Electronics for IoT

Connecting to the Internet

Bernhard E. Boser
University of California, Berkeley
boser@eecs.berkeley.edu

Summary

- Voltage, Current, Power
- Electrical components
  - Resistors, potentiometer
  - Solar cells, batteries, …
- Ohm's law
- Kirchhoff's laws
  - KVL
  - KCL
- Circuit analysis: determine I, V, P, … using
  - KVL, KCL, (nodal analysis)
  - Component characteristics

Efficiency

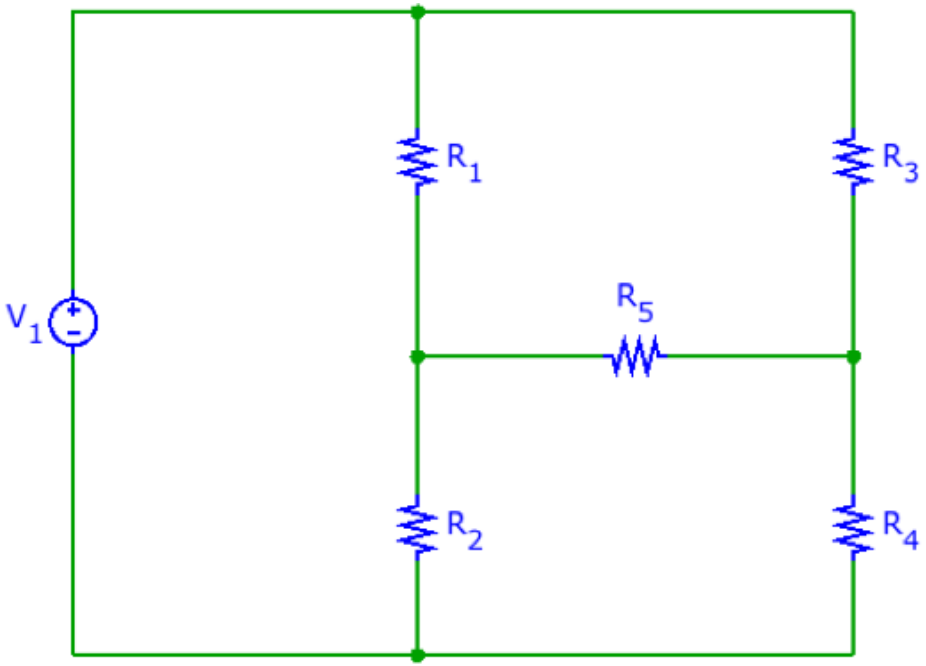
- Solar power (sun overhead): ~ 1 kW/m^2
- Typical solar panel efficiency: ~ 5 … 20%
- → 50 … 200 W/m^2
- Our panel:
  - 12 x 15 cm^2 = 0.018 m^2
  - 0.018 m^2 x 100 W/m^2 = 1.8W … vs 0.45 W
  - Explanation (?): lamp provides less illumination than sun

Equations, equations, …

- KVL & KCL
  - Many more equations than we need
  - Linear dependent
- Systematic way to write just the equations needed?

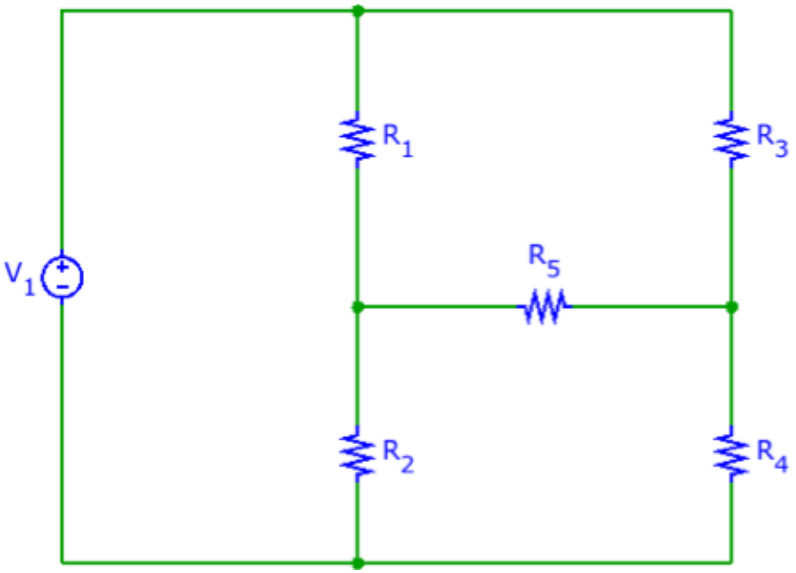
Nodal Analysis

- Objective
  - Find all unknown voltages in circuit

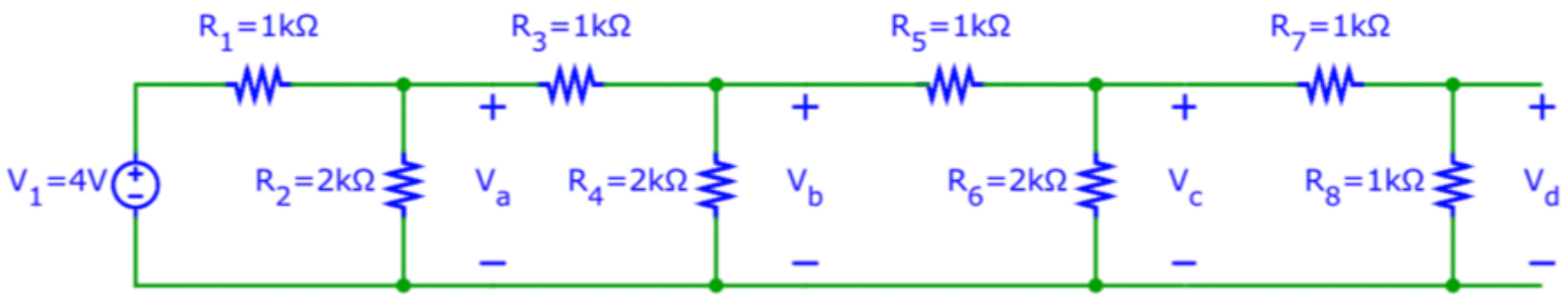


Nodal Analysis

1. Choose ground (GND)
2. Mark unknown node voltages
3. Write KCL equation for each unknown node
4. Solve set of equations



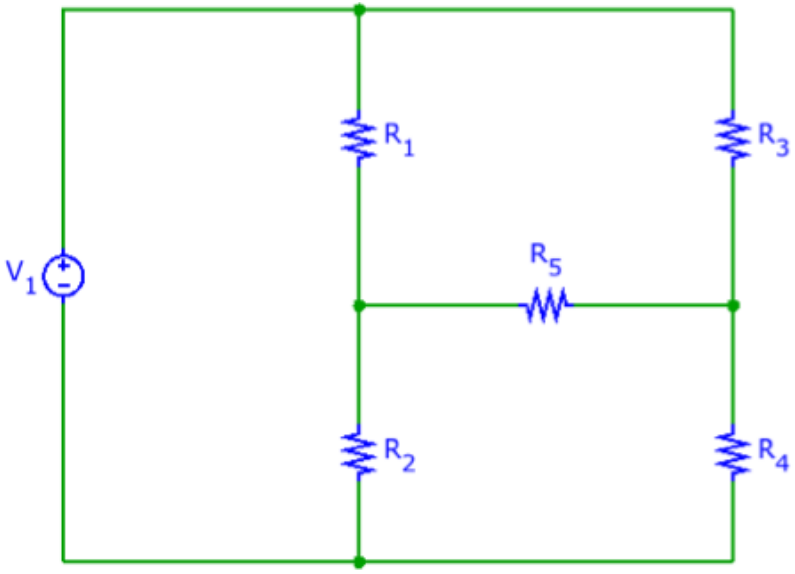
Homework 2, Question 4



Don't use nodal analysis on this problem!
(unless you are a computer)

Hint

Nodal Analysis Example

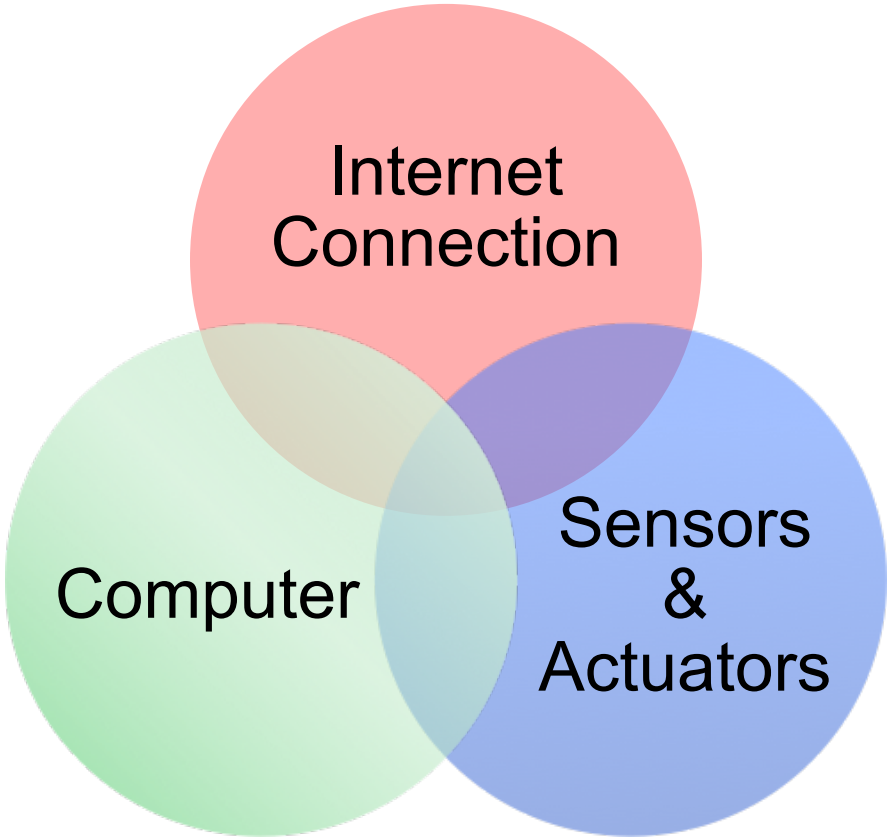


Outline

- Loose ends
  - Solar cell efficiency
  - KVL, KCL equation madness

- Applications … IoT!

IoT Application



Internet Connection

- ESP32 has built-in WiFi radio
- Connects to any 2.4GHz wireless network
  - With WPA2 security
  - No security
  - Other security protocols not supported:
    - E.g. airbears2
- @ UC Berkeley
  - EECS-PSK (available in Cory Hall)
  - CalVisitor (not very usable because of firewall)
- Most “home” wifi routers work as well

MicroPython Language

- Mostly standard Python 3.4
  - Some libraries are missing, or missing functionality
  - Lots of additional stuff available on web
- Documentation
  1. http://docs.micropython.org/en/latest/pyboard/
     - Mostly applies to all boards
     - Some information is specific to pyboard (different from ESP32)
  2. https://github.com/bboser/IoT49
     - Setup instructions (lab 2 points to this)
     - Information specific to ESP32 port
  3. https://github.com/bboser/MicroPython_ESP32_psRAM_LoBo
     - MicroPython source code (in C)

MicroPython Programming

- Connect options
  - Serial (USB)
    - Also powers the chip
  - Wireless (telnet)
    - By the end of class …

- Interacting with the board
  - `shell49`
  - Setup in Lab 2
    - On EECS lab computer
    - Alternative: on your own computer
      - See lab 2 instructions

Programming

1. `repl`
   - Good for trying out stuff
   - Not so great for longer programs
2. `run` command
   - Send program from host to ESP32 for execution
   - Good for testing out stuff
3. Install program on ESP32
   - Good for autonomous operation

We'll cover all 3 techniques today and next lecture

shell49

- Connect board (USB)
- Start shell49 from command line
  - shell49 automatically connects to ESP32
  - `help` lists available command
  - E.g. `repl`
  - We'll learn about other commands as we need them
    - Feel free to try out stuff
    - You won't break anything
- Problems, questions
  - Piazza
- Source code
  - https://github.com/bboser/shell49

Connect …

- shell49
- repl

run

- Prepare program (.py file) in editor
  - Lots of choices
  - Features to look for:
    - Syntax highlighting (colors)
    - Error checker
  - Use whatever you are comfortable with
  - In class
    - Atom
    - See lab 2 instructions
- Then, in shell49
  - `run hello.py`
  - Led on, led off (great for debugging)

WiFi

1. Connection
2. IP-Address
3. RTC
4. Telnet
5. mDNS
6. boot.py
7. Autonomous operation & remote programming

Summary

- Connect ESP32 to Internet
  - WiFi
  - Internet address, mDNS
- Wireless capabilities
  - Network time
  - Telnet (remote programming)
- Autonomous operation
  - `boot.py`
  - Battery power
- We'll add to this list!