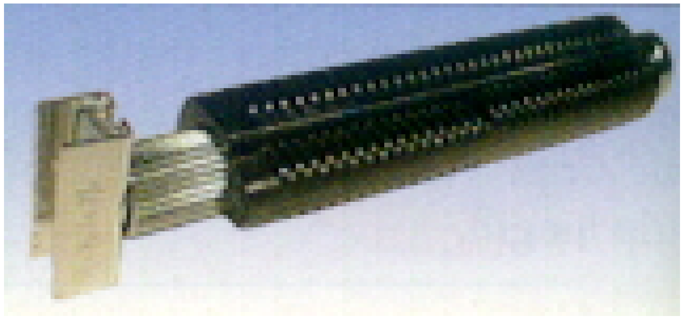
Care with Power

- Only insert components and wires into the breadboard when power is disconnected
- “Wire, check-twice, then power”
  - Never reverse power and ground (this is a very common mistake)
- Most chips that we will use expect +5V
  - More can destroy the chips
  - We will use DC/DC converters to step battery voltages down to +5V

Care of Chips

- Use insertion and extraction tools: never your fingers
- Minimize your contact with pins: static electricity can destroy a chip





www.chantronics.com.au

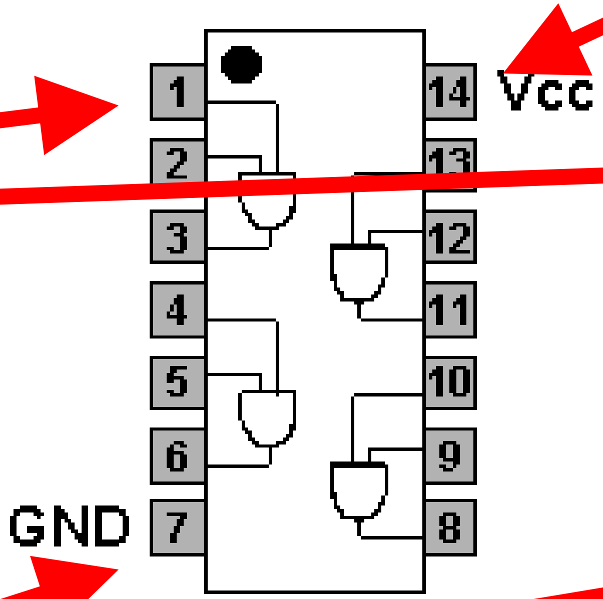
www.a7vtroubleshooting.com

TTL Chips: 2-Input AND Gates

Chip number: 7408

Power

Pin 1 is marked on the chip



Ground

www.dcs.warwick.ac.uk

www2.117.ne.jp

Wiring Procedure (Suggested)

- Power supply
- Power/ground buses
- Insert primary components
- Wire power/ground for components
- Add signals and remaining components
- Test incrementally

Debugging Techniques

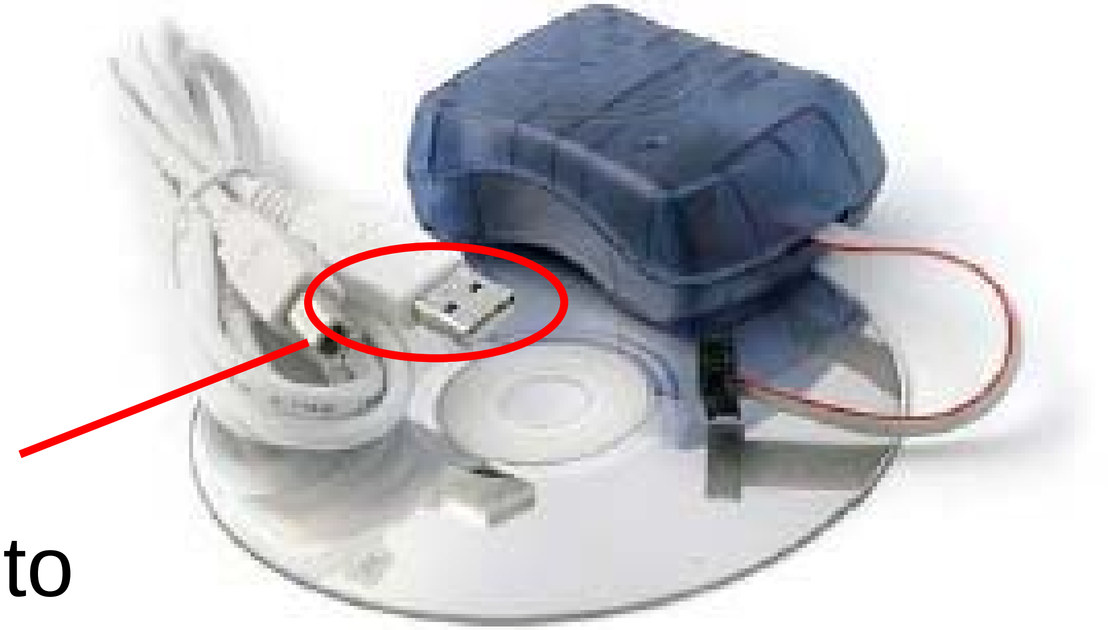
- Multimeter:
  - Use *voltage mode* to check logic levels
  - Use *continuity mode* to confirm connections (but never with power turned on!)
- Oscilloscope:
  - View voltage as a function of time on 2 channels
- Test incrementally
- Test intermediate sub-circuits

Physical Interface for Programming

AVR ISP

Physical Interface for Programming

AVR ISP



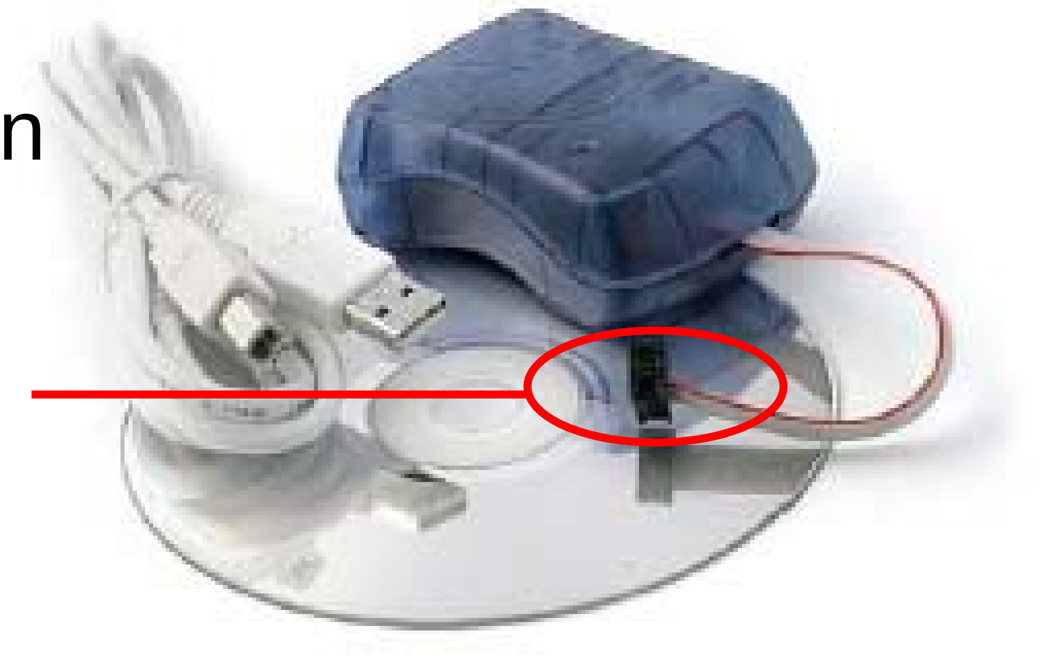
USB connection to your laptop

Physical Interface for Programming

AVR ISP

Header connection will connect to your circuit (through an adapter)

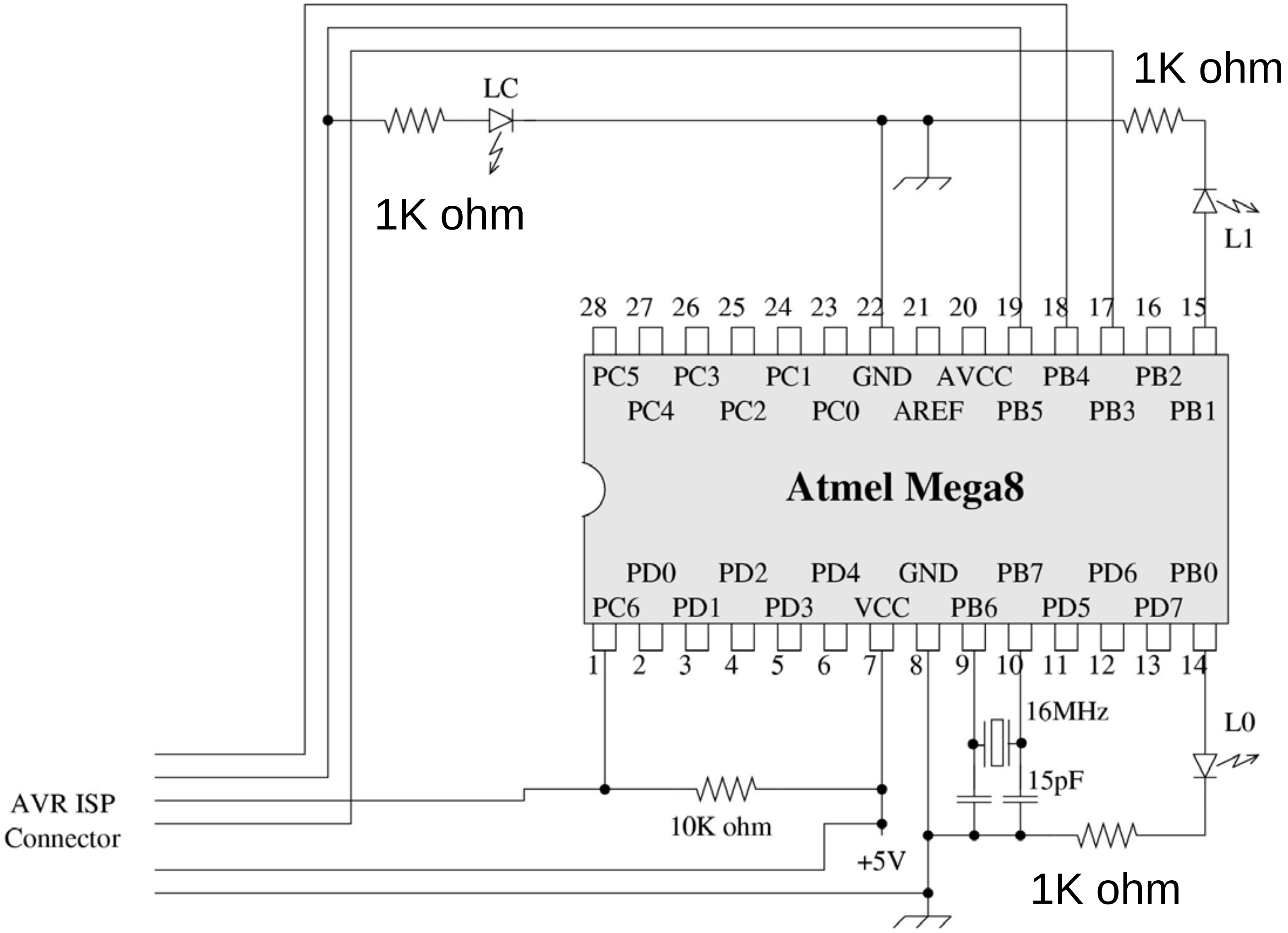
Be careful when you plug your circuit in (check before powering)



AVR ISPs are Cranky

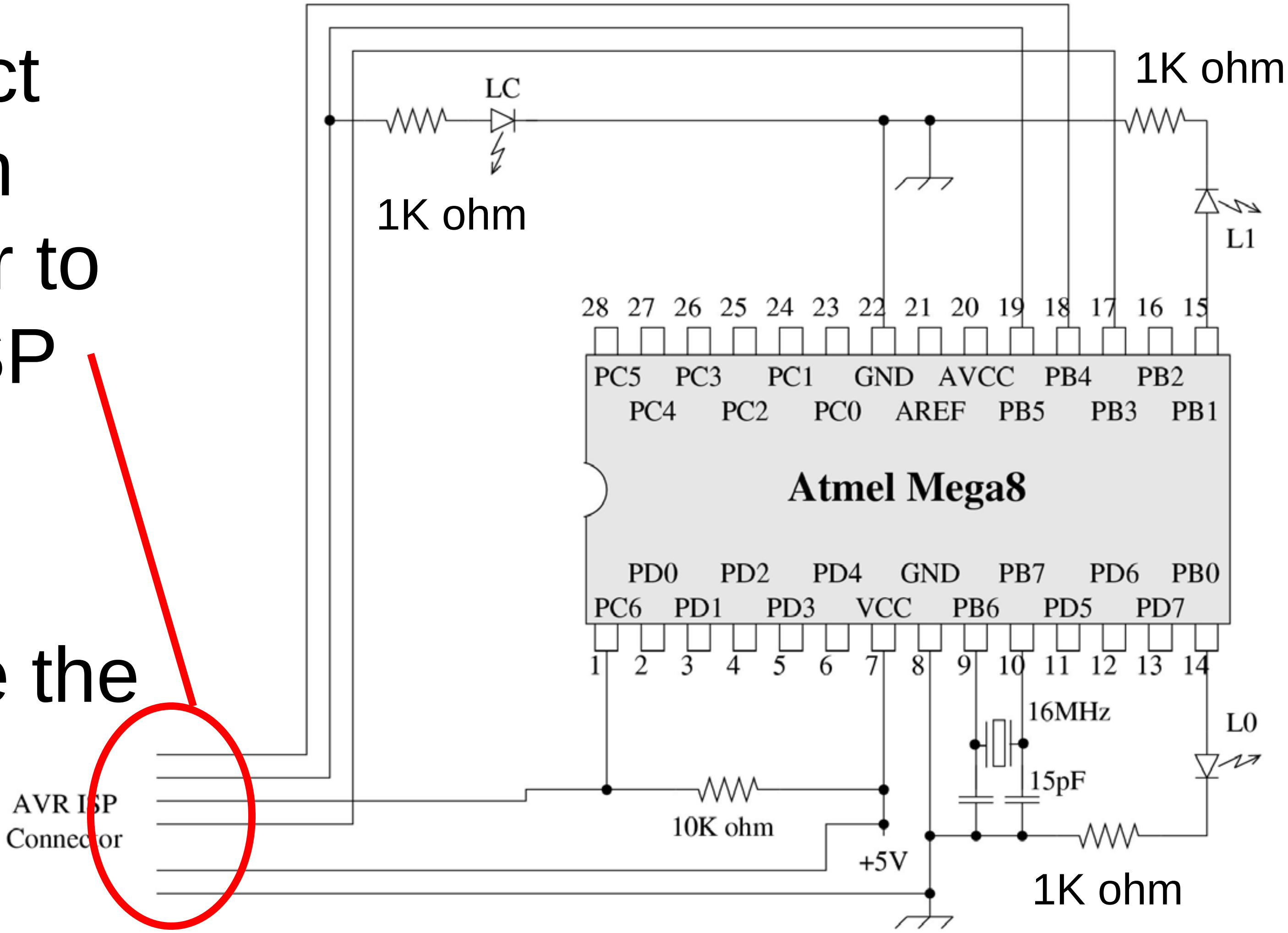
- When things are plugged in and powered, you should see two green LEDs on the ISP (on most units)
- One red: usually means that your circuit is not powered
- Orange: the programmer is confused
  - Could be due to your circuit not being powered at 5V
  - Could be due to other problems
  - Check power and reboot the ISP

A Basic Circuit



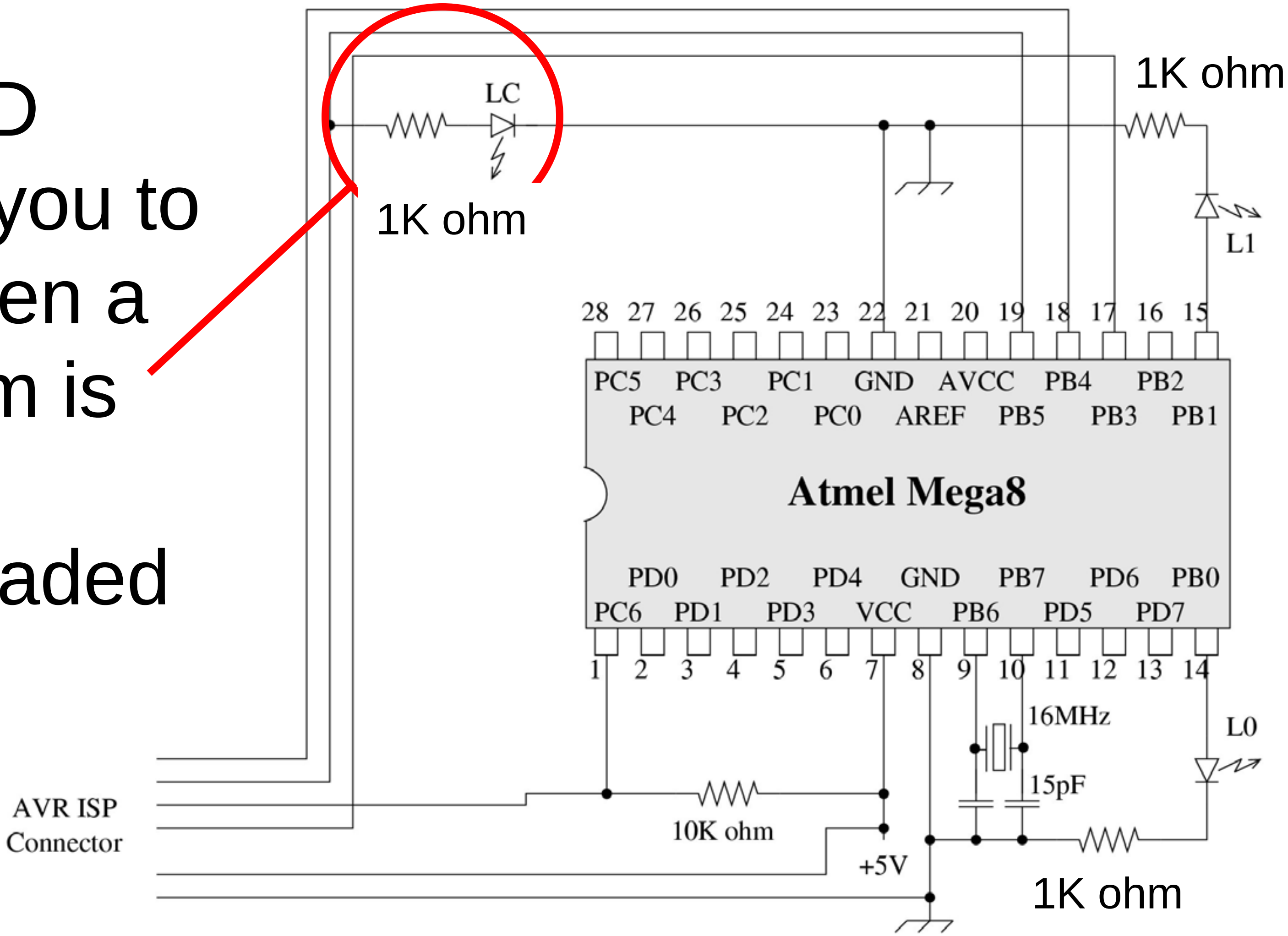
A Basic Circuit

- Connect through adapter to AVR ISP
- Do not reverse the pins!



A Basic Circuit

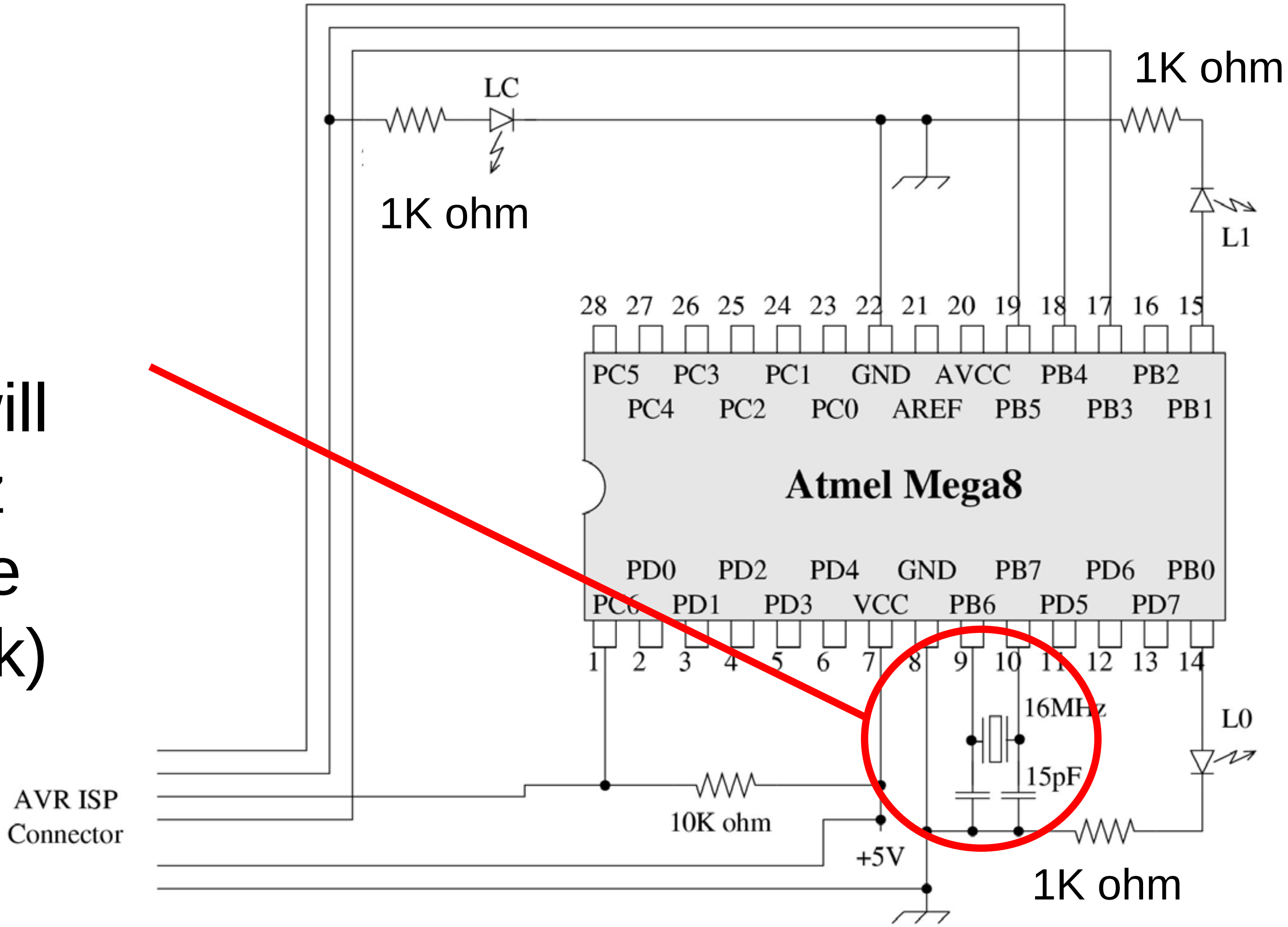
Extra LED allows you to see when a program is being downloaded



A Basic Circuit

16 MHz crystal

- (generally optional)
- Without it, your processor will run at 1MHz (we want the 16MHz clock)



Compiling and Downloading Code

Preparing to program:

- See the Atmel HOWTO (pointer from the schedule page)
- Windoze: Install AVR Studio and WinAVR
- OS X: Install OSX-AVR
  - We will use ‘make’ for compiling and downloading
- Linux: Install binutils, avr-gcc, avr-libc, and avrdude
  - Same as OS X

Compiling and Downloading Code

- Once the chip is programmed, the AVR ISP will automatically reset the processor; starting your program

Hints

- Use LEDs to show status information (e.g., to indicate what part of your code is being executed)
- Have one LED blink in some unique way at the beginning of your program
- Go slow:
  - Implement and test incrementally
  - Insert plenty of pauses into your code (e.g., with delay_ms())

Getting Hardware Help

- Some exercises in class (come ready)
- Office hours
- Appointments

Group Assignments

Next Time

First circuit and program

- Come with laptops installed with the necessary AVR packages

  (see the Atmel HOWTO for details)

- Wire up basic Atmel circuit
- First program: flipping LED states